

filesystem interface 230', where the files 192y comprise both http_response 192b and server B digital signature 307. RU 109b and SRU 109 can use write libraries 246 from SRU firmware 109u in order to write files 192y to directory 192x. In other words, although RU 109b is depicted in Figure 4a as writing http_response 192b to directory 192x in a message 427a, in
5 exemplary embodiments memory processor 109d in SRU 109 could write the data, where (i) memory processor 109d receives the data such as http_response 192b from RU 109b via internal bus 212 and (ii) memory processor 109d operates write libraries 246. As depicted in Figure 4a, the collection of steps and messages beginning with step 421 and ending with step/write operation 427 can comprise a step 315, which is also depicted and described in connection with
10 Figure 3c above.

After writing http_response 192b and server digital signature 307 to files 192x, RU 109b can then conduct a step 428 to detach or disconnect from wireless network 102 and enter a sleep state. A step 428 could also comprise RU 109b using a power saving mode (PSM) with wireless network 102. Note that RU 109b can enter a sleep state without receiving a command
15 from device 101 and CPU 101b both (i) after the receipt of http_response 192b and (ii) before entering the sleep state. SRU 109 can then continue to operate a trigger library 245 to detect when the next set of application data 112a is written by device 101 to files 191y in directory 191y and subsequently return to a step 405 and the following steps and messages depicted in Figure 4a in order to transmit any subsequently written application data 112a. Note that a step
20 409 to derive a new ephemeral private key and ephemeral public key and conduct an ECDE key exchange can be omitted for some following application data 112a, and a previous ephemeral public key 409a could be utilized, such as rotating or deriving key 409a only periodically, such as every day or week, and other possibilities exist as well. The period of time for an ephemeral public key 409a to remain active or valid could be specified in cryptographic parameters 192e or
25 cryptographic settings 191e.

At step 429b, SRU 109 with power control unit 109e can send a signal 429c for device 101 and/or CPU 101b to exit from a sleep state. The sleep state for CPU 101b could be entered in a step 429a as depicted above. The signal 429c could also comprise a signal 192f from Figure 2b. The signal 429c could be transmitted via pins 109x. In an exemplary embodiment, SRU
30 109 could temporarily disconnect a set of pins 109x in order to send a signal 429c, and other possibilities exist as well. SRU 109 could perform a dismount and remount step for signal 429c, such as simulating that SRU 109 has been disconnected and reconnected (simulating that a storage device such as SRU 109 has been removed and then reconnected). A physical interface 101p could detect the signal 429c as a signal 192f and wake CPU 101b, such as waking a

computing device when devices are attached. Other possibilities exist as well for SRU 109 to send a signal 429c to device 101 for waking CPU 101b without departing from the scope of the present disclosure.

In some exemplary embodiments, the trigger library 245 could also function as a
5 detection by SRU 109 and processor 109d that RU 109b has written a response 112a to directory 192x. The trigger library 245 could then signal device 101 through pins 109x according to the specified interface or bus for pins 109x, such as signaling that data is available for a read by device 101. Upon receiving the signal, data reporting libraries 101x could conduct the read of response 112a. Thus, trigger library 245 could operate as a trigger for SRU 109 to both (i) wake
10 RU 109b when application data 112a is written to memory 191 and (ii) wake device 101 and CPU 101b when response data 112b is written to memory 192.

At step 429d, device 101 could receive the signal 429c via pins 109x and wake processor 101b from the sleep state entered in a step 429a above. Device 101 could also conduct a polling
233 of directory 192x for files 192y in shared filesystem 230' in order to detect or determine that
15 http_response 192b with server B digital signature 307 has been written by RU 109b to files 192y (such as if CPU 101b exits a sleep state before signal 429b). Device 101 can then conduct a step 316 to read http_response 192b and server digital signature 307 using data reporting libraries 101x. As depicted and described in connection with a step 316 in Figure 3c, device 101 may then also optionally delete http_response 192b and server B digital signature 307 from files
20 192x.

Device 101 can then conduct a step 429e to process and operate on the data contained in http_response 192b, such as adjusting an actuator 101y based on the data in http_response 192b. Or, http_response 192b could comprise simply an acknowledgement that http_request 191b was received and processed by server B 110a. A step 429e by device 101 could also comprise
25 device 101 verifying server B digital signature 307 using a public key from certificate 110d, before processing and operating on the data contained in http_response 192b. Other possibilities exist as well for device 101 to conduct a step 429e to process data in http_response 192b and server B digital signature 307 without departing from the scope of the present disclosure.

Although an http_request 191b is depicted in Figure 4a, other application data 112a
30 could be written by device 101, read by RU 109b, and transmitted by RU 109b through wireless network 102 to server A 106 and server B 110a as well, such as an SMS, FTP file, MQTT message, SIP message, CoAP message, or simply an XML or JSON file. Likewise, although an http_response 192b is depicted in Figure 4, other server B response 112b data could be written by RU 109b when received by RU 109b from wireless network 102 from server A 108 and

server B 110a, using the exemplary protocols or data types from the previous sentence. Other possibilities exist as well for the protocol or format of application data 112b and server B response data 112b without departing from the scope of the present disclosure. In addition, although the use of a trigger library 245 for SRU 109 is depicted in Figure 4a for a SRU 109 to detect that device 101 has written application data 112a for server B 110a in files 192x, RU 109b or SRU 109 could also use polling 232 as depicted in Figure 2b to determine that application data 112a has been written by device 101 into files 191y.

Further, although the use of files 191y and 192y are described within Figure 4a as the structure of data transferred between device 101 and RU 109b, in some embodiments the use of files could be omitted, and alternatively specified or mutually agreed memory addresses within shared nonvolatile memory 109h could be utilized as the location and structure for passing data between device 101 and RU 109b. In other words, a first specified range of memory 109h could comprise the location for application data 112a written by device 101, and a second specified range of memory 109h could comprise the location for server B response 112b data.

Figure 4b

Figure 4b is a flow chart illustrating exemplary steps for conducting a key exchange using PKI keys in order to derive a shared secret key and a symmetric ciphering key, in accordance with exemplary embodiments. A RU 109b or SRU 109 can conduct a key exchange step 409. At step 409, RU 109b can use a step 430 to calculate the sum of SRU static private key SRU SK 109p and the derived ephemeral private key SKe 409a, and then calculate the modulus of the sum using the value n, where the value n can be defined or determined based upon the selected set of cryptographic parameters 191e.

Note that in some exemplary embodiments, the use of a SRU PKI key pair comprising SRU SK 109p and SRU PK 109w could be omitted, and for these embodiments then the input into ECDH key exchange algorithm 443 for a step 409 can comprise the single value of SKe 409a. In other words, a key 223 can be calculated without requiring a SRU SK 109p key for these embodiments (where SRU public key 109w would also be omitted from key exchange calculations by server A in step 414), although the use of a SRU private key SRU SK 109p can be preferred for a step 409 in other embodiments. The selection for the use of SRU PKI key pair in ECDH key exchange algorithms 441 and 442 can be specified in a set of cryptographic parameters 191e.

At step 431, RU 109b can select and read SRU-SP PK 106b. Exemplary data for a step 430 and step 431 will be shown below for this Figure 4b. The combination of output from step 430 and step 431 can be input into an ECDH key exchange algorithm 441 using parameters 191e in order to calculate the shared secret 223.

5 A summary of ECDH as a key exchange algorithm 441 and 442 is included in the Wikipedia article titled "Elliptic Curve Diffie-Hellman" from March 9, 2018, which is herein incorporated by reference. An exemplary embodiment of key exchange algorithm 441 and 442 could comprise a "One-Pass Diffie-Hellman, C(1, 1, ECC CDH)" algorithm as described in section 6.2.2.2 on page 81 of the National Institute of Standards and Technology (NIST) document "NIST SP 800-56A, Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography" from March, 2007 which is hereby incorporated by reference its entirety. Other key exchange algorithms in NIST SP 800-56A could be utilized as well for a key exchange algorithm 441 and 442 in Figure 4b without departing from the scope of the present disclosure. Example calculations for an ECDH key exchange for a key exchange algorithm 441 are shown below. In exemplary embodiments, the key exchange algorithm 442 used by server A 108 can comprise the equivalent key exchange algorithm 441 used by RU 109b in a key exchange step 409.

Other algorithms to derive a secret keys using public keys and private keys may also be utilized in a key exchange algorithm 409, such as, but not limited to, the American National Standards Institute (ANSI) standard X-9.63. Cryptographic parameters 191e can also include information, values, or settings for conducting (i) a key exchange algorithm 441 in step 409 (and steps 442) and (ii) a key derivation function 443 in order to derive a commonly shared symmetric encryption key 223a. As contemplated herein, the terms "cryptographic settings 192e" and "cryptographic parameters 191e", and "parameters 191e" can be equivalent, and cryptographic settings 192e could be a subset of cryptographic parameters 191e, where the subset is selected by device 101.

Parameters 191e input into a key exchange algorithm 441 and 442 can include (i) a time-to-live for a shared secret 223 and symmetric ciphering key 223a that is derived, and (ii) a supported point formats extension, where the supported point formats extension could comprise uncompressed, compressed prime, or "compressed char2" formats, as specified in ANSI X-9.62 and related IETF standards. In other words, (i) an ECC PKI keys input into a key exchange algorithm 441 and 442 and (ii) shared secrets output from key exchange algorithm 441 and 442 may have several different formats and a set of parameters 191e can be useful to specify the format.

Exemplary data and numbers can be provided to demonstrate the calculations for (i) ECC point addition step 444, (ii) step 430 to combine private keys, and (iii) key exchange step 409. Parameters 191e can include the elliptic curve of “secp128r1” with key lengths of 128 bit long keys. Although the exemplary numbers for keys, points, and a named curve are provided, other values for keys, points, and named ECC curves could be utilized as well. Other example named curves and parameters 191e could comprise a different curve from IETF RFC 5480 or similar standards.

The ephemeral private key SKe 409a can comprise the exemplary following number, and can be stored in RU 109b after a key pair generation step at the beginning of a step 409 as described in Figure 4a:

330225223315954763304200471137380016969

The SRU static private key SRU SK 109p can comprise the exemplary following number, and can be recorded in SRU 109 in PKI keys 109v before a step 409, such as during SRU manufacturing, SRU configuration, or SRU distribution (e.g. step 252 in Figure 2f):

209122135174501513062984245101620420255

Note that the private keys SKe 409a and SRU SK 101p above correspond to the public keys (i) PKe 409b received by server A 108 in message 412 and (ii) SRU PK 109w recorded by a server A 108, as depicted in Figure 1a. The public keys for server A 109 can also have the exemplary numeric values below for step 414 for the example numbers given above.

Note that the public key SRU-SP PK 106b corresponds to the private key SRU-SP SK 106a. The server static public key SRU PK 106b for a step 409 can comprise the following exemplary values with X and Y numbers (or “coordinates”) of:

X: 319423829544285733939020505180109110187

Y: 242179187598040154943588326777101424083

RU 109b can conduct step 430 to calculate the sum of SRU static private key SRU SK 101p and the derived ephemeral private key SKe 409a, and then calculate the modulus of the sum using the value n, where the value n can be defined or determined based upon the selected set of cryptographic parameters 191e. For the exemplary values for keys above, when using the named elliptic curve secp128r1, the value of n can comprise the decimal number:

340282366762482138443322565580356624661

Consequently, the modulus of (i) the sum of private keys SRU SK 101p and SKe 409a and (ii) the value n above will equal the following number for a step 430:

199064991727974137923862150658643812563

5

An ECDH key exchange algorithm 441 in key exchange step 409 can input (i) the static public key for SRU-SP PK 106b and (ii) the value calculated from a step 430 (e.g. $(\text{SRU SK } 101p + \text{SKe } 409a) \bmod n$). The output of ECDH key exchange algorithm 441 in key exchange step 409 can be the secret value or point X1 223. Note that the secret X1 223 as derived by RU 109b in a key exchange step 409 equals or is the same numeric value as the secret X1 223 derived by server A 108 in a key exchange step 414 below. An exemplary point or value for secret X1 223 calculated by RU 109b using a key exchange step 409 using the above exemplary numeric values for SKe 409a, SRU SK 101p, and SRU-SP PK 106b would be:

10

15

X: 283470377637256529257925581999478035172

Y: 117395441238388206677723127104680679540

20

Note that the above exemplary value for X1 223 can also be derived through the operation of $(\text{SKe } 409a * \text{SRU-SP PK } 106b) + (\text{SRU SK } 101p * \text{SRU-SP PK } 106b)$. In other words, the shared secret can be derived through either (i) an addition of secret keys 409a and 101p or (ii) an elliptic curve point addition using points, where the points are determined by at least secret keys 409a and 101p.

25

30

For a key exchange step 409, derived shared secret key X1 223 can be input into a key derivation function 443 where the key derivation function 443 can be equivalent to the key derivation function 443 below for a key derivation step 414 by server A 108. Note that for key derivation steps in the present disclosure, the X coordinate of a derived shared secret can be taken or used as input into the key derivation function 443. The output of a key derivation function 443 can comprise both (i) a symmetric ciphering key 223a and (ii) a MAC key 223b. MAC key 223b can be used with a symmetric ciphering algorithm in order to generate a MAC code, such that the other party using the same key 223a and MAC key 223b can process the ciphertext and calculate the same MAC code in order to verify message integrity. RU 109a can subsequently use keys 223a and 223b in an encryption step 411 and a decryption step 425 in Figure 4a above.

Key derivation function 443 can use a secure hash algorithm such as, but not limited to, SHA-256, SHA-384, SHA-3, etc. and additional values such as a text string with secret X1 223. The specification of a secure hash algorithm and the text string for use with a key derivation function 443 could be commonly shared between server A 108 and RU 109b by commonly shared parameters 191e. The output of a secure hash algorithm within a key derivation function 443 could have a subset of bits selected or possibly a secure hash expanded in order to obtain the number of bits required for a symmetric key with a symmetric ciphering algorithm, such as key 223a. A key derivation function (KDF) 443 could comprise a KDF compatible with or specified by ANSI standards for "X9.63 Key Derivation Function". Other possibilities exist for a key derivation function 443 to convert a secret X1 223 into a symmetric ciphering key 223a and a MAC key 223b without departing from the scope of the present disclosure. As contemplated in the present disclosure, although an ECC public key such as secret X1 223 can comprise a coordinate with an X value and a Y value, in exemplary embodiments a single number comprising the X value can be selected and input into a key derivation function 443.

For a key exchange step 414 by server A 108, server A 108 can select the SRU PK 109w from for the specific SRU 109 and RU 109b using the identity SRU ID 109i received in message 412. For some exemplary embodiments, the identity SRU ID 109i can comprise a secure hash value over SRU PK 109w, and thus server A 108 could select the SRU PK 109w using the secure hash value over SRU PK 109w as the identity (where server A 108 could store the has values for all public keys SRU PK 109w). Server A 108 can also receive the ephemeral public key PKe 409b in message 412, possibly along with the set of cryptographic parameters 191e. Server A 108 could also record set of cryptographic parameters 191e associated with both PKe 409b and SRU PK 109w in a certificate for SRU PK 109w. RU 109b could also record a set of cryptographic parameters 191e in a certificate for SRU-SP PK 106b, and all of the PKI keys depicted in Figure 4b could use the same set of cryptographic parameters as those specified in a certificate for SRU-SP PK 106b recorded by SRU 109. In other words, in exemplary embodiments, a certificate for SRU-SP PK 106b and a certificate for SRU PK 109w could use the same set of cryptographic parameters 191e, such as specifying the same named elliptic curve and same key lengths.

Key exchange step 414 for a sever A 108 depicted in Figure 4b can correspond to key exchange 409 for RU 109b. Key exchange step 414 can comprise (i) inputting or using an ECC point addition operation 444 over PKe 409b and SRU PK 109w, and (ii) the server static private key SRU SK 106a with an ECDH key exchange algorithm 442, which can comprise the same or equivalent key exchange algorithm 441 depicted and described in connection with key exchange

step 409 described above. Other elements or algorithms within a key exchange step 414 can be equivalent to a key exchange step 409 above, including the use of shared parameters 191e. The output of a key exchange algorithm 414 can comprise a secret key or value X1 223 as well as symmetric ciphering key 223a and MAC key 223b.

5 Server A 108 can conduct an ECC point addition step 444 in order to combine (i) SRU static public key SRU PK 109w recorded in a database for server A 108 with (ii) received ephemeral public key PKe 409a received in a message 412. Exemplary calculations for an ECC point addition 444 can comprise the calculations shown for point addition in the Wikipedia article for “Elliptic Curve Point Multiplication” dated May 15, 2018, which is herein
10 incorporated by reference in its entirety. The output of an ECC point addition step 444 can be input into ECDH key exchange algorithm 442. An ECC point addition step 444 can be performed with a named elliptic curve from a selected set of parameters 191e, where the set of parameters 191e could be specified in a certificate for SRU PK 109w.

 At step 445, server A 108 can select the static private key SRU-SP SK 106a. In
15 exemplary embodiments, server A 108 could record a plurality of different private keys for SRU-SP, and server A 108 could select the private key SRU-SP SK 106a corresponding to the public key SRU-SP PK 106b recorded in SRU 109 based on the identity SRU ID 109i received in message 412. In this manner, by using a plurality of different keys SRU-SP SK 106a with different SRU 109, security can be increased, since the compromise of a single key SRU-SP SK
20 106a would affect only a subset of SRU 109. In other words, for exemplary embodiments, the security of an overall system 100 with a plurality of different units SRU 109 is increased by using different keys SRU-SP SK 106a with different SRU 109. For this embodiment, different SRU 109 would record different values for SRU-SP PK 106b. Mapping of SRU-SP SK 106a to different SRU 109 could be obtained by server A 109 associating the identity SRU ID 109i with
25 different values of SRU-SP SK 106a.

 Exemplary data and numbers can be provided to demonstrate the calculations for a key exchange step 414. Parameters 191e can comprise the named elliptic curve of “secp128r1” with key lengths of 128 bit long keys. Although the exemplary numbers for keys, points, and a named curve are provided, other values for keys, points, and named ECC curves could be
30 utilized as well.

 The server static private key SRU-SP SK 106a selected for SRU 109 in a step 445 based on the identity SRU ID 109i from message 412 can comprise the exemplary following number, and can be recorded by server A 108:

246768250079261690512638148137618184294

Note that the above value for SRU-SP SK 106a corresponds to the public key SRU-SP PK 106b in a step 431. The ephemeral public key PKe 409b received in message 412 can comprise the following exemplary values with X and Y numbers (or “coordinates”) of:

5 X: 239356896551017663412726672579682627094

Y: 209570745539973929739512070961905802250

Note that the above ephemeral public key PKe 409b corresponds to the ephemeral private key SKe 409a above from step 409. The SRU static public key SRU PK 109w for a step 10 444 can comprise the following exemplary values with X and Y numbers (or “coordinates”) of:

X: 203473426612520506812902270038827201196

Y: 64318327833120582836973711848343026891

Note that the above value for SRU PK 109w corresponds to the secret or private key for 15 SRU SK 109p from a step 430. ECC point addition step 444 by server A 108 can combine the SRU static public key SRU-SP PK 109w and the ephemeral public key PKe 409a in order to output the following value for point calculated in a step 444:

X: 59121922812458579600446751790662160796

Y: 304934509235778268978955867170200917057

20 An ECDH key exchange algorithm 442 in key exchange step 414 can input (i) the static private key for SRU-SP SK 106a from a step 445 and (ii) the value calculated from the ECC point addition operation in step 444 (e.g. SRU PK 101w + PKe 409b) mod n). The output of ECDH key exchange algorithm 442 in key exchange step 414 can be the secret value or point 25 X1 223. Note that the secret X1 223 as derived by RU 109b in a key exchange step 409 above equals or is the same numeric value as the secret X1 223 derived by server A 108 in a key derivation step 414. An exemplary number or value for secret X1 233 calculated by server A 108 using a key exchange step 414 using the above exemplary numeric values for PKe 409a, SRU PK 101w, and SRU-SP SK 106a would be:

30 X: 283470377637256529257925581999478035172

Y: 117395441238388206677723127104680679540

After conducting an ECDH key exchange algorithm 442 in a key exchange step 414, server A 108 can input the shared secret key X1 223 into a key derivation function 443. The key derivation function 443 can comprise the same key derivation function 443 used by a RU 109b in a step 409 above. The output of a key derivation function 443 can comprise both (i) a symmetric ciphering key 223a and (ii) a MAC key 223b. Symmetric ciphering key 223a and MAC key 223b can be subsequently used by server A 108 in a decryption step 415 and an encryption step 423 in Figure 4a above. MAC key 223b can be used with a symmetric ciphering algorithm in order to generate a MAC code, such that the other party using the same key 223a and MAC key 223b can process the ciphertext and calculate the same MAC code in order to verify message integrity.

The value of shared secret X1 223 could also be computed by server A 108 by the following equivalent calculation: $(\text{SRU-SP SK } 106a * \text{PKe } 409e) + (\text{SRU-SP SK } 106a * \text{SRU PK } 109w)$. In other words, the calculation or generation of shared secret X1 223 by server A 108 can comprise an elliptic curve point addition using at least the device or RU ephemeral public key 409e and the SRU static public key 109w.

Figure 5a

Figure 5a is a graphical illustration of hardware, firmware, and software components for a device and a storage radio unit, including a shared memory interface operating in RAM for the storage radio unit, in accordance with exemplary embodiments. A device 101 can include and operate a storage radio unit (SRU) 109. Although embodiments depicted above such as with Figure 1a, Figure 2a, etc. depict shared memory 109h for device 101 and SRU 109 as operating or within a nonvolatile memory 109a such as flash memory within SRU 109, for some exemplary embodiments a shared memory interface comprising device write memory 191 and RU write memory 192 can reside and operate within SRU RAM memory 213, as depicted in Figure 5a. Shared memory 109h could also record and operate a shared filesystem interface 230' from Figure 2a, Figure 3a, Figure 3b, and Figure 3c, where the filesystem and files 191y and 192y are stored within volatile memory 213. In other words, (i) device write memory 191 (and RU read memory) in Figure 5a within RAM 213 could store a directory 191x and files 191y as depicted in Figure 3a, and (ii) RU write memory (and device read memory) 192 in Figure 5a within RAM 213 could store a directory 192x and files 192y as depicted in Figure 3b.

Memory processor 109d could include the logic, drivers, firmware, etc. to store, configure, and operate the shared filesystem interface 230' within RAM 213. For example, the configuration process 231 could detect the filesystem 229 for nonvolatile memory 101g

recorded for device 101 from Figure 2f, and subsequently use libraries in SRU firmware 109u to write directory 191x in memory 191 and directory 192x in memory 192. The usage of memory 191 and memory 192 for recording files could comprise a RAM drive. Processor 109d in SRU 109 could operate and store a shared filesystem interface 230' in RAM 213 in a manner that (x) directories 191x and 192x appear to device 101 to operate within the same logical drive, partition, or storage unit as filesystem 229 within nonvolatile memory 109g, but (y) SRU 109 records the directories 191x and 192x in RAM 213. Or, in other embodiments, processor 109d in SRU 109 could operate and store a shared filesystem interface 230' in RAM 213 in a manner that directories 191x and 192x appear to device 101 to operate within a different or separate logical drive, partition, or storage unit as filesystem 229 within nonvolatile memory 109g.

Benefits for operating shared memory 109h in RAM 213 can be that read and write operations can be faster and utilize less energy. For standard computing components, read and write functions to RAM are faster and utilize less energy than read and write operations to nonvolatile memory such as flash memory. A tradeoff is that RAM 213 could be flushed or cleared of usable binary data when power is removed from SRU 109. For files 191y in memory 191 and files 192y in memory 192, volatile storage in RAM 213 could require that configuration process 231 for memory 109h in RAM 213 is conducted each time that SRU 109 is powered (as opposed to a single time or very occasionally for configuration process 231 operating on memory 109h in nonvolatile memory 109a)

However, for some exemplary embodiments, an SRU 109 could use persistent memory, known as nonvolatile RAM (NVRAM). NVRAM can use transistor gate technology recording binary states within memory cells when electrical power is removed from the memory. NVRAM for RAM 213 in SRU 109 could use a memory chip supporting the memory technology commercially available in the NVDIMM-P specifications as of 2018 (but not with the DIMM physical form and rather in a single chip or silicon area within SRU 109 with NVRAM capabilities). In exemplary embodiments, SRU 109 could use persistent memory or nonvolatile RAM memory in RAM 213 for shared memory 109h, and thus obtain the benefits of faster read and write times (with lower power consumption) while also having the benefits of nonvolatile storage, such that files 191y and 192y could continue to be recorded in RAM 213 in a shared nonvolatile memory 230 when power is removed from SRU 109.

Other than use of shared memory 191 and 192 in RAM 213 (which can comprise NVRAM or persistent memory), the components, firmware, and software for a device 101 and SRU 109 depicted in a Figure 5a could be the same or equivalent those depicted in Figure 1b, Figure 2a, etc. Note that SRU firmware 109u in Figure 5a is configured to support the operation

of a shared memory interface 230 between device 101 and SRU 109 in RAM 213, which would be another difference for an SRU 109 in Figure 2a and Figure 5a. In exemplary embodiments, the same device 101 drivers 101g and libraries 101x can be used with either configuration of (i) files 191y and 192y operating in nonvolatile flash memory 109a within SRU 109, or (ii) files 191y and 192y operating in RAM 213 within SRU 109. In other words, the firmware of 109u can operate internal drivers and controlling software such that device 101 does not need to be aware of the location for storage and/or recording of files 191y and 192y within SRU 109. In addition, although the use of SR RAM 213 within SRU 109 in Figure 5a, in some exemplary embodiments the RAM 213 could reside or operate outside of a SRU 109 and within device 101, such as a RAM 213 functioning equivalent to the memory 109h in Figure 7 below.

Figure 5b

Figure 5b is a flow chart illustrating exemplary steps for configuring a storage radio unit to communicate with a device using a shared nonvolatile memory interface in RAM, in accordance with exemplary embodiments. Although embodiments depicted above such as with Figure 1a, Figure 2a, etc. depict shared memory 109h for device 101 and SRU 109 as operating or within a nonvolatile memory 109a such as flash memory within SRU 109, for some exemplary embodiments a shared memory interface comprising device write memory 191 and RU write memory 192 can reside and operate within SRU RAM memory 213, as depicted in Figure 5a. Figure 5b depicts exemplary steps for SRU 109 and RU 109b to use a shared memory interface 230 in RAM 213. A filesystem recorded in RAM 213 could comprise or be referred to as a flash drive. RAM memory 213 can also comprise persistent RAM or NVRAM, as described in Figure 5a.

At step 253, SRU 109 can be manufactured with both RAM 213 and nonvolatile memory 109a. In exemplary embodiments, RAM 213 can comprise portions of both (i) volatile RAM such as SRAM or DRAM and (ii) persistent memory RAM, such as NVRAM. A shared memory interface 230 could use either volatile RAM or persistent RAM for the steps in Figure 5a. At step 254, a filesystem 229 could be written to general memory 101g for device 101, and files for device 101 such as an OS 101h could be written as well. A step 254 could be conducted by a configuration unit 110d in Figure 1a, or by device 101 (similar to a personal computer formatting a flash memory drive such as a USB memory stick). If a configuration unit 110d performs a step 254, then SRU 109 could be inserted inside device 101 after a step 254. At step 401c, device 101 can power SRU 109. SRU 109 can then conduct a boot process in step 402 conducted along with step 401c, at the conclusion of a step 401c, device 101 can mount

filesystem 229. In exemplary embodiments, a power control unit 109e sets the power for RU 109b in a sleep state both (i) after RU 109b completes a boot process along with SRU 109 in a step 402 until (ii) device 101 writes data for RU 109b in files 191y. In other words, for some embodiments, a trigger 245 could wake RU 109b if data other than application data 112a is written to files 191y, such as, but not limited to, radio parameters input 191a, command script 191g, device date and time 191h, etc.

At step 255, SRU 109 detects the type of filesystem 229 recorded and operating with nonvolatile memory 109g as general device memory in nonvolatile memory 109a of SRU 109. A step 255 is also depicted and describe in connection with Figure 2f above. SRU 109 can select a filesystem driver 242 from SRU firmware 109u that is compatible with the detected filesystem type for filesystem 229 in a step 255. Note that in exemplary embodiments, when either (a) RAM 213 comprises volatile memory or (b) a step 255 and subsequent steps are being conducted the first time with a NVRAM for RAM 213, then a directory 191x may not be present or readable for device 101. Consequently, SRU 109 after boot and detection of type of filesystem 229 in memory 109g, SRU 109 can in a step 501 conduct a configuration process 231 for SRU 109 to operate shared filesystem interface 230' for device 101 and SRU 109 in RAM 213. A step 501 can include SRU 109 selecting, loading, and operating drivers or packages from firmware 109u, including possibly kernel modules.

At step 502, SRU 109 can conduct a step 257 from Figure 2f, where the actions can be performed for RAM memory 213 instead of nonvolatile flash memory 109a for a step 257 from Figure 2f. At step 257 with a step 502, SRU 109 can write SRU 109 device write directory 191x to shared memory 109h in RAM 213 using the selected file system driver 242. SRU 109 can use (i) memory write libraries 217c from boot firmware and a RAM 213 memory interface in order to change transistor states, or memory blocks in order to write binary data to RAM 213, where the resulting binary data written can comprise a device write directory 191x for the file system 229. Device write directory 191x at step 257 with a step 502 can also include base data such as the exemplary files and data depicted in Figure 3a above. At step 502, SRU 109 can also write RU write directory 192x and associated files for startup in shared memory interface 230 in RAM 213.

As mentioned above in Figure 5a, the operation of a shared memory interface 230 in RAM 213 can be transparent to device 101, and directories 191x and 192x could be viewed and written to by CPU 101b under a collective root directory that includes all of 191x, 192x, and general files for device 101 in memory 109g. For example, a root directory of "/" in shared filesystem interface 230' can have directories of "/device-write/", "/ru-write/", and "/general/",

where device 101 could write files that don't pertain to the operation of RU 109b in the "/general/" directory. After device 101 mounts filesystem 229, the directory 191x and 192x can appear within the filesystem 229. In exemplary embodiments with shared filesystem interface 230' in either RAM 213 or nonvolatile memory 109g, SRU 109 can operate and control the shared filesystem interface 230' so that directory 191x and directory 192x appear to reside within filesystem 229, although directory 191x and directory 192x operate or are stored in shared memory 109h, which could be in RAM 213.

SRU 109 using processor 109d can begin polling 232 RAM 213 for changes or updates written by device 101, in order to detect data written and ready to be processed by RU 109. Note that either processor 109c or processor 109d could conduct a polling 232. Or, a trigger library 245 operating in a processor 109d could be used instead of polling 232, since memory processor 109d normally processes all data communicated with device 101 through bus 202. A polling process 232 or trigger library 245 could be integrated with a step 503, such that the polling process 232 or trigger library 245 conducts a step 503 to determine if device 101 has written files 191y or data to directory 191x.

At step 503, SRU 109 using processor 109d could determine if device 101 has written files 191y or data to directory 191x. If no data or files 191y are written, then SRU 109 could continue (i) the polling 232 process as depicted in Figure 5b, or (ii) monitoring of directory 191x via trigger library 245. Note that processor 109d could continue to operate separately as a memory controller for device nonvolatile memory 109g during step 232 and step 503, such that device 101 could read and write files to memory 109g for other purposes than communicating with RU 109b, such as device 101 logging data or reading files for device 101 in memory 109g. Trigger library 245 could be activated when device 101 writes data for a memory address that resides in shared memory 109h operating in RAM 213.

SRU 109 upon determining that at least one file 191y for directory 191x has been written to shared memory 109h could proceed to a step 504. At step 504, SRU 109 using processor 109d could (i) read file 191y in RAM 213 written by device 101 in a step 313, (ii) use power control unit 109e to power up RU 109b or change RU 109b from an off or sleep state to an active state, (iii) instruct or provide data for RU 109b through internal bus 212 to transmit data from files 191y, which could comprise application data 112a. In a sub-steps (iv) of step 504, RU 109b could receive server B response data 112b for the application data 112a transmitted in sub-step (iii). RU 109b in conjunction with SRU 109 could write the server B response data 112b to 192y, and (v) power control unit 109e could either power off RU 109b or return RU 109b to a sleep state. In exemplary embodiments, after a sub-step (v), the power down of RU

109b could comprise RU 109b entering a sleep or deep sleep state, where RU 109b continues to accept wake signaling through internal bus 212 and from a power control unit 109e. In other words, RU 109b does not need to conduct a boot process in order to return to an active state and subsequently read or receive data from files 191y.

5 At step 505, SRU 109 can continue operating until device 101 disconnects SRU 109 from electrical power, such as when device 101 shuts down or enters a deep sleep state where SRU 109 is no longer powered. If power is continued to be provided to SRU 109, then SRU 109 could continue to step 232 in Figure 5b, where SRU 109 continues to monitor directory 191x and files 191x for data input from device 101, and then subsequent take the steps depicted in
10 in order to process and transmit the data, as well as receive and record a response from a server B 110a.

Figure 6a

Figure 6a is a flow chart illustrating exemplary steps for storage radio unit to
15 communicate with a device using a shared nonvolatile memory interface, in accordance with exemplary embodiments. Figure 6a depicts exemplary steps for an SRU 109 to communicate with device 101, the steps depicted in Figure 6a also continue with Figure 6b below. In the Figure 6b below, SRU 109 uses application data 112a received from device 101 in Figure 6a to operate a radio unit (RU) 109b to transmit the application data 112a and receive a server B
20 response 112a. The steps for SRU 109 in Figure 6a include steps previously depicted and described, as well as providing additional details for the operation of a SRU 109 and device 101.

At step 601 SRU 109 can complete manufacturing and configuration for use with a device 101. SRU 109 can be manufactured with the components from a system 200a. SRU 109 can also undergo a step 260 from Figure 2f, where SRU 109 is configured with boot firmware
25 217, a boot configuration 218, SRU firmware 109u, and radio unit firmware 222. A file system 229 for device memory 109g can also be written and formatted into the nonvolatile memory 109a of SRU 109. SRU 109 can also conduct a shared filesystem configuration 231 step. Note that the use of a shared filesystem configuration step 231 could omit the use of a particular file system, and an SRU 109 and device 101 could communicate through shared memory 191 and
30 192 using pre-agreed memory addresses in a shared memory 101h for SRU firmware 109u and device driver 101g for SRU 109. A step 601 can include distribution of SRU 109 and also insertion of SRU 109 into device 101.

At step 401c, device 101 powers on and load device driver 101g for SRU 109 and libraries 101x for in order to for device 101 to communicate using device driver 101g using

shared filesystem 230'. Libraries 101x can include the logic, instructions for CPU 101b, or machine executable code for device 101 to write data such as application data 112a to a directory 191x within shared memory 109g in the form of a file 191y. Libraries 101x can include the logic, instructions for CPU 101b, or machine executable code for device 101 to read data such as server B response 112b data to from directory 192x in the form of a file 192y. A step 401c is also depicted and described in connection with Figure 4a. A step 401c can also comprise OS 101h for device 101 mounting a "drive" for memory 109f, where memory 109f from Figure 1c can include both device general memory 109g and shared memory 109h. Memory 109f can include a filesystem 229.

At step 402, device 101 can provide power to SRU 109 via pins 109x. SRU 109 can boot with boot firmware 217, boot configuration 218, where the boot firmware and configuration loads SRU firmware 109u, and an SRU operating configuration 249b. SRU 109 can power RU 109b and RU 109b can also boot and load radio unit firmware 222. An eUICC 113 can also boot up in a step 403 and load and operate firmware for the eUICC in a step 402. Upon power up of SRU 109, power control unit 109e can power RU 109b. At the conclusion of a boot process in step 402 and steps 602 and 603 below, power control unit 109e can command RU 109b to enter a sleep or deep sleep state, such that waking RU 109b does not require a full boot process for RU 109b.

At step 602, RU 109b can monitor wireless network 102 and read or receive a value for network time UTC from a system information block (SIB) broadcast by wireless network 102.

At step 603, RU 109b can write in 192x values or files 192y for network status 308, also update network date-time 309b. As mentioned above, the steps for RU 109b to write files 192y in directory 192x could comprise RU 109b sending data to processor 109d via an internal bus 212, and processor 109d using libraries 246 can write the data to directory 192x on behalf of RU 109b. Note that other startup data for device 101 to read can be written in a step 602 as well, such as files 192y for radio parameters 192a, a log file 306, power control settings 192g, cryptographic parameters available or operating 192e, and other exemplary data depicted or described in Figure 3b. Although not depicted for files 192y written by RU 109b, RU 109b could also obtain geographical coordinates or information from in a step 602 and write the coordinates or location data to files 192y.

Network status 308 in a step 603 can indicate which wireless network 102 is available to RU 109b as well as signal strength for RF signals received from wireless network 102. Note that SRU 109 could also record and use network date-time 309b. SRU 109 could use the network date-time 309b with RF clock 205 and PLL 211 going forward to properly track time.

In exemplary embodiments, the date/time values 304a (from Figure 3a) can be recorded by SRU 109 using the system time for SRU 109 which can be initially set with network date-time 309b from RU 109b in a step 602.

After RU 109b records files 192y from a step 603, in a step 604 power control unit 109e can sleep RU 109b, such that RU 109b is no longer in an active state and powering RF front end 203 (excluding RF clock 205 which can be used to drive processor 109d). SRU 109 can then instruct memory processor 109d to load and operate with a trigger library 245 in a step 245a, where trigger library 245 can detect when device 101 writes files 191y to directory 191x. Trigger library 245 can operate in a continuous manner when processor 109d is in an active state. For embodiments where processor 109d enters a sleep state, then trigger library 245 can resume operation upon change of processor 109d back to an active state.

At step 605, device 101 can read from 192y the radio network status 308 and network date-time 309b, and device 101 can also apply network date-time 309b to clock 101m. In this manner, clock 101m for device 101 can be synchronized with network date and time broadcast by wireless network 102. Device 101 could use a device polling 233 with either driver 101g or library 101x to determine when data is written to files 192y. Device 101 can then proceed with normal operation, such as collecting data from a sensor 101f for operating an actuator 101y. Device 101 can then process application data 112a for transmission to server B 110a. The application data 112a can comprise any of the exemplary application data 112a described in Figure 1a or Figure 3a.

At step 313, device 101 can write the application data 112a for server B 110a in files 191y. The use of a step 313 by device 101 is also depicted and described in connection with Figure 3c above. At step 429a, device 101 can enter CPU 101b in a sleep state, where step 429a is also depicted and described in connection with Figure 4a above. At step 405a trigger library 245 with memory processor 109d in SRU 109 can detect updated files 191y with application data 112a in directory 191x from Step 313. Note that data for files 191y can normally pass through processor 109d since processor 109d controls both (i) external bus 202 to receive the application data 112a and (ii) memory core interface 109m in order to write the data to memory 109h. Consequently, trigger library 245 in processor 109d can detect when data is written to memory addresses or files 109y allocated to memory 109h from a configuration step 231.

Other possibilities for the operation and function of a trigger library 245 exist as well, without departing from the scope of the present disclosure. At the conclusion of the step 405a, SRU 109 has received application data 112a from device 101 and can subsequently take steps for RU 109b to transmit the data. The subsequent steps are depicted in Figure 6b below. For

some embodiments, SRU 109 could omit the use of a trigger library 245 and trigger process or step to detect when device 101 writes application data 112a to files 191x and instead RU 109b with processor 109c could conduct a polling step 232 as depicted in Figure 2b above in order to detect when device 101 writes application data 112a to files 191x.

5

Figure 6b

Figure 6b is a flow chart illustrating exemplary steps for a storage radio unit to operate a radio unit and a shared nonvolatile memory interface, in accordance with exemplary embodiments. Figure 6b depicts exemplary steps for an SRU 109 to operate an RU 109b to send and receive data, and also continues from Figure 6a above, where SRU 109 receives application data 112a from device 101. In Figure 6b, SRU 109 can operate a radio unit (RU) 109b to transmit the application data 112a and receive a server B response 112a. The steps for SRU 109 in Figure 6b include steps previously depicted and described, as well as providing additional details for the operation of a SRU 109 and RU 109b.

10

15

At step 405b, power control unit 109e can receive a signal from trigger library 245 that files 191y or data is available for RU 109b. Power control unit 109e can comprise a module or library operating within memory processor 109d. Power control unit 109e can change RU 109b to an active state from a sleep state entered in a step 604 above. Note that RU 109b can remain continuously in a sleep state after a step 604 until a step 405b, which is a sleep state for RU 109b until device 101 has written application data 112a. RU 109b can conduct an attach step 407b to connect and register with wireless network 102 using eUICC 113. Attach step 407b can use network access credentials and network parameters recorded with an operational profile 113d in eUICC 113. For some embodiments, the use of an eUICC 113 is not required, and RU 109b could connect with wireless network 102 without an eUICC 113, such as connecting to a WiFi network using a pre-shared key or EAP-TLS for the WiFi network.

20

25

For a step 405b, power control unit 109e can also optionally temporarily disable the write or erase function of device 101 for memory 101g or memory 101h when either (a) RU 109b is in a transmitting or attached state with wireless network 102, or (b) when RU 109b is in a transmitting state and drawing a current above a threshold, such as an exemplary 100 mA. This exemplary functionality for a power control unit 109e to disable write or erase functionality for device 101 can be used by power control unit 109e any time that RU 109b transmits, and not just in a step 405b.

30

At step 408b, RU 109b can read application data 112a for server B 110a from files 191y. Note that memory processor 109d could read the data from memory 109h and write the data to

RU 109b via internal bus 212. In other words, for some embodiments, memory processor 109d could read the data on behalf of RU 109b. The use of (a) a step 408b and memory 109h to store application data 112a supports (b) RU 109b and CPU 101b in device 101 separately using sleep states, such that both RU 109b and processor 101b do not have to be active or in active states at the same time, in order to communicate data between CPU 101b and RU 109b. If data is passed directly between CPU 101b in device 101 and RU 109b (such as with AT commands and related conventional technology), then both CPU 101b and RU 109b would normally need to be in active states at the same time. In exemplary embodiments and through the use of separate and independent sleep states for RU 109b and CPU 101b, overall power consumption for device 101 can be reduced and a maximum current draw on battery 101n for device 101 could also be reduced, thereby extending the battery life (e.g. battery life can be longer when maximum current draw is lower).

At step 409 in Figure 6b, RU 109b can use cryptographic algorithms 109q and a set of cryptographic parameters 191e (or cryptographic settings) to derive an ephemeral ECC PKI key pair comprising SKe 409a and PKe 409b. Note that cryptographic parameters 191e could be written to files 191y by device 101, or SRU 109 could record the cryptographic parameters 191e without using or requiring a file 191y. RU 109b can use a random number generated and information entropy accessible to RU 109b (including via RF clock 205 and RF front end 203) in order to generate a random number to process a number for SK 409a. RU 109b can then conduct a sub-step 441 of a step 409, as depicted and described in connection with a key exchange step 409 in Figure 4b above. In a sub-step 411, RU 109b can conduct an ECDH exchange using recorded or stored PKI keys. RU 109b can calculate a secret shared key 223 as equal to $\{ [(SKe\ 409a + SRU\ SK\ 101p) \bmod n] * SRU\text{-}SP\ PK\ 106b \}$. Exemplary values for an ECDH key exchange in step 441 are depicted and described in connection with Figure 4b. A step 441 could also use a set of cryptographic parameters 191e, such as values of A, B, Q, and n for a named ECC curve associated with all of the PKI keys used in a step 441.

Although not depicted in Figure 6b, RU 109b could then also conduct a key derivation function step 443 using the secret shared key 223 in order to calculate a symmetric ciphering key 223a and MAC key 223b. SRU 109 could store the resulting keys 223a and 223b in protected memory 109r, and re-use the keys 223a and 223b at a later time until parameters 191e specify key rotation. In other words, (i) an RU 109b does not need to conduct a step 409 and 441 each time application data 112a is received by SRU 109 for RU 109b, but rather (ii) a step 409 and 441 may be conducted only periodically, such as an exemplary time of once a day, or once a week, and other possibilities exist as well for the periodic timing of conducting a step 409

and 441 without departing from the scope of the present disclosure. For times when a step 409 and 441 are skipped, then RU 101b could read the keys 223a and 223b from protected memory 109r. The need or use of a rotated key 223a could be signaled to server A 108 via the use of a new, different ephemeral public key 409b in a message 412 in Figure 4a. Omission of key 409b or re-use of the same key 409b in a message 412 (depicted in Figure 4a) can indicate to server A 108 that a previously calculated symmetric key 223a is used to encrypt data.

As noted above in connection with Figure 4a, for some exemplary embodiments a step 409 and step 441 could also be omitted, and RU 109b could use a recorded or stored shared secret symmetric ciphering key 127b instead of a derived key 223a as an encryption key for subsequent steps. Server A 108 could also record the shared secret key 127b, and select key 127b using an SRU identity 109i from a message 412 in Figure 4a.

At step 410, RU 109b can compress application data 112a with standard compression libraries, and the name or identifier of a compression library could be included with parameters 191e. At step 411, RU 109b also encrypts the compressed application data 112a with symmetric ciphering key 223a and a symmetric ciphering algorithm such as AES-128 from a set of cryptographic algorithms 109q. In some exemplary embodiments, steps 409, 441, and 411 could be omitted, if device 101 performs the compression and encryption, and in these cases application data 112a received by SRU 109 from device 101 could be in the form of a ciphertext. For these embodiments, device 101 could specify that encryption by RU 109b is omitted using values or fields set in cryptographic parameters 191e, which could be written by device in files 191y.

At step 412, RU 109b can transmit the compressed, encrypted application data 112a through wireless network 102 to server A 108. A step or message 412 is also depicted and described in connection with Figure 4a. Although not depicted in Figure 6b (but depicted and described in connection with Figure 4a), server A 108 could also send a response or confirmation that message 412 has been received by server A 108. Server A 108 and server B 110a can then conduct the series of steps and messages to transfer the application data 112a to server B 110a and also for server A 108 to receive server B response 112b data from server B 110a. Server A 108 can then use IP network 111 (which was used by server A 108 to receive application data 112a from RU 109b) to send the server B response 112b data to RU 109a via wireless network 102.

At step 424, RU 109b can receive encrypted, compressed server B response data 112b from Server A 108 via wireless network 102. At step 425, RU 109b can decrypt response data 112b with the mutually derived symmetric ciphering key 223a, where decryption can use the

same symmetric ciphering algorithm used in a step 411. A step 411 and step 425, RU 109b can also use MAC key 223b in order to confirm message and data integrity within transmitted or received. At step 426, RU 109b can decompresses server B response data 112b. At step 427, RU 109b can write the plaintext data for server B response data 112b to files 192y in shared
5 memory 109h. Writing data for RU 109b to shared memory 109b can comprise RU 109b passing the data via internal bus 212 to processor 109d, and processor 109d could physically write the server B response data 112b to shared memory 109h. For some exemplary embodiments, SRU 109 and processor 109d could conduct the steps 425, 426, and 427 instead of RU 109b.

10 At step 428, after a step 427, power control unit 109e could instruct RU 109b to enter a sleep state. In exemplary embodiments, a trigger library 245 could also monitor or detect when RU 109b writes server B response 112b data to memory 109h, and trigger library could signal power control unit 109e to sleep RU 109b. Or, RU 109b could be programmed to detach from wireless network 102, or enter a long discontinuous reception (DRX) state, and enter a sleep
15 state after writing server B response 112b data to memory 109h, which could comprise a file 192y written for server B response 112b data. Trigger library 245 could continue to detect if new data is written by device 101 to files 191y, where SRU 109 could return to step 405a depicted in Figure 6b.

At step 429b, power control unit 109e or SRU 109 could signal device 101 or CPU 101b
20 that CPU 101b should exit a sleep state entered at step 429a (in Figure 6a and Figure 4a above). Or, (i) if CPU 101b exited a sleep state after 429a and then re-entered a sleep state or (ii) or if CPU 101b comprises a sleep state at step 429b, then power control unit 109e or SRU 109 could signal device 101 that CPU 101b should exit the sleep state in order to receive server B response 112b data. A signal 429c to device 101 for a step 429b is also depicted and described in
25 connection with Figure 4a above. At step 429d, CPU 101b or device 101 could receive the message or signal 429c to wake and read sever B response 112b data written by SRU 109 to files 192y.

Although not depicted in Figure 6b, device 101 could use polling 233 (from Figure 2b) during periodic wake states for CPU 101b to read the server B response 112b data written by
30 SRU 109 in files 192y. In other words, a continuous sleep for CPU 101b from a step 429a to until a step 429d is not required, and for some embodiments a CPU 101b could periodically wake and poll 233 files 192y for server B response data 112b. For these embodiments, if no new files 192b for device 101 are detected, then CPU 101b could return to a sleep state and then later wake and poll 233 again for files 192s.

Device 101 could process the server B response 112b data with a device program 101i using a step 429e from Figure 4a. In addition, although not depicted in Figure 6b, a step 425 could include RU 109b receiving a server B digital signature 307 for the server B response 112a data. RU 109b could also write the server B digital signature 307 to files 192y, which is also depicted and described in connection with Figure 3b and Figure 4a above. Device 101 could use a certificate for server B 110 such as a public key in certificate 110c in order to verify the received server B digital signature 307.

Figure 7

Figure 7 is a graphical illustration of hardware, firmware, and software components for a device, in accordance with exemplary embodiments. A device 101 could operate a shared memory interface 109h for the device 101 and RU 109b without requiring a separate radio storage unit. In other words, the benefits of a shared nonvolatile memory 109h could be obtained for a device where nonvolatile memory 109h operates within a separate housing or package than radio unit 109b. Thus, for some exemplary embodiments, the components and logic for a storage radio unit 109 could be utilized in a device 101 without device 101 operating the components inside a storage radio unit 109 with an enclosure, package, or housing 227 (as depicted in Figure 2a).

Device 101 for embodiments depicted in Figure 7 could use components depicted and describe in connection with Figure 1b, including CPU 101b, device bus 101d, RAM 101e, an antenna 101t, a power control unit 109e, and a WiFi radio or system 101z. Device 101 for embodiments depicted in Figure 7 could also include radio unit (RU) 109b, and nonvolatile memory 101q. Radio unit 109b could be connected to bus 101d instead of residing or operating within a SRU 109. Nonvolatile memory 101q could comprise nonvolatile memory such as flash memory that is operated by device 101 and connected to bus 101d. Nonvolatile memory 101q could also comprise nonvolatile RAM or persistent RAM. Transducer 701 for device 101 could include a sensor 101f and/or an actuator 101y depicted and described in connection with Figure 1b. Nonvolatile memory 101q can be similar to nonvolatile memory 109a for SRU 109 in Figure 1c, but nonvolatile memory 101q could be operated by CPU 101b or controlled by CPU 101b instead of an SRU 109.

Benefits of the present disclosure can be achieved through the use of operating shared memory interface 109h in device nonvolatile memory 101q. For example, the benefits of sending data between CPU 101b and radio unit 109b through a shared memory interface 109h in nonvolatile memory 101q can include the previously noted benefits where CPU 101b and RU

109b can operate with separate and independent sleep states, and both CPU 101b and RU 109b do not have to both be in or operate with active states in order to pass data between the two units. A benefit from transferring the data via shared nonvolatile memory 109h is that the (x) data output from either CPU 101b or RU 109b can be recorded in shared memory 109h at any point in time (when the writing unit is active), and (y) the receiving or reading component for the data can be in sleep states. Trigger library 245 operated by CPU 101b can control the power of RU 109b (e.g. deep sleep, sleep, active, attached to wireless network 102, etc.) in conjunction with a power control unit 109e. Other benefits of the use of a shared memory interface 109h operated by device 101 in a nonvolatile memory 101q for device 101 can be obtained as well, without departing from the scope of the present disclosure.

A shared memory 109h in Figure 7 can comprise a portion of nonvolatile memory nonvolatile memory 109h. Or, a shared memory 109h could operate in RAM 101e for some embodiments, such as those depicted and described in connection with Figure 5a and Figure 5b. Shared memory 109h when operating in RAM 101e could also be recorded in persistent RAM memory or nonvolatile RAM memory such as NVRAM depicted and described in connection with Figure 5a above. Shared memory 109h could store or record a shared filesystem interface 230'. For some embodiments, the use of a shared filesystem interface 230' could be omitted and data for device write memory 191 and RU write memory 192 be recorded using memory addresses instead of file names 305a (from Figure 3a) and 305b (from Figure 3b) and directories 191x and 192x. Drivers 101g (in Figure 1b) and RU firmware 222 (in Figure 2e) could operate in order for CPU 101b and RU 109b to communicate through shared memory 109h via bus 101d. In some preferred exemplary embodiments, drivers 101g and RU firmware 222 can utilize a shared filesystem interface 230' operating in shared memory 109h.

A device 101 in Figure 7 could use the steps for an SRU 109 to send and receive data from RU 109b through shared memory 109h. For these embodiments, CPU 101b could perform the steps of memory processor 109d for an SRU 109. Shared memory 109h could undergo a configuration process 231 in order to allocate shared memory 109h to (i) device write and RU read memory 191 and (ii) RU write and device read memory 192. A configuration unit 110d for device service provider 110 could conduct the configuration process 231 in order to format and structure memory 109g. Or, CPU 101b could operate with firmware to conduct the configuration process 231 before CPU 101b communicates with RU 109b through shared memory 231.

After a configuration process 231, device write memory 191 and RU write memory 192 could be structured with addresses and/or directories to store or record (i) data from CPU 101b

for RU 109b and (ii) data from RU 109b for CPU 101b. Memory 191 and memory 192 are depicted and described in connection with Figure 2b above. In exemplary embodiments a file system could be written to memory 191 and memory 192, and (i) memory 191 could record directory 191x and store files 191y (as in Figure 3a) and (ii) memory 192 could record directory 192x and store files 192y (as in Figure 3b). Exemplary data for files 191y are depicted and described in connection with Figure 3a and exemplary data for files 192y are depicted and described in connection with Figure 3b. Or, for embodiments where a file system is not used in memory 109h, then the data for files 191y could be recorded at memory addresses in memory 191 and the data for files 192y could be recorded in memory addresses in memory 192 (without the file structures depicted in Figure 3a and Figure 3b).

For exemplary embodiments depicted in Figure 7, CPU 101b can perform the steps for a memory processor 109d depicted in Figures above, such as, but not limited to, (i) a step 405 in Figure 4a and Figure 6a (with separate steps 405a and 405b depicted in Figure 6a), (ii) sending message 406a or commands to RU 109b using a message 406a in Figure 4a, (iii) writing a command_script 191g from Figure 3a to memory 191 for RU 109b to subsequently read, (iii) operating a trigger library 245 in a step 245a in Figure 6a, and (iv) controlling a power control unit 109e for changing state of RU 109b, and (v) conducting a shared nonvolatile memory 109h configuration process 231. CPU 101b can perform other actions and steps for a processor 109d depicted and described for a processor 109d in SRU 109 without departing from the scope of the present disclosure. In exemplary embodiments for Figure 7, clock 101m (in a device 101 from Figure 1c) could comprise RF clock 205, where clock 101m drives both CPU 101b and RU 109b. Clock 101m could be connected to a PLL 211 to convert the frequency from clock 101m to a driving clock frequency for CPU 101b.

In summary, a CPU 101b and RU 109b within a device 101 could communicate through a shared nonvolatile memory 109h storing a shared nonvolatile memory interface 230 and achieve various benefits described herein to address needs in the art identified above. Benefits can include independent sleep states for CPU 101b and RU 109b (e.g. they don't have to be active at the same time to transfer control data such as 191g or application data 112a or server B response data 112b). Benefits also include transferring data between CPU 109b and RU 109b in a nonvolatile state, such that the data in memory 191 and memory 192 is available to both CPU 101b and RU 109b immediately after transitioning from a sleep state to an active state (thereby allowing time in an active state to be reduced and consequently conserve power or energy). Other benefits are provided as well, and these benefits are exemplary as opposed to being limiting.

In addition, although an RU 109b is depicted and described in Figure 4a and Figure 4b as conducting cryptographic operations such as conducting an ECDH key exchange 409 and encrypting application data 112a with a symmetric ciphering key 223a, a CPU 101b in device 101 as depicted in Figure 7 could conduct the those and related steps (such as data compression 410, data decompression 426, data decryption 425, etc.) In other words, although a SRU 109 with RU 109b may conduct steps 409, 410, 426, 425, etc. for some embodiments, for other embodiments without an SRU 109 then CPU 101b could conduct the steps.

Figure 8

Figure 8 is a graphical illustration of hardware, firmware, and software components for a device, in accordance with exemplary embodiments. A device 101 could operate a shared memory interface 109h for the device 101 and radio components without requiring a separate storage radio unit and a separate radio unit. In other words, the benefits of a shared nonvolatile memory 109h for storing application data 112a for transmission to a wireless network 102 could be obtained where device 101 operates components a “system on a chip” 801. For some exemplary embodiments, the components and logic for a storage radio unit 109 could be utilized with a “system on a chip” 801 for a device 101 without device 101 operating (i) the components inside a storage radio unit 109 with an enclosure, package, or housing 227 (from Figure 2a), and (ii) a separate housing or package for a radio unit RU 109b. For example, the RU 109b' could comprise a wafer level chip scale package for a RU 109b within a system on a chip 801, or RU 109b' could comprise circuits for performing the functions of a RU 109b within the system on a chip 801.

Device 101 in Figure 8 and include a system on a chip 801, a bus 101d, a transducer 701, an antenna 101t, and an RF front end 203. Bus 101d can be equivalent to bus 101d depicted and described in connection with Figure 1b, and bus 101d can connect components external to the system on a chip 801 to the system on a chip 801. Transducer 701 can comprise a sensor 101f and/or an actuator 101y (as depicted and described in connection with Figure 1b), as well as comprising a plurality of sensors and actuators. Antenna 101t can comprise an internal antenna within the housing of device 101 or a connection to an external antenna. RF front end 203 can comprise an RF front end as depicted and described in connection with Figure 2a. RF front end 203 can include a power amplifier 203b in order to amplify RF signals transmitted, such as providing an exemplary gain of 30 dB for transmit signals input from system on a chip 801, up to reasonable maximum transmit power of an exemplary 30 dBm. Other values could be used as

the gain and maximum transmit power of PA 203b without departing from the scope of the present disclosure, such as a maximum exemplary transmit power of 23 dBm, which would support NB-IoT networking technology 104 according to 3GPP Releases 13, 14, and 15.

The system on a chip 801 can include CPU 101b', an iUICC 113', nonvolatile memory 101q, a system bus 802, nonvolatile RAM memory 101e', regular RAM 101e, radio unit 109b', and transducer interface 803. A "system on a chip" 801 could also be referred to as a system 801 or chip 801 or an integrated circuit 801. The "system on a chip" 801 could also be referred to as a module. CPU 101b' can be similar to CPU 101b for device 101 in Figure 1b. Note that CPU 101b' in Figure 8 could operate with multiple cores, and CPU 101b' could include functionality of processor 109d for an SRU 109 and processor 109c for an RU 109b from Figures above, such as Figure 1c and Figure 2a. The system on a chip 801 can include an integrated universal integrated circuit card (iUICC) 113', which could be equivalent to an eUICC 113 depicted above, but iUICC 113' could be operated within system on a chip 801. Regular RAM 101e could comprise a standard RAM memory such as SRAM or DRAM. RU 109b' in chip 801 can comprise an RU 109b as depicted in Figure 2a, with the exception that RF front end 203 could operate within device 101 instead of chip 801. Or, in other exemplary embodiments RF front end 203 could be integrated onto chip 801 as well.

Nonvolatile memory 101q could comprise flash memory for system 801 and include general nonvolatile memory 109g for device 101, such as operating firmware for device 101 (similar to SRU firmware 109u in Figure 2e). A radio baseband unit 206 and RF transceiver 204 can be operated by the radio unit 109b', similar to those depicted and described in connection with Figure 2a above. Transducer interface 803 can comprise an electrical connection plus signaling or control logic for connection to transducer 701. System bus 802 can comprise a plurality of internal busses within chip 801 in order to connect the various internal components. System bus 802 can be similar to bus 101d, except operating within chip 801.

Integrated circuit 801 can include a nonvolatile RAM (NVRAM) memory 101e'. NVRAM memory 101e can be similar to RAM 101e depicted and described in connection with Figure 1b, except that NVRAM can comprise persistent memory, such that data recorded in NVRAM 101e' is recorded when device 101 is in a deep sleep state or connected to a battery 101n. Some versions of NVRAM could support battery 101n being removed or completely drained. NVRAM 101e' can be different than flash memory 101q, where flash memory 101q typically has a more limited number of write cycles, and NVRAM 101e' has a less limited number of write cycles. Further, write operations to flash memory 101q typically take longer and require more power, while write operations to NVRAM 101e' require less power and are

faster. NVRAM 101e' typically costs more per byte than flash memory for nonvolatile memory 101q. Further, data in nonvolatile memory 101q can remain stored in device 101 when a battery 101n is disconnected or essentially completely drained, while data in NVRAM 101e' could be erased if battery 101n is removed for extended periods.

5 Consequently, some less frequently changing data such as an operating system 101h for device 101 can be stored in memory 101q, while a smaller set of data such as application data 112a and server B response data 112b can be stored in NVRAM 101e'. NVRAM 101e' can also be referred to as a "persistent memory" RAM device. NVRAM 101e' could also simply be a portion of RAM 101e that receives power from battery 101n when integrated circuit 801
10 comprises a sleep or deep sleep state (such as requiring less than an exemplary microamp of current).

NVRAM 101e' can include shared nonvolatile memory 109h. Although shared nonvolatile memory 109h is depicted within NVRAM 101e' for the system on a chip 801, memory 109h could be recorded in memory 101q or in a volatile state in RAM 101e. As
15 depicted in Figure 8, NVRAM 101e can store or record a shared memory interface 230 within shared nonvolatile memory 109h. Shared nonvolatile memory 109h can include a shared memory interface 230, such as that depicted and described in connection with Figure 2b. Shared memory interface 230 could operate as a filesystem interface between radio unit 109b and CPU 101b'. Shared nonvolatile memory interface 230 does not require a file system and can use
20 memory addresses for (i) device write and radio unit read memory 191 and (ii) radio unit write and device read memory 192, where memory 191 and memory 192 are depicted and described in connection with Figure 2b. For embodiments that do use a filesystem within shared nonvolatile memory interface 230, then memory 191 could include a directory 191x with files 191y as depicted and described in connection with Figure 3a. For embodiments that do use a
25 filesystem within shared nonvolatile memory interface 230, then memory 192 could include a directory 192x with files 192y as depicted and described in connection with Figure 3b.

For exemplary embodiments depicted in Figure 8, CPU 101b' can perform the steps for a memory processor 109d depicted in Figures above, such as, but not limited to, (i) a step 405 in Figure 4a and Figure 6a (with separate steps 405a and 405b depicted in Figure 6a), (ii) sending
30 message 406a or commands to RU 109b using a message 406a in Figure 4a, (iii) writing a command_script 191g from Figure 3a to memory 191 for RU 109b to subsequently read, (iii) operating a trigger library 245 in a step 245a in Figure 6a, and (iv) controlling a power control unit 109e for changing state of RU 109b, and (v) conducting a shared nonvolatile memory 109h configuration process 231. CPU 101b can perform other actions and steps for a processor 109d

depicted and described for a processor 109d in SRU 109 without departing from the scope of the present disclosure. In exemplary embodiments for Figure 8, a clock 101m (in a device 101 from Figure 1c) could comprise RF clock 205, where clock 101m drives both CPU 101b' and RU 109b. Clock 101m could be connected to a PLL 211 to convert the frequency from clock 101m to a driving clock frequency for CPU 101b'.

In summary, a CPU 101b' and RU 109b' within a system on a chip or integrated circuit 801 could communicate through a shared nonvolatile memory 109h storing a shared nonvolatile memory interface 230 and achieve various benefits described herein to address needs in the art identified above. Benefits can include independent sleep states for CPU 101b' and RU 109b' (e.g. they don't have to be active at the same time to transfer control data such as 191g or application data 112a or server B response data 112b). Benefits also include transferring data between CPU 109b' and RU 109b' in a nonvolatile state, such that the data in memory 191 and memory 192 is available to both CPU 101b' and RU 109b immediately after transitioning from a sleep state to an active state (thereby allowing time in an active state to be reduced and consequently conserve power or energy). Other benefits are provided as well, and these benefits are exemplary as opposed to being limiting.

Conclusion

Various exemplary embodiments have been described above. Those skilled in the art will understand, however, that changes and modifications may be made to those examples without departing from the scope of the claims.

CLAIMS

1. A storage radio unit (SRU) operating within a device, the SRU comprising:

a nonvolatile memory comprising (i) a shared memory for the device and the SRU, (ii) a general memory for the device, wherein the general memory stores a filesystem, and (iii) a
5 protected memory for the SRU, wherein the protected memory stores a firmware for the SRU, network access credentials, and a symmetric ciphering key;

a processor configured to conduct a filesystem detection step from the firmware to detect a type of the filesystem, configured to conduct a configuration process, wherein the configuration process writes (i) the detected type of the filesystem to the shared memory and (ii)
10 a directory in the shared memory, and configured to operate a trigger to detect when application data from the device is written to the directory;

an external bus configured to receive the application data from the device as a file in the directory and configured to send a response to the device, wherein the response is received by the SRU from a server;

15 an internal bus connected to the shared memory configured to store the application data in the directory and configured to store the response in the directory;

a radio unit configured to read the application data from the directory via the internal bus, configured to connect to a wireless network using the network access credentials, configured to encrypt the application data using the symmetric ciphering key, configured to
20 transmit (i) the encrypted application data to the wireless network, configured to receive the response to the transmitted application data, and configured to write the response to the directory via the internal bus; and

a power control unit configured to use the trigger to wake the radio unit after the external bus receives the application data, and configured to sleep the radio unit after the radio unit writes
25 the response to the directory.

2. The SRU of claim 1, further comprising an embedded universal integrated circuit card (eUICC) configured to record an operational profile, wherein the operational profile includes the network access credentials for the wireless network.

3. The SRU of claim 1 or 2, wherein the directory comprises (i) a device write directory for the application data and (ii) a radio unit write directory for the response, and wherein the radio unit reads the device write directory and the device reads the radio unit write directory.

4. The SRU of claim 3, wherein the device is configured to periodically poll the radio unit write directory for the response.

5. The SRU of claim 3 or 4, wherein the device write directory comprises at least one file for a radio parameters input, a http request, a mobile originated SMS, an upload file, a command script for the SRU, power control settings for the SRU, a device identity for the device, cryptographic parameters for the SRU, and a time value for device current time.

6. The SRU of any one of claims 3 to 5, wherein the radio unit write directory comprises at least one file for radio parameters loaded, a http response, a mobile terminated SMS, a download file, a log file for the SRU, a digital signature from the server, power control settings for the SRU, cryptographic parameters for the SRU, a radio network status, an error log, and a time value from the wireless network.

7. The SRU of any one of claims 1 to 6, wherein the device writes to the shared memory an elliptic curve cryptography (ECC) server public key, wherein the radio unit (i) derives an ephemeral ECC private key, (ii) conducts an elliptic curve Diffie-Hellman (ECDH) key exchange with the ECC server public key and the ephemeral ECC private key in order to derive the symmetric ciphering key.

8. The SRU of any one of claims 1 to 7, further comprising a configuration unit configured to write the firmware and the symmetric ciphering key to the protected memory before the device is received by an end user.

9. The SRU of any one of claims 1 to 8, wherein the device configures the general memory by writing the filesystem before the SRU conducts the configuration process, and wherein the type of the filesystem comprises one of FAT16, FAT32, exFAT, NTFS, Ext2, Ext3, Ext4, UDF, ISO-9660, HFX+, HFSX, Apple, UFS, XFS, JFS, YAFFS, JFFS2, and UBIFS.

10. The SRU of claim 9, wherein the firmware includes a filesystem driver for the type of the filesystem and the SRU is configured to select the filesystem driver based on the detected type of filesystem in the general memory.

11. The SRU of any one of claims 1 to 10, wherein the firmware includes an interface driver for the external bus, and wherein the external bus comprises one of a serial peripheral interface (SPI) bus, a Peripheral Component Interconnect (PCI) bus, a Peripheral Component Interconnect Express (PCIe) bus, a SD bus, an I2C bus, a USB bus, and a serial bus.

12. The SRU of any one of claims 1 to 11, further comprising a memory controller, wherein the memory controller is connected to the internal bus and the nonvolatile memory, and wherein the memory controller prevents the device from directly reading or directly writing to the protected memory.

13. The SRU of any one of claims 1 to 12, further comprising a micro SD card form factor.

14. The SRU of claim 13, wherein the SRU connects with the device using an SD bus, and wherein a pin 1 and a pin 8 of the external bus connect to an antenna within the device.

15. A system for supporting wireless communications, the system comprising:

a device configured to operate (i) a central processing unit (CPU), (ii) a sensor configured to collect application data, and (iii) a first filesystem driver for a device operating system, wherein the first filesystem driver is configured to write a first file with a file type and configured to read a second file with the file type;

a radio unit configured to operate a radio processor that is (i) configured to connect with a wireless network through an antenna, (ii) configured to store a private key, a network certificate and a device identity in a protected memory, and (iii) configured to transmit the application data, wherein the radio unit is configured to mutually authenticate using the private
5 key, the network certificate, and the device identity;

a firmware for the radio unit, wherein the firmware includes a second filesystem driver, wherein the second filesystem driver is configured to read the first file and to write the second file;

a nonvolatile memory configured to (i) connect to both the radio unit and the device, (ii)
10 receive and store the application data as the first file, and (iii) receive and store a response from the network as the second file; and

a power control unit of the radio unit configured to (i) sleep the radio unit before the device writes the application data to the nonvolatile memory, (ii) wake the radio unit after the device writes the application data, and (iii) sleep the radio unit after the radio unit writes the
15 response.

16. The system of claim 15, wherein the CPU is configured to (i) sleep after writing the first file with the application data, (ii) wake after the radio unit writes the second file with the response, and (iii) sleep after reading the second file with the response.

17. The system of claim 15 or 16, wherein the firmware includes a filesystem detection
20 library, wherein the radio unit is configured to (i) detect the file type using the filesystem detection library and (ii) select the second filesystem driver based on the detected file type.

18. The system of claim 17, wherein the device configures the nonvolatile memory by writing a filesystem before the firmware detects the file type, and wherein the file type comprises one of FAT16, FAT32, exFAT, NTFS, Ext2, Ext3, Ext4, UDF, ISO-9660, HFX+,
25 HFSX, Apple, UFS, XFS, JFS, YAFFS, JFFS2, and UBIFS.

19. The system of any one of claims 15 to 18, wherein the nonvolatile memory is configured to store (i) the first file in a device write directory and (ii) the second file in a radio unit write directory, and wherein the radio unit reads the device write directory and the device reads the radio unit write directory.

5 20. The system of claim 19, wherein the device periodically polls the radio unit write directory for the response.

21. The system of claim 19 or 20, wherein the device write directory comprises at least one file for a radio parameters input, a http request, a mobile originated SMS, an upload file, a command script for the SRU, power control settings for the SRU, a device identity for the
10 device, cryptographic parameters for the SRU, and a time value for device current time.

22. The system of any one of claims 15 to 21, wherein the nonvolatile memory stores an elliptic curve cryptography (ECC) server public key, wherein the radio unit is (i) configured to derive an ephemeral ECC private key, (ii) configured to conduct an elliptic curve Diffie-Hellman (ECDH) key exchange with the ECC server public key and the ephemeral ECC private
15 key in order to derive a symmetric ciphering key, and wherein the radio unit is configured to encrypt the application data with the symmetric ciphering key.

23. The system of any one of claims 15 to 22, wherein the device includes the nonvolatile memory and wherein the radio unit connects with the nonvolatile memory through an external bus for the radio unit.

20 24. The system of claim 23, wherein the firmware includes an interface driver for the external bus, and wherein the external bus comprises one of a serial peripheral interface (SPI) bus, a Peripheral Component Interconnect (PCI) bus, a Peripheral Component Interconnect Express (PCIe) bus, a SD bus, an I2C bus, a USB bus, and a serial bus.

25 25. The system of any one of claims 15 to 22, wherein the radio unit includes an internal bus and the nonvolatile memory, and wherein the nonvolatile memory is connected to the internal bus.

26. A method for a storage radio unit (SRU) to communicate with a device and a wireless network, the method performed by the SRU, the method comprising:

a) storing in a protected nonvolatile memory of the SRU a firmware, a server public key, and an identity;

5 b) receiving a filesystem format command for a general nonvolatile memory of the SRU;

c) connecting to the device through an external bus;

d) using the firmware to detect a type of the filesystem format;

e) selecting a filesystem driver compatible with the type of the filesystem format to write a directory in a shared nonvolatile memory, wherein the device has write permissions for the
10 directory and wherein the SRU creates the directory;

f) operating a trigger to detect that a device has written application data to the directory;

g) waking a radio unit within the SRU based on the trigger, wherein the radio unit authenticates with a server through the wireless network using at least the server public key and the identity, and wherein the radio unit transmits the application data to the wireless network;

15 h) receiving a response from the server, wherein the SRU writes the response to the shared nonvolatile memory; and

i) sleeping the radio unit after the SRU writes the response to the shared nonvolatile memory.

27. The method of claim 26, wherein the directory comprises (i) a device write directory for
20 the application data and (ii) a radio unit write directory for the response, and wherein the radio unit reads the device write directory and the device reads the radio unit write directory.

28. The method of claim 27, wherein the device periodically polls the radio unit write directory for the response.

29. The method of claim 27 or 28, wherein the device write directory comprises at least one file for a radio parameters input, a http request, a mobile originated SMS, an upload file, a command script for the SRU, power control settings for the SRU, a device identity for the device, cryptographic parameters for the SRU, and a time value for device current time.

5 30. The method of any one of claims 27 to 29, wherein the radio write directory comprises at least one file for radio parameters loaded, a http response, a mobile terminated SMS, a download file, a log file for the SRU, a digital signature from the server, power control settings for the SRU, cryptographic parameters for the SRU, a radio network status, an error log, and a time value from the wireless network.

10 31. The method of any one of claims 26 to 30, further comprising (j) receiving by the shared memory an elliptic curve cryptography (ECC) server public key, wherein the radio unit (i) derives an ephemeral ECC private key, (ii) conducts an elliptic curve Diffie-Hellman (ECDH) key exchange with the ECC server public key and the ephemeral ECC private key in order to derive a symmetric ciphering key, and (k) encrypting by the SRU the application data with the
15 symmetric ciphering key before the SRU transmits the application data.

32. The method of any one of claims 26 to 31, wherein the type of the filesystem format comprises one of FAT16, FAT32, exFAT, NTFS, Ext2, Ext3, Ext4, UDF, ISO-9660, HFX+, HFSX, Apple, UFS, XFS, JFS, YAFFS, JFFS2, and UBIFS.

33. The method of any one of claims 26 to 32, wherein the firmware includes an interface
20 driver for the external bus, and wherein the external bus comprises one of a serial peripheral interface (SPI) bus, a Peripheral Component Interconnect (PCI) bus, a Peripheral Component Interconnect Express (PCIe) bus, a SD bus, an I2C bus, a USB bus, and a serial bus.

34. The method of any one of claims 26 to 32, wherein the storage radio unit comprises a micro SD card form factor.

35. The method of claim 34, wherein the storage radio unit connects with the device using an SD bus, and wherein a pin 1 and a pin 8 of the micro SD card form factor connect to an antenna within the device.

36. The method of any one of claims 26 to 35, further comprising (j) sleeping a central
5 processing unit (CPU) for the device after the device writes the application data and (k) waking the CPU after the radio unit writes the response.

37. The method of any one of claims 26 to 36, wherein the device sends the filesystem format command after the SRU is connected to the device.

38. The method of any one of claims 26 to 36, wherein a configuration unit sends the
10 filesystem format command before the SRU is connected to the device.

39. The method of any one of claims 26 to 38, wherein the device reads the response from the shared nonvolatile memory after the radio unit sleeps.

40. The method of any one of claims 26 to 39, wherein the device mounts the general
nonvolatile memory as a first storage device and the shared nonvolatile memory as a second
15 storage device.

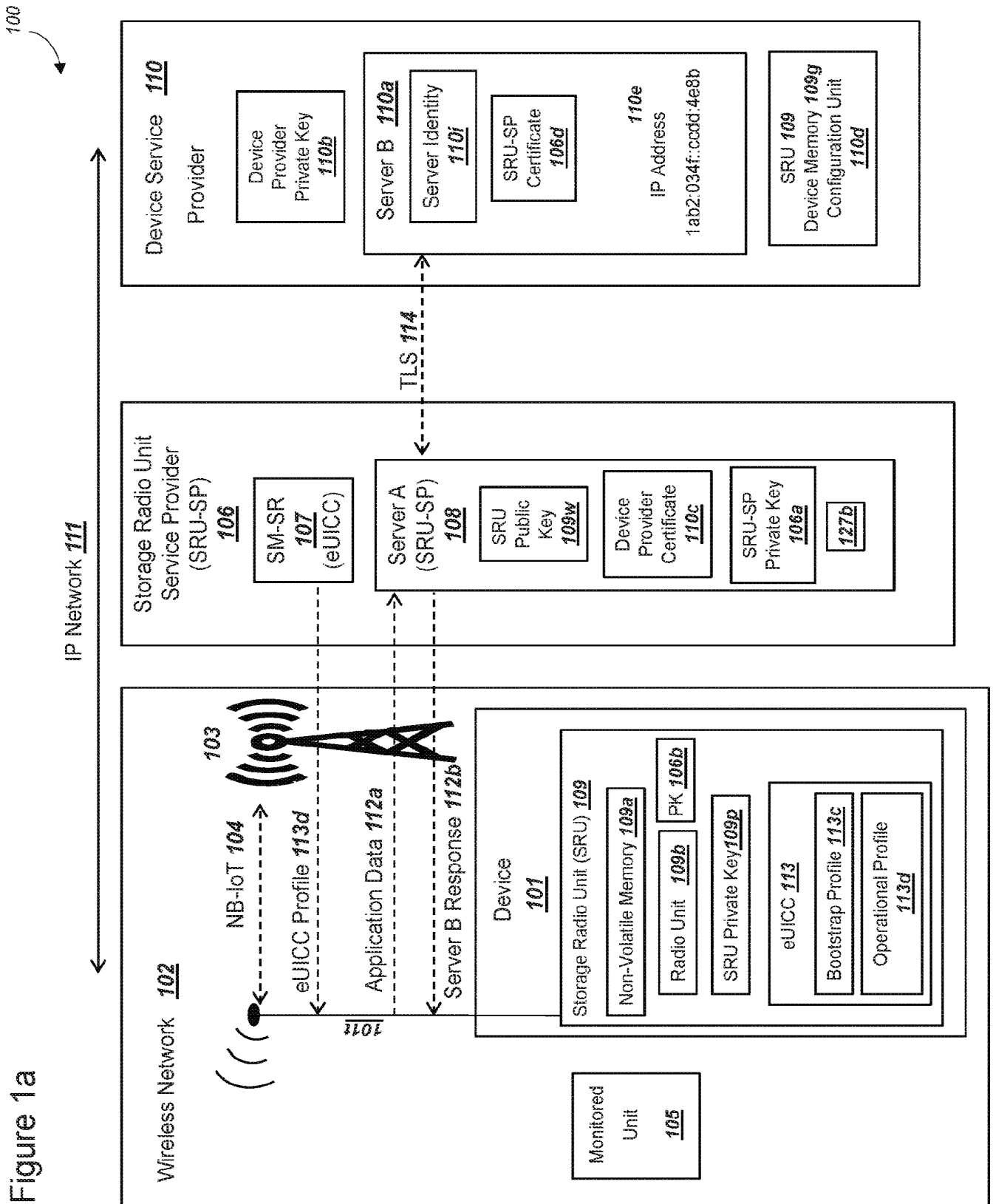


Figure 1c

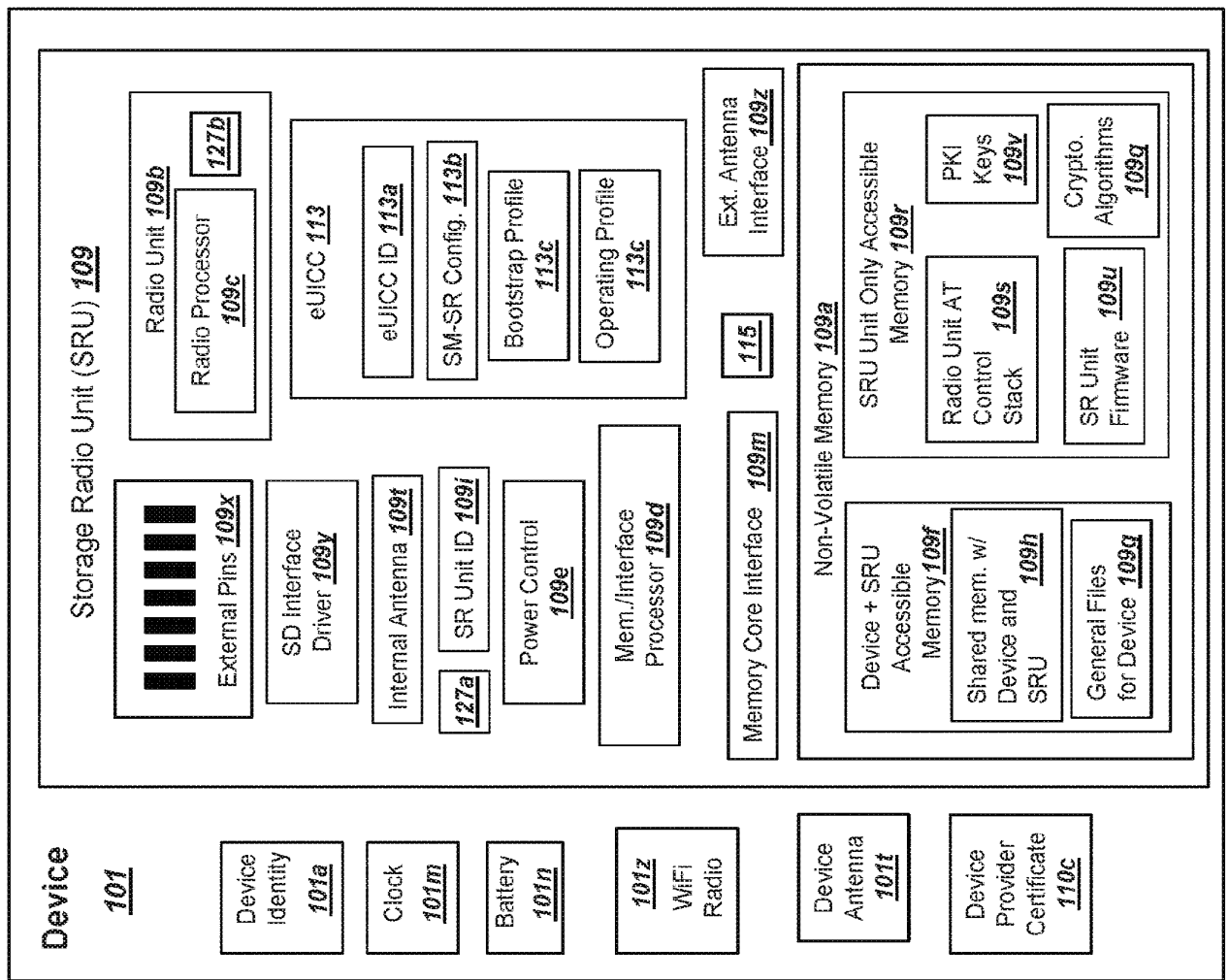


Figure 1b

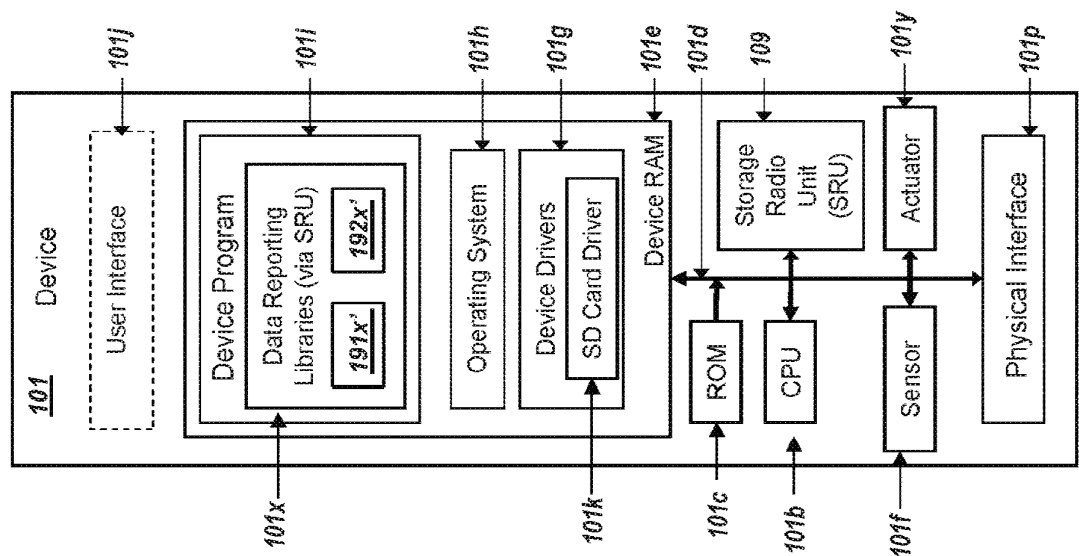
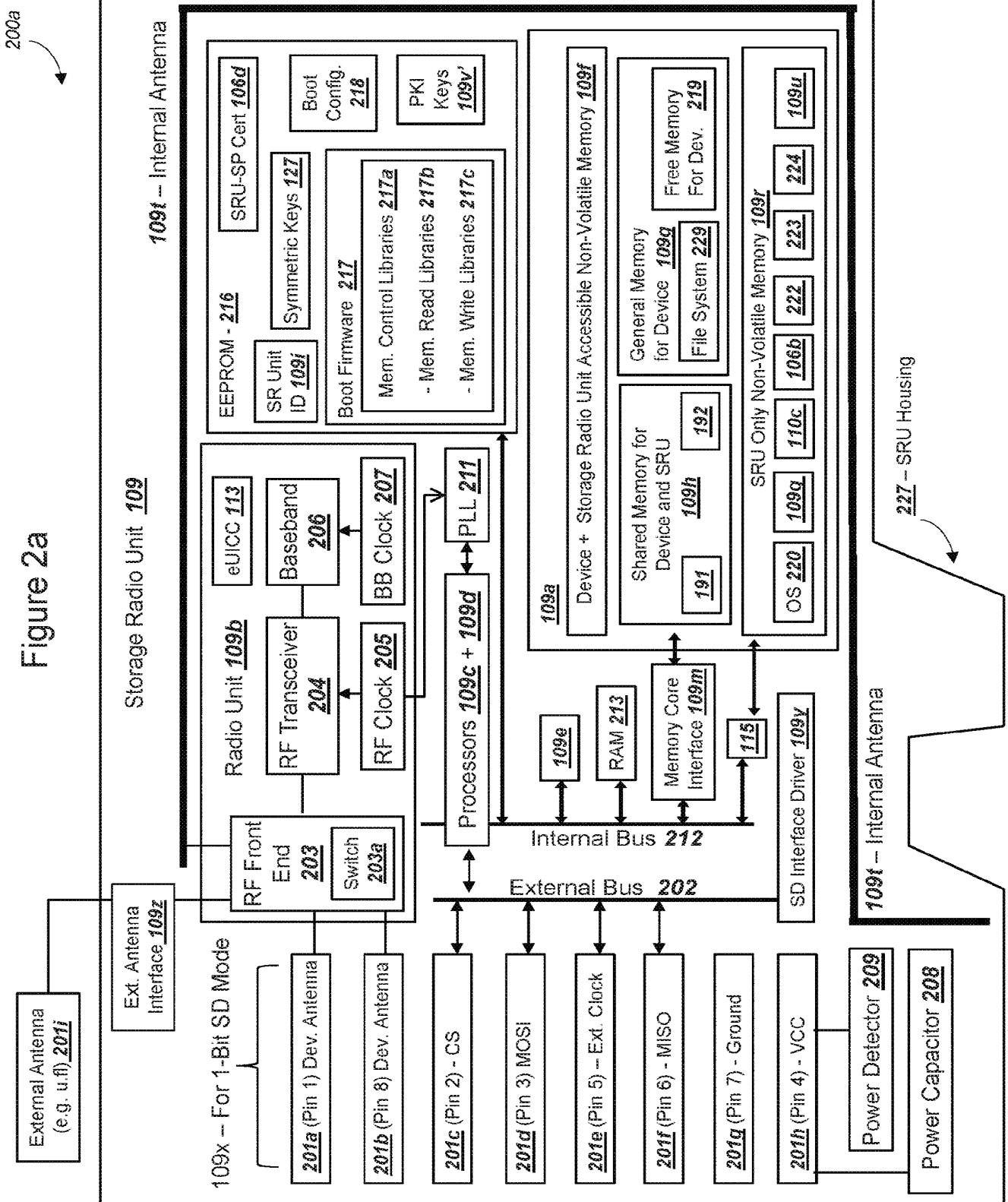


Figure 2a



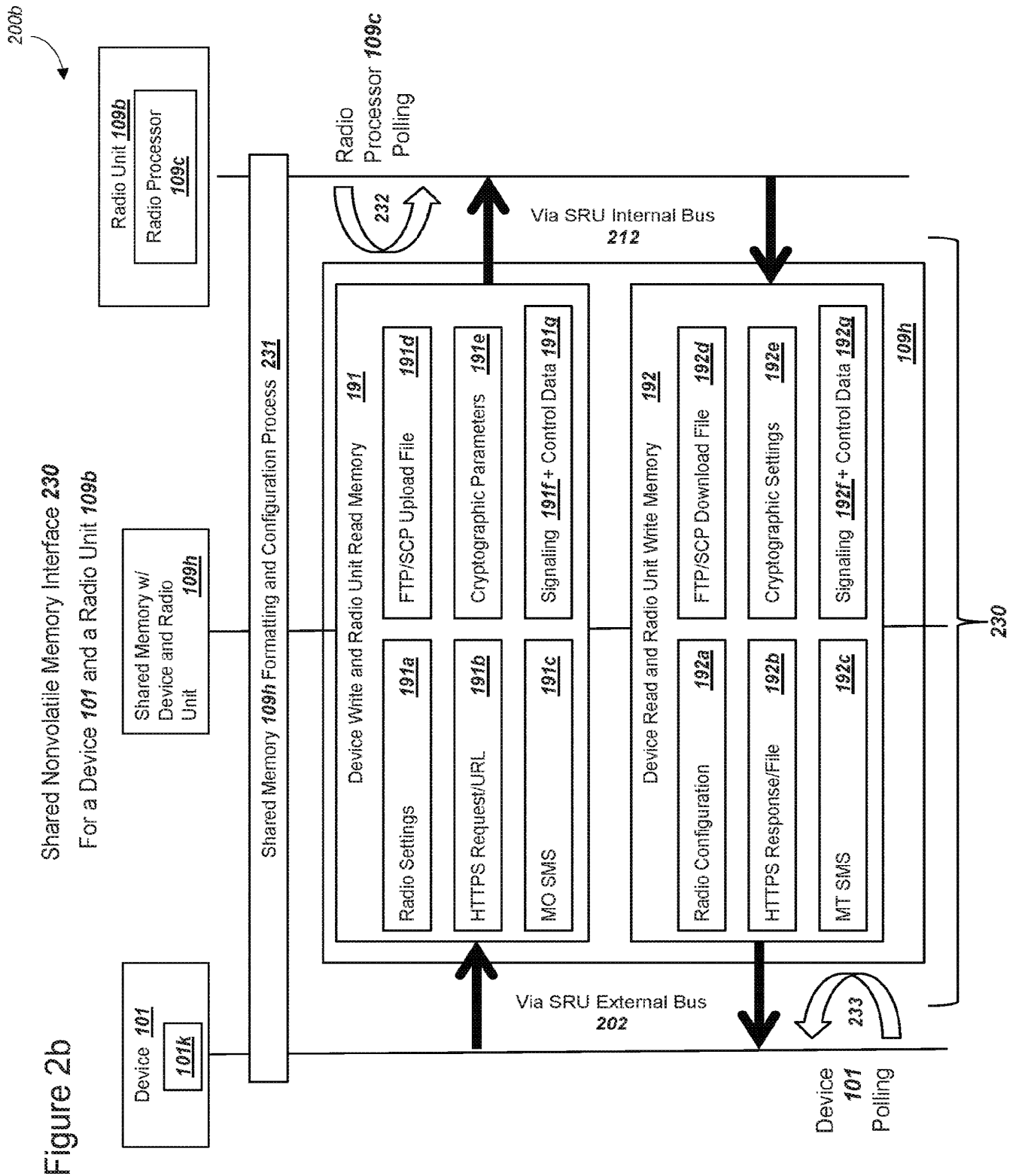


Figure 2c

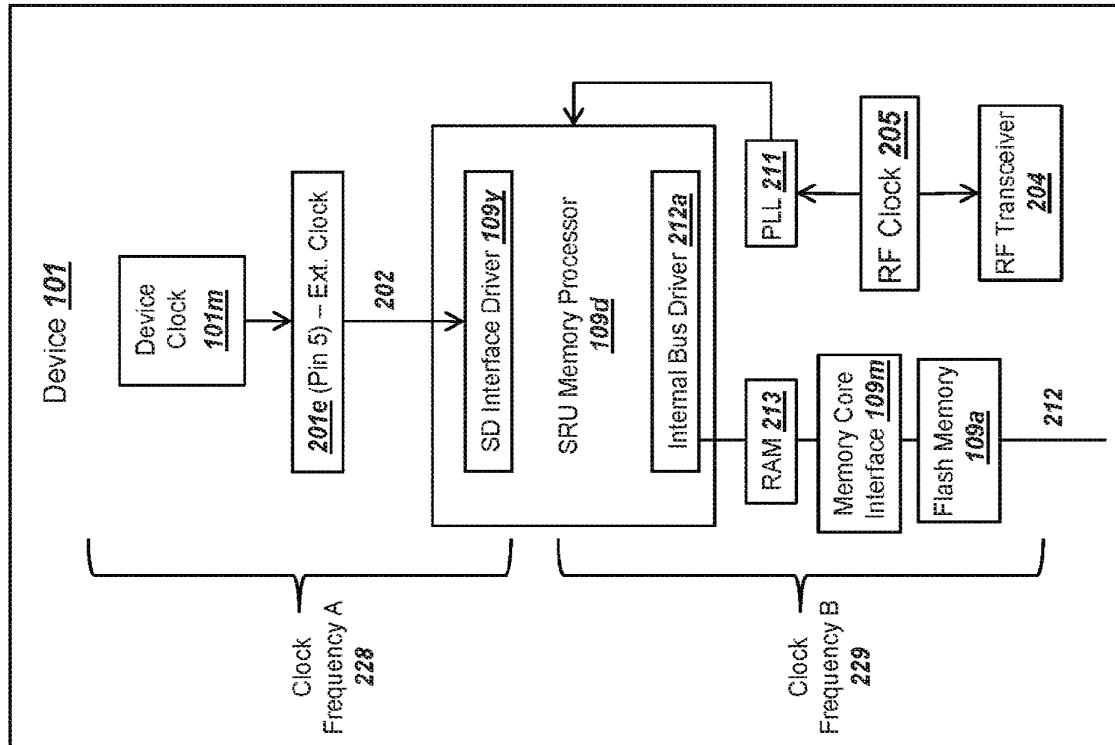


Figure 2d

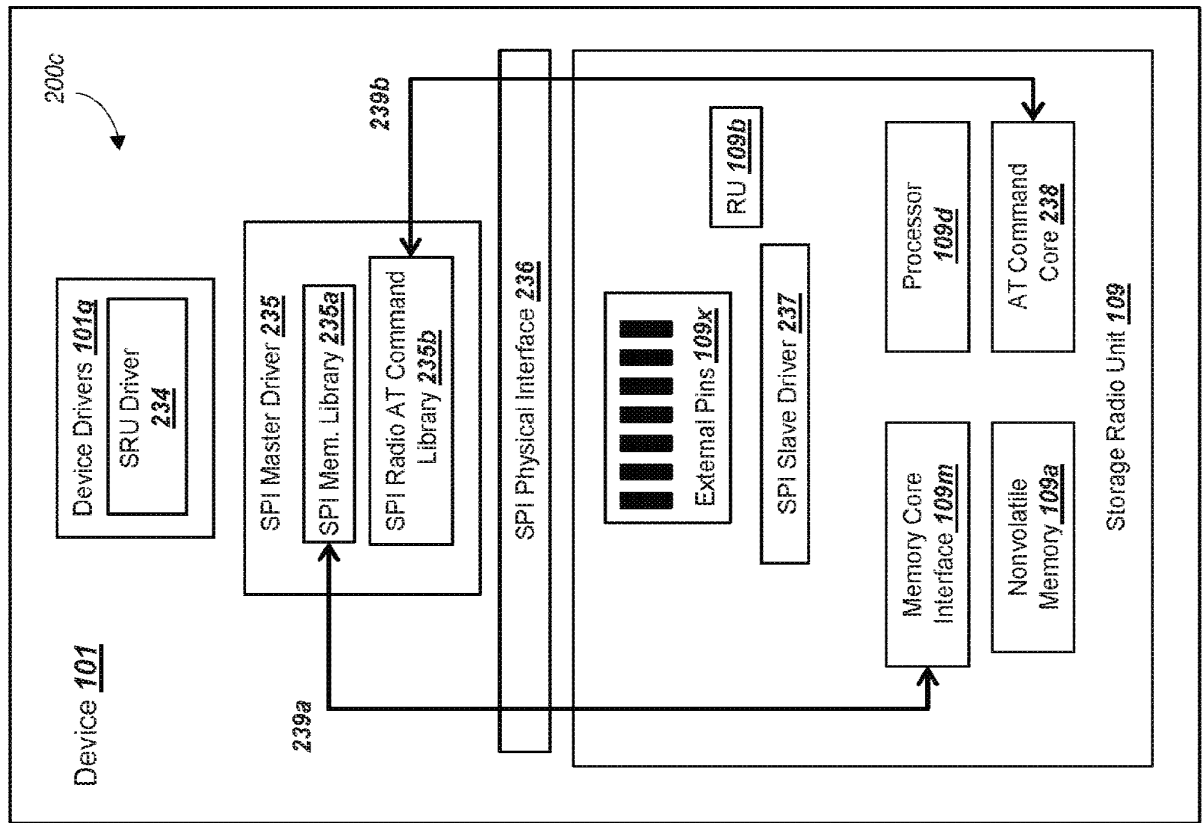


Figure 2e

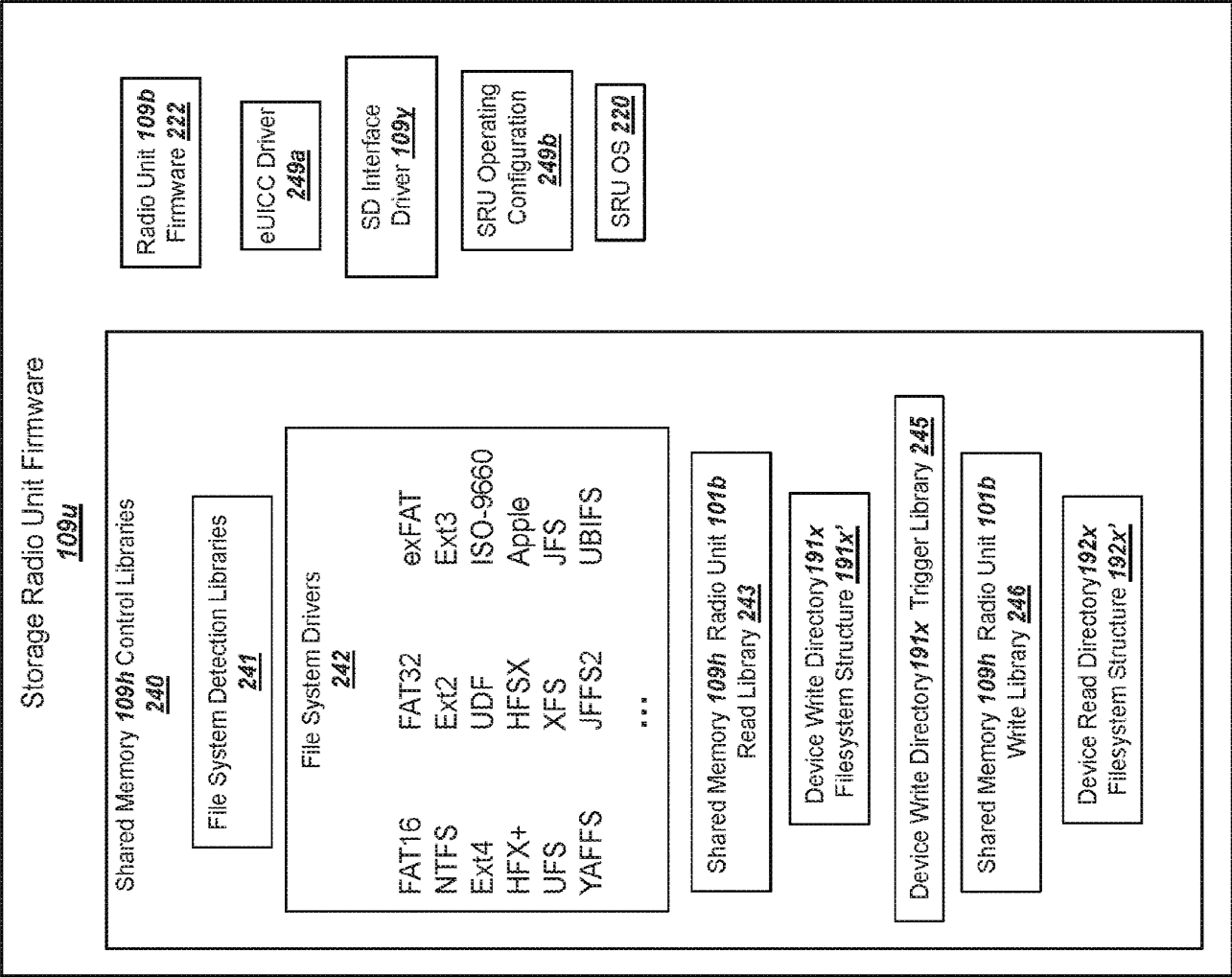


Figure 2f

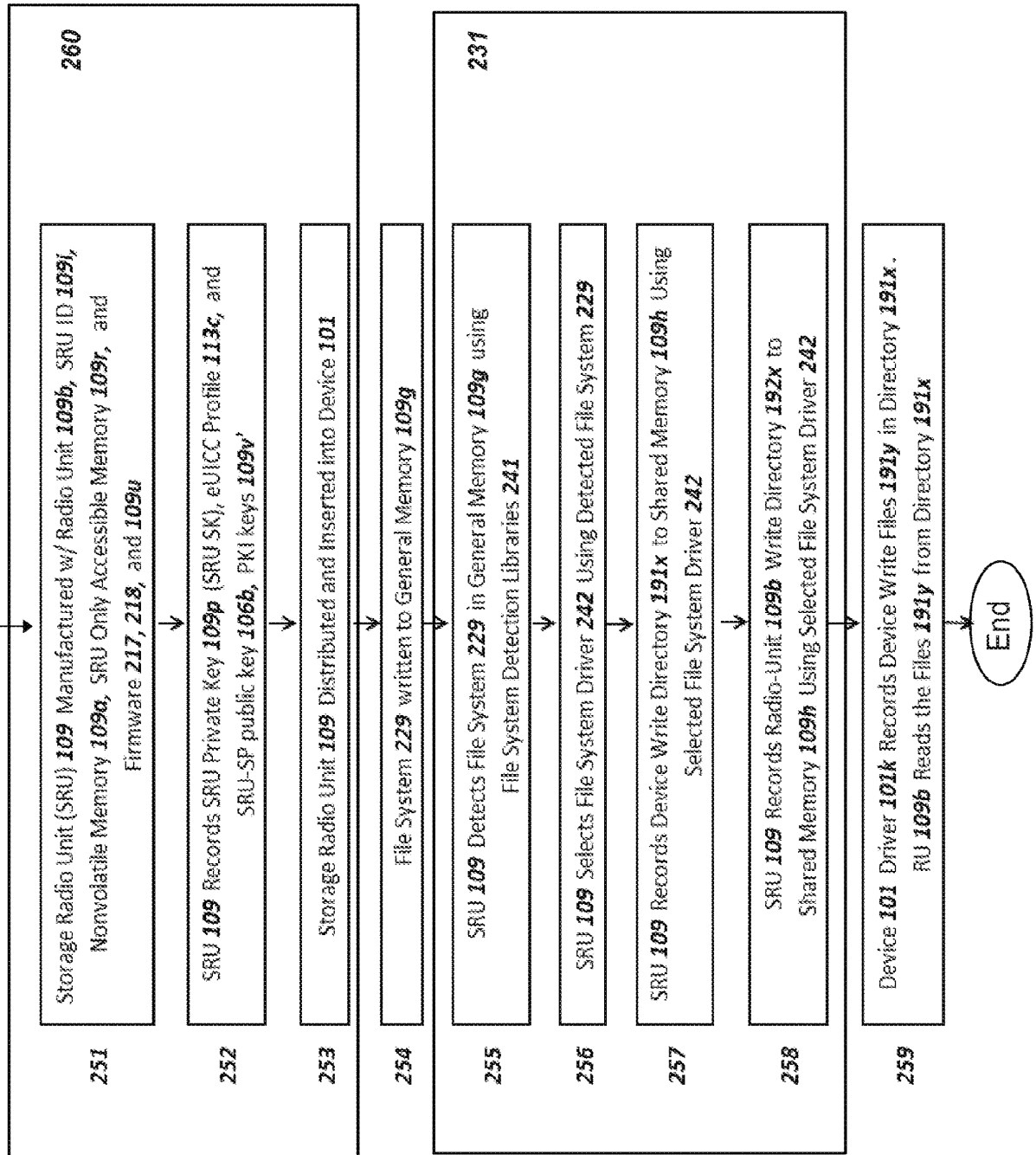


Figure 3a

Shared File System Interface **230'**: Device Write and Radio Unit Read Files **191y** in Directory **191x**

File System Interface 230' : .../device_write/.. Directory 191x				191y
-rw-rw-----	1	device_OS SRU_OS	403 Apr 12 2018	radio_parameters_input 191a
-rw-rw-----	1	device_OS SRU_OS	112a 93 Apr 22 2018	https_request 191b
-rw-rw-----	1	device_OS SRU_OS	75 Apr 22 2018	MO_sms 191c
-rw-rw-----	1	device_OS SRU_OS	2451 Apr 22 2018	FTP_file_upload 191d
-rw-rw-----	1	device_OS SRU_OS	842 Mar 31 2018	command_script 191g
-rw-rw-----	1	device_OS SRU_OS	395 Mar 31 2018	power_control_settings 191f
-rw-rw-----	1	device_OS SRU_OS	16 Mar 30 2018	device_identity 101a
-rw-rw-----	1	device_OS SRU_OS	2542 Feb 8 2017	cryptographic_parameters 191e
-rw-rw-----	1	device_OS SRU_OS	22 Feb 8 2017	date-time-set 191h
301a		302a	302b	303a
				304a
				305a

Figure 3b

Shared File System Interface 230' -- Device Read and Radio Unit Write Files 192y in Directory 192x

File System Interface 230' : ../radio-unit_write/..						Directory 192x		192y	
-rw-rw-----	1	device_OS	SRU_OS	6582	Oct 4	2017	radio_parameters_loaded	192a	
-rw-rw-----	1	device_OS	SRU_OS	766	Apr 22	2017	https_response	192b	
-rw-rw-----	1	device_OS	SRU_OS	423	Apr 22	2017	MT_sms	192c	
-rw-rw-----	1	device_OS	SRU_OS	9884	Apr 22	2017	FTP_file_download	192d	
-rw-rw-----	1	device_OS	SRU_OS	98244	Mar 31	2017	log_file	306	
-rw-rw-----	1	device_OS	SRU_OS	98	Mar 31	2017	Server_B_digital_sig.	307	
-rw-rw-----	1	device_OS	SRU_OS	56	Feb 8	2017	power_control_settings	192g	
-rw-rw-----	1	device_OS	SRU_OS	4877	Oct 4	2017	cryptographic_parameters	192e	
-rw-rw-----	1	device_OS	SRU_OS	566	Feb 8	2017	radio_network_status	308	
-rw-rw-----	1	device_OS	SRU_OS	3844	Feb 8	2017	error_log	309	
-rw-rw-----	1	device_OS	SRU_OS	32	Feb 8	2017	network-date-time	309b	
				301b		302b		305b	
								•	
								•	
								•	

Figure 3c

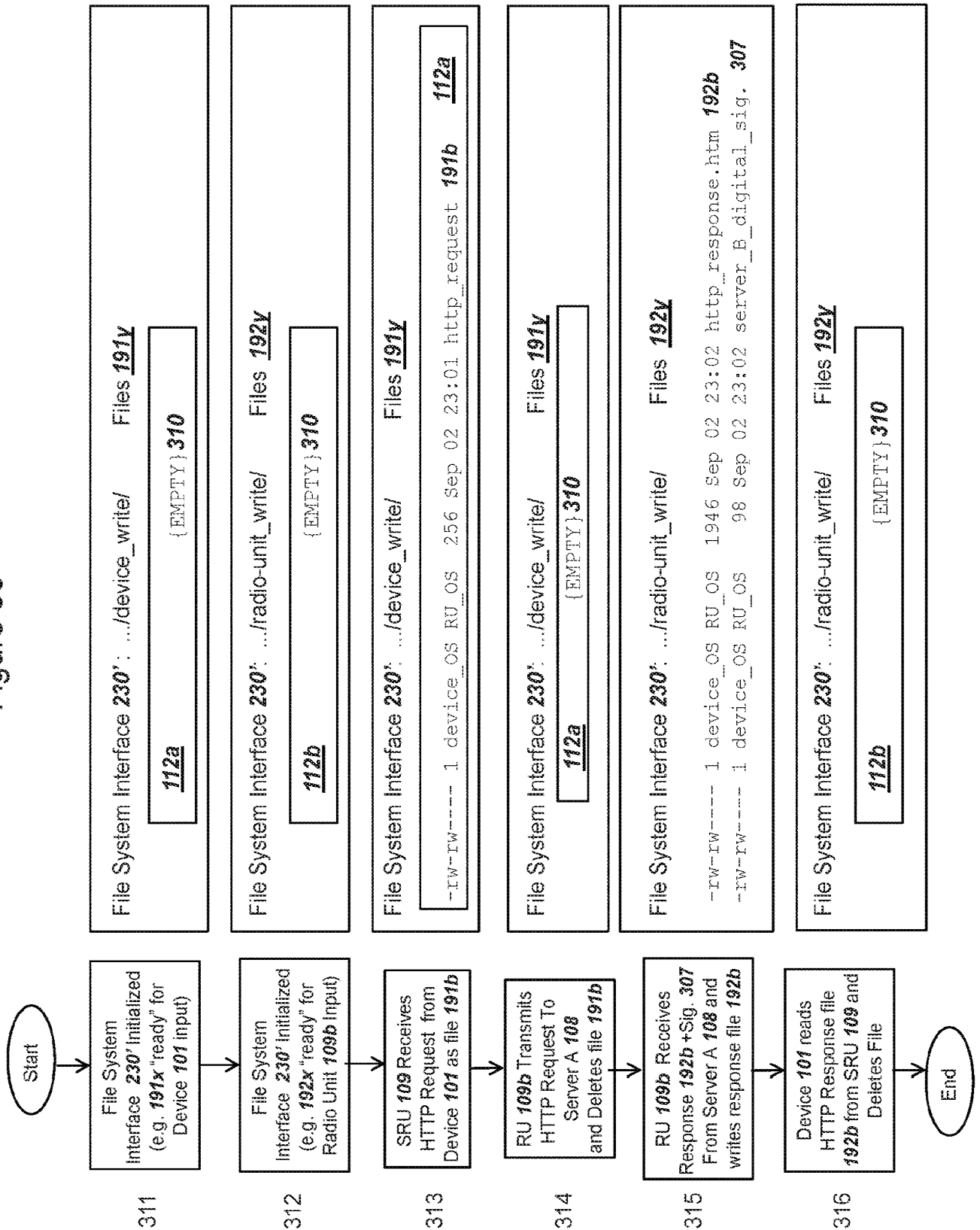


Figure 4a

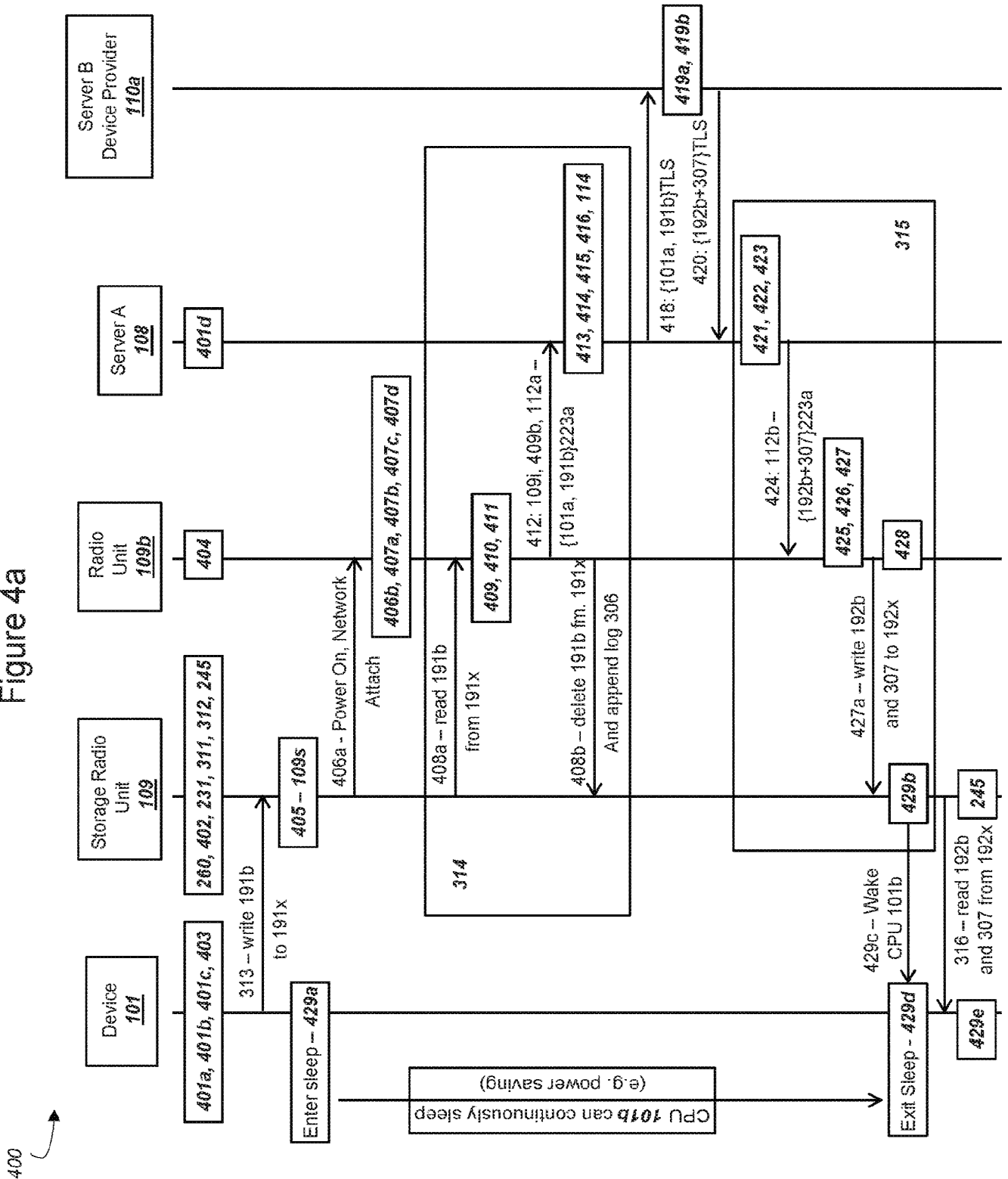


Figure 4b

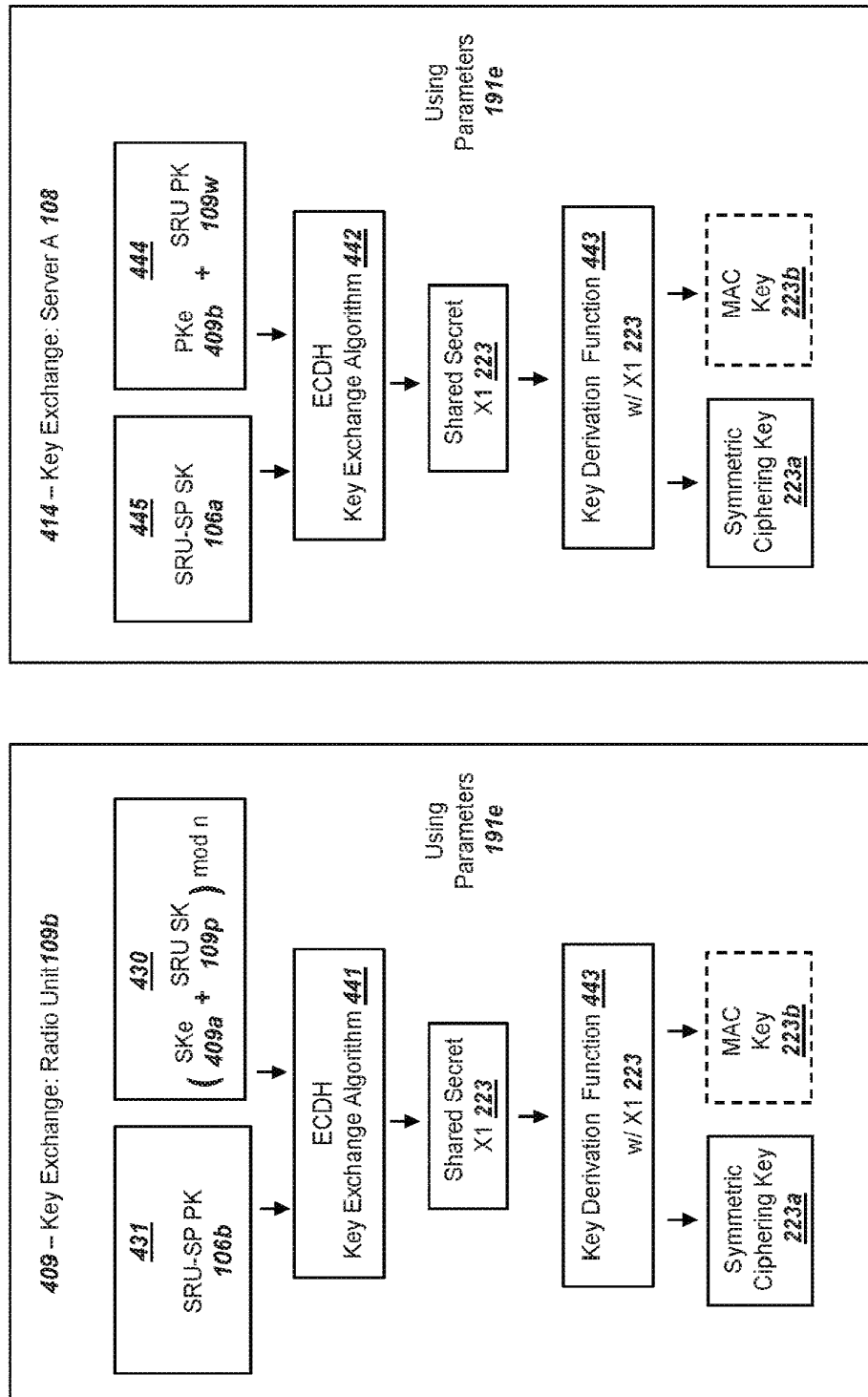


Figure 5b

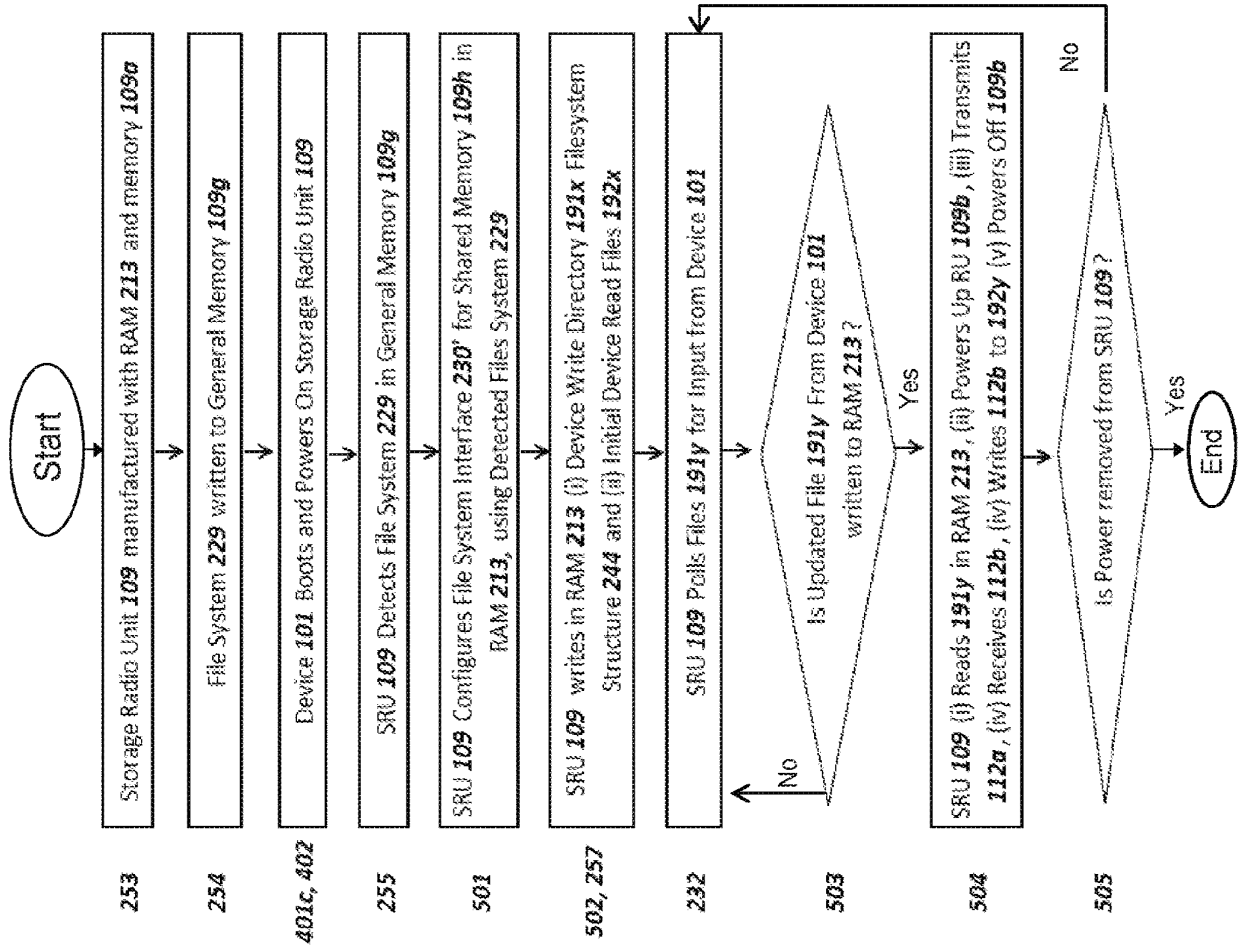
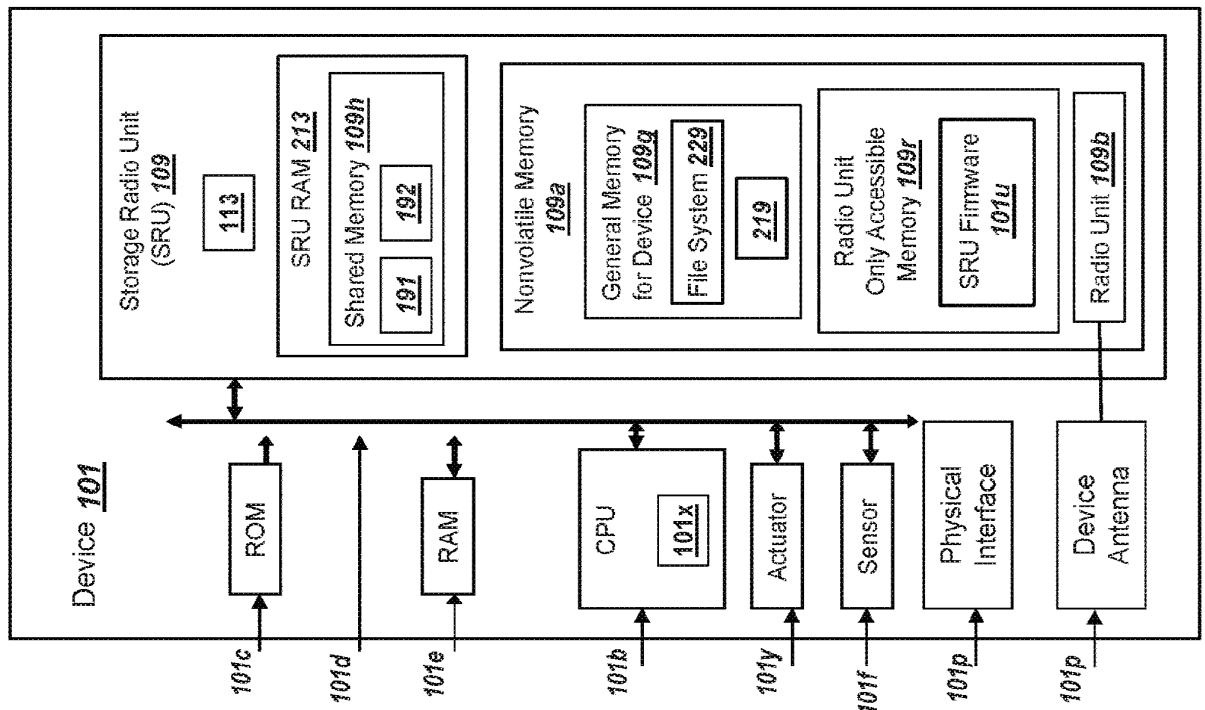


Figure 5a



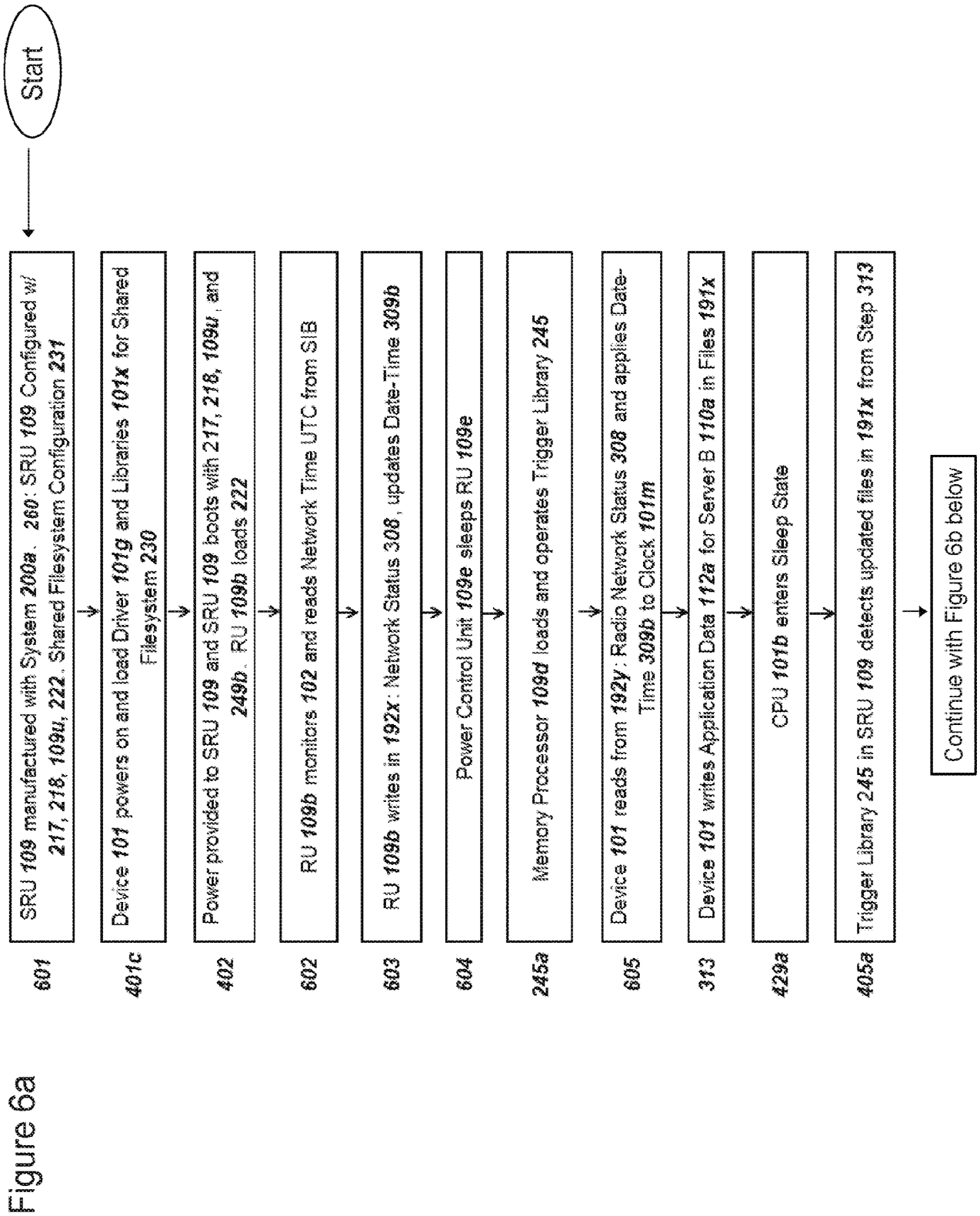


Figure 6b

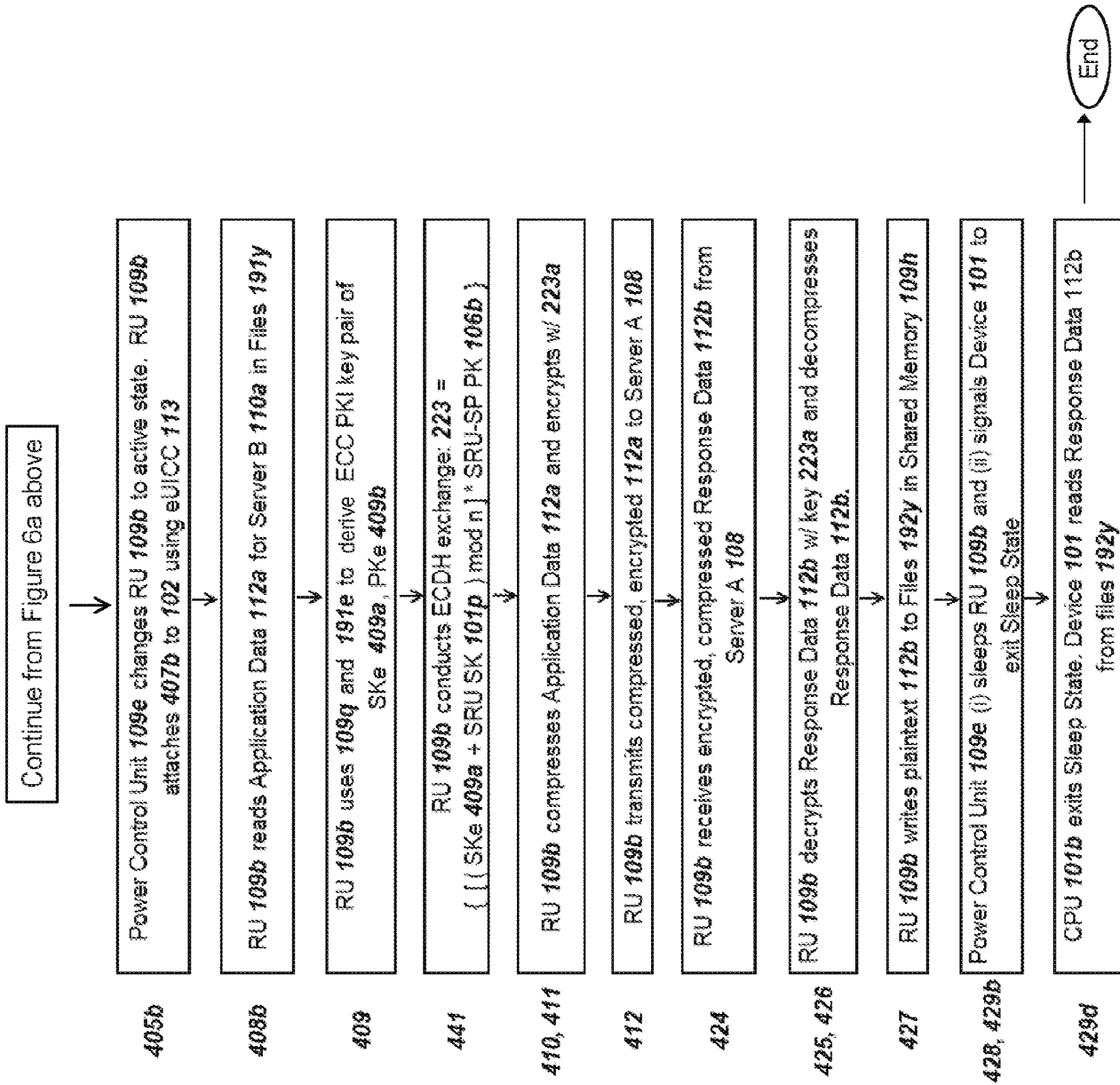


Figure 8

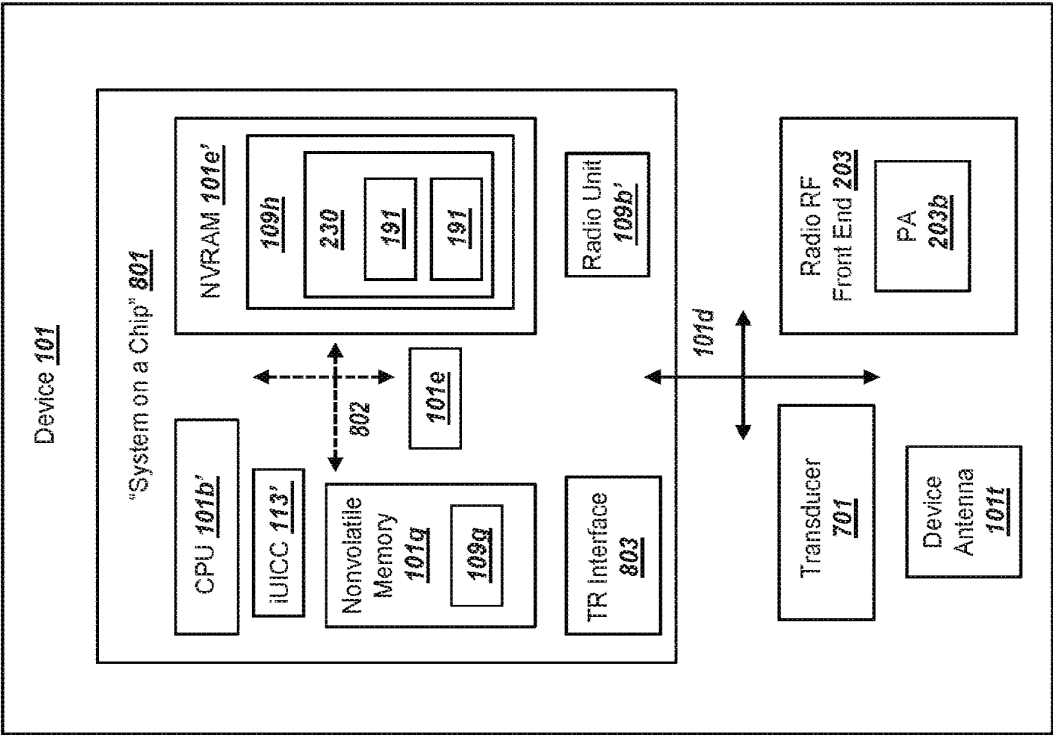
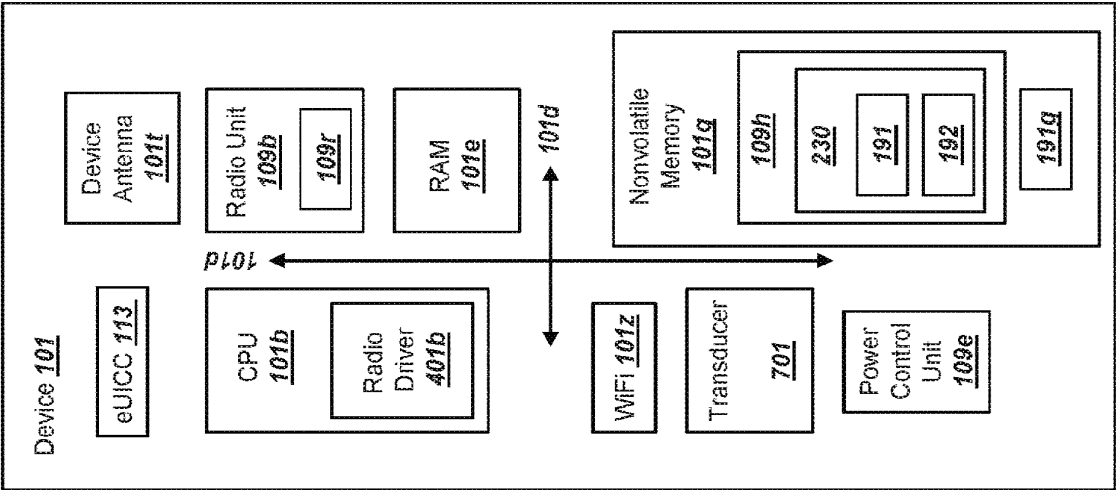


Figure 7



INTERNATIONAL SEARCH REPORT

International application No.

PCT/US 2019/051481

A. CLASSIFICATION OF SUBJECT MATTER

H04W 4/70 (2018.01)**H04L 9/28 (2006.01)**

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

H04W 4/00,4/70, 12/00-12/10, H04L 9/00,9/28, 12/00,12/28, 29/00-29/08, G06F 15/00,15/16-15/167, 1/00,1/26,1/32,1/3203, 16/00,16/10

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

PatSearch (RUPTO internal), USPTO, PAJ, Esp@cenet, Information Retrieval System of FIPS

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	US 2018/0144147 A1 (JOHN A. NIX) 24.05.2018, abstract, paragraphs [0002], [0015], [0016], [0021], [0023]-[0026], [0048], [0056], [0073], [0074], [0076], [0082], [0085], [0087], [0088], [0093]-[0095], [0111], [0112], [0121], [0122], [0124], [0129], [0135], [0146], [0155]-[0158], [0161], [0162], [0165], [0182], [0196], [0198]-[0200], [0202], [0250], [0258]-[0260]	1-4, 15-18, 26-29
Y	US 2016/0270000 A1 (M2M AND LOT TECHNOLOGIES, LLC) 15.09.2016, paragraphs [0010]-[0012], [0014], [0060], [0120], [0194]	1-4, 15-18, 26-29
A	US 2016/0135241 A1 (QUALCOMM INCORPORATED) 12.05.2016	1-4, 15-18, 26-29



Further documents are listed in the continuation of Box C.



See patent family annex.

* Special categories of cited documents:

“A” document defining the general state of the art which is not considered to be of particular relevance

“E” earlier document but published on or after the international filing date

“L” document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

“O” document referring to an oral disclosure, use, exhibition or other means

“P” document published prior to the international filing date but later than the priority date claimed

“T” later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

“X” document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

“Y” document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art

“&” document member of the same patent family

Date of the actual completion of the international search

02 December 2019 (02.12.2019)

Date of mailing of the international search report

12 December 2019 (12.12.2019)

Name and mailing address of the ISA/RU:

Federal Institute of Industrial Property,
Berezhkovskaya nab., 30-1, Moscow, G-59,
GSP-3, Russia, 125993
Facsimile No: (8-495) 531-63-18, (8-499) 243-33-37

Authorized officer

I. Kryazhev

Telephone No. (495) 531-64-81

Form PCT/ISA/210 (second sheet) (January 2015)

INTERNATIONAL SEARCH REPORT

International application No.

PCT/US 2019/051481

Box No. II Observations where certain claims were found unsearchable (Continuation of item 2 of first sheet)

This international search report has not been established in respect of certain claims under Article 17(2)(a) for the following reasons:

1. ☐ Claims Nos.:
because they relate to subject matter not required to be searched by this Authority, namely:
2. ☐ Claims Nos.:
because they relate to parts of the international application that do not comply with the prescribed requirements to such an extent that no meaningful international search can be carried out, specifically:
3. ☒ Claims Nos.: 5-14, 19-25, 30-40
because they are dependent claims and are not drafted in accordance with the second and third sentences of Rule 6.4(a).

Box No. III Observations where unity of invention is lacking (Continuation of item 3 of first sheet)

This International Searching Authority found multiple inventions in this international application, as follows:

1. ☐ As all required additional search fees were timely paid by the applicant, this international search report covers all searchable claims.
2. ☐ As all searchable claims could be searched without effort justifying additional fees, this Authority did not invite payment of additional fees.
3. ☐ As only some of the required additional search fees were timely paid by the applicant, this international search report covers only those claims for which fees were paid, specifically claims Nos.:
4. ☐ No required additional search fees were timely paid by the applicant. Consequently, this international search report is restricted to the invention first mentioned in the claims; it is covered by claims Nos.:

Remark on Protest

- ☐ The additional search fees were accompanied by the applicant's protest and, where applicable, the payment of a protest fee.
- ☐ The additional search fees were accompanied by the applicant's protest but the applicable protest fee was not paid within the time limit specified in the invitation.
- ☐ No protest accompanied the payment of additional search fees.

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Applicant: Network-1 Technologies, Inc. Confirmation No.: 2186
Title: Embedded Universal Integrated Circuit Card Supporting Two-Factor Authentication
Appl. No.: 16/453,682
Filing Date: June 26, 2019
Examiner: HO, Dao Q.
Art Unit: 2432
Docket No.: 63121/0184
Customer No.: 01912

SUPPLEMENTAL INFORMATION DISCLOSURE STATEMENT
LETTER UNDER 37 C.F.R. § 1.56

Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Sir:

In accordance with the duty of disclosure under 37 C.F.R. §1.56, the attention of the Patent and Trademark Office is hereby directed to the information item(s) listed on the attached Form PTO-SB08. Unless otherwise indicated herein, one copy of each item(s) is attached. It is respectfully requested that the information be expressly considered during the prosecution of this application, and that the item(s) be made of record therein and appear among the “References Cited” on any patent to issue therefrom.

Applicant wishes to bring to the attention of the Office that the Form PTO-SB08 previously submitted on July 27, 2021 contained an erroneous certification statement under 37 C.F.R. § 1.197(e)(1) and that the present submission is identical in substance.

Reference to Related Patents and Co-Pending Applications

The below chart provides information regarding related issued patents and co-pending applications filed by the present Applicant:

Serial No.	Attorney Docket No.	Filing Date	Status
17/304,922	63121/0209	2021-06-28	Application Undergoing Preexam Processing
17/249,242	63121/0202	2021-02-24	Application Dispatched from Preexam
16/879,325	63121/0198	2020-05-20	Application to issue as USP 11,082,218 on 2021-08-03
16/843,107	63121/0197	2020-04-08	Non-Final Rejection mailed 2021-07-02
16/721,153	63121/0195	2019-12-19	Final Office Action mailed 2021-04-15
16/593,561	63121/0190	2019-10-04	Non-Final Rejection mailed 2021-07-06
16/453,682	63121/0184	2019-06-26	Notice of Allowance mailed 2021-06-29
16/271,455	63121/0181	2019-02-08	USP 10,594,679
16/271,428	63121/0180	2019-02-08	USP 10,523,432
16/201,401	63121/0179	2018-11-27	USP 10,700,856
16/200,118	63121/0178	2018-11-26	USP 10,498,530
16/110,804	63121/0168	2018-08-23	USP 10,382,422
16/036,506	63121/0167	2018-07-16	USP 10,530,575
15/983,957	63121/0163	2018-05-18	USP 10,177,911
15/973,012	63121/0162	2018-05-07	USP 10,250,386
15/972,914	63121/0161	2018-05-07	USP 10,652,017
15/928,848	63121/0160	2018-03-22	USP 10,084,768
15/010,905	63121/0157	2016-01-29	USP 9,641,327
15/680,758	63121/0147	2017-08-18	USP 10,187,206
15/162,292	63121/0146	2016-05-23	USP 10,362,012
14/751,119	63121/0145	2015-06-25	USP 9,961,060
15/457,700	63121/0144	2017-03-13	USP 9,998,281
15/583,968	63121/0143	2017-05-01	USP 10,057,059
15/043,293	63121/0142	2016-02-12	USP 9,998,280
15/642,088	63121/0141	2017-07-05	USP 10,003,461
15/130,146	63121/0140	2016-04-15	USP 9,742,562
15/162,302	63121/0139	2016-05-23	USP 9,698,981
14/789,255	63121/0138	2015-07-01	USP 9,596,078
14/139,419	63121/0137	2013-12-23	USP 9,351,162
14/136,711	63121/0136	2013-12-20	USP 9,300,473
14/099,329	63121/0135	2013-12-06	USP 9,100,175
14/084,141	63121/0134	2013-11-19	USP 9,319,223
14/064,618	63121/0133	2013-10-28	USP 9,118,464
14/055,606	63121/0132	2013-10-16	USP 9,276,740

14/039,401	63121/0131	2013-09-27	USP 9,288,059
14/023,181	63121/0130	2013-09-10	USP 9,350,550
AU 2019246774	63121-0199	2019-10-08	Pending
GB 1608573	63121-0151	2014-10-27	Published on 2016-08-03
GB 2100530	63121-0201	2014-10-27	Published on 2021-05-12
CA 2,965,119	63121-0152	2014-10-27	Published on 2015-05-07
HK 17101082.8	63121-0153	2014-10-27	Published on 2017-10-20
EU 14868381.6	63121-0154	2014-12-04	Published on 2017-01-04
CA 2,969,829	63121-0155	2014-12-04	Published on 2015-06-11
HK 17106540.3	63121-0156	2014-12-04	Published on 2018-01-19

The Examiner is invited to review the prosecution history of each of these applications, including the cited references of record and related office actions, as the Examiner deems appropriate.

* * * * *

Since this Supplemental Information Disclosure Statement is being filed more than three months after the U.S. filing date and after the mailing date of a Notice of Allowance, but accompanies a Request for Continued Examination, no certification or fee is required. Please charge any fee deficiency or credit any overpayment to Deposit Account No. 01-1785, under Docket No. 63121/0184.

Respectfully submitted,

AMSTER, ROTHSTEIN & EBENSTEIN LLP
Attorneys for Applicant
90 Park Avenue
New York, NY 10016
(212) 336-8000

Dated: August 5, 2021

By: /Albert J. Boardman/
Albert J. Boardman
Reg. No. 70,601



(12) **DEMANDE DE BREVET CANADIEN
CANADIAN PATENT APPLICATION**

(13) **A1**

(86) Date de dépôt PCT/PCT Filing Date: 2019/04/17
(87) Date publication PCT/PCT Publication Date: 2019/10/31
(85) Entrée phase nationale/National Entry: 2021/02/24
(86) N° demande PCT/PCT Application No.: US 2019/027968
(87) N° publication PCT/PCT Publication No.: 2019/209598
(30) Priorité/Priority: 2018/04/27 (US62/664,057)

(51) Cl.Int./Int.Cl. *H04L 9/08* (2006.01),
G06F 21/62 (2013.01), *H04L 29/06* (2006.01),
H04L 29/08 (2006.01), *H04L 9/14* (2006.01),
H04L 9/32 (2006.01), *H04W 12/00* (2021.01),
H04W 12/04 (2021.01), *H04W 12/06* (2021.01),
H04W 12/08 (2021.01), *H04W 84/12* (2009.01),
H04W 88/02 (2009.01)

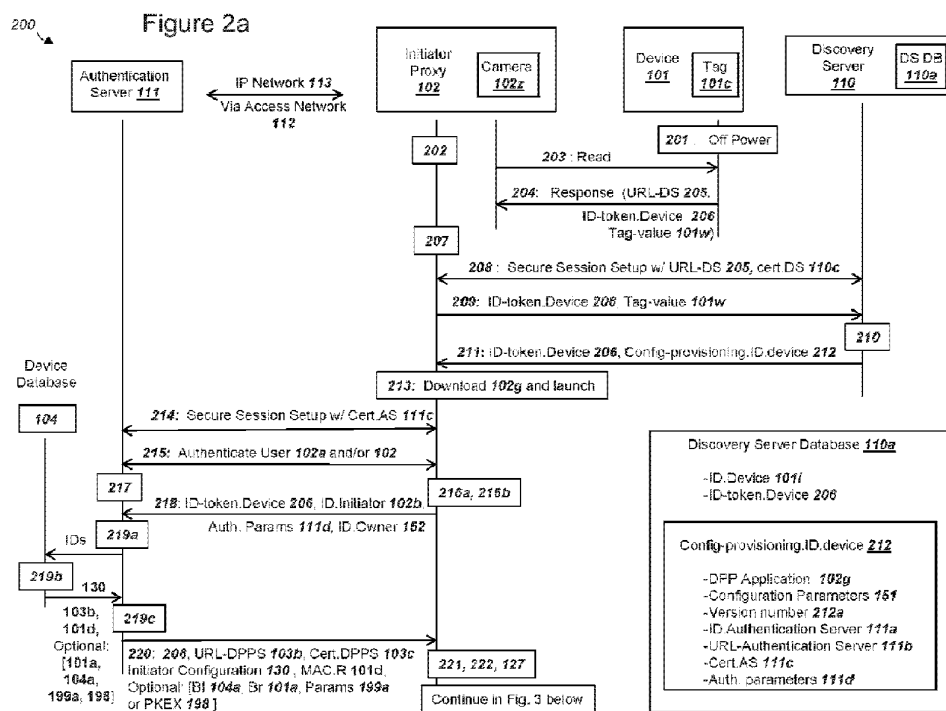
(71) Demandeur/Applicant:
IOT AND M2M TECHNOLOGIES, LLC, US

(72) Inventeur/Inventor:
NIX, JOHN A., US

(74) Agent: HICKS & ASSOCIATES

(54) Titre : PROTOCOLE DE FOURNITURE DE DISPOSITIF HEBERGE AVEC SERVEURS ET UN INITIATEUR RESEAU

(54) Title: A HOSTED DEVICE PROVISIONING PROTOCOL WITH SERVERS AND A NETWORKED INITIATOR



(57) Abrégé/Abstract:

A network can operate a WiFi access point with credentials. An unconfigured device can (i) support a Device Provisioning Protocol (DPP), (ii) record responder bootstrap public and private keys, and (iii) be marked with a tag. The network can record initiator bootstrap public and private keys, as well as derived initiator ephemeral public and private keys. An initiator can (i) operate a DPP application, (ii) read the tag, (iii) establish a secure and mutually authenticated connection with the network, and (iv) send the network data within the tag. The network can record the responder bootstrap public key and derive an encryption key with the (i) recorded responder bootstrap public key and (ii) derived initiator ephemeral private key. The network can encrypt credentials using the derived encryption key and send the encrypted credentials to the initiator, which can forward the encrypted credentials to the device, thereby supporting a device configuration.

(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property
Organization
International Bureau



(10) International Publication Number
WO 2019/209598 A1

(43) International Publication Date
31 October 2019 (31.10.2019)

(51) International Patent Classification:

H04L 9/08 (2006.01) H04W 12/00 (2009.01)
H04L 9/32 (2006.01) G06F 21/62 (2013.01)
H04L 29/06 (2006.01) H04L 9/14 (2006.01)
H04W 12/06 (2009.01) H04W 12/04 (2009.01)
H04W 12/08 (2009.01) H04W 84/12 (2009.01)
H04L 29/08 (2006.01) H04W 88/02 (2009.01)

(71) Applicant: **IOT AND M2M TECHNOLOGIES, LLC**
[US/US]; 807 Davis Street, Unit 2207, Evanston, IL
60201-7104 (US).

(72) Inventor: **NIX, John, A.**; c/o Iot and M2M Technolo-
gies, LLC, 807 Davis Street, Unit 2207, Evanston, IL
60201-7104 (US).

(21) International Application Number:

PCT/US2019/027968

(74) Agent: **THOMSON, Kirsten, L.**; McDonnell Boehnen
Hulbert & Berghoff LLP, 300 South Wacker Drive, Chica-
go, IL 60606 (US).

(22) International Filing Date:

17 April 2019 (17.04.2019)

(25) Filing Language:

English

(26) Publication Language:

English

(30) Priority Data:

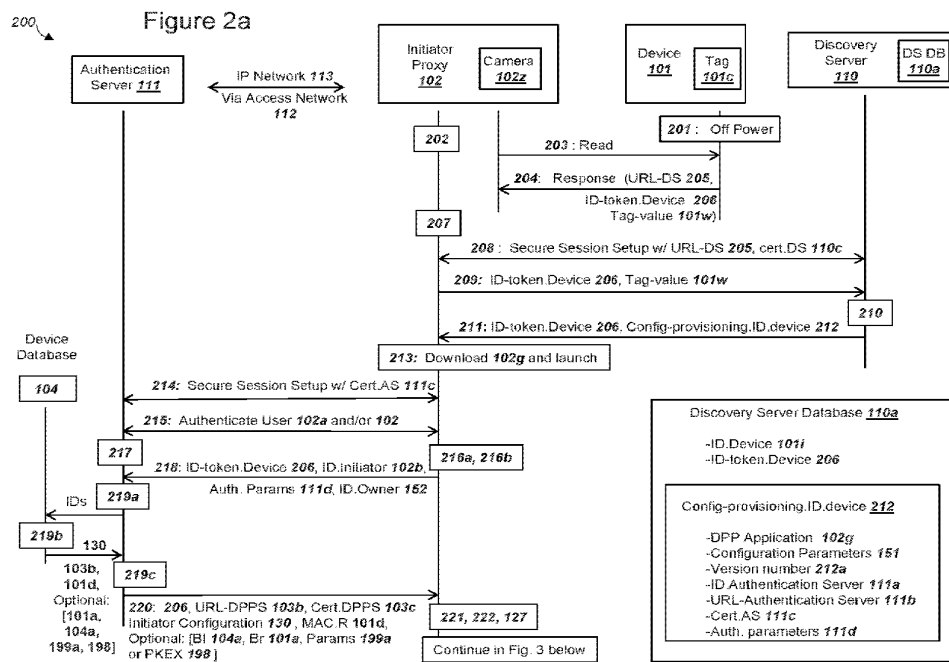
62/664,057

27 April 2018 (27.04.2018)

US

(81) Designated States (unless otherwise indicated, for every
kind of national protection available): AE, AG, AL, AM,
AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ,
CA, CH, CL, CN, CO, CR, CU, CZ, DE, DJ, DK, DM, DO,
DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN,
HR, HU, ID, IL, IN, IR, IS, JO, JP, KE, KG, KH, KN, KP,
KR, KW, KZ, LA, LC, LK, LR, LS, LU, LY, MA, MD, ME,

(54) Title: A HOSTED DEVICE PROVISIONING PROTOCOL WITH SERVERS AND A NETWORKED INITIATOR



(57) Abstract: A network can operate a WiFi access point with credentials. An unconfigured device can (i) support a Device Provisioning Protocol (DPP), (ii) record responder bootstrap public and private keys, and (iii) be marked with a tag. The network can record initiator bootstrap public and private keys, as well as derived initiator ephemeral public and private keys. An initiator can (i) operate a DPP application, (ii) read the tag, (iii) establish a secure and mutually authenticated connection with the network, and (iv) send the network data within the tag. The network can record the responder bootstrap public key and derive an encryption key with the (i) recorded responder bootstrap public key and (ii) derived initiator ephemeral private key. The network can encrypt credentials using the derived encryption key and send the encrypted credentials to the initiator, which can forward the encrypted credentials to the device, thereby supporting a device configuration.

[Continued on next page]

WO 2019/209598 A1

MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ,
OM, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SA,
SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN,
TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(84) Designated States (*unless otherwise indicated, for every kind of regional protection available*): ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, ST, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, RU, TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, KM, ML, MR, NE, SN, TD, TG).

Published:

— with international search report (Art. 21(3))

A HOSTED DEVICE PROVISIONING PROTOCOL WITH SERVERS AND A NETWORKED INITIATOR

CROSS-REFERENCE TO RELATED APPLICATIONS

5 This international PCT application claims the benefit of the filing date of U.S. Provisional Patent Application Serial No. 62/664,057, filed on April 27, 2018, which is hereby incorporated by reference in its entirety.

BACKGROUND

Technical Field

10 The present systems and methods relate to authentication and configuration of WiFi radios, and more particularly, to using a Device Provisioning Protocol with a networked initiator and a set of servers that record PKI keys, in order securely transfer a set of credentials to a device.

Description of Related Art

15 The ability to connect transducers such as sensors and actuators with a network is a growing field with many economical applications. As the costs for both electronic hardware and bandwidth continue to decrease, the use of networked transducers is expected to continue increasing over the coming decades. Connecting transducers to a network can be referred to as “machine-to-machine (M2M) communications” or “the Internet of Things (IoT).” Among many potential benefits, IoT technologies allow automated monitoring and/or control of assets, equipment, personnel, or a physical location where manual monitoring may not be economical. Many applications for the “Internet of Things” significantly reduce costs by using automated monitoring instead of manual techniques. Prominent examples of IoT applications today include monitoring and control for building heating/air-conditioning, automobiles, alarm systems, and tracking devices. Fast growing markets for IoT applications today include health applications such as the remote monitoring of a person’s fitness activity, heartbeat, or glucose levels, monitoring of industrial equipment deployed remotely in the field, and also security systems.

20 Many IoT applications can leverage wireless networking technologies, in addition to wired technologies such as Ethernet. Wireless technologies such as wireless local area networks and wireless wide area networks have proliferated around the world over the past 30 years, and usage of these wireless networks is also expected to continue to grow. Wireless local area network (LAN) technologies include WiFi based on the series of 802.11 standards from the Institute of Electrical and Electronics Engineers (IEEE). The many options to connect a transducer device to a wireless network creates opportunities for new products and services, but also creates several classes of

problems that need to be solved. Many of these problems or needs in the art pertain to securely and efficiently configuring the transducer devices. A need exists in the art to allow a user or a network to securely and easily upload to the device a set of network access credentials for a wireless network.

Manually configuring devices to (i) use different security keys, (ii) apply new network access credentials to connect with the owner's selected wireless network, and/or (iii) update running firmware can be time consuming and difficult for users. Entering network access credentials is also prone to errors and especially difficult for devices with limited user interfaces, where a limited user interface is common for many devices supporting applications for the "Internet of Things". Loading credentials and network configuration into a device in order to obtain connectivity to a network can also be known as device provisioning or device configuration. In an attempt to address these needs to simplify or automate device provisioning, the WiFi Alliance™ released a Device Provisioning Protocol (DPP) specification on April 9, 2018 as version 1.0 (DPPv1.0). Although the DPPv1.0 defines a series of messages between an initiator and a responder in order to support a device configuration, several unmet needs remain in the art in order to securely and easily deploy a Device Provisioning Protocol at wide scale with potentially billions of devices over many years.

To highlight one example unmet need by a DPPv1.0, WiFi capable devices currently being designed and deployed may have an expected life of more than a decade. During that timeframe, the device may need to undergo a configuration or device provisioning step more than one time. Reasons could include either (i) the device undergoes a "factory reset", or (ii) the device needs to be provisioned a second time, such as if the device owner changes or the device is moved to a different network, and other reasons exist as well. But, as stated in DPPv1.0, "threats against this bootstrapping method are possible if public keys and/or codes are reused." A need exists in the art to support a Device Provisioning Protocol in a manner where bootstrapping public keys can be securely reused. A need exists in the art for bootstrapping PKI keys to be securely updated after a first instance of conducting a DPP, such that the updated bootstrapping PKI keys can be subsequently used at a later time for a second instance of conducting a DPP.

In order to securely provision a device, a device may prefer to conduct mutual authentication with an initiator that records and operates with initiator PKI keys. The device could be a high value piece of equipment, such as an automobile, health monitoring equipment in a hospital, or industrial equipment, etc. that could be targets for potential hackers seeking to "provision" the devices to a network under their control or a network providing the hackers access to the device. In these cases, an owner or user of a device may prefer that a device provisioning protocol use mutual authentication. The DPPv1.0 supports mutual authentication in order to securely authenticate a device and an initiator before transferring network access credentials to the device, but creates a new and shifted problem of both (i) securely providing an initiator bootstrap private key to an initiator and (ii) the corresponding initiator bootstrap public key to a device. In other words, recording an

initiator bootstrap public key in a device can be difficult if there is no pre-existing relationship or communication between a device and an initiator before the start of a device provisioning protocol. In addition, an insecure initiator may have no pre-existing relationship with a target network to provide connectivity to the device, yet the network may need to securely transfer credentials to the device through the insecure initiator. A need exists in the art to securely and readily enable an initiator to have access to an initiator bootstrap private key while also enabling a device to have access to the corresponding initiator bootstrap public key. A need exists in the art for a network to securely transfer credentials to a device through an insecure initiator.

With the widespread deployment of WiFi access points, a device may have access to potentially a plurality of different access points associated with potentially different networks. Although DPPv1.0 supports configuration of a device, DPPv1.0 does not suggest (i) which of a possible plurality of potential different access points should be selected, nor (ii) how credentials for a selected access point could be obtained by an initiator. Need exists in the art for an initiator to automatically determine a list of available networks and to automatically and securely receive a set of credentials for a preferred network for the device. A need exists in the art for the credentials to be encrypted in a manner such that the initiator cannot feasibly read the credentials, but the device can receive the encrypted credentials and convert the encrypted credentials into plaintext form in order for the device to load and use the plaintext credentials.

A network can use DPPv1.0 in order to connect devices to the network using an initiator. The security for conducting a device provisioning protocol according to DPPv1.0 depends on the security of an initiator, since as envisioned by DPPv1.0 the initiator (i) records an initiator bootstrap private key and (ii) conducts key exchanges with the initiator bootstrap private key. However, an initiator for conducting a DPP may be outside the control of a network and may also be an insecure device. An initiator could be a “rooted” mobile phone operated by a hacker, as one example. Or, the initiator could simply be a mobile device that is insecure or operates with security policies that do not meet the standards of the network. A need exists in the art for an insecure initiator to conduct a device provisioning protocol that is compatible with DPPv1.0, while simultaneously keeping (i) the initiator bootstrap private key and (ii) cryptographic operations with the initiator bootstrap private key under the control or within the security policies of a network.

The secure use of a device provisioning protocol can depend on an initiator having access to a responder bootstrap public key. Access to the responder bootstrap public key for the initiator is described in DPPv1.0 as supported in an “out of band” operation. The security of a device provisioning protocol can depend on the security of the steps for an initiator to gain access to the responder bootstrap public key. The initiator could also comprise an insecure device, as described in the paragraph above. If the responder bootstrap public key is externally readable by an initiator, such as contained in a QR code, then potentially many different initiators could read the responder

bootstrap public key and attempt a DPP with a device. Although this ease may be preferred for some devices and applications, many devices may require a higher level of security such that a responder bootstrap public key can only be accessed by an authorized initiator. A need exists in the art for an initiator to have access to a responder bootstrap public key, while simultaneously keeping
5 the responder bootstrap public key secured.

Many other examples exist as well for needs in the art to support efficient yet secure device configuration or provisioning, and the above are examples are just a few and intended to be illustrative instead of limiting.

SUMMARY

Methods and systems are provided for a networked initiator to conduct a device provisioning protocol (DPP) with a responder operating in a device. The device provisioning protocol can support the Device Provisioning Protocol specification version 1.0 and subsequent versions from the WiFi Alliance™. Conducting a DPP with a device can transfer a set of network credentials to the device, in order for the device to connect with a network using the network credentials. The set of network credentials can also include a network configuration, where the network configuration includes settings to use with a WiFi radio in the device, in order for the WiFi radio to communicate with other nodes with the credentials, including an access point which can provide connectivity to an IP network for the device.

A device can operate a responder, which listens for an authentication request from an initiator. The responder can record and operate an initiator configuration, specifying a frequency band to utilize, a WiFi networking technology to implement (e.g. 802.11 ac, 802.11ah, etc.) and a channel list. When the device is awake or powered on, the responder can listen for incoming authentication requests using a WiFi radio for the device. The responder can record a responder bootstrap public key, a responder bootstrap private key, and, when supporting mutual authentication with an initiator, an initiator bootstrap public key. The PKI keys can be associated with a selected set of cryptographic parameters, where the parameters specify settings to use for cryptographic operations by a responder. The responder can also include a random number generator, a PKI key pair generation algorithm, a secure hash algorithm, an elliptic curve Diffie Hellman (ECDH) key exchange algorithm, and a symmetric ciphering algorithm. The device with the responder can be marked with a tag, such that the tag records an identity for the device and a uniform resource locator (URL) for a server.

A network can operate an access point, where the access point operates with a set of credentials. The credentials can include an identifier for the network such a service set identifier (SSID), a pairwise master key (PMK) or a preshared key (PSK), a pairwise master key identifier (PMKID), and also a configuration. The access point can support wireless networking standards in the series of 802.11 standards. A network can also operate a plurality of access points. The network can operate or be associated with a Device Provisioning Protocol (DPP) server, a device database, an authentication server, and a discovery server. The network can be connected to an IP network, which can comprise the globally routable public Internet. Different servers within a network can be connected using an Intranet, a virtual private network (VPN), Internet Protocol Security (IPSEC), or similar techniques to secure communication between the servers.

The network can operate or communicate with the device database. The DPP server in the network can also communicate with the device database. The device database can record information for a plurality of devices before responders in the devices conduct the device provisioning protocol with the initiator. The device database can record the device identity, a responder identity, which could be in the form of a responder medium access control (MAC) address, an initiator configuration, and a key table. The key table in a device database can record for each device a responder bootstrap public key, an initiator bootstrap public key, and an initiator bootstrap private key. Recorded data within a device database can be obtained by a network (i) upon device manufacturing from a device manufacturer, or (ii) before distribution of a device to the location of the network's access point.

The DPP server for a network can operate a server database, where the server database records information for a plurality of devices connecting with the network. The server database can record the device identity, medium access control (MAC) addresses for devices and initiators, an initiator mode, a selected set of cryptographic parameters for each device, an initiator bootstrap private key, the initiator bootstrap public key, the responder bootstrap public key, an initiator ephemeral public key, and initiator ephemeral private key, an initiator bootstrap key sequence number to identify and track the initiator bootstrap PKI keys, a public key exchange (PKEX) key, a first, second, and third derived shared secret key for a responder. Depending on the initiator mode, not all PKI keys and secret shared keys may be recorded in the server database. The DPP server can also include a set cryptographic algorithms, where the cryptographic algorithms can include a random number generator, a PKI key pair generation algorithm, a secure hash algorithm, an elliptic curve Diffie Hellman key exchange algorithm, and a symmetric ciphering algorithm. The cryptographic algorithms within a DPP server can be compatible with the equivalent set of cryptographic algorithms in the responder and also the initiator.

An initiator can be a computing device that includes a WiFi radio and can be associated with an initiator user. The initiator can operate the WiFi radio with an initiator configuration in order for the initiator to send and receive messages with the responder. The initiator is located in sufficient proximity with the responder so that the WiFi radio for the initiator and the WiFi radio for the responder can communicate using 802.11 standards. The initiator can operate a DPP application, where the DPP application can also include the set of cryptographic algorithms, which are compatible with cryptographic algorithms in both the responder and the DPP server. The initiator can use an access network in order to communicate with the network via an IP network. The initiator can record a user configuration for the WiFi radio, where the user configuration can be temporarily replaced with the initiator configuration in order to conduct the device provisioning protocol (DPP) session with the responder. After the DPP session, the initiator configuration can be

replaced with the user configuration, such that the initiator is restored to its previous state before the DPP session.

The discovery server for a network can operate a discovery server database. The discovery server can receive requests from initiators, where the requests include a device identity from the tag of the device. The discover server database can record configuration provisioning data for the device with the device identity, and a URL for an authentication server for use by the initiator in order for the initiator to (i) authenticate with the network before conducting the device provisioning protocol and (ii) receive information from the DPP server. The initiator can send a request to the discovery server with the device identity. The discovery server can query the discovery server database with the device identity in order to respond to the initiator with the configuration provisioning data for the initiator, where the configuration provisioning data includes the URL for the authentication server.

The initiator can call the URL for the authentication server and conduct an authentication step with the authentication server. After authentication of the initiator, the authentication server can query the device database in order to obtain information for the initiator to use in order to conduct a device provisioning protocol. The information for the initiator can include a URL for the DPP server, an initiator configuration, and optionally the initiator bootstrap public key and the responder bootstrap public key, where the optional information can depend on the initiator mode for a DPP selected by the network.

The initiator can call the URL for the DPP server and setup a secure session via TLS or similar standards. The initiator can then use the WiFi radio and wireless WAN radio to conduct a radio frequency scan in order to collect a networks available list. The networks available list can include identities broadcast by wireless networks and available at the physical location of the initiator near the device. The initiator can send the networks available list to the DPP server. The DPP server can use the networks available list to select a primary access network for the device and obtain credentials for the primary access network for the device to use. The DPP server can also select an initiator mode, which can specify the subsequent series of steps for the initiator to take in order to conduct a DPP session with the responder. The DPP server can send the initiator the initiator mode, where a DPP application operating in the initiator implements the mode. The DPP server can obtain the initiator bootstrap PKI keys and the responder public key from the device database, as well as a selected set of cryptographic parameters in order to use the PKI keys.

In a first initiator mode, the DPP server can then conduct a series of steps in order to generate data for a Device Provisioning Protocol (DPP) authorization request. The DPP server can derive an initiator ephemeral PKI key pair. The DPP server can conduct a first initiator ECDH key exchange with the responder bootstrap public key and the derived initiator private key to derive a key k1. The initiator can derive an initiator nonce and encrypt the initiator nonce using the derived

key k_1 , where the encrypted initiator nonce can comprise a first ciphertext. The initiator can send (i) secure hash values for the initiator bootstrap public key and the responder bootstrap public key, the (ii) derived initiator ephemeral public key, and (iii) the first ciphertext to the initiator. The initiator can send a DPP authentication request to the responder.

5 The responder can receive and process with DPP authentication request. The responder can record a plurality of initiator bootstrap public keys, where the plurality of initiator bootstrap public keys could be recorded during manufacturing or distribution of the device with the responder before the device was placed in the physical proximity of the network access point. At least one of the recorded initiator bootstrap public keys can be selected based on the hash value for the key received
10 in the DPP authentication request. The responder can generate a responder ephemeral PKI key pair. The responder can conduct a first responder key exchange using the recorded responder bootstrap private key and the received initiator ephemeral public key in order to derive the key k_1 . The responder can conduct a second responder key exchange using the derived responder ephemeral private key and the received initiator ephemeral public key in order to derive a key k_2 . The
15 responder can conduct a third key exchange using the selected initiator bootstrap public key and a combination of (i) the responder ephemeral private key and the responder bootstrap private key in order to derive a key k_e . The responder can decrypt the first ciphertext using the key k_1 and read the plaintext initiator nonce. The responder can calculate a responder authentication value using the initiator nonce and encrypt the value using the key k_e , creating a second ciphertext. The responder
20 can generate a responder nonce and encrypt both the responder nonce and the second ciphertext using the key k_2 , creating a third ciphertext. The responder can send a DPP authentication response to the initiator, where the response includes the third ciphertext and the responder ephemeral public key.

The initiator can receive the DPP authentication response from the responder. The initiator
25 can send the third ciphertext and the responder ephemeral public key to the DPP server, along with an identity for the device. The DPP server can conduct a second initiator key exchange using the responder ephemeral public key and the initiator ephemeral private key in order to derive the key k_2 . The DPP server can conduct a third initiator key exchange using (i) a combination of the responder bootstrap public key and the responder ephemeral public key and (ii) the initiator bootstrap private
30 key in order to derive the key k_e . The DPP server can decrypt the third ciphertext using the derived key k_2 . The DPP server can read the plaintext responder nonce and the second ciphertext. The DPP server can decrypt the second ciphertext using the derived key k_e . The DPP server can read the plaintext responder authentication value from the second ciphertext. The DPP server can calculate the responder authentication value, and if the received responder authentication value matches the
35 calculated responder authentication value, then the responder is authenticated.

The DPP server can calculate an initiator authentication value using the received responder nonce. The DPP server can encrypt the initiator authentication value using the key ke , thereby creating a fourth ciphertext. The DPP server can send the fourth ciphertext to the initiator. The initiator can send a DPP authentication confirm message to the responder with the fourth ciphertext.

5 The responder can decrypt the fourth ciphertext using the derived key ke and read the plaintext initiator authentication value. The responder can calculate the same initiator authentication value, and if the received authentication value matches the calculated authentication value then the initiator is authenticated with the responder.

After the mutual authentication, the responder can take the roll of an enrollee and the

10 initiator can take the roll of a configurator. The responder can generate an enrollee nonce and also a configuration attribute, where the configuration attribute describes WiFi capabilities for the device. The responder can create a fifth ciphertext by encrypting the enrollee nonce and the configuration attribute with the derived key ke . The responder can send a DPP configuration request to the initiator, where the DPP configuration request can include the fifth ciphertext. The initiator can send

15 the fifth ciphertext to the DPP server.

The DPP server can receive the fifth ciphertext from the initiator and decrypt the fifth ciphertext using the derived key ke , in order to read the plaintext enrollee nonce and the configuration attribute. Using the selected primary network from above and the configuration attribute, the DPP server can process the selected network credentials. The selected network

20 credentials were described above and can be obtained by a DPP server using the networks available list from the initiator. The DPP server can encrypt the received enrollee nonce and the network credentials using the derived key ke , creating a sixth ciphertext. The DPP server can send the sixth ciphertext to the initiator. The initiator can forward the sixth ciphertext to the responder in a DPP configuration response message.

25 The responder can receive the DPP configuration response message and decrypt the sixth ciphertext using the derived key ke in order to read the plaintext network credentials. The responder can pass the network credentials to the device, which can apply the credentials with the WiFi radio for the device. The device can subsequently connect with the network access point using the received network credentials. The network access point can provide connectivity to an IP network

30 for the device. The device can establish a secure session with the DPP server using the network access point. The DPP server can derive a new initiator bootstrap PKI key pair and send the new initiator bootstrap public key to the device via the secure session. The device can record the new initiator bootstrap public key for use in a subsequent DPP session, if necessary, for a later time. The DPP server can send the device database the new initiator bootstrap PKI keys. The DPP server and

35 the device can record that the DPP session is successfully completed.

These as well as other aspects and advantages will become apparent to those of ordinary skill in the art by reading the following detailed description, with reference where appropriate to the accompanying drawings.

BRIEF DESCRIPTION OF THE DRAWINGS

Various exemplary embodiments are described herein with reference to the following drawings, wherein like numerals denote like entities.

5 Figure 1a is a graphical illustration of an exemplary system, where an initiator communicates with a network and a responder in a device, in order to conduct a device provisioning protocol, in accordance with exemplary embodiments;

Figure 1b is a graphical illustration of a device provisioning protocol for authentication and configuration of a responder, in accordance with conventional technology;

10 Figure 1c is a graphical illustration of a device provisioning protocol for (i) authentication and configuration of a responder and (ii) authentication of an initiator, in accordance with conventional technology;

Figure 1d is a graphical illustration of hardware, firmware, and software components for an initiator, including an initiator configuration step, in accordance with exemplary embodiments;

15 Figure 2a is a simplified message flow diagram illustrating an exemplary system with exemplary data sent and received by an initiator, in accordance with exemplary embodiments;

Figure 2b is an illustration of an exemplary device database, with tables for a device database recording exemplary data, in accordance with exemplary embodiments;

20 Figure 3a is a simplified message flow diagram illustrating an exemplary system with exemplary data sent and received by a device database, a DPP server, an initiator proxy, and a responder, in accordance with exemplary embodiments;

Figure 3b is an illustration of an exemplary server database, with tables for a server database recording exemplary data, in accordance with exemplary embodiments;

25 Figure 4a is a flow chart illustrating exemplary steps for conducting a key exchange using PKI keys in order to derive a shared secret key, and for using the derived shared secret key to encrypt and decrypt data, in accordance with exemplary embodiments;

Figure 4b is a flow chart illustrating exemplary steps for conducting a key exchange using PKI keys in order to derive a shared secret key, and for using the derived shared secret key to encrypt and decrypt data, in accordance with exemplary embodiments;

30 Figure 4c is a flow chart illustrating exemplary steps for conducting a key exchange using PKI keys in order to derive a shared secret key, and for using the derived shared secret key to encrypt and decrypt data, in accordance with exemplary embodiments;

35 Figure 5a is a simplified message flow diagram illustrating an exemplary system with exemplary data sent and received by a device database, a DPP server, an initiator proxy, and a responder, in accordance with exemplary embodiments;

Figure 5b is a simplified message flow diagram illustrating an exemplary system with exemplary data sent and received by a device database, a DPP server, an initiator proxy, and a responder, in accordance with exemplary embodiments;

Figure 6a is a simplified message flow diagram illustrating an exemplary system with exemplary data sent and received by a device database, a DPP server, a WiFi access point, and a device, in accordance with exemplary embodiments;

Figure 6b is a graphical illustration of an exemplary system, where a responder records a plurality of initiator bootstrap public keys and the responder selects and uses an initiator bootstrap public key, in accordance with exemplary embodiments;

Figure 7 is a graphical illustration of an exemplary system, where an access point conducts a configuration step for a device using a hosted device provisioning protocol, in accordance with exemplary embodiments; and,

Figure 8 is a graphical illustration of an exemplary system, where an access point operates as an initiator for a device with a responder, in order transfer a set of network credentials to the device, in accordance with exemplary embodiments.

DETAILED DESCRIPTION OF EXEMPLARY EMBODIMENTS OF THE INVENTION**Figure 1a**

Figure 1a is a graphical illustration of an exemplary system, where an initiator communicates with a network and a responder in a device, in order to conduct a device provisioning protocol, in accordance with exemplary embodiments. The system 100 can include an initiator 102, a device 101, a network 107, and a discovery server 110. In exemplary embodiments, initiator 102 can communicate with network 107 and discovery server 110 through an access network 112, which provides communication to an Internet Protocol (IP) network 113. Initiator 102 can also be associated with an initiator user 102u, which could be an end user, group of people, or business entity controlling or operating initiator 102. Initiator 102 could comprise a computing device that operates a radio and processor in order to transmit and receive wireless data. Additional details for components within an initiator 102 are provided below in Figure 1d. In some exemplary embodiments, initiator 102 could comprise a mobile phone or “smart phone” with an operating system such as Android ® from Google ® or IOS® from Apple ®. In other exemplary embodiments, an initiator 102 could comprise a portable device, such as laptop, tablet, wearable device such as a smart watch, a USB device including a “USB memory stick”, etc. In exemplary embodiments, the function of initiator 102 could be combined with network WiFi access point 105 (such as depicted and described in connection with Figure 8 below), and a separate physical unit for initiator 102 could be omitted for some embodiments of a system 100.

Initiator 102 could operate as an initiator proxy 102 and send and receive messages with device 101 according to a device provisioning protocol (DPP) 191 as specified in the “Device Provisioning Protocol Specification version 1.0” from the WiFi Alliance™ (DPPv1.0), which is hereby incorporated by reference in its entirety. Subsequent and related versions of a DPP 191 could be supported as well by device 101 and an initiator 102. A summary of the message flows between initiator 102 and device 101 according to the DPP 191 are depicted and described in connection with Figure 1b and Figure 1c below, as well as subsequent figures herein. The message flows for a DPP 191 between initiator 102 and device 101 could be via a WiFi data link 192, where the WiFi data link 192 can use a configuration 130. Initiator proxy 102 could operate a camera 102z and a WiFi radio as depicted, where the WiFi radio could record a medium access control (MAC) address MAC.I 102n. MAC.I 102n for initiator 102 could comprise a string of octets to identify initiator 102 with any of a WiFi access point 105, device 101, network 107, and discovery server 110. In order to acquire the data to function as an initiator according to a DPP 191 and/or DPPv1.0, initiator 102 in Figure 1a can communicate and transmit/receive data from network 107 via access network 112.

Additional details regarding the electrical components within an initiator 102 are depicted and described in connection with Figure 1d below.

Embodiments of the present invention contemplate both (i) some of the functionality of an initiator in the DPP specification DPPv1.0 operates within initiator proxy 102 depicted in Figure 1a, and (ii) other functionality of an initiator according to the DPP protocol in DPPv1.0 resides within network 107. Depictions and descriptions below in the present invention may utilize the term “initiator 102”, which can also comprise an “initiator proxy 102” depicted in Figure 1a. In other words, (x) from the perspective of device 101, initiator proxy 102 can operate as an initiator according to the DPP Specification DPPv1.0, but (y) in the present invention some of the functionality and data for an initiator as contemplated by the DPP Specification remains in the network 107. This configuration of operating an initiator proxy 102 in a system 100 (along with proper operation of a network 107) can address and solve the needs in the art described above. Initiator proxy 102 can communicate with network 107 via access network 112.

Access network 112 could be either a Local Area Network (LAN) or a Wide Area Network (WAN), or potentially a combination of both. Access network 112 could comprise a network supporting either IEEE 802.11 (WiFi) standards. Initiator 102 and access network 112 could also utilize a variety of WAN wireless technologies to communicate, including Low Power Wide Area (LPWA) technology, 3rd Generation Partnership Project (3GPP) technology such as, but not limited to, 3G, 4G Long-Term Evolution (LTE), or 4G LTE Advanced, NarrowBand – Internet of Things (NB-IoT), LTE Cat M, proposed 5G networks, and other examples exist as well. A wired initiator 102 can connect to the access network 112 via a wired connection such as, but not limited to, an Ethernet, a fiber optic, or a Universal Serial Bus (USB) connection (not shown). In some exemplary embodiments, access network 112 for initiator proxy 102 can be provided by access point 105, where initiator proxy 102 may also record a compatible set of network credentials 109.

IP network 113 could be a public or private network supporting Internet Engineering Task Force (IETF) standards such as, but not limited to, such as, RFC 786 (User Datagram Protocol), RFC 793 (Transmission Control Protocol), and related protocols including IPv6 or IPv4. A public IP network 113 could utilize globally routable IP addresses, and a private IP network 113 could utilize private IP addresses which could also be referred to as an Intranet. Other possibilities for IP Network 113 exist as well without departing from the scope of the invention. Although not depicted in Figure 1a, elements within a system 100 could each operate or be associated with a firewall, such that packets from IP network 113 are filtered or restricted in order to secure the communication between the nodes.

Device 101 can comprise a computing device with a WiFi radio for transmitting and receiving data such as via WiFi data link 192. Device 101 can include a tag 101c and a WiFi radio 101h. WiFi radio 101h can operate with a medium access control (MAC) address MAC.R 101d.

The tag 101c can record an identity for the device 101, which can be ID-token.device 206 as described in Figure 2a below. For the system depicted in Figure 1a, device 101 operating a responder 101x can have a roll of an enrollee, and initiator proxy 102 can operate with a roll of a configurator. Device 101 can operate a responder 101x and, after a configuration step 106, also operate a set of network WiFi credentials 109 that are compatible with an access point 105. Device 101 could be powered via any of (i) traditional “wall power” potentially with an AD/DC adapter, (ii) a battery which may be periodically recharged or replaced, (iii) power over a wired LAN connection such as “power over Ethernet”, and other possibilities exist as well. A device 101 can record a certificate cert.device 101m, which can record a device public key, which can be associated with a device private key also recorded in device 101, but not shown in Figure 1a.

The responder 101x in device 101 can record a responder bootstrap public key Br 101a, a responder bootstrap private key br 101b, an initiator bootstrap public key Bi 104a, a set of cryptographic parameters 199a for the PKI keys, and optionally (i) an initiator bootstrap public key sequence number 197, and (ii) a public key exchange protocol (PKEX) shared secret key 198. The responder 101x in device 101 can comprise a process operated by a processor and recorded in memory for device 101. Although not depicted in Figure 1a, a responder 101x can also include a random number generator, a PKI key pair generation algorithm, a secure hash algorithm, an elliptic curve Diffie Hellman key exchange algorithm, and a symmetric ciphering algorithm.

A network 107 can operate an access point 105, where the access point operates with a set of credentials 109. The credentials 109 can include an identifier for the network such as SSID.network-AP 109a, a pairwise master key (PMK) PMK.network-AP 109b or a preshared key (PSK), a pairwise master key identifier (PMKID) (not shown), and also a configuration Config.network-AP 109c. The access point can support wireless networking standards in the series of 802.11 standards. A network 107 can also operate a plurality of access points 105. In addition, access point 105 can comprise a plurality of different WiFi radios to collectively function as one access point 105, such as with “mesh” WiFi networks. Or, access point 105 can operate as a single WiFi router. The network 107 can include or be associated with a Device Provisioning Protocol (DPP) server 103, a device database 104, an authentication server 111, and a discovery server 110. Different servers within the network 107 can be connected using an Intranet, a virtual private network (VPN), Internet Protocol Security (IPSEC), or similar techniques to secure communication between the servers.

The network 107 can operate or communicate with the device database 104. The DPP server 103 in the network 107 can also communicate with the device database 104. The device database 104 can record information for a plurality of devices 101 before responders 101x for the device 101 conducts the device provisioning protocol 191 with the initiator 102. The device database 104 can record the device identity 206, a responder identity 101d, which could be in the form of a responder

medium access control (MAC) address MAC.R 101d, an initiator configuration 130, and a key table. The key table in a device database 104 can record for each device 101 a responder bootstrap public key Br 101a, an initiator bootstrap public key Bi 104a, and an initiator bootstrap private key bi 104b. Recorded data within a device database 104 can be obtained by a network from either (i) upon
5 manufacturing from a device manufacturer, or (ii) before distribution of a device to the location of the network's access point. Some data in a device database 104 could also be acquired by an initiator 102 prior to the start of a DPP 191.

The DPP server 103 for a network 107 can operate a server database 103a, where the server database 103a records information for a plurality of devices 101 connecting with the network 107.
10 The server database 103a can record a medium access control (MAC) address MAC.R 101d in the device 101 a set of cryptographic parameters 199a for each device, an initiator bootstrap private key bi 104b, the initiator bootstrap public key Bi 104a, and the responder bootstrap public key Br 101a. A server database can also record additional data for responders 101x in devices 101, as depicted and described in connection with Figure 3b below. Depending on an initiator mode 306a in Figure 3
15 below, not all PKI keys and secret shared keys may be recorded in the server database 103a. The DPP server 103 can also include a set cryptographic algorithms, where the cryptographic algorithms can include a random number generator, a PKI key pair generation algorithm, a secure hash algorithm, an elliptic curve Diffie Hellman key exchange algorithm, and a symmetric ciphering algorithm. The cryptographic algorithms within a DPP server 103 can be compatible with the
20 equivalent set of cryptographic algorithms in the responder 101x and also the initiator proxy 102.

Note that the security of communications between a device 101 and an access point 105 may generally depend on the security of a configuration step 106, which in turn may rely on a device provisioning protocol 191. Further, a manufactured device 101 may not have the credentials 109 or configuration in order to communicate with an access point 105 without successfully completing a
25 DPP 191 and a configuration step 106 in order for device 101 to secure receive and use network credentials 109. After a DPP 191 and a configuration step 106 described in the present invention and in multiple figures below, device 101 can communicate with access point 105 using credentials 109.

30 **Figure 1b**

Figure 1b is a graphical illustration of a device provisioning protocol for authentication and configuration of a responder, in accordance with conventional technology. Figure 1b depicts a summary of the WiFi Device Provisioning Protocol (DPP) specification, version 1.0 which was
35 published on April 9, 2018. The summary depicted in Figure 1b highlights recorded bootstrap PKI

keys, derived ephemeral PKI keys, and messages transmitted and received between an initiator 102* and a responder 101x. The use of keys and messages for a DPP 191 can accomplish both (i) authenticating the responder 101x and (ii) transfer a configuration object to responder 101x, where the configuration object could comprise a set of network access credentials 109 from Figure 1a.

Initiator 102* is depicted with a "*" and has several differences than an initiator proxy 102 depicted in Figure 1a and subsequent figures below, although some functionality is shared. Both initiator 102* and initiator proxy 102 (from Figure 1a and other Figures below) can send and receive the full and complete set of a DPP messages for a Device Provisioning Protocol 191.

In other words, initiator 102* and an initiator proxy 102 can send equivalent messages such that a responder 101x would process the messages in the same manner and would not normally be able to detect to discern whether a initiator proxy 102* or an initiator 102 was transmitting and receiving messages for the Device Provisioning Protocol 191. In the present invention, initiator proxy 102* can operate as an initiator according to the DPP Specification 1.0. Initiator proxy 102* or an initiator 102 can be (x) a process, program, firmware, or software application operating in a computing device with a WiFi compatible radio, in order to (y) communicate with a responder 101x via a WiFi data link 192 (in Figure 1a). Exemplary additional configurations for an initiator 102 and a responder 101x within computing devices is depicted and described in connection with Figures 7 and 8 below, and other possibilities exist as well without departing from the scope of the present invention.

Other differences between an initiator 102* and an initiator proxy 102 can exist in the present invention. For a system 100 with an initiator proxy 102 in Figure 1a and Figures below, the recordation of PKI keys, PKI key derivation, and cryptographic processes such as symmetric encryption and decryption can be distributed between an initiator proxy 102 and a DPP server 103, while an initiator 102* can internally record the keys necessary for symmetric key exchange, key derivation, and symmetric encryption and decryption. The distributed storage and operation with PKI keys for a system that includes an initiator proxy 102 is depicted and described in connection with Figure 3, Figure 5a, Figure 5b, and Figure 6a below. Additional details for internal components, memory structures, values, and operation of an initiator proxy 102 in order to provide equivalent functionality as initiator 102* for a responder 101x are described in Figures below.

Initiator 102* can record bootstrap keys, ephemeral keys, and derive symmetric encryption keys in order to authenticate the responder 101x and transfer the network access credentials 109 according to a Device Provisioning Protocol 191. For a system with responder authentication 141, an initiator proxy 102* can record a medium access control (MAC) address MAC.I 102n. MAC.I 102n can be associated with a WiFi radio within Initiator proxy 102* and comprise a string of octets in order to uniquely identify initiator 102* within a wireless network. With current, convention technology, MAC.I 102n can comprise a string of 6 octets, although other possibilities exist as well

for a MAC.I 102n. Initiator 102* can also operate a key pair generation function 102y in order to derive ephemeral PKI keys such as an initiator ephemeral public key Pi 102a and an initiator ephemeral private key pi 102b. As contemplated herein, an ephemeral PKI key can also be referred to as a “protocol” key. In other words, key Pi 102a can be referred to as initiator protocol public key
 5 Pi 102a or initiator ephemeral public key Pi 102a, which can both represent the same numeric value, number, or sequence of digits.

Although not depicted in Figure 1b, a key pair generation function 102y could use (i) a random number generator in order to derive the PKI private key and (ii) a selected set of cryptographic parameters 199a (depicted in Figure 2b below) in order to specify an elliptic curve
 10 cryptography (ECC) curve name, key length, key formatting, etc. As contemplated herein, the algorithms used with ECC PKI keys in the present invention can be compatible and substantially conform with ECC algorithms and keys as specified in (i) the IETF Request for Comments (RFC) 6090 titled “Fundamental Elliptic Curve Cryptography Algorithms”, and (ii) IETF RFC 5915 titled “Elliptic Curve Private Key Structure”, and also subsequent and related versions of these standards.
 15 With a random number generated by initiator 102* or an initiator proxy 102, a key pair generation function 102y, a derived ephemeral initiator private key pi 102b can be used to derive a corresponding ephemeral initiator public key Pi 102a. The two derived PKI keys can be used in conjunction for subsequent operations such as an Elliptic Curve Diffie Hellman key exchange. As contemplated in the present invention, the use of a capital letter as the first character for a PKI key
 20 can represent a public key, the use of a lower case letter as the first character for a PKI key can represent a private key, and the second letter for a PKI key can represent the entity the key is associated with or belongs to. Further, the letter “B” or “b” as the first character for a PKI key can represent a recorded bootstrap key, which could be static or relatively static over the lifetime of a responder or initiator, and the letter “P” or “p” the first character for a PKI key can represent an
 25 ephemeral key.

Responder 101x in Figure 1b, Figure 1c, and other Figures below can include functionality to operate as a responder according to the DPP Specification version 1.0. As depicted in Figure 1a, responder 101x could operate within a device 101, although other possibilities exist as well for the location or equipment operating a responder 101x without departing from the scope of the present
 30 invention. For a DPP 191, a responder 101x can record a responder bootstrap public key Br 101a, a responder bootstrap private key br 101b, a MAC address MAC.R 101d, and also operate a key pair generation algorithm 101y. Responder bootstrap public key Br 101a and responder bootstrap private key br 101b could comprise an ECC key pair selected and formatted according to a selected set of parameters 199a (in Figure 1a above).

35 Ephemeral bootstrap PKI keys for responder 101x could also be formatted and selected with a compatible set of cryptographic parameters 199a as Pr 101e and pr 101f. Bootstrap PKI keys for

responder 101x could be recorded in responder 101x or nonvolatile memory for a device 101 operating a responder 101x. The bootstrap PKI keys could be recorded in responder 101x during manufacturing or a configuration of responder 101x before delivery of a device 101 to the location of an access point 105 from Figure 1a. MAC address MAC.R 101d can comprise an identity of responder 101x or a device 101 on a WiFi network, and can be similar to MAC.I 102d for an initiator 102* as described above. Key pair generation algorithm 101y can be equivalent to key pair generation algorithm 102y described above for initiator proxy 102*. Key pair generation algorithm 101y can derive ephemeral PKI keys for responder 101x comprising public key Pr 101e and private key pr 101f, which could also be derived using a compatible set of parameters 199a for Br 101a, Br 101b, Pi 102a and pi 102b.

An initiator 102* and a responder 101x can conduct a Device Provisioning Protocol (DPP) 191. Again, as contemplated herein, an exemplary Device Provisioning Protocol 191 according to conventional technology could comprise the “Device Provisioning Protocol Specification” version 1.0 as published by the WiFi Alliance TM. In message 121, an initiator 102* can receive the bootstrap public key Br 101a for responder 101x. The “out of band” transfer of Br 101a in message 121 could comprise several different possible methods for initiator 102* to receive Br 101a, which are also described in the DPP specification 1.0, such as via reading a QR code, reading a near-field communications tag, using a Bluetooth wireless connection to read Br 101a from responder 101x, and other possibilities exist as well. A DPP 191 also contemplates that associated information for responder 101x could be transferred in message 121, such as a device identity for device 101, MAC.R 101d, in addition to Br 101a and possible parameters 199a.

Initiator 102* can receive Br 101a and derive PKI keys Pi 102a and pi 102b. Initiator 102* can conduct an Elliptic Curve Diffie Hellman key exchange algorithm 401a as depicted and described below in key exchange 314 step for Figure 4a in order to derive a shared secret key with responder 101x. Initiator 102* can derive a random number comprising an initiator nonce and also determine a set of capabilities for the initiator, such as a configurator or enrollee as specified in DPP specification version 1.0. Initiator 102* can encrypt the initiator nonce and the initiator capabilities using the derived shared secret key as depicted and described in connection with an encryption step 315 in Figure 4a below. Initiator 102* can send the derived public key Pi 102a and a ciphertext with the encrypted data in a message 122. Although not depicted in Figure 1b, initiator 102* can also send a hashed value of Br 101a in message 122.

As specified in DPP specification version 1.0, message 122 could also include a hashed value for initiator bootstrap public key Bi 104a, but for a responder authentication 141 the value of Bi 104a may not be used (since responder 101x may not record the corresponding initiator bootstrap public key Bi 104a for a responder authentication 141 as depicted in Figure 1b). The hash algorithm on a bootstrap public key could comprise the SHA-256 hash algorithm according to IETF RFC 5754

titled "Using SHA2 Algorithms with Cryptographic Message Syntax". Other hash algorithms could be used as well and specified in a selected set of cryptographic parameters 199 as depicted and described in connection with Figure 2b below, such as SHA-3, SHA-384, etc. Message 122 can comprise a Device Provisioning Protocol Authentication Request according to DPP specification version 1.0. Additional details for a message 122 are provided in section 6.2.2 of DPP specification version 1.0.

Responder 101x can receive message 122 and follow the steps for a responder according to the section 6.2.3 of DPP specification version 1.0, in order to generate a Device Provisioning Protocol Authentication Response, which could comprise message 123. Responder 101x can conduct an Elliptic Curve Diffie Hellman key exchange algorithm 401a as depicted and described below in key exchange 319 step for Figure 4a in order to mutually derive a first shared secret key with initiator 102*. Responder 101x could decrypt the initiator nonce and the initiator capabilities using the derived shared secret key as depicted and described in connection with an decryption step 320 in Figure 4a below. Responder 101x can use the key pair generation algorithm 101y in order to derive an ephemeral responder public key Pr 101e and a corresponding ephemeral private key pr 101f. Responder 101x can use the derived responder ephemeral private key pr 101f and the received initiator ephemeral public key Pi 102a in order to conduct a key exchange algorithm 314b as depicted below in Figure 4b and derive a second secret shared key. The first mutually derived symmetric key can comprise k1 and the second mutually derived symmetric key can comprise k2 according to the DPP specification version 1.0. Responder 101x can derive a random number comprising a responder nonce and also determine a set of capabilities for the responder, such as a configurator or enrollee as specified in DPP specification version 1.0.

Responder 101x could also generate a responder authentication value, which could comprise a hash value over the initiator nonce received in message 122 and the generated responder nonce. Responder 101x could use the first shared secret key and the second shared secret key in order to derive a third shared secret key. Responder 101x could encrypt the responder authentication value with the third shared secret key and also then include the encrypted responder authentication value with the responder nonce and responder capabilities in a ciphertext. The ciphertext could be encrypted with the second symmetric key, as depicted and described in connection with an encryption step 315b in Figure 4b below. Responder 101x could send the derived ephemeral responder public key Pr 101e and the ciphertext in a message 123.

Initiator 102* can receive message 123 and record the received ephemeral responder public key Pr 101e. Initiator 102* can follow the steps in section 6.2.4 of DPP specification version 1.0 in order to process message 123 and confirm the authentication response from responder 101x in message 123. Initiator 102* can mutually derive the second symmetric key using the received ephemeral responder public key Pr 101e and the derived initiator ephemeral private key pi 102b,

using a key exchange algorithm 319a as depicted in Figure 4b. Initiator 102* can decrypt the ciphertext in message 123 using the second mutually derived symmetric key and a decryption step 320b in Figure 4b. Initiator 102* can derive a third symmetric key using at least the first mutually derived symmetric key and the second mutually derived symmetric key. Initiator 102* can use the
 5 third symmetric key to decrypt the encrypted responder authentication value using a decryption step 328 in Figure 4c. Initiator 102* can calculate a value for the responder authentication, and if the calculated value for the responder authentication matches the received value for the responder authentication, then responder 101x can be considered authenticated.

Although not depicted in Figure 1b, after authenticating responder 101x, initiator 102* can
 10 send an authentication confirm message, which could comprise a DPP authentication confirm message according to section 6.2.4 of the DPP specification version 1.0. The authentication confirm message can also include an initiator authentication value, which could be encrypted with the third symmetric key. The responder 101x can receive the encrypted authentication confirm value and decrypt the value and also internally calculate the initiator authentication value. If the received
 15 initiator authentication value and the calculated authentication confirm values are equal, then the initiator is authenticated with the responder 101x. Note that for a responder authentication 141 in Figure 1b, the authentication of initiator 102* by responder 101x is weak and only comprises verifying that initiator 102* records the bootstrap public key Br 101a for responder 101x. Also, although not depicted in Figure 1b, responder 101x could have the capabilities of an enrollee and
 20 send a DPP configuration request message to initiator 102*, which could take the role of a configurator. The DPP configuration request message can be according to section 6.3.2 of the DPP specification version 1.0, and can include an enrollee nonce, which could be a random number generated by the enrollee.

As depicted in Figure 1b, initiator 102* can then send a configuration object in a ciphertext
 25 in message 124. Message 124 could comprise a DPP configuration response message according to section 6.3.3 of the DPP specification version 1.0. Initiator 102* can record the configuration object before sending message 124. In the present invention, the configuration object comprises the network credentials 109 depicted in Figure 1a above. In other words, a configuration object can be configuration information for a device 101 with responder 101x to use in order to connect with other
 30 devices via a wireless network. Configuration object could contain a list of network identifiers, device identifiers, RF band and channel information, configuration parameters, pre-share keys, PKI keys, names, passwords, group temporal keys, a shared secret 198 for a PKEX key exchange, etc. An exemplary list of possible values for a configuration object is depicted in table 14 of DPP specification version 1.0. In message 124, the network credentials 109 as a configuration object can
 35 be sent in a ciphertext using the third mutually derived symmetric key. As depicted in Figure 1b, the ciphertext can include both the enrollee nonce and the network credentials. Responder 101x can

receive message 124, decrypt the ciphertext using the third mutually derived symmetric key, and record the network credentials 109. Device 101 can then apply the network credentials 109 and communicate with access point 105.

Figure 1c

Figure 1c is a graphical illustration of a device provisioning protocol for (i) authentication and configuration of a responder and (ii) authentication of an initiator, in accordance with conventional technology. Figure 1c depicts a summary of the WiFi Device Provisioning Protocol (DPP) specification, version 1.0 which was published on April 9, 2018, supporting a mutual authentication 142 by both initiator 102* and responder 101x. The summary depicted in Figure 1c highlights recorded bootstrap PKI keys, derived ephemeral PKI keys, and messages transmitted and received between an initiator 102* and a responder 101x. Many of (i) the PKI keys for initiator 102* and responder 101x, and (ii) the messages transmitted between the nodes are equivalent to those depicted and described in connection with Figure 1b. This description of Figure 1c herein focuses upon the differences from Figure 1b in order for initiator 102* and responder 101x to mutually authenticate.

In order to support a mutual authentication between initiator 102* and responder 101x, initiator 102* can record an initiator bootstrap public key Bi 104a and an initiator bootstrap private key bi 104b. Although bootstrap public key Bi 104a is depicted within initiator 102*, another entity besides initiator 102* could record the public key and make it available to other nodes such as a responder 101x. As contemplated by the DPP Specification, the initiator bootstrap keys can be recorded with initiator 102* before conducting a DPP 191 authentication and configuration with responder 101x. For example, the bootstrap PKI keys for the initiator could be recorded during manufacturing of the initiator, or loaded by an initiator user 102u before conducting the depicted communications with responder 101x. Bootstrap public keys for the initiator can support a selected set of cryptographic parameters 199a, where the selected set of cryptographic parameters 199a can match the parameters used by keys in responder 101x. The DPP specification version 1.0 identifies the minimal set of cryptographic parameters in order to ensure compatibility, which can comprise the “set A” for a set of cryptographic parameters 199 depicted in Figure 2b below.

In order to conduct a mutual authentication 142 in addition to the responder authentication 141 depicted in Figure 1c, a responder 101x should have access to the initiator bootstrap public key Bi 104a and may also record the initiator bootstrap public key Bi 104a. In message 126, a responder 101x can receive the initiator bootstrap public key Bi 104a for initiator 102*. The “out of band” transfer of Bi 104a in message 126 could comprise several different possible methods for

responder 101x to receive Bi 104a, which are also described in the DPP specification 1.0, such as via reading a QR code, reading a near-field communications tag, using a Bluetooth wireless connection to read Bi 104a from initiator 102*, and other possibilities exist as well, including using WiFi direct to first establish a “peer-to-peer” WiFi connection. A DPP 191 also contemplates that associated information for responder initiator 102* could be transferred in message 126, such as a device identity for initiator 102*, MAC.I 102n, in addition to Bi 104a. As depicted in Figure 1c, the responder 101x can also record the initiator bootstrap public key Bi 104a, which can be used to authenticate the initiator 102* as described below.

For a mutual authentication 142 between device 102* and responder 101x, the two nodes can exchange the same or equivalent messages as message 122 and message 123, which were also described above in connection with Figure 1b. For mutual authentication, the third shared encryption key ke in DPP specification version 1.0 can be derived by a responder 101x using a key exchange step 321 and the received initiator bootstrap public key Bi 104a as depicted and described in connection with Figure 4c below. The use of public key Bi 104a for the key exchange step 321 is described in section 6.2.3 of the DPP Specification version 1.0. The difference between the third mutually derived symmetric key ke for a mutual authentication 142 and a responder authentication 141 is the inclusion of initiator bootstrap public key Bi 104a in the key exchange algorithm 401c and key derivation function 402c in Figure 4c. The first mutually derived symmetric key can comprise k1 and the second mutually derived symmetric key can comprise k2. The third mutually derived symmetric key ke is used by the responder 101x to encrypt the responder authentication value as depicted in an encryption step 322 in Figure 4c. The responder authentication value can include a random number “initiator nonce” or I-nonce from the initiator 102* in message 122. The responder authentication value can comprise the value “R-auth” in step 3 of section 6.2.3 of DPP specification version 1.0.

Initiator 102* can also mutually derive the third symmetric encryption key ke using the initiator bootstrap private key bi 104b and key exchange step 327 depicted and described below in connection with Figure 4c. Initiator 102* can subsequently decrypt the responder authentication value using the mutually derived third symmetric encryption key ke and a decryption step 328 as depicted and described in connection with Figure 4c below. Initiator 102* can calculate the same value for the responder authentication value as described in section 6.2.4 of DPP specification version 1.0. If the received, decrypted responder authentication value matches the calculated responder authentication value, then the responder 101x is authenticated. The initiator 102* can calculate an initiator authentication value using the received responder nonce in message 123 and encrypt the initiator authentication value using the mutually derived third symmetric key ke (from a key exchange step 327) and send the initiator authentication value in a message (not shown in Figure 1c). The responder 101x can (i) decrypt the encrypted initiator authentication value using the third

mutually derived symmetric key ke (from a key exchange step 321) and (ii) also calculate the initiator authentication value. If the calculated initiator authentication value by the responder 101x matches the received, decrypted initiator authentication value then the initiator 102* is authenticated with the responder 101x for a mutual authentication. The responder 101x can then receive a configuration object in a message 124. Message 124 can include network credentials 109 encrypted with the third mutually derived symmetric key ke . The responder 101x can pass the decrypted network credentials 109 to device 101, which can then apply the network credentials 109 in order to communicate with network access point 105.

Figure 1d

Figure 1d is a graphical illustration of hardware, firmware, and software components for an initiator, including an initiator configuration step, in accordance with exemplary embodiments. Figure 1d is illustrated to include several components that can be common within an initiator 102. Initiator 102 may consist of multiple electrical components in order to both (i) communicate with a network 107 and (ii) transmit and receive messages with a responder 101x according to a device provisioning protocol 191. Initiator 102 can undergo an initiator configuration step 127 in order to configure the components in an initiator 102 to function as the enhanced initiator or initiator proxy as contemplated herein, such that an initiator bootstrap private key bi 104b can remain recorded with a network 107 as depicted in Figure 1a above.

In exemplary embodiments and as depicted in Figure 1d, initiator 102 can include an initiator identity 102i, a processor 102c (depicted as “CPU 102c”), random access memory (RAM) 102d, an operating system (OS) 102e, storage memory 102f (depicted as “nonvolatile memory 102f”), a WiFi radio 102w, a system bus 102j, and a wireless wide area network (WAN) radio 102h. Figure 1d depicts initiator 102 as both a default initiator 102 and a configured initiator 102'. The default initiator 102 can comprise the state of initiator 102 before a configuration step 127 and initiator 102' can comprise the state of initiator 102' after the configuration step 127. As depicted, both a default initiator 102 and a configured initiator 102' can contain the same hardware components such as CPU 102c, RAM 102d, etc., where a difference between initiator 102 and 102' can be in both (i) data recorded in storage memory 102f and (ii) the configuration of WiFi radio 102w.

Initiator identity 102i could comprise a preferably unique alpha-numeric or hexadecimal identifier for initiator 102, such as an Ethernet MAC address, an International Mobile Equipment Identity (IMEI), an owner interface identifier in an IPv6 network, a serial number, or other sequence of digits to uniquely identify each of the many different possible units for initiator 102. A system 100 depicted in Figure 1a could operate with a plurality of different initiators 102. Initiator identity

102i can preferably be recorded in a non-volatile memory or written directly to hardware in initiator 102 by (i) an initiator manufacturer upon device manufacturing, or (ii) a pre-configuration step for initiator 102. The CPU 102c can comprise a general purpose processor appropriate for typically low power consumption requirements for a initiator 102, and may also function as a microcontroller.

5 CPU 102c can comprise a processor for initiator 102 such as an ARM® based processor or an Intel® based processor such as belonging to the Atom or MIPS family of processors, and other possibilities exist as well. CPU 102c can utilize bus 102j to fetch instructions from RAM 102d and operate on the instruction. CPU 102c can include components such as registers, accumulators, and logic elements to add, subtract, multiply, and divide numerical values and record the results in RAM 102d

10 or storage memory 102f, and also write the values to an external interface such as WiFi radio 102w.

RAM 102d may comprise a random access memory for initiator 102. RAM 102d can be a volatile memory providing rapid read/write memory access to CPU 102c. RAM 102d could be located on a separate integrated circuit in initiator 102 or located within CPU 102c. The RAM 102d can include data recorded in initiator 102' for the operation when collecting and communicating

15 DPP messages 122 through 124 with responder 101x depicted in Figure 1b and Figure 1c above. The system bus 102j may be any of several types of bus structures including a memory bus or memory controller, a peripheral bus, and a local bus using any of a variety of bus architectures including a data bus. System bus 102j connects components within initiator 102 as illustrated in Figure 1d, such as transferring electrical signals between the components illustrated. Initiator 102

20 can include multiple different versions of bus 102j to connect different components, including a first memory bus 102j between CPU 102c and RAM 102d, and a second system bus 102j between CPU 102c and WiFi radio 102w, which could be an I2C bus, an SPI bus, PCIe bus, or similar data busses.

The operating system (OS) 102e can include Internet protocol stacks such as a User Datagram Protocol (UDP) stack, Transmission Control Protocol (TCP) stack, a domain name system

25 (DNS) stack, a TLS stack, a datagram transport layer security (DTLS) stack, etc. The operating system 101e may include timers and schedulers for managing the access of software to hardware resources within initiator 102, including CPU 102c and WiFi radio 102w. The operating system shown of 102e can be appropriate for a low-power device with more limited memory and CPU resources (compared to a server such as DPP server 103, discovery server 110, etc.). Example

30 operating systems 102e for a initiator 102 includes Linux, Android® from Google®, IOS from Apple®, Windows® 10 IoT Core, or Open AT® from Sierra Wireless®. Additional example operating systems 102e for an initiator 102 include eCos, uC/OS, LiteOs, Contiki, OpenWRT, Raspbian, and other possibilities exist as well without departing from the scope of the present invention. Although depicted as a separate element within initiator 102 in Figure 1d, OS 102e may

35 reside in RAM 102d and/or storage memory 102f during operation of initiator 102.

Storage memory 102f (or “nonvolatile memory 102f”) within initiator 102 can comprise a nonvolatile memory for storage of data when initiator 102 is powered off. Memory 102f may be a NAND flash memory or a NOR flash memory, and other possibilities exist as well without departing from the scope of the present invention. Memory 102f can record firmware for initiator 102.

5 Memory 102f can record long-term and non-volatile storage of data or files for initiator 102. In an exemplary embodiment, OS 102e is recorded in memory 102f when initiator 102 is powered off, and portions of memory 102f are moved by CPU 102c into RAM 102d when initiator 102 powers on. Memory 102f (i) can be integrated with CPU 102c into a single integrated circuit (potentially as a “system on a chip”), or (ii) operate as a separate integrated circuit or a removable card or device,

10 such as a removable SD card. Memory 101f may also be referred to as “device storage” and can include exemplary file systems of FAT16, FAT 32, NTFS, ext3, ext4, UDF, or similar file systems.

As depicted in Figure 1d, nonvolatile memory 101f for a default initiator 102 may also contain a secret key SK0.initiator 102s, a certificate cert0.initiator 102m, and a certificate cert.CA.root 102r. A secret key SK0.initiator 102s in a nonvolatile memory 102f could comprise the

15 private key for a PKI key pair, where the corresponding public key could be recorded in a certificate cert0.initiator 102m. A secret key SK0.initiator 102s and the corresponding public key in a certificate 102m could comprise values to use with asymmetric public key infrastructure (PKI) cryptography, and could utilize exemplary algorithms based on either Rivest Shamir Adleman (RSA) or elliptic curve cryptography (ECC). For use of ECC algorithms, parameters within

20 certificate 102m (and other certificates herein) can specify elliptic curve names (e.g. NIST P-256, sect283k1, sect283r1, sect409k1, sect409r1, etc.). The same ECC parameters can be used with bootstrap PKI keys in the form of parameters 199a. Further, elliptic curves that do not depend on parameters specified by NIST could be utilized as well, such as Curve22519 or FourQ. For use of

25 RSA algorithms, parameters within certificate 102m can specify a modulus and other associated values for using an RSA PKI key pair. For either asymmetric algorithms RSA or ECC for a PKI key pair herein, parameters in a certificate such as cert0.initiator 102m can specify key lengths, a duration for key validity, uses of the PKI key pair such as for key derivation or signatures, encoding rules, message digest algorithms, etc. Parameters in certificate 102m (and other certificates herein) may also include identifying information associated with the PKI key pair such as a sequence

30 number, a serial number, a certificate revocation list or URL, a domain name of the server associated with the PKI key pair.

Certificate cert.CA.root 102r can comprise a root certificate for initiator 102 to authenticate a server certificate received from DPP server 103, authentication server 111, or discovery server 110, where these servers are depicted and described in connection with Figure 1a, Figure 2a, and Figure

35 3, and other Figures below. Certificate cert.CA.root 102r can also be used to authenticate other servers or nodes or access points in a network 107 as well. After authenticating a server certificate

received with cert.CA.root 102r, initiator 102 can conduct additional authentication steps contemplated by secure protocol standards such as Transport Layer Security (TLS), Datagram Transport Layer Security (DTLS), IPSec, etc. Root certificate cert.CA root 102r can comprise the list or a subset of the list of included root certificates from the Mozilla Foundation with Mozilla projects, where the aggregate list of community approved root certificates and associated parameters is in the widely distributed file certdata.txt from the Mozilla web site.

Initiator 102 and 102' can include a WiFi radio 102w to communicate wirelessly with networks such as AP 105, access network 112, and device 101 depicted and described in Figure 1a above, including via WiFi data link 192. In exemplary embodiments, access network 112 can comprise all available wireless WAN and LAN networks in the range of initiator 102, such as the exemplary networks listed in a networks available list 304c depicted in Figure 3a below. WiFi access point 105 can comprise a member of the set of all access networks 112 for initiator 102. WiFi radio 102w could connect with an antenna in order to transmit and receive radio frequency signals.

As depicted in Figure 1d, a WiFi radio 102w in a configured initiator 102' can record or operate with an initiator configuration 130. The values for initiator configuration 130 could also specify a version of the 802.11 standards to utilize, such as, but not limited to, 802.11n, 802.11ac, 802.11ah, 802.11ax, or related and subsequent versions of these standards. The initiator configuration 130 can be compatible with the configuration of a target WiFi device 101 with responder 101x. The initiator configuration 130 could be obtained during an authentication step for initiator 102, such as depicted below in Figure 2a, although initiator configuration 130 could be obtained at other times as well, such as possibly directly from reading a tag 101c on device 101 or from a DPP server 103.

The initiator configuration 130 can include a device identity ID-token.device 206 (or alternatively ID.device 101i), a MAC address for responder 101x comprising MAC.R 101d, a device operating class 130a which can be selected from the plurality of potential global WiFi operating classes in table E-4 of IEEE standard specification 802.11ac, which is incorporated by reference herein. Recording a device identity 206 and MAC.R 101d with initiator configuration 130 can be used by an initiator 102' to select which set of initiator configuration 130 is associated with which devices 101. In other words, the present invention contemplates that an initiator 102' communicates with multiple responders 101x either concurrently, or over time. MAC.R 101d can be required for initiator 102' to send message DDP Authentication Request 122 as a unicast frame to responder 101x, such that a device 101 could receive message 122 by listening in a unicast mode for messages that include the device's MAC.R 101d. The device operating class 130 could specify if device 101 operates at 2.4 GHz, 5 GHz, etc.

The initiator configuration 130 can also include Device DPP Channel Authentication List 130b, which can comprise a list of channels for initiator 102' to send DPP authentication request 122

messages in order to establish initial connectivity with device 101 and responder 101x. In exemplary embodiments, initiator configuration 130 for initiator 102' provides data for an initiator 102' to send data such as message 122 to responder 101x, where responder 101x may operate with different possible values for operating classes, listening channels, or MAC addresses MAC.R 101d used by a responder 101x in device 101. In other words, without initiator configuration 130, an initiator 102' may not normally be able to (i) select the proper WiFi operating channel, and (ii) send unicast data to responder 101x. For embodiments where a Device DPP Channel Authentication List 130b is unknown or not available, then initiator 102' can periodically send DPP 122 messages over time on all available channels for a radio 102w.

WiFi radio 102w can include standard radio components such as RF filters, RF amplifiers, a clock, and phased loop logic (PLL). WiFi radio 102w can support protocols and specifications according to the IEEE 802.11 family of standards. For example, if WiFi radio 102w operates according to 802.11n standards, WiFi radio 102w could operate at a radio frequency of 2.4 or 5 Ghz. In other embodiments where WiFi radio 102w supports 802.11ac standards, then WiFi radio 102w could also support radio frequencies of 0.054 through 0.79 Ghz in order to operate in "white space" spectrum. Other possibilities exist as well for WiFi radio to support wireless LAN standards or 802.11 standards and radio frequencies, without departing from the scope of the present invention. WiFi radio 102w can access a nonvolatile memory 102f in order to record the depicted configuration values of user configuration 131 where the nonvolatile memory could be within WiFi radio 102w, or WiFi radio 102w could access memory 102f via bus 102j in order to record and read the values. User configuration 131 can include credentials for an initiator 102 with initiator user 102u, such as a list of SSIDs, identities, and passwords for operation of the initiator 102 with other networks besides network AP 105. For example, user configuration 131 may include a user operating class 131a, which could correspond to the WiFi global operating class, bands, and channel numbers for initiator 102 to communicate with other WiFi nodes besides device 101, such as a different access point for user 102u than access point 105.

As depicted in Figure 1d, a WiFi configured initiator 102' can include configuration data resulting from a configuration step 127, in order for initiator 102' to conduct a device provisioning protocol 191 with a device 101. A nonvolatile memory 102f for a WiFi configured initiator 102' can include data recorded for a default initiator 102, with the addition of recording a DPP application 102g and configuration parameters 151, and optionally shared secret key 198 for a PKEX key exchange, if bootstrap public keys between configured initiator 102' and device 101 are shared "in band". The contents and source of DPP application 102g and configuration parameters 151 for initiator 102 will be further described in connection with Figure 2a below, where the steps and message flows in Figure 2a below also conclude with initiator 102 completing a configuration step 127 in order to transition to a configured initiator 102'.

Initiator 102 may also optionally include user interface 102k which may include one or more devices for receiving inputs and/or one or more devices for conveying outputs. User interfaces are known in the art and may be simple for many devices such as a few LED lights or and LCD display, and thus user interfaces are not described in detail here. User interface 102k could comprise a touch screen if initiator 102 has sophisticated interaction with user 102u. Initiator 102 can optionally omit a user interface 102k, if no user input or display is required for conducting a device provisioning protocol 191 between initiator 102' and device 101 with responder 101x. Although not depicted in Figure 1d, initiator 102 can include other components to support operation, such as a clock, power source or connection, antennas, etc. Other possibilities exist as well without departing from the scope of the present invention.

In exemplary embodiments, initiator 102 can also include a wide area network radio 102h. WAN radio 102h could support 4G LTE, 5G, and subsequent or related standards for use of devices with licensed radio spectrum operating potentially as public land mobile networks (PLMN). WAN radio 102h could also support wide area narrow-band technologies such as NB-IoT or LPWAN, and other examples exist as well. WAN radio 101h can provide connectivity through an access network 112 in order for initiator 102 to communicate with network 107, DPP server 103, and discovery server 110 as depicted and described in connection with Figure 1a. In other words, in the present invention, WAN radio 102h can provide a radio connection to an access network 112, which provides connectivity to an IP network 113 in order for initiator 102 to communicate with remote servers in order to conduct a device provisioning protocol as contemplated herein. WAN radio 102h can include a MAC address MAC.N 102q, although the use of a MAC address for WAN radio 102h is not required. Further, WAN radio 102h can be omitted or not utilized by an initiator 102', in embodiments where initiator 102' can communicate with network 107 via network AP 105. For these embodiments, WAN radio 102h or WiFi radio 102w can also record or utilize a set of network credentials 109 in order to obtain access to network 107 while conducting a device provisioning protocol. Other possibilities exist as well for a configured initiator 102' to have access to network 107 while conducting a device provisioning protocol 191 without departing from the scope of the present invention.

Although not depicted in Figure 1d, the various servers shown above in Figure 1a such as DPP server 103, authentication server 111, and discovery server 110 and other servers as well can include equivalent internal electrical components depicted for an initiator 102 in order to operate as servers. The servers in Figure 1a could include a processor similar to CPU 102c, with primary differences for the processor in a server being increased speed, increased memory cache, an increased number and size of registers, with the use of a 64 bits for datapath widths, integer sizes, and memory address widths, etc., In contrast, an exemplary 32 bit datapaths could be used for CPU 102c in initiator 102 (although CPU 102c could also use 64 bit wide datapath widths if initiator 102

comprises a mobile phone such as a smart phone). For embodiments where initiator 102 comprises a home WiFi router, then a CPU 102c in initiator 102 could comprise an exemplary 32 bit processor, although other possibilities exist as well. Similarly, RAM in a server could be a RAM similar to RAM 102d in initiator 102, except the RAM in a server could have more memory cells such as supporting exemplary values greater than an exemplary 8 gigabytes, while RAM 102d in initiator 102 could support fewer memory cells such as less than an exemplary 4 gigabytes. Non-volatile memory for storage in a server in Figure 1a could comprise disks, “solid state drives” (SSDs) or “storage area networks” (SAN) for a server. Instead of a radio 102w, in exemplary embodiments, a server in Figure 1a could use a physical, wired LAN interface such as a high-speed Ethernet or fiber optic connection.

Figure 2a

Figure 2a is a simplified message flow diagram illustrating an exemplary system with exemplary data sent and received by an initiator, in accordance with exemplary embodiments. System 200 can include an initiator 102, device 101, a discovery server 110 and authentication server 111. Initiator 102 can communicate with discovery server 110 and authentication server 111 via IP network 113, where access to IP network 113 can be provided via access network 112 for initiator 102 as depicted and described in Figure 1a and Figure 1d. Initiator 102 can comprise a smart phone such as, but not limited to, a phone based on an Android operating system from Google ® or IOS from Apple ®, and other possibilities exist as well. In addition, a mobile computing device with both a wireless WAN and wireless LAN connectivity could be used for an initiator 102 in system 200, such as a tablet, laptop computer, e-reader, etc. The present invention also contemplates that a fixed station initiator 102 could be utilized instead of a mobile unit in order to conduct the device provisioning protocol 191 and configuration step 106 with device 101. In exemplary embodiments, a fixed station initiator 102 could comprise a WiFi access point operated by a device owner, initiator user 102u, or network 107 in order to configure a plurality of device 101 in nearby physical proximity in order for the plurality of devices 101 to receive a plurality of sets of network access credentials 109.

Initiator 102 can include a battery, radio, and touch screen, as well as a camera 102z. Initiator 102 can connect with access network 112 in order to communicate with discovery server 110 and authentication server 111, as depicted in Figure 1a above. The access network 112 used in a system 200 can be a different access network than access point 105 used by device 101 in Figure 1a, although access network 112 for initiator 102 could also be through access point 105. For example, if device 101 is an access point to be configured, then initiator 102 could use WAN radio 102h in order to connect with network 107 via access network 112. If (i) an access point 105 is already

operating with network access credentials 109, and (ii) a device 101 is to be configured to connect with access point 105, then initiator 102 could also use access point 105 using a set of compatible network access credentials 109. In exemplary embodiments, access network 112 can comprise a wireless WAN such as 4G LTE, 5G, etc, which can comprise a primary access network for initiator 102 and backup network for device 101 (where the primary network for device 101 can comprise a wireless LAN through network access point 105).

Device 101 in system 200 can comprise a device 101 depicted in Figure 1a and device 101 can also include a tag 101c. Tag 101c can comprise a recorded image on device 101 or packaging of device 101 and could be a bar code, QR code, or a serial number for device 101. Tag 101c can provide a unique identifier for device 101. In exemplary embodiments, if device 101 is a medical device, then tag 101c can include a Unique Device Identification (UDI). Although a device 101 could include a plurality of different tags 101c or markings for different purposes (such as shipping, inventory, a distributor code, etc.), for the purposes of conducting (i) a device provisioning protocol 191 and (ii) a read step 203 below by an initiator 102, the tag 101c as contemplated herein for device 101 could include distinct markings for an initiator user 102u to identify as the proper code to read for conducting a configuration step 127 of initiator 102. The distinct markings could comprise a pictogram or icon for a mobile phone or a wireless network next to the tag 101c, an outline around the tag 101c in a bright paint or distinguished color, and other possibilities exist as well for a device 101 to distinguish a tag 101c for a initiator user 102u from other markings, codes, serial numbers, etc. that may exist on a device 101.

In another embodiment for Figure 2a, tag 101c could comprise a “near field communications” NFC tag such as a tag compatible with the NFC Forum standards for type 1 through type 5 tags (and subsequent or related standards). The NFC technology could also be NFC-A, NFC-B, or NFC-V, or subsequent standards. For these exemplary embodiments where tag 101c comprises an NFC tag, then initiator 102 could use an NFC reader instead of camera 102z, and the NFC reader could operate a NFC radio. Initiator 102 could utilize an NFC chip and antenna, where the NFC chip operates in read mode with tag 101c for device 101 as an NFC tag. In exemplary embodiments, discovery server 110 can include or be associated with a discover server database 110a. Database 110a could record information about a plurality of devices 101, such as ID.device 101i, and ID-token.device 206, and a dataset config-provisioning.ID.device 212, which will be described in further detail below for Figure 2a.

At step 201 in Figure 2a, device 101 can be in a powered off state. For a step 201 device 101 could be shipped from a manufacturer to a distributor and then to a device user or an initiator user 102u. A device 101 could also be taken to the physical proximity (e.g. WiFi RF range) of network access point 105 in a step 201. In some embodiments, device 101 could be powered on at step 201, although power for device 101 is not required for embodiments with the steps and message flows

outlined in Figure 2a. At step 202, initiator 102 could be powered on and placed in proximity of device 101. Initiator 102 could be operated by an initiator user 102u at step 202. At step 203, initiator user 102u could use initiator 102 to conduct a “read” step 203, where initiator 102 can (i) take a picture of a tag 101c for device 101, or (ii) read a bar code, QR code, printed serial number, or similar code or markings on device 101. As mentioned above, tag 101c could comprise a bar code, QR code, or other encoding of data for device 101, including a serial number. For the alternative embodiment where (i) initiator 102 uses an NFC reader and (i) tag 101c comprises an NFC tag, the initiator 102 could conduct an NFC read of tag 101c for step 203 and step 204.

At step 204, initiator 102 can receive data from tag 101c, such as a response, where the response can consist of a uniform resource locator (URL) for discovery server 110, URL-DS 205, an ID-token.device 206, and a tag value 101w. The response at step 204 could be recorded in the tag 101c. URL-DS 205 could include a domain name for discovery server 110 and parameters such as the use of HTTP or HTTPS and a destination TCP or UDP port number for sending packets to discovery server 110. ID-token.device 206 can be an identifier for device 101 and/or responder 101x and could be a token for ID.device 101i, or a value or unique pseudo-random number associated with device 101 and/or ID.device 101i. ID-token.device 206 could be a value that obfuscates ID.device 101i but can preferably uniquely map to ID.device 101i in a system 100 or 200 with a plurality of devices 101. Through the use of ID-token.device 206 that is readable by initiator 102, then a normal operating value for ID.device 101i with a network 107 can remain confidential at step 204. Or, in some exemplary embodiments an ID.device 101i can be the same as ID-token.device 206.

Tag value 101w in response 204 can include data associated with tag 101c and device 101, such as versions of software supported, product validity or expiration dates, a responder bootstrap public key Br 101a, cryptographic parameters 199 supported (e.g. RSA or ECC), a curve name utilized for an ECC cryptographic algorithm, a name or URL for Certificate Authority 107 used by manufactured device 101, a certificate for device cert0.device 101m (in Figure 1d), or a subset of the values in config-provisioning.ID.device 212. In an exemplary embodiment, tag value 101w could comprise some values of config-provisioning.ID.device 212 upon manufacturing of device 101, which could be subsequently updated at a later date in discovery server DB 110a before installation or operation of device 101. For exemplary embodiments where tag 101c comprises a bar code or QR code or similar markings, then tag value 101w can be limited to fewer bytes such as listing URL-DS 205 and ID-token.device 206. For exemplary embodiments where tag 101c comprises an NFC tag capable of storing an exemplary few thousand bytes, then tag value 101w could include more values and data listed in the first sentence of this paragraph.

In other exemplary embodiments where a lower level of security can be required in order to have simplified functionality, tag value 101w can comprise MAC.R 101d, responder bootstrap

public key Br 101a, a parameter set 199a, configuration 130, and/or a version number 212a. For these simplified embodiments, initiator 102 can (i) read tag value 101w, load a DPP application 102g compatible with version number 212a and parameter set 199a, (iii) optionally send Br 101a to a DSS server 103, and (iii) load configuration 130 for WiFi radio 102w, and thereby skip the remaining steps and message flows depicted in Figure 2a. In other exemplary embodiments where a higher level of security is desired and a system 100 or 200 could comprise millions of devices, or high-value devices, and thousands or more initiators 102, then the full set of steps and message flows depicted in Figure 2a may be preferred.

At step 207, initiator 102 could process the data received in response 204. For embodiments where tag 101c is an image such as a bar code or QR code, at step 207 initiator 102 could process the image and extract the data or values for ID-token.device 206, URL-DS 205, and tag value 101w. Continuing at step 207, initiator 102 could connect with an access network 112 in order to call the URL in URL-DS 205, which could be an HTTPS/TLS session in exemplary embodiments. In exemplary embodiments, URL-DS 205 is combined with ID-token.device 206, such that the HTTPS session is a unique request to discovery server 110 for each different device 101. In exemplary embodiments, the value ID-token.device 206 can have enough information entropy or pseudo randomness that it cannot be easily guessed, such as (i) appearing to be an exemplary 32 byte random number, and (ii) not related to a value ID-token.device 206 for another device 101 in any externally observable manner such as in sequence, or sharing similar common factors, etc. By combining a URL-DS 205 with a unique and a sufficiently randomized ID-token.device 206, only an initiator 102 physically in the presence of device 101 (or by reading tag 101c) could feasibly query discovery server 110 with a fully and properly formed query/URL to fetch the subsequent data from discovery server 110. At step 207, initiator 102 could perform additional checks and steps in preparation for communication with device 101, such as verifying data in tag value 101w is compatible with software or settings in initiator 102, including DPP application 102g (if DPP application 102g is already installed in initiator 102).

In message flow 208, initiator 102 can establish a secure HTTPS session with discovery server 110 using the URL-DS 205, and the secure session could comprise a transport layer security (TLS) session such as TLS version 1.2, TLS version 1.3, or similar and subsequent standards. As part of the setup of message flow 208, initiator 102 can receive a certificate cert.DS 110c from discovery server 110. Cert.DS 110c could comprise an X.509 certificate and include a signed public key for discovery server 110. In exemplary embodiments, initiator 102 verifies the certificate cert.DS 110c and parent certificates through root certificate cert.CA.root 102r. The verification of signatures using public keys in certificates can use the signature verification steps as specified in the US National Institute of Standards and Technology (NIST) FIPS 186-4 titled "Digital Signature Standard" and subsequent or related versions. Note that a secure session 208 is not required for some

exemplary embodiments, and confidentiality and/or data integrity could be acquired via other means besides HTTPS and/or TLS or DTLS, including embodiments where data flow from discovery server 110 to initiator 102 be signed. Other possibilities exist as well for a message flow 208 to be adequately secured.

5 In exemplary embodiments initiator 102 can also authenticate with discovery server 110, such as an initiator user 102u associated with initiator 102 entering user credentials in a web site for discovery server 110. Or initiator 102 could send discovery server 110 a certificate for initiator 102 cert.initiator 102m, which the discovery server 110 could verify using a signature verification step. As contemplated herein, a message flow 208 and all subsequent message flows depicted and
10 described can comprise a series of parts or sub-messages, where the collection of sub-messages or parts can comprise the message. In other words, a message depicted with multiple elements for sets of data within the message does not require that all data be transferred together or in a single flow of TCP or UDP data. Transferring the message could comprise sending exemplary data within the message in parts, where upon conclusion of the message, (i) exemplary data depicted and described
15 could be transferred in parts or portions and (ii) the collection of sending different parts or portions for the message could comprise sending the message.

In message 209 and after the secure HTTPS/TLS session setup in message flow 208, initiator 102 can send discovery server 110 ID-token.device 206 and data from tag value 101w. Or, the value for ID-token.device 206 could be included in the URL called by initiator 102 when accessing
20 discovery server 110, and in this manner a value ID-token.device 206 does not need to be sent again in a separate message 209. Initiator 102 could alternatively send discovery server 110 a subset of the data for ID-token.device 206 and/or tag value 101w.

At step 210, discovery server 110 can use data in ID-token.device 206 to query discovery server database 110a for the values or data sets ID.device 101i and config-provisioning.ID.device
25 212. Discovery server database 110a may comprise a database with tables to track configuration information for a plurality of devices 101. A device manufacturer or device owner could send initial data for a device 101 to discovery server 110 and discovery server database 110a. In exemplary embodiments, config-provisioning.ID.device 212 contains data used by initiator 102 for the configuration of device 101, including conducting a device provisioning protocol 191. Exemplary
30 data depicted for config-provisioning.ID.device 212 in Figure 2a includes DPP application 102g, configuration parameters 151, version number 212a, ID.Authentication Server 111a, URL-authentication Server 111b, cert.AS 111c, auth. parameters 111d. Additional or other data for initiator 102 and device 101 could be included in discovery server database 110a and values for config-provisioning.ID.device 212. The data for config-provisioning.ID.device 212 can comprise a
35 set of values or a table of data for initiator 102 and/or a DPP application 102g to use in order to

conduct a configuration step 106 with device 101, where a specific device 101 can be identified by either ID-token.device 206 or ID.device 101i.

DPP application 102g in a config-provisioning.ID.device 212 can specify the name or URL of a software application for initiator 102 to utilize (e.g. a field with less than a few hundred bytes), instead of comprising the full software package for DPP application 102g in exemplary embodiments (where the full DPP application 102g could comprise several megabytes or more). For these exemplary embodiments, initiator 102 could utilize the name or URL to fetch the software for DPP application 102g. DPP application 102g could be a public or private application for the operating system 102e of initiator 102. For embodiments where initiator 102 utilizes an Android-based operating system, DPP application 102g could be downloaded from the Google Play store or a private web site associated with network 107. For embodiments where initiator 102 utilizes an Apple ® / IOS –based operating system, DPP application 102g could be distributed through a developer enterprise program or downloaded through iTunes ®. Other possibilities exist as well without departing from the scope of the present invention for an initiator 102 to download a DPP application 102g, where the name of DPP application 102g is received from a discovery server 110, after initiator 102 sends a message 209 to a discovery server 110.

As depicted in Figure 2a, values for config-provisioning.ID.device 212 sent to initiator 102 may also include configuration parameters 151 and version number 212a. Configuration parameters 151 may include values and data for initiator 102 to utilize with configuration application 102g and communicate with device 101 using a WiFi radio 102w. Exemplary uses of parameters 151 with initiator 102 conducting a configuration step 106 can comprise (i) cryptographic algorithms to utilize, (ii) ECC curve names and key lengths and associated data, (iii) selection and parameters for a WiFi radio 102w in initiator 102 to communicate with device 101, including the selection of one of multiple possible radios in initiator 102, (iv) addresses, names, and port numbers to utilize with IP network 113 including IP version number such as IP4 or IP6, (v) timers and retry counts for communication with device 101 or DPP server 113 or authentication server 111, (vi) parameters for network 107 including a certificate authority, RSA PKI keys, selected cryptographic algorithms, etc, and (vii) parameters for communicating with device 101, etc.

Version number 212a for config-provisioning.ID.device 212 may specify a version number of DPP application 102g to utilize, or a list or dataset of version numbers of standards supported, such as IP version of IP network 113, WiFi radio 101h type such as 802.11ac or 802.11ax, TLS or DLTS version supported by device 101, etc. In exemplary embodiments, not all of the values depicted for config-provisioning.ID.device 212 are sent from a discovery server 110 to an initiator 102, and only a subset of the depicted values are sent instead. In an exemplary embodiment, similar but different to that depicted in Figure 2a, the data within config-provisioning.ID.device 212 can include all the data for a message 220 below, and in these embodiments an initiator 102 could use

config-provisioning.ID.device 212 to conduct a configuration step 127 for initiator 102 and then proceed to step 301 in Figure 3a, thereby omitting the subsequent steps depicted in Figure 2a.

As depicted in Figure 2a, config-provisioning.ID.device 212 sent to initiator 102 may also include ID.authentication Server 111a, URL-authentication Server 111b, cert.AS 111c, auth. parameters 111d. ID.Authentication Server 111a could comprise an identity or name for authentication server 111, such as a domain name or multiple domain names for multiple servers. URL-Authentication Server 111b could comprise a URL for configuration application 102g to call in order to authenticate with authentication server 111. In some exemplary embodiments, URL-authentication server 111b includes either ID.Device 101i or ID-token.Device 206, so that when initiator 102 using DPP application 102g calls URL-authentication Server 111b, the URL includes a unique string that identifies device 101. Or, in another exemplary embodiment, initiator 102 can (a) call URL-authentication server 111b without an identity for device 101, and subsequently (b) send either ID.Device 101i or ID-token.Device 206. Cert.AS 111c could comprise a certificate for authentication server 111 where cert.AS 111c could include (i) a public key for authentication server 111 and (ii) a signature of a certificate authority. Authentication parameters 111d can include parameters for DPP application 102g to utilize with authentication server 111, such as, but not limited to, hash or message digest algorithms, digital signature algorithms, padding schemes, key lengths, nonce or challenge lengths, timers, and retry counts, encoding rules, TLS or DTLS versions supported, cipher modes supported (e.g. AES-CBC, AES-CTR), etc.

Message 211 from discovery server 110 to initiator 102 can communicate the above fields and values described for config-provisioning.ID.device 212 and ID.device 101i, as depicted in Figure 2a. In an exemplary embodiment, discovery server 110 can send initiator 102 the full DPP application 102g with the response 211 (e.g. several megabytes or more) instead of a name or URL for DPP application 102g (e.g. less than a few hundred bytes), if the current, updated DPP application 102g is not already downloaded by initiator 102. In another exemplary embodiment, initiator 102 can download DPP application 102g in a step 213 as depicted in Figure 2a, if initiator 102 has not already installed the software for DPP application 102g after receiving message 211. For example, response or message 211 could include a version number 212a that is higher or different than initially utilized by initiator 102 and/or an initial DPP application 102g, and consequently the higher version number 212a could indicate to initiator 102 that a different DPP application 102g would be required in order to communicate with device 101. In exemplary embodiments, DPP application 102g downloaded in a step 213 can include a hash signature that can be verified by initiator 102 before using the software for a DPP 191 with device 101.

Step 213 may comprise initiator 102 loading and launching the DPP application 102g and applying a subset of configuration parameters 151 or tag-value 101w. If any of the steps depicted in Figure 2a fail, error codes or messages could be displayed through initiator 102 to initiator user

102u. After successful launch of DPP application 102g, initiator 102 can connect with authentication server 111. The subsequent steps shown performed by an initiator 102 after a step 213 in Figure 2a could be performed by a DPP application 102g using data within config-provisioning.ID.device 212. In exemplary embodiments, DPP application 102g could be included
5 with the operating system 102e for initiator 102, and in this case a separate download of DPP application 102g may not be required.

Secure session setup 214 could comprise a secure HTTPS session with authentication server 111 using the URL-authentication server 111b, and the secure session could comprise a transport layer security (TLS) session such as TLS version 1.2, TLS version 1.3, or similar and subsequent
10 standards. Other networking standards besides HTTPS and TLS could be utilized as well for secure session setup 214, such as an IPSec tunnel. As part of the setup of secure session setup 214, initiator 102 can receive a certificate cert.AS 111c from authentication server 111, if initiator 102 had not already received cert.AS 111c from discovery server 110. Initiator 102 could verify cert.AS 111c
15 after receiving the certificate, such as verifying a signature from a certificate authority and parent certificates of cert.AS 111c as well all the way to recorded cert.CA.root 102r, which is depicted above in Figure 1d. In other words, a secure session setup 214 can mutually authenticate both authentication server 111 and initiator 102, such as using certificates from both sides with TLS. An additional layer of authentication can be performed by an initiator user 102u, although authentication for all of authentication server 111, initiator 102, and initiator user 102u is not required for some
20 embodiments.

In message 215, initiator 102 transmits authentication information to authentication server 111, and authentication server 111 verifies the authentication information in a step 217. The authentication information transmitted in a message 215 could comprise information to authenticate (i) initiator user 102u, (ii) initiator 102, or (iii) both initiator user 102u and initiator 102. The
25 authentication information in message 215 may comprise an initiator user 102u entering identification and credential information in a touch screen of initiator 102, and the information is passed to authentication server 111 (possibly in an encrypted or hashed form). In exemplary embodiments, initiator user 102u provides biometric data such as a fingerprint, an image of the user's face, or similar data, where initiator 102 transmits confirmation or signatures for the data
30 through secure session setup 214 to authentication server 111, and the authentication server 111 verifies the biometric data or signatures of the biometric data is associated to initiator user 102u.

Message 215 may also request the verification of initiator 102 with authentication server 111. In exemplary embodiments, initiator 102 records a certificate Cert0.initiator 102m for initiator 102. The certificate for initiator 102 includes a public key for the initiator 102 and a signature by a
35 certificate authority. In an authentication step 217, authentication server can use the information in message 215 to verify the identity of an initiator user 102u and/or initiator 102. In embodiments

where initiator 102 uses a device certificate for initiator 102 (which could be a Cert0.initiator 102m), authentication server 111 could verify that initiator 102 also records the private key SK0.initiator 102s, by (i) sending a challenge/random nonce and (ii) verifying a signature for the challenge/nonce from initiator 102 using the private key 102s for cert0.initiator 102m. Authentication server 111 can
5 verify the signature using the public key for initiator 102 from Cert0.initiator102m. Message 215 and authentication 217 could be conducted using authentication parameters 111d, which could be previously included in Config-provisioning.ID.device 212.

In step 216a of Figure 2a, initiator 102 may also authenticate authentication server 111, using cert.AS 111c. In step 216b, initiator user 102u can enter device owner information into
10 initiator 102, such as an owner name, a DNS name associated with owner, or a code or other value to identify a device owner. Device owner information could be automatically recorded for DPP application 102g or initiator 102 before a configuration step 106 from Figure 1a. For example, initiator 102 with DPP application 102g may operate on a mobile phone, and the owner of device 101 can be associated with or specified by the initiator user 102u. For example, if a home owner
15 with a mobile phone wishes to configure a plurality of devices 101 for his or her home, an owner identity 152 can be selected based on the user of a mobile phone which could be the initiator user 102u. Other possibilities exist as well without departing from the scope of the invention. The value identifying a device owner could be ID.owner 152, although the use of ID.owner 152 could be optionally omitted.

In some embodiments, the value of ID.owner 152 could be recorded in a tag 101c, such as if
20 device 101 comprises leased equipment that is owned by a different entity than an initiator user 102u. Note that a tag 101c could also comprise a collection of different tags, such that a first tag could specify a URL-DS 205 (which could be a label or sticker applied by a manufacturer), and a second tag could specify ID.owner 152 (which could be a label or sticker or mark applied to device
25 101 by an owner). A step 216b can function to associate device 101 with device owner for a network 107. In exemplary embodiments, millions of devices 101 could be distributed out to millions of device owners. Network 107 may prefer to know which devices 101 are associated with which device owners by using a value of ID.owner 152. In an exemplary embodiment, the value ID.owner 152 could be included in a set of Config-provisioning.ID.device 212 for cases where device owner
30 can register the ID.device 101i with discovery server 110 before initiator 102 sends message 209 to discovery server 110.

In message 218, initiator 102 can send a first set of identities to authentication server 111 through the secure session setup in 214, and also send authentication parameters 111d. Note that in
35 some embodiments the set of authentication parameters 111d can be sent from authentication server 111 to an initiator 102 in a response to message 218. The list of identities in a message 218 can

include an ID.device 101i (or ID-token.device 206), ID.initiator 102bi, and ID.owner 152, as depicted in Figure 2a.

In step 219a, authentication server 111 can process the first set of identities received in message 218. A step 219a could comprise authentication server 111 verifying that initiator 102 with initiator user 102u is authorized to configure or provision device 101. In other words, the prior step 217 could be an initial step to verify or authenticate identities, and then the subsequent step 219a could be verifying that the authenticated identities have privileges or authorization to conduct a configuration step 106 or a DPP 191 with device 101. For a step 219a, authentication server 111 may communicate with a device database 104 in order to confirm that initiator 102 with an initiator user 102u is authorized to conduct configuration step 106 and a DPP 191 with device 101 identified by ID.device 101i or ID-token.device 206. Authentication server 111 can utilize any or all of the identities received in message 218 in order to identify and communicate with the correct device database 104 and determine authorization for initiator 102 and/or initiator user 102u.

Although not depicted in Figure 2a, authentication server 111 could utilize a mutually authenticated secure channel such as TLS v1.3 with a device database 104 in a step 219a. In some exemplary embodiments, a step 219a could be optionally omitted, for example if a device owner is also an initiator user 102u, such as a home owner using initiator 102 and installing a device 101 in their home. In these embodiments, an authentication step 217 may be sufficient to allow/authorize initiator 102 to conduct the DPP 191 and configuration step 106. Other possibilities exist as well for an authentication server 111 use a step 219a to authorize that an authenticated initiator 102 and/or initiator user 102u can conduct a DPP 191 and configuration step 106 without departing from the scope of the present invention.

After the authorization step 219a for initiator 102, authentication server 111 can query device database 104 for (i) initiator configuration 130 for device 101 using ID.device 101i or ID-token.device 206, and (ii) a DPP server URL 103b for initiator 102 to utilize in order to conduct a device provisioning protocol 191 and configuration step 106 with device 101, (iii) MAC address for responder 101x of MAC.R 101d, and (iv) optionally, responder public bootstrap keys Bi 104a and Br 101a or a PKEX key 198 for conducting a public key exchange protocol. The optional sending of bootstrap public keys and MAC address for responder 101x can depend on the selected operation of initiator 102 for conducting a DPP 191. For embodiments depicted in Figure 3a below, the bootstrap public keys Bi 104a and Br 101a can remain in DPP server 103, and in these embodiments then the query to device database 104 after step 219a can query for (i) through (iii) in the first sentence of this paragraph. In other words, for the embodiments depicted in Figure 3a and 5b, the public bootstrap keys Bi 104a and Br 101a, may optionally not be queried and sent back to authentication server 111 and initiator 102, but remain in a network 107 for enhanced security purposes.

For embodiments depicted in Figure 5a, the public bootstrap keys Bi 104a and Br 101a, may optionally be queried and sent back to authentication server 111 and initiator 102, in order for initiator 102 and DPP server 103 to conduct the steps and message flows depicted and described in connection with Figure 5a. Other possibilities exist as well for a query and communication of data between an authentication server 111 and a device database 104 after an authorization step 219a without departing from the scope of the present invention. Note that in exemplary embodiments, initiator bootstrap private key bi 104b can continue to reside within device database 104 and/or network 107 and may optionally not be sent back to initiator 102 in order to enhance security of the operation of a DPP 191 by combined operation of an initiator 102 and DPP server 103 as depicted in Figure 3a, Figure 5a, and Figure 6a.

A device database 104 in Figure 2c can comprise a database for a plurality of devices 101, and device database 104 was depicted in Figure 1a above and also Figure 2b below. Device database 104 can record data for the plurality of devices 101 in a system 100, including data for device identities, responder MAC addresses 101d, device public bootstrap keys Br 101a, initiator bootstrap PKI keys, PKEX keys 198, and initiator configuration 130, and other data as well in order to support the operation of a DPP 191 using an initiator 102 with DPP server 103. A device database 104 could also record additional data pertaining to device 101 with ID.device 101i, such as (i) a DPP server 103 to use with device 101 and/or initiator 102, where the location or identity of DPP server 103 could be in the form of a DPP server URL 103b (depicted as URL-DPPS 103b), and also (ii) a certificate for DPP server 103 comprising cert.DPPS 103c. In a step 219b, device database 104 can use the identities received from authentication server 111 in order to query for data such as (i) through (iv) described in two paragraphs above.

For a step 219b, if device database 104 does not record or have the device 101 identity ID.device 101i or ID-token.device 206, or initiator configuration 130, and/or URL-DPPS 103b, then for step 219b device database 104 could point authentication server 111 to another server or location for the exemplary queried data (i) through (iv) described in three paragraphs above and depicted in Figure 2a. The other server or location for queried data (i) through (iv) could be recorded possibly by a device manufacturer which could also record a second device database 104. Note that a device manufacturer could record some of the data for a device database 104, since the data such as MAC address, public bootstrap keys, etc. could be recorded upon manufacturing, although other possibilities exist as well.

Recording URL-DPPS 103b in a database 104 can provide greater flexibility for conducting a DPP 191 and configuration step 106 with a device 101 since an initiator user 102u or device owner may prefer that a DPP 191 or configuration step 106 be performed using a particular DPP server 103. However, the DPP server 103 may not be known by a device manufacturer at the time of manufacturing of device 101. In other words, (a) a device manufacturer cannot write a value for a

URL-DPPS 103b to memory of device 101 (or on a tag 101c) if (b) the device manufacturer does not know the value before the device 101 is shipped away from the device manufacturer facilities. Further, an initiator 102 may not know which of a plurality of possible DPP servers 103 to utilize when conducting a DPP 191 with a device 101 without receiving the data for a URL-DPPS 130b via IP network 113 such as from a network 107 or authentication server 111.

At step 219c, authentication server 111 could receive and process data from device database 104. The received data could comprise (i) initiator configuration 130 for ID.device 101i or ID-token.device 206, and (ii) a DPP server URL 103b for initiator 102 to utilize in order to conduct a device provisioning protocol 191 and configuration step 106 with device 101 (possibly with a certificate cert.DPPS 103c), (iii) MAC address for responder 101x of MAC.R 101d, and (iv) optionally, public bootstrap keys Bi 104a and Br 101a or a PKEX key 198 for conducting a public key exchange protocol. Although not depicted, an identity for device 101 or initiator 102 could also be returned in the data received for step 219c, since authentication server 111 may process data for a plurality of different initiators 102 and devices 101. In another exemplary embodiment, authentication server 111 could record a device database 104 for a plurality of devices 101 in a system 100 and system 200 and in this embodiment then step 219c could represent querying a locally stored device database 104 for authentication server 111 to obtain the depicted data. Further, in exemplary embodiments where initiator 102 can operate as an initiator 102* depicted in Figure 1b (and not as an initiator proxy 102' depicted in Figure 3a, Figure 5a, and Figure 5b below), then the present invention also contemplates that initiator bootstrap private key bi 104b could also be received by authentication server 111 and subsequently sent in a message 220 to initiator 102.

As depicted in Figure 2a, authentication server 111 can send initiator 102 a message 220 through secure connection 214, where message 220 can include a device 101 identity such as ID-token.device 206 (since initiator 102 may communicate with a plurality of devices 101), (i) initiator configuration 130 for ID.device 101i or ID-token.device 206, and (ii) a DPP server URL of URL-DPPS 103b for initiator 102 to utilize in order to conduct a device provisioning protocol 191 and configuration step 106 with device 101 (possibly with a certificate cert.DPPS 103c), (iii) MAC address for responder 101x of MAC.R 101d, and (iv) optionally, public responder bootstrap keys Bi 104a and Br 101a or a PKEX key 198 for conducting a public key exchange protocol. For embodiments where public bootstrap keys Bi 104a and Br 101a are sent in message 220, then also associated data for the keys could be sent such as a selected set of parameters 199a, such that an initiator 102 could determine which cryptographic parameters are associated with the keys (such as an ECC curve name, key lengths, hash algorithms, etc.) An authentication server 111 could receive a selected set of cryptographic parameters 199a for bootstrap keys from device database 104.

Initiator 102 could receive message 220 and read and record the data received. ID-token.device 206 can be useful in a message 220 for initiator 102, because initiator 102 could

communicate with a plurality of devices 101 over time and thus keep track of which message 220 is for which device 101. Note that ID.device 101i could be sent instead of or in addition to ID-token.device 206.

At step 221, initiator 102 could process data received in message 220. Initiator 102 could determine if data for initiator configuration 130 is supported by initiator 102, such as if a WiFi radio 102w supports a specified operating class 130a. For example, responder 101x may operate on WiFi at 60 Ghz and WiFi radio 102w may not support 60 Ghz, and in this case initiator 102 could in step 221 abort attempts to conduct a device provisioning protocol 191 with responder 101x. At step 221, initiator 102 could also determine if other data received in message 220 is supported by initiator 102, such as if key lengths or associated parameter sets 199a for any keys received in message 220 are supported by initiator 102 and potentially DPP application 102g. For example, if a bootstrap public key Br 101a is received in message 220 with a selected set of cryptographic parameters 199a that are not supported by DPP application 102g (such as a curve name not supported by initiator 102 with DPP application 102g), then initiator 102 could terminate further attempts at conducting a DPP 191 and also notify initiator user 101a.

Note that as contemplated herein, the transfer of a PKI key in a message can also be with the transfer of a corresponding selected set of cryptographic parameters 199a. If a message is depicted in Figure 2a (and other figures below), as transferring a PKI key, such as, but not limited to, Br 101a, then a selected set of cryptographic parameters 199a can be sent with the key as well. If parameters 199a are not received by a node when receiving a PKI key, or the parameters are otherwise unknown (such as the parameters not being previously selected and sent/received), then initiator 102 can operate with the assumption the PKI keys are for set "A" in a set of parameters 199 below in Figure 2b, which would also comprise the basic set of parameters for device interoperability as specified by cryptographic suite 1 in section 3.2.3 of DPP specification version 1.0.

At step 222, initiator 102 could determine an initiator mode 306a (from Figure 3a below) for conducting a device provisioning protocol 191 and configuration step 106, using data received in a message 220. If the key comprising Br 101a, with possibly parameters 199a is received in message 220 along with a command for initiator 102 to conduct responder only authentication 141, then initiator 102 could operate as an initiator 102* depicted in Figure 1b. If the keys comprising Br 101a, Bi 104a, bi 104b are received in a message 220 along with a command for initiator 102 to conduct a mutual authentication 142, then initiator 102 could operate as an initiator 102* depicted in Figure 1c. If no bootstrap PKI keys are received by initiator 102 in a message 220 (but other data is valid such as initiator configuration 130), then initiator 102 can begin operating as an initiator proxy 102' as depicted in Figure 3a. If only bootstrap PKI keys Br 101a and Bi 104a are received in a message 220, then initiator 102 could proceed with a configuration mode to support subsequent DPP 191 and configuration step 106 as depicted in Figure 5a below. Other possibilities exist as well for

an initiator 102 to select the steps for conducting a device provisioning protocol 191 using data or commands in a message 220 without departing from the scope of the present invention. In another exemplary embodiment, a DPP server 103 sends initiator 102' the initiator mode 306a (in Figure 3a) for initiator 102' to use for a DPP 191 with DPP server 103. Note that as contemplated herein, steps or actions that are described as initiator 102 or initiator 102' performing the action could also be performed by (i) a DPP application 102g or (ii) the computing device operating initiator 102, such as a mobile phone, smart phone, tablet, or even (iii) a WiFi access point 105 as depicted in Figure 8 below.

Initiator 102 could then conduct configuration step 127, which was depicted and described in connection with Figure 1d above. In summary, initiator 102 could backup or record any previously active user configuration 131 operating for a WiFi radio 102w in initiator 102, such as (a) recording user operating class 131a in a nonvolatile memory 102f in order to (b) restore the configuration 131 at a later time. In exemplary embodiments, a step 127 comprises initiator 102 reading the initiator configuration 130 and activating them with WiFi radio 102w in order to transmit DPP message 122 to responder 101x, optionally using MAC.R 101d in order to send unicast messages to responder 101x.

In an alternative exemplary embodiment to that depicted in Figure 2a, discovery server 110 could be optionally omitted, and data within tag 101c or tag value 101w could specify authentication server 111 as well as data for Config-provisioning.ID.device 212. However, the use of a discovery server 110 can be preferred for some exemplary embodiments since a discovery server 110 could provide increased flexibility over the lifetime of device 101, which could be a decade or longer, since parameters Config-provisioning.ID.device 212 and related data could change over time. Including all data for config-provisioning.ID.device 212 in tag value 101w or tag 101c may reduce flexibility for changing or updating the data over time, such as specifying (i) an updated DPP application 102g (where app versions for smartphones may be updated relatively frequently such as more than once a year), or other (ii) updated data for network 107 over time. Further, the use of a discovery server 110 can support potentially millions of devices communicating potentially with many different authentication servers 111 and DPP servers 103.

For another alternative exemplary embodiment to that depicted in Figure 2a, although message 220 with data for initiator 102 is depicted as flowing from authentication server 111 to initiator 102, other possibilities exist as well for initiator 102 to receive the data within message 220 without departing from the scope of the present invention. In some embodiments, data within message 220 could be obtained by initiator 102 from a DPP server 103 directly, and for these embodiments, a mutually authenticated secure session similar to secure session 214 could be set up between initiator 102 and DPP server 103. Further, the functionality for different servers in Figure 2a could be combined, such that an authentication server 111 also operates as a discovery server 110.

In another embodiment, a DPP server 103 could operate as all of a discovery server 110 and authentication sever 111, and other possibilities exist as well without departing from the scope of the present invention.

Figure 2b

Figure 2b is an illustration of an exemplary device database, with tables for a device database recording exemplary data, in accordance with exemplary embodiments. A device database 104 depicted and described above in connection with Figure 1a and Figure 2a can record data for (i) an unconfigured initiator 102 or (ii) a configured initiator 102' and a DPP server 103 to conduct a device provisioning protocol 191 with a device 101 operating a responder 101x. A database 104 could record a plurality of tables for a plurality of devices 101 and/or responders 101x, such as the depicted values for a key table 104k, a cryptographic parameters table 104p, and a device table 104d. Other possibilities exist as well for the organization, tables, and recorded data within a device database 104 without departing from the scope of the present invention. Data within device database 104 could be encrypted using a symmetric key. As depicted in Figure 1a, a network 107 could operate and record a device database 104, although another entity besides network 107 could also operate and record the device database 104, such as a device manufacturer or a device owner.

Device database 104 can record sets of ID-token.device 206, ID.device 101i, MAC.R 101d for the MAC address of the responder 101d, global operating class 130a, channel list 130b, cryptographic parameters 199, and a secret shared key 198 for a public key exchange protocol (PKEX). These values within a device database 104 could be recorded in a device 101 and/or responder 101x upon manufacturing of a device 101. The exemplary data in device database 104 could be obtained from the manufacturer. For embodiments where some data is not previously recorded in a device database 104 before an initiator 102' conducts a step 301 as depicted below in Figure 3a, then a network 107 operating device database 104 could query the device 101 manufacturer or a device 101 distributor in order to receive and record values for a device database 104, and other possibilities exist as well for the source of device 101 data within a device database 104.

In device database 104, ID-token.device 206 can comprise a unique, obfuscated identity for device 101, such as the identity read from a tag 101c on device 101. ID.device 101i can comprise an identity for device 101i that is not obfuscated, and could be a serial number for device 101. Although Figure 2a above depicts a tag 101c as recording ID-token.device 206, the tag 101c could alternatively record ID.device 101i and send the non-obfuscated identity to initiator 102 in Figure 2a. The two identity values shown for a device 101 in a device database 104 allows a DPP server 103 to map between ID-token.device 206 and ID.device 101i, and also convert one value into another.

MAC.R 101d can comprise the MAC address used by a responder for conducting a device provisioning protocol 191. In other words, MAC.R 101d can be the MAC address used by responder 101x in order to listen for DPP authentication request 122 messages from an initiator 102'. In other words, MAC.R 101d can be used by an initiator 102' in order to send a unicast message. In some exemplary embodiments, MAC.R 101d can be omitted from a device database 104 or otherwise not available to a network 107, and in this case an initiator 102' can send a DPP authentication request message 122 as a broadcast message on a WiFi radio link 192 to responder 101x.

Global operating class 130a and channel list 130b can specify the settings a device 101 WiFi radio 101h will use with responder 101x, where responder 101x periodically listens or monitor for DPP authentication request messages 122 over time as using different RF frequencies or channel numbers within ISM band RF radio spectrum. As depicted in Figure 2b, an initiator configuration 130 can include channel list 130b and cryptographic parameters 199. An initiator configuration 130 could also include the global operating class 130a and other information could be included as well in an initiator configuration, such as timer values for retry of sending DPP 191 messages, timer values for timeouts for sending and receiving messages, as well as counters for a maximum number of retries on sending messages.

Although not depicted in Figure 2b, an initiator configuration 130 in a device database 104 could also specify an initiator mode 306a (in Figure 3a and Figure 3b below) for initiator 102' to operate with responder 101x, where different modes could comprise (i) DPP server 103 records and operates on all bootstrap PKI keys for initiator 102', or (ii) initiator 102' records and operates on bootstrap public keys and derived ephemeral keys, but not initiator bootstrap private key bi 104b (e.g. as depicted in Figure 5a), and other possibilities exist as well. Sets of cryptographic parameters 199 in device database can represent different sets of cryptographic parameters supported by responder 101x in order to conduct a device provisioning protocol 191. For example, some devices 101 may use a first set of cryptographic parameters with PKI keys such as bootstrap and ephemeral keys, and other devices may use a second set of cryptographic parameters, which could be specified by cryptographic parameters 199. Note that devices 101 can optionally support more than one set of cryptographic parameters, as depicted.

Device database 104 can record a key table 104k, which could comprise exemplary keys as depicted. A key table 104k can include a secret shared PKEX key 198, responder bootstrap public key Br 101a, initiator bootstrap public key Bi 104a and initiator bootstrap private key bi 104b. PKEX key 198 could be used with a public key exchange protocol as specified in IETF draft document "Public Key Exchange", which is hereby incorporated by reference in its entirety. In some exemplary embodiments, a PKEX key 198 can be omitted, such as for the example data depicted with an "NA" in Figure 2b. The values depicted of X1, X2, etc. for a PKEX key 198 could

comprise the secret key for each device 101 identified by ID-token.device 206 or MAC.R 101d. For the exemplary database 104 depicted in Figure 2a and other tables herein, a numeric value following a constant name can depict a subsequent or different instance or numeric value for the constant. In other words, “X1” and “X2” for a PKEX key 198 can represent different values of “X” for the PKEX. Similarly, values of “Br-1” and “Br-2” depict different values for a responder bootstrap public key Br 101a, but exemplary values of “Br1” and then a second “Br1” can depict that the same value or number may be used for two different responders 101x.

Responder bootstrap public key Br 101a in a device database 104 can represent the bootstrap public key for responder 101x. Br 101a was depicted and described above in connection with Figure 1a and Figure 1b above. In exemplary embodiments, a device database 104 can record a plurality of different responder bootstrap public keys Br 101a, where each key can be associated with different responders 101x. Although not depicted in Figure 2b, for some embodiments different responders 101x could also share the same numeric value for a responder bootstrap public key Br 101a. Different responder bootstrap public keys Br 101a can be associated with a different selected set of parameters 199a. The manufacturer or entity configuring responder 101x in device 101 may not know which of a plurality of different possible sets of cryptographic parameters 199 may be utilized by an initiator 102 for conducting a DPP 191. For example, DPP specification version 1.0 identifies several possible sets of cryptographic parameters 199 for bootstrap keys that could be utilized in a DPP 191, such as the different values for cryptographic parameters shown in Appendix A of DPP specification version 1.0. As depicted in Figure 2b, a responder bootstrap public key Br 101a of “Br-3 (A)” can represent a third instance or number for Br 101a which uses the selected set “A” of cryptographic parameters 199 (e.g. curve p256 with SHA-256), while responder bootstrap public key Br 101a of “Br-3 (B)” can represent a third instance of Br 101a which uses the selected set “B” of cryptographic parameters 199 (e.g. curve p384 with SHA-384).

Initiator bootstrap public key Bi 104a and corresponding initiator bootstrap private key bi 104b in a device database 104 can represent the bootstrap public key and the bootstrap private key for initiator 102'. Bi 104a and bi 104b were depicted and described above in connection with Figure 1a and Figure 1b above. In exemplary embodiments, a device provisioning protocol 191 using an initiator 102' and a DPP server 103 can utilize a plurality of different initiator bootstrap PKI keys, and different values for the initiator bootstrap PKI keys Bi 104a and bi 104b can be associated with different responders 101x. As discussed above in connection with Figure 1a and 1b, a responder 101x could optionally record an initiator bootstrap public key Bi 104a, and subsequently the value recorded for the responder 101x could also be stored in a device database 104. The initiator bootstrap public key Bi 104a and corresponding initiator bootstrap private key bi 104b could be stored by a device database 104 concurrently with the responder bootstrap public key Br 101a, where the responder bootstrap key could be received from a device manufacturer or device distributor. Or,

in exemplary embodiments a device manufacturer or a device distributor could operate a device database 104 and subsequently record the exemplary keys and values depicted for a responder 101x in a device database 104 in Figure 2a. As depicted in Figure 2b, some responders 101x could share a common initiator bootstrap public key Bi 104a, which could be associated with a common initiator bootstrap private key bi 104b.

The sharing of initiator bootstrap public keys Bi 104a among multiple responders 101x can simplify the manufacturing and distribution of devices by using a common initiator bootstrap public key Bi 104a for some embodiments. For the present invention, the potential tradeoff of greater simplicity for reduced security (by reusing an initiator bootstrap public key Bi 104a in multiple devices 101) can be avoided, since the initiator bootstrap private key bi 104b may optionally not be sent to the initiator 102'. Yet, with the present invention, the full set of device provisioning protocol messages 191 can be sent and received with a responder 101x using mutual authentication 142. Embodiments for the present invention depict and describe the initiator bootstrap private key bi 104b as residing within a DPP server 103 when an initiator 102' conducts a DPP 191 (as depicted and described in connection with Figure 3a, Figure 5a, and Figure 5b below). Consequently, (i) the initiator bootstrap private key bi 104b does not need to be recorded by an initiator 102* (as in Figure 1c), and (ii) an initiator bootstrap public key Bi 104a can be commonly shared between multiple responders 101x.

Thus, the present invention resolves the potential tradeoff of security versus simplification by supporting manufacturing and distribution of devices 101 with responders 101x with commonly shared initiator bootstrap public key Bi 104a (depicted in Figure 2b as "shared keys"). Additional details will be provided in Figure 3a, Figure 5a, and Figure 5b below. The initiator bootstrap private key bi 104b can be recorded in a device database 104 and subsequently sent to a DPP server 103, but optionally not sent to initiator 102', and initiator 102' can still conduct a device provisioning protocol 191 with mutual authentication 142 as depicted in Figure 1c. In exemplary embodiments, only an initiator bootstrap public key Bi 104a is recorded inside an initiator 102, while a device database 104 can record both the initiator bootstrap public key Bi 104a and the initiator bootstrap private key bi 104b.

Cryptographic parameters 199 can specify sets of cryptographic parameters that are supported by responder 101x when using bootstrap PKI keys. Cryptographic parameters 199 can be recorded in a device database 104 as a parameters table 104p. Cryptographic parameters 199 can record a collection of cryptographic algorithms or specifications such as a set identifier 199a, a key length 199b, an ECC curve name 199c, a hash algorithm 199e, a key length for a symmetric ciphering algorithm 199f, and a nonce length 199g. As contemplated herein, when a selected set of cryptographic parameters such as using the words or description "parameters 199a" or "cryptographic parameters 199a" can specify a row of parameters or values in a set of cryptographic

parameters 199, such that the collection of values in the row can be used with a device provisioning protocol 191. Set identifier 199a can be an identity for a row or set of values for cryptographic parameters 199. For example, set “A” can comprise cryptographic suite 1 as specified in section 3.2.3 of DPP specification version 1.0. Key length 199b can be the length of keys in bits for PKI
 5 keys used with a device provisioning protocol with a responder 101x. ECC Curve name 199b can be a name for an ECC curve used with PKI keys for a device provisioning protocol with a responder 101x.

Hash algorithm 199e in cryptographic parameters 199 can be the name of a secure hash algorithm, such as the exemplary SHA-256 algorithm depicted, which may also be referred to as
 10 “SHA-2”. Hash algorithm 199e can be used in a key derivation function and also to generate a signature based on a received nonce. A key length for a symmetric ciphering algorithm 199f can specify the length of a symmetric key in bits to use with a symmetric ciphering algorithm. The depicted algorithm in Figure 2b for algorithm 199f is “AES-SIV”, although other symmetric ciphering algorithms could be specified in a set of cryptographic parameters as well. In other words,
 15 an additional, different column in cryptographic parameters 199, such as an exemplary “symmetric algorithm” 199z (not depicted) could specify the symmetric ciphering algorithm to utilize with a responder for a device provisioning protocol. Nonce length 199g can specify the length in bits for nonces or random numbers generated by both initiator 102’ and responder 101x, where the nonces can be used for generating signatures and encryption keys.

As depicted in Figure 2b, a device database 104 can also record a device table 104d. Device
 20 table 104d can record a plurality of different PKI keys for the same responder 101x. Device table 104d can include ID-device 206, initiator bootstrap keys sequence number 197, secret shared PKEX key 198, responder bootstrap public key Br 101a, initiator bootstrap public key Bi 104a and initiator bootstrap private key bi 104b. Devices 101 may be designed to be utilized for a decade or longer,
 25 and during that time it may be desirable for a device 101 with responder 101x to undergo more than one instance of a device provisioning protocol. As one of many possible examples, a device 101 could have performed a “factory reset” or equivalent clearing of nonvolatile memory, such that device 101 is restored to an unconfigured state. Or, device 101 may be sold or transferred to a different owner, where the new owner wishes to conduct a device provisioning protocol 191 with the
 30 same device 101 a second time. These are just a few of the many possible reasons why a device 101 could undergo more than one instance of a device provisioning protocol 191 during the lifetime of device 101, and other examples are possible as well. Through the use of a device table 104d, different values for an initiator bootstrap PKI keys could be utilized with the same device 101 with responder 101x over time, where the different values for PKI keys used with responder 101x could
 35 be identified by initiator bootstrap keys sequence number 197. Note that a PKEX key 198 could also

change for a device 101 over time, and initiator bootstrap keys sequence number 197 could also identify a new or different PKEX key 198.

In a device table 104d, a responder public bootstrap key Br 101a can remain relatively static over time, such as the depicted same numeric value for key Br 101a of “Br-1” used for each sequence of initiator bootstrap keys. In other words, although a device 101 may undergo a factory reset or similar second instance of a device provisioning protocol, in exemplary embodiments the same responder key Br 101a can be reused. If the responder bootstrap public key Br 101a is changed for some reason (such as a device user reprograms device 101), then a second, new responder bootstrap public key Br 101a could be recorded in device table 104d with a designation “Br-2” for the same device with a new value for sequence number 197.

In device table 104d, different values for initiator bootstrap public key Bi 104a and initiator bootstrap private key bi 104b could be utilized over time, represented by increasing numbers for sequence number 197 for the same device 101 identified by ID-token.device 206. A device table 104d could also identify devices by ID.device 101i instead. For a first sequence number 197 with a value of “1”, initiator bootstrap public key Bi 104a could record a value represented as “Bi-1a”. For a second sequence number 197 with a value of “2”, initiator bootstrap public key Bi 104a could record a value represented as “Bi-1b”, etc. In other words, the depicted number plus letter represents a different key for the same device. A different device could have initiator bootstrap public keys of “Bi-2a” for a first sequence number “1” and then a key “Bi-2b” for the second sequence number 197. Similar designations and tracking or mapping of keys could be utilized for an initiator bootstrap private key bi 104b as well.

In exemplary embodiments, the initiator bootstrap public key Bi 104a within a responder 101x can be updated by a DPP server 103, as depicted in Figure 6a below. One benefit for updating key Bi 104a is that although (i) multiple manufactured devices 101 with responders 101x could initially share a common initiator bootstrap public key Bi 104a (e.g. Bi-1a) in order to conduct a mutual authentication 142 (depicted with key table 104k in Figure 2b as “shared keys”), (ii) a DPP server 103 could subsequently update a device 101 with a different, new initiator bootstrap public key Bi 104a (e.g. Bi-1b). The tracking of (a) which initiator bootstrap public key Bi 104a (identified by a sequence number 197) is used with which responder 101x can be accomplished by (b) using a device table 104d. Alternatively, a device table could be omitted, and a device database 104 could record both (a) initial keys 104a and 104b for a manufactured responder 101x (for initial use or a “factory reset”) and then (b) the subsequent, current keys 104a and 104b, and in this case a sequence number 197 could be optionally omitted.

In another embodiment keys 104a and 104b could be omitted from a device database 104, and only the responder bootstrap public key Br 101a is recorded. For these embodiments, key Br 101a may preferably not be sent to an initiator 102’ using the embodiment depicted and described in

connection with Figure 3 below. Responder 101x and device 101 could be designed for use where all of (i) tag 101c does not record the responder bootstrap public key Br 101a, (ii) the entity recording key Br 101a is not designed to send or transfer the key Br 101a outside of a database 104 and a DPP server 103, and (iii) access to data, calculations, or algorithm output where key Br 101a is used as input would only be provided to an authenticated initiator 102'. Subsequently, responder 101x can know that any initiator 102' that has access to the responder bootstrap public key Br 101a (indirectly via DPP server 103) can then be mutually authenticated. In this manner, each of PXEX key 198, and initiator bootstrap PKI keys could also be omitted and still responder 101x can trust data from initiator 102', such as a configuration object 109. In other words, in the present invention a system 100 can (i) operate without a mutual authentication 142 from DPP Specification version 1.0 shown in Figure 1c, and (ii) without an initiator 102' receiving the responder bootstrap public key Br 101a, and (iii) with DPP server 103 operating on the key and securely storing the key such that no other entities unnecessarily receive the responder bootstrap public key Br 101a, and in this manner a responder 101x can strongly mutually authenticate the initiator 102' (to a sufficient level to receive credentials 109), since only an initiator 102' that is authenticated with DPP server 103 could successful send and receive DPP message 191 with responder 101x.

Figure 3a

Figure 3a is a simplified message flow diagram illustrating an exemplary system with exemplary data sent and received by a device database, a DPP server, an initiator proxy, and a responder, in accordance with exemplary embodiments. Before starting steps and message flows depicted in Figure 3a, the initiator proxy 102' and device database depicted for system 300 in Figure 3a may previously complete exemplary message flows and steps depicted in Figure 2a above. System 300 can include a device database 104, DPP server 103, initiator proxy 102', and responder 101x operating in device 101. Initiator proxy 102' can communicate with DPP server 103 via IP network 113 using an access network 112, as depicted in Figure 2a. As contemplated herein, an initiator proxy 102' depicted in Figure 3a may also be referred to as initiator 102', and an initiator 102' and initiator proxy 102' may be equivalent. In other words, an initiator 102' may operate as an initiator with some functions and operations by proxy with DPP server 103.

Initiator 102' can be different than initiator 102* in Figure 1b and Figure 1c, since (a) some PKI and cryptographic computations (plus recording of PKI keys) that would normally be performed by an initiator 102* as contemplated by DPP specification version 1.0 can (b) be performed by DPP server 103 and thus initiator 102' is different than an initiator 102*. For system 300, DPP server 103 and device database 104 may establish a secure session setup 302a, which could comprise establishing a secure communications link between the two servers using protocols such as TLS,

IPSec, a virtual private network (VPN), a secure shell (SSH), or similar networking, transport, or application layer technologies in order to establish secure communications between DPP server 103 and database 104. Secure session setup 302a can be (i) equivalent or similar to secure session setup 214 depicted in Figure 2a above, and (ii) utilize certificates cert.DPPS 103c for DPP server 103 and a certificate for database 104 in order to provide mutual authentication and mutual key derivation for a symmetric encryption key.

Initiator 102' in system 300 can operate a running DPP application 102g and WiFi radio 102w operating with initiator configuration 130 from Figure 2a and Figure 1d, in order to communicate with a WiFi radio 101h operating in device 101 with responder 101x, where the communications can use a WiFi data link 192. DPP application 102g may previously be downloaded in a step 213 as depicted in Figure 2a and subsequently launched or activated for operation in a system 300. In exemplary embodiments, DPP application 102g may be (i) included or distributed along with an operating system 102e running on initiator 102' or (ii) acquired at a different time than a step 213, and thus from a separate download of DPP application 102g. In exemplary embodiments, WiFi radio 102w operating in initiator 102' can send either broadcast or unicast messages or datagrams to device 101 with responder 101x when using initiator configuration 130. In exemplary embodiments, responder 101x listens for incoming messages using the same initiator configuration 130.

DPP server 103 and initiator 102' can establish a secure session 302b, where secure session 302b could comprise a session using TLS, DTLS, IPSec, a VPN, and other possibilities exist as well. Initiator 102' can previously record the URL to reach DPP server 103 based on the value URL-DPPS 103b received in a message 220 in Figure 2a. Initiator 102' can also use the certificate for DPP server 103 comprising cert.DPPS 103c also previously received in a message 220. As depicted in Figure 3a, secure session 302b could also comprise DPP server 103 receiving and using an initiator certificate cert0.initiator 102m in order to authenticate initiator 102'. Alternatively, DPP server 103 could authenticate an initiator user 103a via a user ID and password entered by user 102u into a user interface 102k for initiator 102'. In some exemplary embodiments, authentication of an initiator 102' or an initiator user 102u could be omitted, such as if an initiator certificate cert0.initiator 102m is not present in initiator 102.

At step 302c in Figure 3, device 101 may power on from an unpowered or "deep sleep" state. Power could be provided from a battery or converted "wall power". Device 101 could (i) load the equivalent or responder version of information contained in initiator configuration 130 using a bootloader program and (ii) power on WiFi radio 101h. Responder 101 using WiFi radio 101h in device 101 could periodically cycle through a channels list 130b in order to listen for incoming DPP authentication request messages 122. In exemplary embodiments, before initiator 102' has conducted a step 127 to load initiator configuration 130, a device 101 can periodically power up and

attempt to listen for incoming messages 122. In other words, device 101 can use equivalent information for initiator configuration 130 as the default state or condition, even without initiator 102' being present or configured, since device 101 may not know the time or place where initiator 102' with initiator configuration 130 may be present. Responder 101x could also read the values for responder bootstrap public key Br 101a and initiator bootstrap public key Bi 104a, if present, in a step 302c.

As depicted in Figure 3a, initiator 102' using DPP application 102g can conduct a step 303a to collect an identity list 304, where identity list 304 comprises a list of identities for elements of systems 100, 200, and 300 around or nearby to device 101 during installation and/or operation. Identity list 304 can include a networks available list 304c. In order to conduct a step 303a, initiator 102' may need to temporarily disconnect from an access network 112, such as a mobile phone network, in order to perform a full frequency scan of all available mobile network operators and network access technologies around device 101 in order to collect a networks available list 304a. For embodiments where initiator 102' temporarily disconnects from an access network 112 or other cases where secure session 302b may be dropped, initiator 102' and DPP server 103 can pause secure session 302b and then restore the session when connectivity for initiator 102' through access network 112 is restored.

The networks available list 304c can be useful for DPP server 103 to select the best or a preferred access point 105 from the networks available list 304c for device 101. As depicted in Figure 3a, a networks available list 304c can also include the access technology available from a plurality of access networks 112 and access points 105 around device 101, such as 2G, 3G, 4G LTE, 5G, WiFi, NB-IoT, LoRaWAN, LPWAN, etc. Identification of all available wireless networks around device 101 can support DPP server 103 or network 107 in a later step 308b below to receive or determine a working set of credentials 109 for device 101 to utilize in order to obtain network connectivity via the access network 112. In exemplary embodiments, the networks available list 304c of surrounding wireless networks also includes an associated RF signal strength for each, such as RSSI as well as a frequency band, frequency list in Mhz, or RF channel for each surrounding wireless network. In exemplary embodiments, initiator 102' can observe WiFi broadcast packets from network Wifi Access Point 105 depicted in Figure 1a using SSID.Network-AP 109a and include the value in the networks available list 304c.

A step 303a may also comprise initiator 102' collecting a list of identities for a monitored unit associated with a device 101, and include the identities in the identity list 304. In exemplary embodiments, a step 303a in Figure 3a may comprise initiator 102' using a camera 102z to scan for a bar code or QR code on a surrounding monitored unit in order to collect a monitored unit identity 304b. In exemplary embodiments, device 101 can include sensors and transducers for monitoring and controlling external devices or equipment. A monitored unit for a device 101 could comprise

exemplary values of industrial equipment, health care equipment, security systems, etc. A step 303a could also comprise initiator 102' obtaining physical location 304a for device 101, which could comprise a room number, zone in a building or floor, shelf number, rack number, etc. Physical location 304a could also comprise a set of geographical coordinates for a value of physical location 304a or (ii) estimated geographical coordinates or similar physical location of device 101. In exemplary embodiments, initiator 102' can comprise a mobile phone or smart phone and can have a GPS or similar receiver for outdoor location, or use WiFi or Bluetooth for approximate indoor locations, and record the location in a physical location 304a.

Although step 303a to collect an identity list 304 for device 101 is depicted as after secure session setup 302a (confirming mutual authentication between initiator 102' and DPP server 103 in session 302a), step 303a to collect an identity list 304 could be conducted at other times, such as before Figure 3a or after a initiator user 102u takes initiator 102 to the approximate physical location for installation or configuration of device 101, including possibly during a step 202 above in Figure 2a. Or, in other exemplary embodiments the use of an identity list 304 can optionally be omitted, if initiator 102' and DPP server 103 can record the set of credentials 109 for a selected network access point 105, where the selected network access point 105 can be recorded without an identity list 304 (such as if the correct or preferred network access point 105 is known to DPP server 103 before a read step 202 in Figure 2a). As depicted in Figure 3a, initiator 102' can then send DPP server 103 a message 305 through secure session 302a, where message 305 can include ID-token.device 206 (depicted and described for message 204 in Figure 2a above) and the identity list 304. DPP server 103 can receive message 305 and record the data received in a server database 103a. The values of Physical Location 304a and Monitored Unit ID 304b can be recorded for support of subsequent operation of device 101 after a configuration step 106, and DPP server 103 could also send the values to another server supporting device 101 during operation such as a reporting server (not shown). DPP Server 103 can then send ID-token.device 206 through secure session 302a to device database 104 in a message 306.

A step 302d for DPP server 103 can also comprise DPP server 103 querying a server database 103a for an initiator bootstrap keys sequence number 197 for device 101 using ID-token.device 206 (or ID.device 101i). In exemplary embodiments, a device 101 with responder 101x could use multiple different initiator bootstrap public keys Bi 104a over time, such as updating the initiator bootstrap public key Bi 104a using the steps depicted and described in connection with Figure 6a below after a DPP 191 and configuration step 106 has been previously completed. The multiple possible reasons and benefits for updating and initiator bootstrap public keys Bi 104a are discussed with a step 608 in Figure 6a below. A server database 103a could record a value initiator bootstrap keys sequence number 197 in order to track which initiator bootstrap public key Bi 104a is currently recorded in responder 101x for device 101. For other embodiments, a responder 101x in

device 101 could also record a plurality of different initiator bootstrap public keys Bi 104a concurrently, such as a first initiator bootstrap public key Bi 104a for a first network 107 and a second initiator bootstrap public key Bi 104a for a second network 107. Responder 101x could use the multiple different, concurrently available initiator bootstrap public keys Bi 104a to concurrently
5 listen for a DPP authentication request message 122 from different possible networks 107. In a step 302d, A DPP server 103 could use an initiator bootstrap keys sequence number 197 to select the current, preferred initiator bootstrap public key Bi 104a for conducting a DPP 191 by an initiator 102'. DPP server 103 can send a database 104 a message 307 with an identity for device 101 of ID-token.device 206 (or identity ID.device 101i), and the selected initiator bootstrap keys sequence
10 number 197.

At step 306, DPP server 103 can select the mode of operation for initiator 102' to conduct a DPP 191 with device 101. The selected mode of operation for initiator 102' can comprise a mode 306a or an initiator mode 306a, which could be recorded in a server database 103a as depicted and described in connection with Figure 3b below. The selected mode 306a can specify the distribution
15 of PKI keys between DPP server 103 and initiator 102'. The selected mode 306a can determine the subsequent series of steps and message flows between DPP server 103 and initiator 102'. A first mode 306a could (a) specify that DPP server 103 records all public and private bootstrap and ephemerals keys for initiator 102', plus the public keys Br 101a and Pr 101e for responder 101x such that (b) resulting cryptographic operations and ECDH key exchanges and symmetric ciphering reside
20 within DPP server 103 and results are transmitted to initiator 102'. This first mode 306a could represent the message flows and steps in Figure 3a. A second mode 306a could specify that (i.a) initiator bootstrap PKI keys 104a and 104b reside within DPP server 103 and (i.b) responder bootstrap and ephemeral public keys for responder 101x reside within initiator 102', such that (ii) resulting cryptographic operations and key exchanges are shared between DPP server 103 and
25 initiator 102'. This second mode 306a could represent the message flows and steps depicted in Figure 5a.

A third mode 306a selected in a step 306 for DPP server 103 could specify the same recording of bootstrap and ephemeral keys as the second mode, with the addition that DPP server 103 receives and uses the initiator ephemeral private key pi 102b, also with resulting cryptographic
30 operations and key exchanges are shared between DPP server 103 and initiator 102' (but with a different allocation than the second mode). This third mode 306a could represent the message flows and steps depicted in Figure 5b. In exemplary embodiments, all of the first, second, and third modes operate with initiator bootstrap private key bi 104b residing within DPP server 103 and not shared with initiator 102'. By not sharing initiator bootstrap private key bi 104b with initiator 102' (which
35 could be a device with much less security than DPP server 103), the initiator bootstrap private key bi 104b can remain secured within network 107. These embodiments where initiator bootstrap private

key bi 104b remains in network 107 (e.g. in DPP server 103 and device database 104) support securely sharing the same initiator bootstrap public key Bi 104a potentially across a plurality of device 101 with responders 101x. Sharing the same initiator bootstrap public key Bi 104a across multiple devices supports recording the key Bi 104a in a device 101 during manufacturing. This simplifies and enables the use of a device provisioning protocol 191 across multiple devices 101 with mutual authentication, where the initiator 102' does not need to receive or be provisioned with the initiator bootstrap private key bi 104b.

In other words, without the technology described in the present invention, a device provisioning protocol with mutual authentication can essentially shift the problem of (a) securely mutually authenticating a device and providing credentials 109 to a device 101 to a new, but related problem of (b) securely providing an initiator bootstrap private key bi 104b to an initiator 102* . The first, second, and third modes 306a can solve this shifted/new problem since the initiator bootstrap private key bi 104b can permanently reside in a network 107 and does not need to be sent to the initiator 102' or an initiator 102*. However, with the technology in the present invention, the initiator 102' can still transmit and receive the full set of a device provisioning protocol 191 messages with a responder 101x as specified in DPP specification version 1.0.

A fourth mode 306a could specify that initiator 102' records all of the initiator bootstrap and ephemeral PKI keys, as well as the responder 101x public keys, such that resulting cryptographic operations and key exchanges reside in initiator 102'. This fourth mode 306a could represent the message flows and steps depicted in Figure 1b or Figure 1c, and in this case initiator 102' could comprise an initiator 102*. Other possibilities exist as well without departing from the scope of the present invention. Additional details for the different modes 306a selected in a step 306, with the resulting different allocation or recording of PKI keys are described below in Figure 3b for a server database 103a.

At step 308a, DPP server 103 can process data received in message 304 and the received identity list 304. DPP server 103 can use the values in a networks available list 304c in order to select an access network 112 for device 101. A step 308a could comprise DPP server 103 selecting a primary access network 308aa and a backup access network 308ab for a device 101, as depicted and described in connection with Figure 3b below. The selected access networks 308aa and 308ab for device 101 in a step 308a could be based on criteria such as bandwidth cost, expected energy or power requirements (i.e. 3G/4G may use more power than LPWAN), signal strength measurements recorded with a networks available list 304c, and commercial terms such as agreements between a (i) device owner for device 101 and (ii) networks in a networks available list 304c. DPP server 103 could also query other servers not depicted in Figure 3a to select an access network 112 from the networks available list 304c. In other words, a preferred access network 112 in the form of a primary access network 308aa for device 101 may need to be selected by DPP server 103 in a step

308a before the credentials 109 for the access network (or network access point 105) can be obtained. Although Figure 3a depicts DPP server 103 as conducting a step 308a in order to select a primary access network 308aa and backup access network 308ab, in exemplary embodiments a different server than DPP server 103 in a network 107 could (i) securely receive the networks
5 available list 304c and (ii) select the primary access network 308aa and backup access network 308ab.

After the selection of a primary access network 112 (308aa) and a backup access network 112 (308ab) in a step 308a, DPP server 103 could perform a step 308b depicted in order to obtain a set of network credentials 109 for both the selected primary access network 308aa and the selected
10 backup access network 308ab. For embodiments where (a) a network 107 operating DPP server 103 comprises a selected access network 308ab or 308ab, then (b) the credentials 109 for device 101 can be obtained from another server in a network 107 from Figure 1a and recorded in DPP server 103. For example, if a network access point 105 for network 107 is selected as an access network 112 for device 101, then network 107 would normally already record the set of credentials 109 for access
15 point 105. For embodiments where a selected access network 112 (e.g. selected primary access network 308aa) for device 101 other than network 107 operates the access point 105, then for a step 308b DPP server 103 could query the access network 112 for the equivalent of credentials 109. Additional details regarding embodiments where a different network than network 107 provide credentials for device 101 are depicted and described in connection with Figure 3b and Figure 6b
20 below.

Note that in exemplary embodiments, a DPP server 103 can preferably conduct a step 308a to select an access network 112 and a step 308b to obtain credentials 109 for the selected access network 112 before continuing with a device provisioning protocol 191. The reason is that if either
25 (i) a network cannot be selected potential from a networks available list 304c or (ii) credentials 109 cannot be obtained for the selected network, then there may be no reason to continue with a device provisioning protocol 191 since DPP server 103 may not have credentials 109 to send to device 101. In Figure 3a, although a step 308a and step 308b are depicted as taking place after DPP server 103 sends message 307 with a query for PKI keys to device database 104, in some embodiments a step 308a and step 308b could take place before DPP server 103 sends message 307 to device database
30 104.

Further, in some exemplary embodiments where initiator 102' uses network access point 105 with SSID.network-AP 109a with a preshared secret key (PSK) or similar authentication (as an access network 112 for initiator 102'), then a message 305 could comprise initiator 102' sending DPP server 103 the set of network credentials 109 used by initiator 102'. DPP server 103 could then
35 (i) record the credentials 109 in a server database 103a further described below in Figure 3b, (ii) use the value SSID.network-AP 109a as a selected primary access network 308aa, and (ii) use the

received credentials 109 from initiator 102' in a message 305 as the credentials 109 for device 101. Other possibilities regarding the source of credentials 109 exist without departing from the scope of the present invention.

Device database 104 can receive message 307 with ID-token.device 206 (or similarly ID.device 101i) and initiator bootstrap keys sequence number 197. Device database in step 309 can conduct a query of device database 104 for data associated with ID-token.device 206 and initiator bootstrap keys sequence number 197. In exemplary embodiments, a query in a step 309 can select an initiator bootstrap private key bi 104b and an initiator bootstrap public key Bi 104a using the initiator bootstrap keys sequence number 197 from a device table 104d for device 101. In this manner, for embodiments where initiator bootstrap PKI keys for device 101 are updated over time (e.g. such as depicted and described in connection with Figure 6a), then the most recently updated initiator bootstrap PKI keys can be selected by a database query in a step 309. The query in step 309 can return values for a message 310 from device database 104 to DPP server 103. The values in a message 310 can comprise data for ID-token.device 206 and the values can include responder bootstrap public key Br 101a, initiator bootstrap public key Bi 104a, initiator bootstrap private key bi 104b, a device identity ID.device 101i, a selected set of cryptographic parameters 199a for the PKI keys, and optionally PKEX key 198. Although not depicted in Figure 3a, message 310 could also include ID-token.device 206 so that DPP server 103 can track which of a possible plurality of initiators 102' and devices 101 that message 310 is associated with. In some embodiments, a step 302d and the use of a sequence number 197 can be omitted, and a query in step 309 and message 310 can return the most recently updated or recorded values for responder bootstrap public key Br 101a, initiator bootstrap public key Bi 104a, and initiator bootstrap private key bi 104b.

For embodiments where (A) a DPP session 191 between initiator 102' and device 101 fails due to a possible mismatch between (i) initiator bootstrap PKI keys selected in a query for step 309 and (ii) the actual initiator bootstrap public key Bi 104a recorded and used by a responder 101x in device 101 (such as the case where device 101 has undergone a "factory reset" to restore previously recorded initiator bootstrap public key Bi 104a), then (B) DPP server 103 could send a second message 307 with a request to use the oldest recorded initiator bootstrap PKI keys from a device table 104d (such as possibly the lowest or first recorded sequence number 197 for device 101 in device table 104d). For these embodiments where (A) a DPP session 191 between initiator 102' and device 101 fails due to a possible mismatch between (i) initiator bootstrap PKI keys selected in a query for step 309 and (ii) the actual initiator bootstrap public key Bi 104a recorded and used by a responder 101x in device 101, then (B) a query in step 309 can be for the oldest recorded initiator bootstrap PKI keys from a device table 104d. In other exemplary embodiments, DPP server 103 could send message 307 with a specific initiator bootstrap keys sequence number 197, and a query in a step 309 could be for a specific set of responder bootstrap public key Br 101a, initiator bootstrap

public key Bi 104a, initiator bootstrap private key bi 104b designated by the initiator bootstrap keys sequence number 197 in a device table 104d. Further, a key table 104k in a device database 104 can record the most recently updated values and keys for a device 101, while a device table 104d can record the historical values and PKI keys for a device 101 used over time, where the historical different values or PKI keys can be tracked, queried, or selected using an initiator bootstrap keys sequence number 197.

DPP server 103 can receive the message 310 and record the data received in a server database 103a. As depicted in Figure 3a, DPP server 103 can then send message 311 to initiator 102', where message 311 can include (i) the selected mode 306a of operation for initiator 102' and DPP server 103 for conducting a DPP 191 with responder 101x, (ii) a selected set of cryptographic parameters 199a for use with PKI keys and a device provisioning protocol, and (iii) optionally a PKEX key 198 for conducting a public key exchange with responder 101x. Message 311 can also optionally include device identity ID-token.device 206 in order to associate the data in message 311 with a particular responder 101x, since initiator 102' could potentially communicate with a plurality of different devices 101 over time.

At step 303b initiator 102' can receive and process the message 311. Initiator 102' and DPP server 103 can begin operating in the selected mode 306a for conducting a device provisioning protocol 191. Initiator 102' and DPP server 103 can apply and begin using the selected set of cryptographic parameters 199a. For a step 308b, if a PKEX key 198 is present in a message 311, then initiator 102' can conduct a public key exchange protocol with responder 101x in order to obtain responder bootstrap public key Br 101a via WiFi radio 102w and "in band". The public key exchange protocol is specified in the DPP specification version 1.0 section 5.6. If PKEX is optionally used between responder 101x and initiator 102', then initiator 102' could receive the responder bootstrap public key Br 101a and send the key Br 101a to DPP server 103 (and in this case DPP server 103 or database 104 may not have previously recorded responder bootstrap public key Br 101a). The use of a PKEX protocol and key 198 can also be optionally omitted in some embodiments, if responder bootstrap public key Br 101a can be recorded by a network 107 prior to a step 303b.

As depicted in Figure 3a, the collection of steps and messages before a step 313 can be grouped together into a step 301. Step 301 with the collective group of sub-steps and messages depicted for a step 301 can also be performed in other embodiments depicted in Figure 5a and Figure 5b below. Subsequent messages and steps depicted in Figure 3a starting after step 301 can represent a first mode 306a for DPP server 103 and initiator 102', where the allocation of recording PKI keys and symmetric keys from key exchanges for the first mode 306a are depicted below in Figure 3b.

A step 313 can comprise deriving an initiator ephemeral private key pi 102b and an initiator ephemeral public key Pi 102a, and in Figure 3a the DPP server 103 is depicted as conducting the

step 313 to derive the initiator ephemeral PKI keys. Step 313 can use a random number generator to produce a random number, which could be input into a key pair generation algorithm using the set of cryptographic parameters 199a. The resulting PKI key values from conducting a step 313 can comprise the initiator ephemeral private key π_i 102b and the initiator ephemeral public key P_i 102a. The resulting PKI key values from conducting a step 313, if conducted by a DPP server 103, can be recorded in a server database 103a, as depicted in Figure 3b below. Other embodiments depicted below contemplate a step 313 to derive initiator ephemeral PKI keys being conducted by an initiator 102'. In summary, some steps depicted as conducted by a DPP server 103 in Figure 3a could be conducted instead by an initiator 102' for other embodiments such as those depicted in Figure 5a and Figure 5b, with a corresponding change in supporting message flows between DPP server 103 and initiator 102'.

Step 314 can comprise DPP server 103 conducting a first initiator key exchange step 314 using an Elliptic Curve Diffie Hellman (ECDH) key exchange 401a, as depicted and described in connection with Figure 4a below. Data input for a step 314 can comprise the initiator ephemeral private key π_i 102b and the recorded responder bootstrap public key Br 101a. Data output from a step 314 can comprise a key k_1 as described in Figure 4a below. Step 314 can comprise the key exchange and symmetric key derivation step in DPP specification version 1.0 section 6.2.2 in order to for an initiator to obtain key k_1 .

At step 315, DPP server 103 can generate a random number for an initiator nonce 317a using parameters 199a. As used herein and throughout the present invention, the words "random number" can also mean a pseudo random number, since random numbers generated by nodes may not be truly random. However, their "randomness" may be sufficient in order to securely conduct the cryptographic operations contemplated herein. DPP server 103 can also determine the initiator capabilities, which could comprise a configurator for the initiator as depicted in Figure 3a (where responder 101x would be the enrollee). In other embodiments, the initiator 102' could have initiator capabilities as an enrollee (where the responder 101x could be the configurator). A step 315 can comprise encrypting the initiator nonce 317a and initiator capabilities using the derived key k_1 from step 314 in order to create a ciphertext 315a. An encryption step 315 step is further depicted and described in connection with Figure 4a below.

At step 316, DPP server 103 can calculate secure hash values for initiator bootstrap public key Bi 104a and responder bootstrap public key Br 101a, using a secure hash algorithm 199c. The values are depicted in Figure 3a as " $H(Bi)$ " and " $H(Br)$ ", respectively. DPP server 103 can then send initiator 102' message 317, which can include the hash value for the initiator bootstrap public key Bi 104a and the responder bootstrap public key Br 101a, the derived initiator ephemeral public key P_i 102a, and ciphertext 315a, where ciphertext 315a is encrypted with key k_1 and includes an

initiator nonce 317a and the initiator capabilities. Initiator 102' can receive message 317 via the secure session 302b.

For a step 316a (not depicted in Figure 3a, but conducted by initiator 102' after receipt of message 317 and also depicted in Figure 5a below), initiator 102' can then send DPP Authentication Request message 122 to responder 101x, using WiFi radio 102w with initiator configuration 130. In exemplary embodiments, initiator 102' sends message 122 as a unicast message to responder 101x using MAC.R 101d as the destination/recipient, and initiator 102' received MAC.R 101d in a previous message such as message 220. Alternatively, initiator 102' sends message 122 as a broadcast message (and MAC.R 101d would not normally be required for broadcast). However, unicast for WiFi can be preferred, since unicast messages are acknowledged at the data link layer. In a step 316a, initiator 102' may need to repeat message 122 multiple times on several different channels in a channel list 130b from initiator configuration 130, since (i) responder 101x may be periodically sleeping in order to conserve battery power, and (ii) responder 101x may also be cycling through a channel list 130b in order to listen for message 122 on different channels (and the cycling by initiator 102' and responder 101x may not be synchronized).

In addition, DPP authentication request message 122 in Figure 3a can have some differences than message 122 in Figure 1b and Figure 1c. Message 122 in Figure 3a can originate from an initiator proxy 102' (where initiator proxy 102' may not record an initiator bootstrap private key bi 104b for responder 101x), while message 122 in Figure 1c can be from an initiator 102*, where initiator 102* can record and operate on the initiator bootstrap private key bi 104b. In other words, from the perspective of a responder 101x message 122 in Figure 1c and Figure 3a could be identical, but the initiator 102* and 102' can be different, depending on the recording and processing of PKI keys and shared secret keys. Further, although a message 122, message 123, etc. as depicted herein shows key values of "k1", "ke", etc., the depiction of the keys is to illustrate that ciphertext is encrypted with the corresponding key by the use of brackets. In other words, although the key values are shown with the messages, the actual key values are not transmitted inside or with the message in exemplary embodiments. The equivalent syntax is used in DPP specification version 1.0.

Responder 101x can receive message 122 and perform a series of steps to process the message and respond to the initiator 102'. Responder 101x in device 101 can compare (i) hash values received in message 122 with (ii) the calculated hash values for bootstrap public keys from a step 302c. If (a) the hash values match or are equal, then responder 101x can proceed to subsequent steps, and otherwise (b) continue to cycle through a channel list 130b in order to listen for incoming messages 122. At step 318, responder 101x can also generate two random numbers using a random number generator. The first random number can be used as a responder ephemeral private key pr 101f. The responder ephemeral private key pr 101f can be input into an ECC key pair generation

algorithm in order to output a responder ephemeral public key Pr 101e. The set of cryptographic parameters 199a used for the responder ephemeral PKI key pair can match the recorded parameters 199a used with the bootstrap public keys such as responder bootstrap public key Br 101a. The second random number generated by responder 101x can be used as a responder nonce 318b.

5 Responder 101x can also determine responder capabilities 319b, which could comprise the capabilities for an enrollee.

Responder 101x can conduct a first responder key exchange step 319, where the details for a key exchange step 319 are depicted in Figure 4a below. In summary, the received initiator ephemeral public key Pi 102a (from message 122) and the responder bootstrap private key br 101b can be input

10 into an ECDH key exchange in order to mutually derive a shared secret key M 410. The shared secret key M 410 can be input into a key derivation function 402a in order for responder 101x to derive mutually shared key $k1$ 314a. A responder 101x can then conduct a decryption step 320 as depicted in Figure 4a below. Responder can input the received ciphertext 315a into a symmetric ciphering algorithm 405b and key $k1$ 314a using parameter 199f in order to read the plaintext 403,

15 which could comprise the initiator nonce 317a and the initiator capabilities. In exemplary embodiments, the initiator 102' can have capabilities of a configurator, and responder 101x can select responder capabilities or an enrollee in a step 320. Steps 319 and steps 320 can also comprise the first four paragraphs of section 6.2.3 of DPP specification version 1.0.

Responder 101x can then conduct a second responder key exchange step 314b, where the

20 details for a key exchange step 314b are depicted and described in connection with Figure 4b below. In summary, the initiator ephemeral public key Pi 102a received in message 122 can be input into an ECDH key exchange algorithm 401a along with the responder ephemeral private key pr 101f derived in a step 318 above in order to output a secret shared key N 411. The value for shared secret key N 411 can be recorded and used in the derivation of a key ke 327a in a third key exchange step 321

25 below. For a step 314b, the value for the shared secret key N 411 can be input into a key derivation function 402b in order to derive a mutually shared symmetric key $k2$ 314c.

Responder 101x can conduct a third responder key exchange step 321, where the details for a key exchange step 321 are depicted in Figure 4c below. In summary, both (i) the derived responder ephemeral private key pr 101f from a step 318 with the recorded responder bootstrap private key br 101b, and (ii) the recorded initiator bootstrap public key Bi 104a can be input into an ECDH key

30 exchange algorithm 401c in order to output a secret shared key L 412. For embodiments where a DPP responder authentication 141 is conducted by responder 101x without the use of an initiator bootstrap public key Bi 104a, then calculation of a secret shared key L 412 can be optionally omitted in a step 321. Step 321 for responder 101x can then comprise a key derivation function 402c using

35 the mutually derived secret shared keys M 410, N 411, and optionally L 412 in order to mutually

derive secret shared key ke 327. An example for a key exchange step 321 is described in point 2 of section 6.2.3 of DPP specification version 1.0 on page 48.

Responder 101x can then conduct an encryption step 322, where the details for an encryption step 322 are depicted and described in connection with Figure 4c below. Step 322 can also comprise responder 101x creating or calculating a value R-auth 404, where R-auth 404 can comprise a secure hash value of the initiator nonce 317a, the second random number R-nonce 318b from a step 318 above, and public PKI keys. The value for R-auth 404 in a step 322 can comprise the value in point 3 of section 6.2.3 of DPP specification version 1.0. An encryption step 322 can comprise responder 101x encrypting the value for R-auth 404 using the mutually derived key key 327 from the third key exchange step 321 above, where the output of encryption step 322 can comprise ciphertext 322a.

Responder 101x can then conduct an encryption step 315b, where the details for an encryption step 315b are depicted and described in connection with Figure 4b below. Ciphertext 322a from an encryption step 322 in the paragraph above can also be encrypted again for an encryption step 315b. The symmetric ciphering key for encryption step 315b can comprise symmetric key k2 314c. Encryption step 315b, encryption step 322, and decryption step 320 can use a common symmetric ciphering algorithm 405a, which could comprise the Advanced Encryption Standard with Synthetic Initialization Vectors (AES-SIV) (and deciphering algorithm) also with a common set of symmetric ciphering parameters 199f. Symmetric ciphering parameters 199f can also specify the use of a block chaining mode such as cipher block chaining (CBC), counter mode (CTR), or Galois/Counter mode (GCM) and other possibilities exist as well. In addition, symmetric ciphering parameters 199f could specify a mode for message authentication, which could comprise a CMAC mode as specified in NIST publication SP-800-38B. In exemplary embodiments, a symmetric ciphering algorithm 405a can comprise the AES-SIV algorithm as specified in IETF RFC 5297. The output from an encryption step 315b can be ciphertext 315c, as depicted in Figure 4b below. As depicted in Figure 3a, the collection of steps for responder 101x comprising steps 318, 319, etc. through step 315b can comprise a step 325.

Responder 101x can then send an DPP Authentication Response message 123, where message 123 can include hash values of the bootstrap public keys H(Br 101a), H(Bi 104a), the responder ephemeral public key Pr 101e, and the ciphertext 315c. The ciphertext 315c can have encrypted values for (i) R-nonce 318b, (ii) I-nonce 317a, (iii) R-capabilities 318c, and (iv) ciphertext 322a. The ciphertext 315c can be encrypted with mutually derived symmetric encryption key k2 314c as shown for an encryption step 315b in Figure 4b. Ciphertext 322a can be encrypted with the mutually derived symmetric key ke 327a and can include a plaintext value for R-auth 404. As described above, R-auth 404 can comprise a hash value over an initiator nonce 317a received and the responder nonce 318b.

As depicted in Figure 3a, initiator 102' can receive DPP Authentication Response message 123. In exemplary embodiments, initiator 102' can conduct a step 323 to process the received responder ephemeral public key Pr 101e, such as evaluating that the received key in message 123 uses the parameters 199b specified in cryptographic parameters 199a. In addition, a step 323 can
5 comprise initiator 102' can conduct public key validation step to ensure that received responder ephemeral public key Pr 101e is properly formatted and does not contain weak or compromising cryptographic data. For example, a step 323 could comprise initiator 102' performing the steps for an ECC Full Public-Key Validation Routine in section 5.6.2.3.2 of FIPS publication SP 800-56A (revision 2). Alternatively, step 323 can comprise initiator 102' performing the steps ECC Partial
10 Public-Key Validation Routine in section 5.6.2.3.3 of the same FIPS publication. Other example steps within a public key validation step 323 can comprise (i) verifying the public key is not at the "point of infinity", and (ii) verifying the coordinates of the point for the public key are in the range $[0, p-1]$, where p is the prime defining the finite field. Other possibilities exist as well for evaluating and validating a received public key is cryptographically secure in a public key validation step 323,
15 without departing from the scope of the present invention. As contemplated in the present invention, an initiator 102', a DPP server 103, or a responder 101x can conduct a public key validation step 323 each time a public key is received.

In exemplary embodiments, initiator 102' can send DPP server 103 a message 324 using secure session 302b. Message 324 can include the ID-token.device 206 (or alternatively ID.device
20 101b), and the ciphertext 315c. Hash values for public bootstrap keys can be useful over link 192 between responder 101x and initiator 102' in order to confirm identities, but the hash values can be substituted with an identity for device 101 in message 324 between initiator 102' and DPP server 103. The ciphertext 315c in message 324 can be the same as ciphertext 315c in message 123 in Figure 3a. In other embodiments, the hash values for the public bootstrap keys of initiator 102 and
25 responder 101x can be included in a message 324, thereby increasing the message size by at least 64 bytes.

DPP server 103 can receive message 324 from initiator 102' and conduct a series of steps in order to process the message and reply to the initiator 102'. In exemplary embodiments, DPP server 103 can also perform a public key validation step 323 on the received responder ephemeral public
30 key Pr 101e. Although a step 323 by DPP server 103 may appear redundant with a step 323 by initiator 102', DPP server 103 may trust its own computation for a step 323 with a higher level than relying solely on a step 323 performed by initiator 102'. Initiator 102' as operating in Figure 3a could also comprise a device with a lower level of security than DP server 103. After a step 323, DPP server 103 can record the validated, received responder ephemeral public key Pr 101e in a
35 server database 103a, as depicted in Figure 3b below.

At step 319a, DPP server 103 can conduct a second initiator key exchange step 319a, where DPP server 103 can input the received, validated responder ephemeral public key Pr 101e and the derived initiator ephemeral private key pi 102b into an ECDH key exchange algorithm 401b in order to calculate a shared secret key N 411. The second initiator key exchange step 319a is also depicted and described in connection with Figure 4b below. Shared secret key N 411 can also be used by DPP server 103 in a step 327 below. DPP server 103 can input shared secret key N 411 into a key derivation function 402b in order to mutually derived the symmetric encryption key $k2$ 314c.

At step 320b, DPP server 103 can use the derived key $k2$ 314c from a step 319a in order to conduct a decryption step 320b on received ciphertext 315c. A decryption step 320b on received ciphertext 315c using key $k2$ 314c is depicted and described in connection with Figure 4b below. DPP server 103 can read the plaintext 403b values from decryption step 320b of ciphertext 315c. The plaintext values 403b can include the initiator nonce 317a from a message 317 and message 122 above. The plaintext values 403b can also include a second ciphertext 322a (e.g. ciphertext 322a can be inside ciphertext 315c). DPP server 103 verifies that the initiator nonce 317a received in message 123 and decrypted in step 320b is equal to the initiator nonce 317a value sent in message 122 above. If the two initiator nonce 317a values are not equal, then DPP server 103 can abort the DPP procedure and subsequent steps.

At step 327, DPP server 103 can conduct a third initiator key exchange step 327, where DPP server 103 can determine the secret shared key L 412 by inputting (i) the sum or combination of the responder bootstrap public key Br 101a and the responder ephemeral public key Pr 101e, and (ii) the initiator bootstrap private key bi 104b into an ECDH key exchange algorithm 401c. A third server key exchange step 327 is depicted and described in connection with Figure 4c below. The shared secret key L 412 along with shared secret keys M 410 and N 411 can be used with a key derivation function 402c in order to output the mutually derived symmetric ciphering key ke 327a.

At step 328, DPP server 103 can conduct a decryption step 328 on the ciphertext 322a using the key ke 327a. A decryption step 328 is depicted and described in connection with Figure 4c below. DPP server 103 can use the decryption step 328 in order to read the plaintext value of R-auth 404. R-auth 404 can comprise a hash 199e value over the combination of initiator nonce 317a and the responder nonce 318b. At step 328, DPP server 103 can then also internally calculate the same value for R-auth 404, and if the internally calculated value and the received, decrypted value are equal, then the responder 101x can be consider authenticated with DPP server 103. The calculation of R-auth 404 by DPP server 103 can be according to section 6.2.3 of DPP specification version 1.0.

DPP server 103 can then conduct a step 322b, where step 322b can comprise calculating an initiator authentication value I-auth 404a. Initiator authentication value 404a can be similar to responder authentication value R-auth 404, except the values of initiator nonce 317a can be sequenced before a responder nonce 318b inside a hash 199e calculation over the two nonces. A

DPP server 103 can then encrypt the initiator authentication value 404a using an encryption step 322 using key ke 327a, as depicted in Figure 4c below. In other words, an encryption step 322 in Figure 4c depicts the encryption of R-auth 404, and encryption step 322b can comprise a DPP server 103 encrypting the initiator authentication value I-auth 404a using the key ke 327a derived in a step 327.

5 DPP server 103 can then send initiator 102' a message 329, where message 329 can include (i) an identity for device 101, such as ID-token.device 206, although ID.device 101i could be used instead, and (ii) the encrypted initiator authentication value I-auth 404a. Initiator 102' can receive message 329 over secure session 302b. In an exemplary embodiment, the value MAC.R 101d could also be within message 329, such that initiator 102' can use the MAC.R 101d to send data in
10 message 329 to responder 101x using the MAC.R 101d. Initiator 102' can determine which of a plurality of responders 101x is directed towards based on the value of ID-token.device 206 or MAC.R 101d. Initiator 102' can previously record the hash 199e values for the initiator and responder bootstrap public keys. Initiator 102 can take the recorded hash 199e values for the initiator and responder bootstrap public keys and append to them the encrypted value for I-auth 404a
15 received in message 329. Initiator 102' can send responder 101x a DPP authentication confirm message 123a, as depicted in Figure 3a. The data and steps to determine and calculate an DPP authentication confirm message 123a can be according to section 6.2.4 of DPP specification version 1.0.

Note that although the DPP authentication confirm message 123a can be equivalent to the
20 DPP authentication confirm message from section 6.2.4 of DPP specification version 1.0, an important difference can be the operation of initiator 102' in Figure 3a versus an initiator as contemplated by DPP specification version 1.0. In the embodiment of the present invention as depicted in Figure 3a, initiator 102' can record neither the responder bootstrap public key Br 101a, nor the initiator bootstrap private key bi 104b. So, using the technology depicted in Figure 3a, an
25 initiator 102' can (i) send and receive messages that are fully compatible with a DPP protocol 191, but (ii) DPP server 103 can record keys 101a and 104b. In other words, a responder 101x may not "know" that an initiator operates as either (i) an initiator 102* from Figure 1b or Figure 1c, or an initiator 102' from Figure 3a since the messages transmitted and received by the responder 101x can be fully compatible with a DPP 191.

30 Responder 101x can receive the DPP authentication confirm message 123a and process the message. Responder 101x can listen for either (i) a broadcast message with the hash 199e values for the bootstrap public keys 101a and 104a, or (ii) for a unicast message to MAC.R 101d. Responder 101x can first conduct a decryption step 328b, where decryption step 328b can use the same steps as a decryption step 328 as depicted in Figure 4c below, with a difference being decryption step 328b
35 operates on the initiator authentication value I-auth 404a instead of the R-auth 404 value.

In step 331, responder 101x can compare the decrypted value for I-auth 404a with a calculated value for I-auth 404a and if the two values match then data from the initiator 102' can be considered by responder 101x to be authenticated. After an authentication step 331, in step 332, responder 101x can take on a roll of enrollee, such that responder 101x can prepare to send a configuration attribute 330. Configuration attribute 330 can be used to specify that responder 101x can take the roll of enrollee and is expecting a set of network credentials 109 from initiator 102'. A configuration attribute 330 can be formatted and contain data fields according to section 6.3.4 of DPP specification version 1.0. For the exemplary embodiment depicted in Figure 1a, device 101 with responder 101x would prefer a role of WiFi client (or "sta") for a configuration attribute 330, but other possibilities exist as well. For the exemplary embodiment depicted in Figure 7 below a responder 101x operating within an access point 105 would prefer a role of WiFi access point (or "ap") for a configuration attribute 330.

In step 332, responder 101x can also generate a third responder random number, for an enrollee nonce E-nonce 332a. E-nonce 332a and other nonce values in Figure 3a such as the initiator nonce I-nonce 317a and responder nonce R-nonce 318b can be processed using nonce parameters 199g. Responder 101x can then conduct an encryption step 322c, where encryption step 322c can comprise (i) an encryption step 322 using key ke 327a, and (ii) the plaintext to be encrypted can be the E-nonce 332a and the configuration attribute 330. The output of an encryption step 322c can be a ciphertext 322d. As depicted in Figure 3a, the collection of steps comprising 328b through 322c for responder 101x can comprise a step 339, where a responder 101x processes both message 123a and message 123b in a step 339. Responder 101x can then send or transmit to initiator 102' a DPP Configuration Request message 123b, where the configuration request 123b can include the ciphertext 322d, where the plaintext in ciphertext 322d can include E-nonce 332a and the configuration object 330. DPP Configuration Request message 123b can be processed by responder 101x according to section 6.3.3 of DPP specification version 1.0.

Initiator 102' can receive DPP Configuration Request message 123b via connection 192 and using DPP application 102g. As contemplated herein, a DPP application 102g can perform the steps for initiator 102' depicted in Figure 3a, such as transmitting/receiving DPP 191 messages using a first MAC.I 102n for connection 192 and transmitting/receiving messages with DPP server 103 using MAC.N 102q via access network 112. Although not depicted in Figure 3a, DPP application 102g could also perform the depicted steps 323 to validate the received responder ephemeral public key Pr 101e from message 123b. For embodiments where initiator 102' uses a WAN radio 102h to connect with access network 112 and DPP server 103, then DPP application 102g could also use a different identity than MAC.N 102q to identify the physical interface of WAN radio 102h, such as an IMEI, device identity ID.device 102i, or other possibilities exist as well. For other embodiments where initiator 102' uses network access point 105 for an access network 112 to connect with DPP

server 103, then MAC.I 102n could also potentially be the same as MAC.N 102q. Other possibilities exist as well without departing from the scope of the present invention for an identity used by initiator 102' to send messages to DPP server 103.

Initiator 102' can send DPP server 103 a message 334 via secure connection 302b, where message 334 can include the ciphertext 322d. Although not depicted in Figure 3a, message 334 can also include an identity for device 101, such as, but not limited to, ID-token.device 206 or ID.device 101b, since both DPP server 103 and initiator 102' can communicate with a plurality of different devices 101 over time, and DPP server 103 may prefer to know which ciphertext 322d is associated with which device 101 in order to process the ciphertext 322d. As depicted in Figure 3a, message 334 can include the encrypted values for E-nonce 332a and configuration attribute 330 in ciphertext 322d.

DPP server 103 can receive message 334 and process the message 334. DPP server 103 can conduct a decryption step 328c using decryption step 328 depicted in Figure 4c below. A difference between decryption step 328 and decryption step 328c is that decryption step 328c uses the ciphertext 322d as input into the symmetric decryption algorithm 405b. DPP server 103 can use the previously calculated key ke 327a in order to read the plaintext within ciphertext 322d. DPP server 103 can record the plaintext E-nonce 332a and configuration attribute 330.

In step 335, DPP server 103 can process the received configuration attribute 330. For the exemplary embodiment depicted in Figure 1a, configuration attribute 330 can specify that device 101 with responder 101x may have a role of WiFi client. In exemplary embodiments, DPP server 103 can previously select a primary access network 308aa in a step 308a above, using the received networks available list 304c from initiator 102'. Also, in a step 308b above, DPP server 103 could receive or record the set of network credentials 109 associated with the selected primary access network 308aa for device 101 using device identity ID-token.device 206 (or ID.device 101i). In step 335, DPP server can process the set of network credentials 109 for primary access network 308aa into a configuration object. The configuration object with network credentials 109 can comprise a connector or configuration object as specified in section 6.3.5 and 6.3.6 of DPP specification version 1.0. The configuration object with network credentials 109 can be signed using the recorded initiator ephemeral private key pi 102b, corresponding to public key Pi 102a sent in message 317 and 122 above. Or, in another embodiment the configuration object with network credentials 109 can be signed using the recorded initiator bootstrap private key bi 104b. The steps to create and verify a signature with ECC PKI keys can be according to FIPS publication 186-4, and other possibilities exist as well.

In a step 335, DPP server 103 can process a configuration object for network credentials 109, and the configuration object can include an SSID value of SSID.network-AP 109a, a pairwise master key value of PMK.network-AP 109b, and configuration values of config.network-AP 109c. The

value of PMK.network-AP 109b can also include a value for PMKID (e.g. PMK identifier), such that PMK.network-AP 109b is used by each of a client and an access point with the PMKID value. The configuration values of config.network-AP 109c in a configuration object could also specify required supporting data for operation of device 101 with an access point 105 after a configuration step 106, such as an operating class 130a, channel list 130b, an authentication mode for device 101 to operate, which could be PSK, PMK, SAE, EAP, also with any required supporting data and/or identities. The configuration values of config.network-AP 109c in a configuration object could also include a passphrase or pre-shared key values of PSK for legacy WPA2 networks. As contemplated herein, a set of network credentials 109 can include either (i) network credentials 109 for a selected primary access network 308aa, or (ii) network credentials 109 for both a selected primary access network 308aa and a selected backup access network 308ab. Or, in another exemplary embodiment, although a single configuration object of network credentials 109 is depicted in Figure 3a, DPP server 103 could process (i) a first set of network credentials 109 for selected primary access network 308aa into a first configuration object and then (ii) a second set of network access credentials 109 for selected backup access network 308ab into a second configuration object.

At step 322e, DPP server 103 can conduct an encryption step 322e using an encryption step 322 depicted and described in connection with Figure 4c below. Encryption step 322e can be performed using key ke 327a for device 101. For encryption step 322e, the plaintext values of (i) the E-nonce 332a and (ii) the configuration object comprising network credentials 109 can be input into a symmetric encryption algorithm 405a in order to generate a ciphertext 322f. DPP server 103 can then send initiator 102' a message 338, where message 338 includes the ciphertext 322f. Message 338 can also optionally include an identity for device 101 such as ID-token.device 206 or ID.device 101i. Message 338 can be sent via secure connection 302b, as depicted in Figure 3a.

Initiator 102' using DPP application 102g can receive message 338 via secure connection 302b. Initiator 102' can use an optional identity for device 101 in a message 338 in order to determine which of a potential plurality of devices 101 associated with ciphertext 322f in order to forward the data. Initiator 102' can then send responder 101x a DPP configuration response message 124 via connection 192, where message 124 can include the ciphertext 322f, and the ciphertext 322f can include the encrypted E-nonce 332a and (i) the encrypted, signed network credentials 109.

Responder 101x can receive message 124 from initiator 102' with ciphertext 322f. Responder 101x can conduct a decryption step 328d using decryption step 328 depicted in Figure 4c below. A difference between decryption step 328 and decryption step 328d is that decryption step 328d uses the ciphertext 322f as input into the symmetric decryption algorithm 405b. Responder 101x can use the previously calculated key ke 327a (from key exchange step 321) in order to read the plaintext within ciphertext 322f. Responder 101x can confirm that plaintext in ciphertext 322f

includes the previously send E-nonce 332a. Responder 101x can read (i) the plaintext values for credentials 109. Note that credentials 109 may be previously signed by DPP server 103 using a private key initiator ephemeral private key pi 102b for the initiator 102' (or initiator bootstrap private key bi 104b), where the private key corresponds to public key Pi 102a. The responder could also verify a signature for the credentials 109 using the ephemeral public key for the initiator Pi 102a recorded with the responder 101x, such as the initiator ephemeral public key Pi 102a or the initiator bootstrap public key Bi 104a.

After reading the network credentials 109 and optionally verifying a signature using a recorded public key 102a or 104a, device 101 can complete the configuration step 106 by loading the network credentials 109 and begin operating a WiFi radio 101h with the network credentials 109, as depicted in Figure 1a. In this manner, device 101 can obtain connectivity to network access point 105. In exemplary embodiments, the initiator 102' can conduct the initiator restore step 129 depicted and described in connection with Figure 1d above, such that initiator 102' can be returned to the previous state where WiFi radio 102w operates with a user configuration 131, thus minimizing impacts of the prior initiator configuration step 127 for an initiator user 102u.

Figure 3b

Figure 3b is an illustration of an exemplary server database, with tables for a server database recording exemplary data, in accordance with exemplary embodiments. A server database 103a depicted and described above in connection with Figure 1a and Figure 3a can record data for a DPP server 103 to work or operate with an initiator 102' in order for the initiator 102' to conduct a device provisioning protocol 191 with a device 101 operating a responder 101x. A server database 103a could record a plurality of tables for a plurality of devices 101 and a plurality of initiators 102', including the depicted device credentials table 340. Other possibilities exist as well for the organization, tables, and recorded data within a device database 103a without departing from the scope of the present invention. Data within server database 103a could be encrypted using a symmetric key. As depicted in Figure 1a, a DPP server 103 operating in a network 107 could operate and record a server database 103a.

Server database 103a can record a device identity of ID-token.device 206, a responder MAC address MAC. R 101d, an initiator MAC address MAC. I 102n (or MAC.I 102q), a mode of operation for the DPP server 103 and initiator 102' comprising initiator mode 306a, a selected set of Cryptographic Parameters 199a, an Initiator Ephemeral Public Key 102a, an Initiator Ephemeral Private Key 102b, a Responder Ephemeral Public Key 101e, Bootstrap keys 399, an optional PKEX Key 198, a Shared Secret k1 314a, a Shared Secret k2 314c, and a Shared Secret ke 327a. For exemplary embodiments, not all depicted values may be recorded in a server database 103a, as

shown by the depicted value of “-”, which can mean that the value is not recorded in a server database 103a. An initiator 102’ could record the value depicted as “-” for a server database 103a in Figure 3b for some initiator modes 306a, as depicted. For example, with the row in server database 103a depicted with a mode “Fig 5a”, the initiator 102’ could derive, record, and operate on the initiator ephemeral private key pr 102b and initiator ephemeral public key Pr 102a, and DPP server 103 may not need or record the keys. Similarly, with the row in server database 103a depicted in Figure 3b as a mode of “Fig 5b”, the initiator 102’ could derive shared secret keys k2 314c and ke 327e, while server database 103a could derive and record key k1 314a. Note that in alternative exemplary embodiments, a server database 103a could record the values M 410, N 411, and L 412 respectively instead of key k1 314a, k2 314c, and/or ke 327e, and either DPP server 103 or initiator 102’ could use the key derivation function 402a, 402b, and/or 402c, respectively in order to convert the values into keys. In exemplary embodiments, the time that a device shared secret key or a derived ephemeral PKI key, or a bootstrap key is recorded in a server database 103a can be minimized to the minimum time necessary to conduct a DPP 191. For example, shared secret keys and ephemeral PKI keys recorded in a server database 103a can be deleted once a configuration step 106 has been completed for a device 101, such as at the time DPP server 103 receives a report 605aa from device 101 in Figure 6a below.

Initiator mode 306a can specify an operating mode for initiator 102 and DPP server 103, such that a sequence of steps and transmitting/receiving PKI and/or shared secret keys can be specified by the initiator mode 306a. An initiator mode 306a could also be referred to as a “server mode” 306a, since the mode specifies the operation of a DPP server 103. Or, initiator mode 306a could be referred to as a “mode 306a”, since both steps for initiator 102’ and DPP server 103 are affected by the mode. Figure 3a above also described a first, second, third, and fourth operating initiator modes 306a. A first mode is depicted as “Fig. 3a” in Figure 3b and can comprise the mode 306a depicted in Figure 3a. Note that all depicted ephemeral PKI keys and bootstrap PKI keys, as well as all depicted shared secret keys are recorded in a server database 103a for a device 101 (except responder private keys 101b and 101f) for the first mode. In other words, when conducting a DPP 191, initiator 102’ and DPP server 103 would have all the depicted keys recorded in a server database 103a for the first mode. This first mode for an initiator mode 306a can be preferred for embodiments where (i) initiator 102’ may have a lower level of trust with network 107, or (ii) initiator 102’ could have a lower level of processing resources, or (iii) an owner for device 101 or a network 107 may prefer that responder bootstrap public key Br 101a is not shared externally from network 107, or (iv) regulatory requirements where device 101 could comprise a high value or sensitive piece of equipment. Other possibilities exist as well for the reasons different modes 306a may be used in a server database 103a to specify the steps for an initiator 102’ and DPP server 103 to conduct a DPP 191 with device 101.

A second mode is depicted as “Fig. 5a” in Figure 3b, where the steps and message flows in Figure 5a for initiator 102’ and DPP server 103 can comprise the second mode. Note that all relevant bootstrap PKI keys 399 are recorded in a server database 103a for the mode depicted in Figure 5a. The mode 306a can also have (i) the responder ephemeral public key Pr 101e and (ii) the shared secret key k1 314a recorded in a server database 103a. Note that initiator ephemeral PKI keys and secret shared keys can optionally not be recorded in a server database 103a for the second mode depicted in Figure 5a. This exemplary embodiment may be preferred for cases where (i) initiator 102’ can handle more of the cryptographic operations (e.g. reducing DPP server 103 processing load), but (ii) a network 107 and initiator user 102u could prefer that the initiator bootstrap private key bi 104b is not transmitted to or recorded in initiator 102’. An exemplary reason the second mode of “Fig 5a” could be preferred may be embodiments where multiple different devices 101 record the same initiator bootstrap public key Bi 104a, and for these embodiments a network 107 may prefer to keep the initiator bootstrap private key bi 104b recorded in a server database 103a. This embodiment was also discussed above with the devices designated with “shared keys” in Figure 2b. As depicted in Figure 3b, devices 101 with the same recorded value or number for an initiator bootstrap public key Bi 104a could also record different responder bootstrap public keys Br 101a.

A third mode 306a is depicted as “Fig. 5b” in Figure 3b, where the steps and message flows in Figure 5b for initiator 102’ and DPP server 103 can comprise the third mode. Note that all relevant bootstrap PKI keys 399 are recorded in a server database 103a for the third mode depicted in Figure 5b. The third mode 306a can be a mix between the first mode of “Fig. 3a” and the second mode of “Fig 5a”. In other words, the third mode 306a of “Fig 5b” can have the same data recorded for operation of the mode in “Fig 5a”, except the addition of recording the initiator ephemeral private key pi 102b and secret key k1 314a in server database 103a. Thus some additional computation resources are shifted from the initiator 102’ to DPP server 103 from using a mode of “Fig. 5a” to the mode of “Fig. 5b”. One reason to use the third mode of Fig. 5b could be a desire or requirement by DPP server 103 or initiator user 102u or a device 101 owner to have the cryptographic operations of a DPP 191 more evenly shared between an initiator 102’ and a DPP server 103. In other words, a mode depicted in Fig. 3a could comprise a mode that is mostly on DPP server 103, a mode depicted in Figure 5a could comprise a mode that is mostly on initiator 102’, and a mode depicted in Figure 5b could comprise a balanced mode where a mix of the keys and cryptographic operations are more evenly shared between DPP server 103 and initiator 102’.

A fourth mode 306a in Figure 3b can comprise initiator 102’ operating as an initiator 102* depicted in Figure 1b and Figure 1c, and in this case initiator 102* can record the keys shown for the fourth mode 306a in Figure 3b. A DPP server 103 can still provide the bootstrap keys 399 shown to initiator 102* in this fourth mode of operation, or the initiator 102* could obtain the bootstrap keys 399 from other means, such as “out of band” messages 121 and 126 in Figure 1c. In addition, an

initiator 102* could obtain the bootstrap keys “in band” (e.g. via WiFi link 192) using PKEX key 198.

Bootstrap keys 399 in a server database 103a can record the bootstrap keys depicted in Figure 3b, which could comprise a responder bootstrap public key Br 101a, an initiator bootstrap public key Bi 104a, and an initiator bootstrap private key 104b. For embodiments where a device 101 can support multiple different sets of cryptographic parameters 199a, a server database 103a can record the selected set of cryptographic parameters 199a for communication between an initiator 102' and device 101 with responder 101x. The keys for “bootstrap keys 399” can be according to the selected set of cryptographic parameters 199a. In other words, a device database 104 in Figure 2b above can record all versions of the bootstrap PKI keys for all supported cryptographic parameters 199a for a device 101. However, since a set of cryptographic parameters 199a will need to be selected for a DPP 191 (e.g. one of the responder bootstrap public keys Br 101a selected), then the bootstrap keys 399 for that selected set of cryptographic parameters 199a can be recorded in a server database 103a.

A server database 103a can also record a preshared PKEX key 198, if PKEX is optionally supported by responder 101x. PKEX key 198 can be used by initiator 102' to securely transfer bootstrap public keys in band (i.e. via connection 192 in Figure 1a and Figure 3a). The use of a PKEX protocol to exchange the public keys is described in section 5.6 of DPP specification version 1.0. For embodiments where initiator 102' uses PKEX key 198 with responder 101x, the initiator bootstrap private key bi 104b preferably continues to reside within a network 107 and can be in a server database 103a. PKEX key 198 can be used with responder 101x MAC address MAC.R 101d. Embodiments can omit the use of PKEX key 198, as depicted with the value of “NA” for the key in Figure 3b. Derived, shared secret keys with responder 101x can also be recorded in a server database 103a. The derived shared secret keys could comprise key k1 314a, key k2 314c, and key ke 327e. The steps to derive and use these keys are depicted and described below in Figure 4a, Figure 4b, and Figure 4c, respectively. As discussed above, some keys may reside in server database 103a, and other keys may reside within an initiator 102' (depicted with a “-”), depending on the mode 306a.

The present invention contemplates other implementations of a device provisioning protocol besides the DPP specification version 1.0. These other implementations could have different mutually derived shared secret keys than those depicted in Figure 3b, and the different mutually derived shared secret keys could be recorded in a server database as well. For example, a different key derivation function 402 could be used to derive a secret shared key than that specified in DPP specification version 1.0. In addition, other or different PKI keys for a responder 101x and/or initiator 102' could be recorded in a server database 103a. For example, the DPP specification version 1.0 could be extended to explicitly include a DPP server 103 public and private key, in order

to extend support with a DPP server 103, and in these embodiments the DPP server 103 public and private keys could be included in a server database 103a. Other possibilities exist as well for (i) a server database 103a to record PKI keys and shared secret keys to support (ii) a device provisioning protocol without departing from the scope of the present invention.

5 A server database 103a could also record a device credentials table 340. Device credentials table 340 can record a device identity ID-token.device 206 (or ID.device 101i), a selected primary network 308aa, a selected backup network 308ab, a set of primary network credentials 109, and a set of backup network credentials 109. The selected primary network 308aa and selected backup network 308ab could be determined in a step 308a in Figure 3a by DPP server 103 or network 107.
10 The set of primary network credentials 109 can comprise the set of credentials (plus configuration) for a device 101 to use with the selected primary network 308aa. The set of backup network credentials 109 can comprise the set of credentials (plus configuration) for a device 101 to use with the selected backup network 308ab. The credentials 109 in a device credentials table 340 could be obtained from during a step 308b as depicted and described above in connection with Figure 3b.

15 For example, with the first device depicted in a device credentials table 340, device 101 could record a selected primary network 308aa for network access point 105 in Figure 1a. The value recorded for 308aa could be SSID.network-AP 109a. The credentials in device table for the first device could comprise the credentials 109 depicted in Figure 1a. The depicted designation of “109a-1” means the first instance of a value of SSID.network-AP 109a. The device in the second row can
20 use the same value for SSID.network-AP 109a. The device in the last depicted row can use a different value for a selected primary network 308aa of “109a-3” which could be the third distinct value in device credentials table 340. Values of “304c-1” for selected primary network 308aa can mean a first value from a networks identity list 304c from Figure 3a, values of “304c-2” can mean a second value from a networks available list 304c, etc.

25 For embodiments where a network other than network 107 is selected for device 101 to be configured with credentials, the credentials in a device credentials table 340 are depicted as “from ...”, where “...” can be the depicted, selected network from a networks available list 304c. In exemplary embodiments, the credentials for networks external to network 107 can be encrypted by the external network using Elgamal encryption and either (i) the responder bootstrap public key Br
30 101a or (ii) the responder ephemeral public key Pr 101e. The DPP server 103 could send either (i) the responder bootstrap public key Br 101a or (ii) the responder ephemeral public key Pr 101e to the external network for the external network to encrypt the credentials using the public key Br 101a or Pr 101e and an Elgamal asymmetric encryption algorithm, which was also described above. Thus, the credentials are stored as ciphertext in a server database 103a and only the device 101 with
35 responder 101x could feasibly decrypt the ciphertext in order to read the credentials for networks external to network 107. Additional details for receiving credentials from an external network, and

recording the encrypted credentials in a server database 103a are depicted and described below in connection with Figure 5a, depicted as credentials 109'.

Note the present invention contemplates that (a) a selected access network 112 for device 101 in a step 308a above in Figure 3a by a DPP server 103 or network 107 could (b) utilize wireless networking technology other than WiFi or 802.11 based technology. For example, surrounding wireless networks for device 101 could use exemplary potential use of 5G wireless technology or LPWAN networking technology depicted in a networks available list 304c in Figure 3a above for potential access networks 112 for device 101. Or, a WiFi access point could also be selected by a DPP server 103 or a network 107 for device 101 in step 308a above in Figure 3a using the networks available list 304c.

For the exemplary embodiment depicted in Figure 3a and also Figure 1a above, the exemplary network access point 105 can be selected in a step 308a using SSID.network-AP 109a from Figure 1a, although different networks in a networks available list 304c could be selected instead, based on exemplary criteria listed for a step 308a above in Figure 3a. In addition, a network access point 105 in Figure 1a can comprise a form of an access network 112 as contemplated herein, such that device 101 obtains access to IP network 113 or the globally routable Internet via an exemplary network access point 105. A selected access network 112 for device 101 in a step 308a does not need to belong to network 107 from Figure 1a, and could belong to a different network (so long as network 107 could obtain or point to credentials 109 for the access network 112 in order to conduct the configuration step 106 from Figure 1a). Further, a step 308a could comprise the selection of both a primary access network 112 for device 101 and a second, backup access network 112 for device 101. The use of a primary and backup network for device 101 can provide increased reliability for the operation of device 101, such that if the primary access network 112 is not available then device 101 could use the selected backup network in order for device 101 to maintain connectivity to an IP network 113.

Figure 4a

Figure 4a is a flow chart illustrating exemplary steps for conducting a key exchange using PKI keys in order to derive a shared secret key, and for using the derived shared secret key to encrypt and decrypt data, in accordance with exemplary embodiments. Exemplary steps for an initiator 102' and a responder 101x to mutually derive keys can comprise (i) a key exchange 314 step for DPP server 103 or initiator 102' (depending on the mode 306a) to derive a symmetric encryption key k1 314a and (ii) a key exchange 319 step for a responder 101x to derive the same symmetric encryption key k1 314a. Exemplary steps for a responder 101x to decrypt data and initiator 102' for DPP server 103 (depending on a mode 306a) to encrypt data can comprise (a) an

5 encryption step 315 for initiator 102' or DPP server 103 (depending on a mode 306a) to utilize the symmetric encryption key k1 314a to convert plaintext to ciphertext, and (b) a decryption 320 step for responder 101x to utilize the symmetric encryption key k1 314a to convert the ciphertext received from initiator 102' into plaintext. The use of the steps for a key exchange 314, a key exchange 319, encryption 315, and decryption 320 were also depicted and described in connection with Figure 3a above. Additional detail regarding the use of these steps will be provided herein with Figure 4a. Further, although the steps are depicted specifically for the use of particular keys and plaintext/ciphertext combinations in Figure 4a, the steps illustrated can be used with different keys and plaintext/ciphertext combinations. For example, Figure 4b below depicts the same functionality used by the nodes as in Figure 4a, but with (i) different PKI keys input to mutually derive a different symmetric encryption key, (ii) a different key derivation function and (iii) different plaintext encrypted and different ciphertext decrypted.

10 The processes and operations, described below with respect to all of the logic flow diagrams and flow charts may include the manipulation of signals by a processor and the maintenance of these signals within data structures resident in one or more memory storage devices. For the purposes of this discussion, a process can be generally conceived to be a sequence of computer-executed steps leading to a desired result.

15 These steps usually require physical manipulations of physical quantities. Usually, though not necessarily, these quantities take the form of electrical, magnetic, or optical signals capable of being stored, transferred, combined, compared, or otherwise manipulated. It is convention for those skilled in the art to refer to representations of these signals as bits, bytes, words, information, elements, symbols, characters, numbers, points, data, entries, objects, images, files, or the like. It should be kept in mind, however, that these and similar terms are associated with appropriate physical quantities for computer operations, and that these terms are merely conventional labels applied to physical quantities that exist within and during operation of the computer.

20 It should also be understood that manipulations within the computer are often referred to in terms such as listing, creating, adding, calculating, comparing, moving, receiving, determining, configuring, identifying, populating, loading, performing, executing, storing etc. that are often associated with manual operations performed by a human operator. The operations described herein can be machine operations performed in conjunction with various input provided by a human operator or user that interacts with the device, wherein one function of the device can be a computer.

25 In addition, it should be understood that the programs, processes, methods, etc. described herein are not related or limited to any particular computer or apparatus. Rather, various types of general purpose machines may be used with the following process in accordance with the teachings described herein.

The present invention may comprise a computer program or hardware or a combination thereof which embodies the functions described herein and illustrated in the appended flow charts. However, it should be apparent that there could be many different ways of implementing the invention in computer programming or hardware design, and the invention should not be construed
5 as limited to any one set of computer program instructions.

Further, a skilled programmer would be able to write such a computer program or identify the appropriate hardware circuits to implement the disclosed invention without difficulty based on the flow charts and associated description in the application text, for example. Therefore, disclosure of a particular set of program code instructions or detailed hardware devices is not considered
10 necessary for an adequate understanding of how to make and use the invention. The inventive functionality of the claimed computer implemented processes will be explained in more detail in the following description in conjunction with the remaining Figures illustrating other process flows.

Further, certain steps in the processes or process flow described in all of the logic flow diagrams below must naturally precede others for the present invention to function as described.
15 However, the present invention is not limited to the order of the steps described if such order or sequence does not alter the functionality of the present invention. That is, it is recognized that some steps may be performed before, after, or in parallel other steps without departing from the scope and spirit of the present invention.

The processes, operations, and steps performed by the hardware and software described in
20 this document usually include the manipulation of signals by a CPU or remote server and the maintenance of these signals within data structures resident in one or more of the local or remote memory storage devices. Such data structures impose a physical organization upon the collection of data stored within a memory storage device and represent specific electrical or magnetic elements. These symbolic representations are the means used by those skilled in the art of computer
25 programming and computer construction to most effectively convey teachings and discoveries to others skilled in the art.

A key exchange 314 step for initiator 102' or DPP server 103 (depending on a mode 306a) to derive a symmetric encryption key 314a can utilize a selected set of cryptographic parameters 199a as depicted and described in connection with Figure 2b above. As depicted in Figure 4a, a key
30 exchange algorithm 401a can receive input both of a responder bootstrap public key Br 101a and an initiator ephemeral private key pi 102b. The key exchange algorithm 401a could comprise a Diffie Hellman key exchange (DH), an Elliptic Curve Diffie Hellman key exchange (ECDH), and other possibilities exist as well without departing from the scope of the present invention. A key exchange algorithm 401a can support either PKI keys based on elliptic curves or RSA algorithms, although
35 support of elliptic curves may be preferred in some exemplary embodiments due to their shorter key lengths and lower processing requirements.

A summary of ECDH as a key exchange algorithm 401a is included in the Wikipedia article titled "Elliptic Curve Diffie-Hellman" from March 9, 2018, which is herein incorporated by reference. An exemplary embodiment of key exchange algorithm 401a could comprise a "One-Pass Diffie-Hellman, C(1, 1, ECC CDH)" algorithm as described in section 6.2.2.2 on page 99 of the
5 National Institute of Standards and Technology (NIST) document "NIST SP 800-56A Revision 3, Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography" from April, 2018 which is hereby incorporated by reference its entirety. Other key exchange algorithms in NIST SP 800-56A could be utilized as well for a key exchange algorithm 401a in Figure 4a, Figure 4b, and Figure 4c without departing from the scope of the present
10 invention. Example calculations for an ECDH key exchange for a key exchange algorithm 401a are shown on pages 88 and 89 of DPP specification version 1.0, where a selected set of parameters 199a can use the ECC curve P-256 with PKI key lengths of 256 bits.

Other algorithms to derive a shared symmetric encryption key using public keys and private keys may also be utilized in a key exchange algorithm 401a (and 401b in Figure 4b and 401c in
15 Figure 4c), such as, but not limited to, the American National Standards Institute (ANSI) standard X-9.63. Cryptographic parameters 199a can also include information for using a key exchange algorithm 401a and a key derivation function 402a in order to derive a commonly shared symmetric encryption key k1 314a. As contemplated herein, the terms "selected set of cryptographic parameters 199a" and "cryptographic parameters 199a", and "parameters 199a" can be equivalent.
20 Parameters 199a input into a key exchange algorithm 401a can include a time-to-live for a key k1 314a that is derived, a supported point formats extension, where the supported point formats extension could comprise uncompressed, compressed prime, or "compressed char2" formats, as specified in ANSI X-9.62. In other words, an ECC keys input into a key exchange algorithm 401a may have several different formats and a set of parameters 199a can be useful to specify the format.

As depicted in Figure 4a, the output of a key exchange algorithm 401a, such as an ECDH
25 key exchange, can comprise a secret shared key M 410. The secret shared key M 410 can be input into both (i) a first key derivation function 402a in order to calculate the key k1 314a and (ii) a second key derivation function 402c in order to calculate a symmetric encryption key ke 327a as depicted in Figure 4c below. Key derivation function 402a could use a hash algorithm 199e from parameters 199a, where an input into the hash algorithm 199e can comprise the value M 410. As
30 contemplated herein, a secret shared key can also be referred to as a value, since the secret shared key can be a numeric value or a number in the form of a sequence of digits. In DPP specification version 1.0, a key derivation function 402a to determine key k1 314a is depicted on the fourth line of page 48. Other possibilities exist as well for a key derivation function 402a without departing from
35 the scope of the present invention. For example, with other embodiments besides those within DPP specification version 1.0, the output of a key derivation function 402a can be both a symmetric

encryption key k1 314a and a message authentication code (MAC) key, where a MAC key can function to verify message integrity of ciphertexts generated with the symmetric encryption key 314a.

As depicted in Figure 4a, an encryption 315 step could be utilized with the derived symmetric encryption key k1 314a in order to convert plaintext 403b into ciphertext 315a. An encryption 315 step can utilize a symmetric encryption algorithm 405a, where symmetric encryption algorithm 405a could comprise an algorithm according to the Advanced Encryption Standard (AES), Blowfish, and Triple Data Encryption Standard (3DES), and other possibilities exist as well. As depicted in Figure 4a, a symmetric encryption algorithm 405a can comprise the AES-SIV algorithm as specified in IETF RFC 5297. Parameters 199f could specify settings for a symmetric ciphering algorithm 405a as well, such as key length and block cipher modes such as, but not limited to, ECB, CBC, OFB, and CFB. Parameters 199f may also specify an authenticated encryption mode such as CCM (NIST SP800-38C), GCM (NIST SP800-38D), and other examples exist as well.

Symmetric encryption algorithm 405a can accept input of (i) plaintext 403b, (ii) symmetric encryption key k1 314a, (iii) parameters 199f, and (iv) optionally an initialization vector (not shown in Fig. 4a), and output ciphertext 315a. Although not depicted in Figure 4a, an encryption 405a step can output a message authentication code (MAC), where the MAC can be calculated with a MAC key from a key derivation function 402a. In exemplary embodiments the use of a MAC and MAC key, as well as an initialization vector can be optionally omitted. The exemplary listed values for plaintext and ciphertext in step 315 were described in connection with depicted step 315 in Figure 3a.

Key exchange 319 step for responder 101x depicted in Figure 4a can correspond to key exchange 314 step by an initiator 102' or DPP server 103 (depending on a mode 306a). Key exchange 319 step can comprise inputting or using the initiator ephemeral Public key Pi 102a (from message 122) and the responder bootstrap private key br 101b into key exchange algorithm 401a, which can comprise the same or equivalent key exchange algorithm 401a depicted and described in connection with key exchange step 314 described above. Other elements or algorithms within a key exchange step 319 can be equivalent to a key exchange step 314 above, including the use of shared parameters 199a. In this manner, initiator 102' or DPP server 103 (depending on a mode 306a) and responder 101x can securely derive the same symmetric encryption key k1 314a as depicted in Figure 4a.

A decryption 320 step allows responder 101x to convert the ciphertext 315a received in a message 122 from Figure 3a into plaintext 403a. Decryption 320 step can utilize a symmetric decryption algorithm 405b, which could comprise the same algorithm used in symmetric encryption algorithm 405a except the algorithm being used for decryption instead of encryption. For example, if symmetric encryption algorithm 405a in Figure 4a comprises an AES-SIV encryption, then

symmetric decryption algorithm 405b can comprise an AES-SIV decryption. Note that the same values are input into symmetric decryption algorithm 405b as symmetric encryption algorithm 405a, such as symmetric encryption key k1 314a and parameters 199f in order to convert ciphertext 315a back into plaintext 403a. Responder 101x can the read and process plaintext 403a after a decryption
 5 320 step.

For embodiments where a MAC code is received with ciphertext 315a, responder 101x can also verify the MAC code using a MAC key from a key derivation function 402a in a step 319 in order to verify the integrity of ciphertext 315a. Note that AES-SIV and similar symmetric ciphering algorithms can have a message integrity check built into the algorithm, and thus a separate MAC
 10 code and MAC key can be optionally omitted in some exemplary embodiments. In exemplary embodiments, the successful decryption of a ciphertext into a plaintext using decryption algorithms 405b supports mutual authentication between the two nodes, since successful decryption can only take place when the two nodes record the depicted private keys used for the depicted key exchanges 314 and 319.

Figure 4b

Figure 4b is a flow chart illustrating exemplary steps for conducting a key exchange using PKI keys in order to derive a shared secret key, and for using the derived shared secret key to
 20 encrypt and decrypt data, in accordance with exemplary embodiments. Exemplary steps for (i) an initiator 102' or a DPP server 103 (depending on the mode 306a) and (ii) a responder 102x to mutually derive keys can comprise (a) a key exchange 319a step for DPP server 103 or initiator 102' (depending on the mode 306a) to derive a symmetric encryption key k2 314c and (b) a key exchange 314b step for a responder 101x to derive the same symmetric encryption key k2 314c. Exemplary
 25 steps for a responder 101x to encrypt data and initiator 102' to decrypt data can comprise (i) an encryption 315b step for responder 101x to utilize the symmetric encryption key k2 314c to convert plaintext to ciphertext, and (ii) a decryption 320b step for either initiator 102' or DPP server 103 (depending on the mode 306a) to utilize the symmetric encryption key k2 314c to convert the ciphertext received from responder 101x into plaintext. The use of the steps for a key exchange
 30 314a, a key exchange 319a, encryption 315b, and decryption 320b were also depicted and described in connection with Figure 3a above.

Note that steps in Figure 4b and the steps in Figure 4a can share some algorithms and values, and the descriptions for the algorithms and values in Figure 4a can be applicable for Figure 4b. For example, a same key exchange algorithm 401a depicted and described in Figure 4a can also be used
 35 in Figure 4b. The key exchange algorithm 401a can comprise an ECDH key exchange. The set of parameters 199a depicted and described in Figure 4a can also be used in Figure 4b. Further, a

symmetric encryption algorithm 405a and a symmetric decryption algorithm 405b can be the same in Figure 4a and Figure 4b. A set of parameters 199f for a symmetric encryption algorithm 405a and a symmetric decryption algorithm 405b can be the same in Figure 4a and Figure 4b

A responder 101x can conduct a key exchange step 314b. In a key exchange step 314b, the initiator ephemeral public key P_i 102a received in message 122 from Figure 3a can be input into an ECDH key exchange algorithm 401a along with the responder ephemeral private key pr 101f derived in a step 318 in Figure 3a in order to output a secret shared key N 411. The value for shared secret key N 411 can be recorded and also used in the derivation of a key ke 327a in a key exchange step 321 below in Figure 4c. For a step 314b, the value for the shared secret key N 411 can be input into a key derivation function 402b in order to derive a mutually shared symmetric key k_2 314c. In exemplary embodiments which support DPP specification version 1.0, a key derivation function 402b can comprise the key derivation function for key k_2 in point 2 of section 6.2.3 of DPP specification version 1.0.

Either an initiator 102' or a DPP server 103 can conduct a key exchange step 319a, depending on a mode 306a from a server database 103a. At step 319a, a received, validated responder ephemeral public key Pr 101e and a derived initiator ephemeral private key pi 102b can be input into an ECDH key exchange algorithm 401a in order to calculate a shared secret key N 411. Shared secret key N 411 can also be used by initiator 102' or DPP server 103 (depending on a mode 306a) in a step 327 below. DPP server 103 can input shared secret key N 411 into a key derivation function 402b in order to mutually derived the symmetric encryption key k_2 314c. In exemplary embodiments which support DPP specification version 1.0, a key derivation function 402b can comprise the key derivation function for key k_2 in section 6.2.4 of DPP specification version 1.0.

Responder 101x can conduct an encryption step 315b, where the use for an encryption step 315b are depicted and described in connection with Figure 3a above. Ciphertext 322a can comprise a ciphertext from an encryption step 322 below in Figure 4c. In other words, a responder 101x can conduct the encryption step 322 below and also key exchange step 321 below in Figure 4c before conducting the encryption step 315b. As depicted in Figure 4b, the ciphertext 322a can also be encrypted again for an encryption step 315b. The symmetric ciphering key for encryption step 315b can comprise symmetric key k_2 314c from a key exchange step 314b in the paragraph above. The output from an encryption step 315b can be ciphertext 315c, as depicted in Figure 4b.

A decryption 320b step can be performed by either an initiator 102' or a DPP server 103, depending on a mode 306a recorded in a server database 103a as depicted in Figure 3b. A decryption 320b step converts the ciphertext 315c received in a message 123 from Figure 3a into plaintext 403b. Decryption 320b step can utilize a symmetric decryption algorithm 405b, which could comprise the same algorithm used in symmetric encryption algorithm 405a except the algorithm being used for decryption instead of encryption. Note that the same values are input into

symmetric decryption algorithm 405b as symmetric encryption algorithm 405a, such as symmetric encryption key k2 314c and parameters 199f in order to convert ciphertext 315c back into plaintext 403b. Initiator 102' or a DPP server 103 (depending on a mode 306a) can read and process plaintext 403b after a decryption 320b step. In exemplary embodiments, the successful decryption of a ciphertext into a plaintext using decryption algorithms 405b supports mutual authentication between the two nodes, since successful decryption can only take place when the two nodes record the depicted private keys used for the depicted key exchange.

Figure 4c

Figure 4c is a flow chart illustrating exemplary steps for conducting a key exchange using PKI keys in order to derive a shared secret key, and for using the derived shared secret key to encrypt and decrypt data, in accordance with exemplary embodiments. Exemplary steps for (i) an initiator 102' or a DPP server 103 (depending on the mode 306a) and (ii) a responder 102x to mutually derive keys can comprise (a) a key exchange 327 step for DPP server 103 or initiator 102' (depending on the mode 306a) to derive a symmetric encryption key ke 327a and (b) a key exchange 321 step for a responder 101x to derive the same symmetric encryption key ke 327a. Note that in some exemplary embodiments, such a mode 306a depicted and described in connection with Figure 5a and Figure 5b below, (i) a key exchange algorithm 401c can be processed by a DPP server 103 a key derivation function 402c can be processed by an initiator 102'. In other words, for some embodiments the key exchange 327 step can be shared or distributed between a DPP server 103 and an initiator 102'.

Exemplary steps in Figure 4c for a responder 101x to encrypt data and initiator 102' to decrypt data can comprise (i) an encryption 322 step for responder 101x to utilize the symmetric encryption key ke 327a to convert plaintext to ciphertext, and (ii) a decryption 328 step for either initiator 102' or DPP server 103 (depending on the mode 306a) to utilize the symmetric encryption key ke 327a to convert the ciphertext received from responder 101x into plaintext. The use of the steps for a key exchange 321, a key exchange 327, encryption 322, and decryption 328 were also depicted and described in connection with Figure 3a above.

Note that steps in Figure 4c and the steps in Figure 4a can share some algorithms and values, and the descriptions for the algorithms and values in Figure 4a can be applicable for Figure 4c. For example, the key exchange algorithm 401c can comprise an ECDH key exchange. The set of parameters 199a depicted and described in Figure 4a can also be used in Figure 4c. Further, a symmetric encryption algorithm 405a and a symmetric decryption algorithm 405b can be the same in

Figure 4a and Figure 4c. A set of parameters 199f for a symmetric encryption algorithm 405a and a symmetric decryption algorithm 405b can be the same in Figure 4a and Figure 4c.

A responder 101x can conduct a key exchange step 321. In a key exchange step 321, (i) the initiator bootstrap public key B_i 104a received in message 126 from Figure 1c can be input into an ECDH key exchange algorithm 401c along with (ii) both the responder ephemeral private key pr 101f and the responder bootstrap private key br 101b. An ECDH key exchange algorithm 401c can use $(pr\ 101f + br\ 101b)$ modulo q , as specified in DPP specification version 1.0 in point 2 of section 6.2.3, so algorithm 401c for responder 101x can be slightly different than algorithm 401a for responder 101x above in Figure 4a and Figure 4b. The output of key exchange algorithm 401c can be a shared secret key L 412.

The responder 101x can then conduct a key derivation function 402c in order to calculate a shared secret key ke 327a. Note that for a key derivation function 402c, the input includes all of key M 410 from a step 314, key N from a step 314b, and the derived key L 412 from step 321. As contemplated herein, a key derivation function 402a, 402b, and 402c can also use other data mutually shared between (i) responder 101x and (ii) initiator 102' or DPP server 103 (depending on a mode 306a). In exemplary embodiments which support DPP specification version 1.0, a key derivation function 402c can comprise the key derivation function for key ke 327a in point 2 of section 6.2.3 of DPP specification version 1.0.

Either an initiator 102' or a DPP server 103 can conduct a key exchange step 327, depending on a mode 306a from a server database 103a and transmitted to DPP server 103 in a message 311 as depicted in Figure 3a. At step 327, (i) a combination of a recorded responder bootstrap public key Br 101a and received responder ephemeral public key Pr 101e, and (ii) the recorded initiator bootstrap private key bi 104b can be input into an ECDH key exchange algorithm 401c in order to calculate a shared secret key L 412. Shared secret key L 412 can be input into a key derivation function 402c, along with shared secret key M 410 from a step 319 and shared secret key N 411 from a step 319a in order to mutually derive the symmetric encryption key ke 327a. In exemplary embodiments which support DPP specification version 1.0, a key derivation function 402c can comprise the key derivation function for key ke 327e in section 6.2.4 of DPP specification version 1.0.

Responder 101x can conduct an encryption step 322, where the use for an encryption step 322 are depicted and described in connection with Figure 3a above and also Figure 4b above. Plaintext 403c in a step 322 can comprise the responder authentication value 404, where value R_{auth} 404 can comprise a secure hash value with input including the initiator nonce 317a, the second random number R_{nonce} from a step 318 above in Figure 3a, and public PKI keys. The value for R_{auth} 404 in a step 322 (also described with step 322 in Figure 3a) can comprise the value in point 3 of section 6.2.3 of DPP specification version 1.0. The symmetric ciphering key for encryption step

322 can comprise symmetric key ke 327a from a key exchange step 321. The output from an encryption step 322 can be ciphertext 322a, as depicted in Figure 4c. Note that ciphertext 322a in a step 322 is also encrypted again in a step 315b above in Figure 4b. In other words, a responder 101x can conduct the encryption step 322 herein for Figure 4c (along with the key exchange 321 herein in Figure 4b) before conducting the encryption step 315b in Figure 4b above.

A decryption step 328 can be performed by either an initiator 102' or a DPP server 103, depending on a mode 306a recorded in a server database 103a as depicted in Figure 3b. A decryption 328 step converts the ciphertext 322a received in a message 123 (after decrypting a first layer of encryption of ciphertext 315b in a step 320b) from Figure 3a into plaintext 403c. Decryption 328 step can utilize a symmetric decryption algorithm 405b, which could comprise the same algorithm used in symmetric encryption algorithm 405a except the algorithm being used for decryption instead of encryption. Note that the same values are input into symmetric decryption algorithm 405b as symmetric encryption algorithm 405a in Figure 4c, such as symmetric encryption key ke 327a and parameters 199f in order to convert ciphertext 322a back into plaintext 403c. Initiator 102' or a DPP server 103 (depending on a mode 306a) can then read and process plaintext 403c after a decryption 328 step. In exemplary embodiments, the successful decryption of a ciphertext into a plaintext using decryption algorithms 405b supports mutual authentication between the two nodes, since successful decryption of ciphertext 322a can only take place when the two nodes record the depicted private keys used for the depicted key exchange.

As depicted and described in connection with Figure 3a, a responder 101x can also conduct both an encryption step 322 and a decryption step 328. The steps for responder 101x to conduct a decryption step 328 can comprise step 328b and step 328d as depicted and described in Figure 3a. When responder 101x conducts decryption step 328 using symmetric encryption key ke 327a, the ciphertext and plaintext will comprise different values than those depicted in Figure 4c. For example, in Figure 3a a responder 101x can use a decryption step 328b (e.g. step 328 but with different ciphertext than that depicted in Figure 4c) in order to decrypt the encrypted initiator authentication value 404a into a plaintext authentication value 404a. Also, in Figure 3a a responder 101x can use a decryption step 328d (e.g. step 328 but with different ciphertext 322f than that depicted in Figure 4c) in order to decrypt the encrypted network credentials 109.

Note that in some exemplary embodiments, the "plaintext" network credentials 109' (for the purposes of a step 328d in Figure 5a) can comprise network credentials 109' that can also be encrypted with an asymmetric ciphering algorithm such as Elgamal using a step 336 in Figure 5a below. In other words, network credentials 109' can comprise credentials that are "double encrypted" with both (i) a symmetric key 327a (which can be decrypted by responder 101x using a decryption step 328d, where decryption step 328d uses the depicted decryption 328 in Figure 4c), and (ii) an asymmetric ciphering algorithm such as Elgamal from a step 336, (which could be

decrypted by responder 101x using a step 337 from Figure 5a below and either (i) responder ephemeral private key pr 101f or (ii) responder bootstrap private key br 101b).

As depicted and described in connection with Figure 3a, a DPP server 103 or initiator 102' (in Figure 5a) can conduct both an encryption step 322 and a decryption step 328. The steps for a DPP server 103 or initiator 102' (depending on a mode 306a) to conduct an encryption step 322 can comprise step 322b and step 322e as depicted and described in Figure 3a. When DPP server 103 or initiator 102' conducts encryption step 322b or 322e using symmetric encryption key ke 327a, the ciphertext and plaintext will comprise different values than those depicted in Figure 4c. For example, in Figure 3a a DPP server 103 can use an encryption step 322b (e.g. step 322 but with different plaintext than that depicted in Figure 4c) in order to encrypt the initiator authentication value 404a. In Figure 5a, an initiator 102' can use an encryption step 322b (e.g. step 322 but with different plaintext than that depicted in Figure 4c) in order to encrypt the initiator authentication value 404a. In Figure 3a a DPP server 103 can use an encryption step 322e (e.g. step 322 but with different plaintext than that depicted in Figure 4c) in order to encrypt the network credentials 109. In Figure 5a, an initiator 102' can use an encryption step 322e (e.g. step 322 but with different plaintext than that depicted in Figure 4c) in order to encrypt the network credentials 109'.

Figure 5a

Figure 5a is a simplified message flow diagram illustrating an exemplary system with exemplary data sent and received by a device database, a DPP server, an initiator proxy, and a responder, in accordance with exemplary embodiments. Before starting steps and message flows depicted in Figure 5a, the initiator, DPP server, and device database depicted for system 500 in Figure 5a may previously complete exemplary message flows and steps depicted for a step 301 in Figure 3a above. System 500 can include a device database 104, DPP server 103, initiator proxy 102', and responder 101x operating in device 101. Initiator proxy 102' can communicate with DPP server 103 via IP network 113 using an access network 112, also as depicted in Figure 3a above. As contemplated herein, an "initiator proxy 102'" depicted in Figure 5a may also be referred to as initiator 102', and an initiator 102' and initiator proxy 102' may be equivalent. A difference between system 500 in Figure 5a and system 300 in Figure 3a can be that (i) responder bootstrap public key Br 101a, (ii) initiator bootstrap public key Bi 104a, (iii) initiator ephemeral public key Pi 102a, and (iv) initiator ephemeral private key pi 102b are recorded and operated within an initiator 102'. In exemplary embodiments depicted in Figure 5a the initiator bootstrap private key bi 104b can remain recorded in a DPP server 103 and not exposed to initiator 102'. The exemplary embodiment depicted in Figure 5a can comprise a second mode 306a of operation for initiator 102' and DPP server 103, as depicted in Figure 3b above.

At step 502, DPP server 103 can select a mode 306a for both initiator 102' and DPP server 103. Note that the mode 306a selected is not normally observable to a responder 101x, since the full and complete set of DPP 191 messages between responder 101x and initiator 102' can be transmitted and received. In other words, (a) the different distribution of PKI and shared secret keys in a server database 103a and initiator 102' for a first mode 306a in Figure 3a and a second mode 306a in Figure 5a (b) would not normally be observable to a responder 101x. The recording of PKI and shared secret keys in a server database is depicted in a server database 103a in Figure 3b, where the value "--" can mean that the key is recorded and/or operated by initiator 102'. Step 502 can also comprise DPP server 103 selecting an initiator bootstrap public key Bi 104a and a responder bootstrap public key Br 101a for device 101 using ID-token device 206.

In exemplary embodiments, a responder 101x for a device 101 could record a plurality of responder bootstrap public keys Br 101a, as depicted in Figure 2a above. Reasons could include the use of different keys Br 101a for different purposes, such as a first key Br 101a for a responder authentication 141 and a second key Br 101a for a mutual authentication 142. Or, a third key Br 101a could be recorded or mapped in a tag 101c, while a fourth key Br 101a in a responder 101x could only be recorded in a network 107. Responder 101x can designate the use of the fourth key Br 101a with a higher level of trust, with the following exemplary results (A) and (B). For (A), device 101 could provide additional privileges for a configuration step 106 for an initiator 102' or initiator 102* that uses the second key Br 101a (e.g. the first key Br 101a could be easier for potentially any initiator 102' to obtain if the first key Br 101a is recorded in a QR code for device 101). As one example of additional privileges, a set of credentials 109 delivered to device 101 using the fourth key Br 101a could allow the installation of a new certificate authority root certificate for device 101. As one example of normal or other privileges, a set of credentials 109 delivered to device 101 using the third key Br 101a could deny the installation of a new certificate authority root certificate for device 101 as well.

In summary for a step 502, (i) a device 101 with responder 101x can use a plurality of responder bootstrap public keys Br 101a (including for reasons other than simply use of different sets of cryptographic parameters 199a for the different keys), and (ii) a DPP server 103 can select the responder bootstrap public key Br 101a in a step 502. Also, although a step 502 is depicted in Figure 5a, a step 502 could be used in other Figures and embodiments herein, include embodiments depicted in Figure 3a and Figure 5b. In addition, for a step 502, DPP server 103 can use an initiator bootstrap keys sequence number 197 in order to select the appropriate initiator bootstrap public key Bi 104a. For example, if a pair of initiator bootstrap keys had been deprecated (such as depicted in Figure 6a below), then an updated pair of initiator bootstrap keys should be selected using a device table 104d and sequence number 197 (also selected with device ID 206).

DPP server 103 can then send initiator 102' a message 503, where message 503 can include an identity 206 for device 101 (since initiator 102' may communicate with a plurality of responders 101x over time), the selected responder bootstrap public key Br 101a from a step 502, the selected initiator bootstrap public key Bi 104a, and optionally a value for the MAC address of the responder 101x comprising MAC.R 101d. Message 503 can be sent via the secure session 302b setup during a step 301 depicted in Figure 3a above.

Initiator 102' can receive message 503 and process the data received in a step 504. A step 504 can include validating the received public keys from a message 503 using public key validation steps from FIPS publication SP 800-56A (revision 2) and subsequent and related versions. Initiator 102' can use a step 504 in order to verify the public PKI keys received in a message 503 are consistent or comply with a selected set of cryptographic parameters 199a. At step 505 the initiator 102' can determine the initiator capabilities for conducting a DPP 191 with device 101 and responder 101x. In the exemplary embodiment depicted in Figure 5a, initiator 102' can select and use the capabilities of the role as a configurator. In other embodiments, an initiator 102' could select and use the capabilities of the role an enrollee for a step 504. In addition, an initiator 102' can generate a random number for an initiator nonce I-nonce 317a in a step 504.

Initiator 102' can then conduct a step 313 to derive the initiator ephemeral public key Pi 102a and the initiator ephemeral private key pi 102b. A step 313 was previously depicted and described above in Figure 3a. Although DPP server 103 conducted the step 313 in Figure 3a, an initiator 102' can also conduct the step 313 in system 500 in order to derive the initiator ephemeral PKI key pairs. Further, subsequent steps depicted in Figure 5a that were previously depicted as conducted by a DPP server 103 can be conducted by an initiator 102' using the embodiment depicted in Figure 5a. For example, depicted subsequent steps such as steps 314, 315, 319a, 320b, 322b, 328, etc. can be conducted, performed, or processed by initiator 102' using a DPP application 102g in an exemplary system 500, which could also represent a different mode 306a than system 300 in Figure 3a.

Initiator 102' in step 314 can use (i) the derived initiator ephemeral private key pi 102b from step 313, and (ii) the received responder bootstrap public key Br 101a from a message 503 to conduct the first initiator key exchange step 314, which is also depicted in Figure 4a above. The output or results from a step 314 can comprise a mutually derived shared secret key k1 with responder 101x. An intermediate step within the key exchange step 314 can include calculating the value M 410 (which could also be a secret shared key). Initiator 102' in step 315 can then conduct the encryption step 315 depicted in Figure 4a, and also described in Figure 3a. The plaintext input into a symmetric encryption algorithm 405a can comprise the initiator nonce 317a and initiator capabilities from a step 504 above. The shared secret key k1 314a can be used with symmetric

ciphering algorithm 405a in a step 315 from Figure 4a. The ciphertext output from an encryption step 315 can comprise ciphertext 315a.

At step 316, initiator 102' can calculate secure hash values for initiator bootstrap public key Bi 104a and responder bootstrap public key Br 101a. The selected hash 199e could be received by initiator 102' in a set of selected parameters 199a from a message 311 in a step 301. In exemplary embodiments, initiator 102' records the full set of cryptographic parameters 199, and a message to select a set of cryptographic parameters 199a could then specify a hash 199e to use with the public keys. For example, a selected set of cryptographic parameters 199a from message 311 in a step 301 could be the value "A", which would then be used by initiator 102' to determine that hash 199e in a set of cryptographic parameters 199 would be "SHA-256", as depicted in Figure 2b. Or, a message 311 in a step 301 could specify the full set of selected cryptographic parameters 199, such as listing out the individual values of key length 199b, curve name 199c, hash 199e, etc. The hash values calculated by initiator 102' in a step 316 in Figure 5a are depicted in Figure 5a as "H(Bi)" and "H(Br)", respectively.

Initiator 102' can then conduct a step 316a to send responder 101x a message 122, where message 122 can include the hash values for (i) the initiator bootstrap public key Bi 104a and (ii) the responder bootstrap public key Br 101a, the derived initiator ephemeral public key Pi 102a, and ciphertext 315a, where ciphertext 315a is encrypted with key k1 and includes an initiator nonce 317a and the initiator capabilities. Sending DPP authentication request message 122 was also depicted and described in connection with Figure 1b, Figure 1c, and Figure 3a above.

In Figure 5a, responder 101x can receive message 122 and conduct the series of steps 325. The series of steps 325 for responder 101x were depicted and described in connection with Figure 3a above, and steps 325 can comprise the steps 318, 319, 320, 314b, 321, 322, and 315b. Responder 101x can process the message 122 using steps 325 and generate a DPP authentication response 123. The steps 325 for a responder 101x to process the received message 122 and generate the response message 123 can comprise section 6.2.2 of DPP specification version 1.0. Other possibilities exist as well without departing from the scope of the present invention. In general, responder 101x can use the recorded bootstrap private keys and the derived ephemeral private keys in order to conduct ECDH key exchanges. Responder 101x can also (i) derive a responder nonce 318b, (ii) decrypt ciphertext in message 122 and (iii) encrypt ciphertext in message 123 using derived secret shared keys from the ECDH key exchanges. Responder 101x can send initiator 102' the DPP authentication response 123 after conducting the steps 325.

Initiator 102' can receive the DPP authentication response 123 and process the message. Using broadcast messages over connection 192, initiator 102' can identify the message from responder 101x based on the hash values for the bootstrap public keys in response 123. For embodiments where message 122 and response 123 are sent as unicast messages, initiator 102' can

identify the message from responder 101x based on the MAC.R 101d in the response 123. Initiator 102' can then extract and record responder ephemeral public key Pr 101e (depicted for a response 123 in Figure 3a). Initiator 102' can also a key verification step on the received key Pr 101e according to FIPS publication SP 800-56A (revision 2).

5 Initiator 102' can then send DPP server 103 a message 506 using secure connection 302b via access network 112, where message 506 can include (i) an identity for device 101 such as ID-token.device 206 (or ID.device 101i), and (ii) the responder ephemeral public key Pr 101e. In an exemplary embodiment a message 506 can include the MAC address MAC.R 101d to also identify device 101 and the associated key Pr 101e. DPP server 103 can receive the message 506 and record
10 the responder ephemeral public key Pr 101e along with the device identity in a server database 103a, as depicted in Figure 3b above.

DPP server 103 can then conduct an ECDH key exchange step 401c as depicted in Figure 4c in order to determine a secret shared key L 412. As depicted in Figure 5a and also Figure 4c, a step 401c can comprise DPP server 103 inputting (i) the recorded initiator bootstrap private key bi 104b
15 and (ii) a combination of the received responder ephemeral public key Pr 101e and the recorded responder bootstrap public key Br 101a into key exchange algorithm 401c in order to (iii) calculate the secret shared key L 412. Secret shared key L 412 can also comprise a secret shared key L according to section 6.2.4 of DPP specification version 1.0.

As depicted in Figure 5a, an initiator 102' can then conduct a second key exchange step
20 319a, where the second key exchange step 319a was depicted and described above in Figure 4b and also Figure 3a. Although a DPP server 103 was depicted as conducting a key exchange step 319a in Figure 3a, for the embodiments depicted in Figure 5a, an initiator 102' can conduct the key exchange step 319a. At step 319a, initiator 102' can input the received, validated responder ephemeral public key Pr 101e and the derived initiator ephemeral private key pi 102b into an ECDH key exchange
25 algorithm 401b in order to calculate a shared secret key N 411. Shared secret key N 411 can also be used by initiator 102' in a key derivation function 402c step below in Figure 5a. For a key exchange step 319a in Figure 5a, initiator 102' can input shared secret key N 411 into a key derivation function 402b in order to mutually derived the symmetric encryption key k2 314c. Note that secret shared key N 411 can also be used by initiator 102' for later use in a key derivation function step
30 402c, as depicted in Figure 5a.

As depicted in Figure 5a, initiator 102' can then conduct a decryption step 320b on received ciphertext 315c using the derived symmetric encryption key k2 314c from a step 319a above by initiator 102' in Figure 5a. The use and operation of a decryption step 320b was also depicted and described above in connection with Figure 4b and Figure 3a. Initiator 102' can read the plaintext
35 403b values from decryption step 320b. The plaintext values 403b can include the initiator nonce 317a from a message 122 above. The plaintext values 403b can also include a second ciphertext

322a (e.g. ciphertext 322a can be inside ciphertext 315c). Initiator 102' verifies that the initiator nonce 317a received in message 123 and decrypted in step 320b is equal to the initiator nonce 317a value sent in message 122 above. If the two initiator nonce 317a values are not equal, then initiator 102' can abort the DPP procedure and subsequent steps.

Initiator 102' can then receive a message 510 from DPP server 103, where message 510 can include an identity for device 101 such as ID-token.device 206 and the value L 412, where the value or secret shared key L 412 was calculated by DPP server 103 in a step 401c as depicted in Figure 5a. The value L 412 can be calculated by DPP server 103 using the initiator bootstrap private key bi 104b, and thus initiator 102' does not need to record or receive the initiator bootstrap private key bi 104b for embodiments depicted in Figure 5a.

Initiator 102' can then conduct a key derivation function step 402c using (i) the received value L 412 from message 510, (ii) the calculated shared secret key M 410 from a step 314, and (iii) the calculated shared secret key N 411 from a step 319a. A key derivation function step 402c is depicted and described in connection with Figure 4c above. The output from a key derivation function step 402c conducted by initiator 102' can be a shared secret key ke 327a, as depicted in Figure 4c.

At step 328, initiator 102' can conduct a decryption step 328 on the ciphertext 322a using the key ke 327a from the previous step 402c. A decryption step 328 is depicted and described in connection with Figure 4c above. Initiator 102' can use the decryption step 328 in order to read the plaintext value of R-auth 404 from the responder 101x in message 123. R-auth can comprise the value R-auth 404 from Figure 3a and Figure 4c and also comprise a hash 199e value that includes the combination of at least the initiator nonce 317a and the responder nonce 318b. At step 328, initiator 102' can then also internally calculate the same value for R-auth 404, and if the internally calculated value for R-auth 404 and the received, decrypted value for R-auth 404 are equal, then the responder 101x can be considered authenticated. The calculation of R-auth 404 by initiator 102' in a step 328 can be according to section 6.2.3 of DPP specification version 1.0.

Initiator 102' can then conduct a step 322b, where step 322b can comprise calculating an initiator authentication value I-auth 404a. Initiator authentication value 404a can be similar to responder authentication value R-auth 404, except the values of initiator nonce 317a can be sequenced before a responder nonce 318b inside a hash 199e calculation over the two nonces. Initiator 102' can then encrypt the initiator authentication value 404a using an encryption step 322 using key ke 327a, as depicted in Figure 4c above. In other words, an encryption step 322 in Figure 4c depicts the encryption of R-auth 404, and encryption step 322b can comprise initiator 102' encrypting the initiator authentication value I-auth 404a using the key ke 327a derived in a step 402c.

Initiator 102' can previously record the hash 199e values for the initiator and responder bootstrap public keys. Initiator 102 can take the recorded hash 199e values for the initiator and responder bootstrap public keys and append to them the encrypted value for I-auth 404a calculated by initiator 102 in step 322b. Initiator 102' can send responder 101x a DPP authentication confirm message 123a, as depicted in Figure 5a. The data and steps to determine and calculate a DPP authentication confirm message 123a can be according to section 6.2.4 of DPP specification version 1.0. As depicted in Figure 5a, the collection of steps and messages after a message 506 through transmitting a message 123a can collectively comprise a step 512.

Note that although the DPP authentication confirm message 123a can be equivalent to the DPP authentication confirm message from section 6.2.4 of DPP specification version 1.0, an important difference can be the operation of initiator 102' in Figure 5a versus an initiator as contemplated by DPP specification version 1.0. In the present invention as depicted in Figure 5a, initiator 102' does not need to record the initiator bootstrap private key bi 104b. So, using the technology depicted in Figure 5a, an initiator 102' can (i) send and receive messages that are fully compatible with a DPP protocol 191, but (ii) DPP server 103 can record key bi 104b. In other words, a responder 101x may not "know" that an initiator operates as either (i) an initiator 102* from Figure 1b or Figure 1c, or an initiator 102' from Figure 5a since the messages transmitted and received by the responder 101x can be fully compatible with a DPP 191.

The technology depicted in Figure 5a can enhance security since an initiator bootstrap private key bi 104b does not need to be shared with an initiator 102'. The initiator 102' can also not feasibly be able to determine the initiator bootstrap private key bi 104b using value L 412. By only recording the initiator bootstrap private key bi 104b in DPP server 103 (or also a device database 104), then a common, shared initiator bootstrap public key Bi 104a can be more securely shared across multiple devices 101, as contemplated by key table 104k in Figure 2b (with the depiction of "shared keys"). In other words, a manufacturer of device 101 could record a common initiator bootstrap public key Bi 104a across a plurality of devices 101, since a network 107 could share the initiator bootstrap public key Bi 104a with a device manufacturer of devices 101 (or a device distributor) before the devices are deployed near network access points 105. In other embodiments besides those depicted in Figure 5a, the initiator bootstrap private key bi 104b can be shared with initiator 102'.

In Figure 5a, responder 101x can receive message 123a and conduct the series of steps 339. The series of steps 339 for responder 101x were depicted and described in connection with Figure 3a above, and steps 339 can comprise the steps 328b, 331, 332, and 322c. Responder 101x can process the message 123a using steps 339 and generate a DPP configuration request 123b. In summary, responder 101x can decrypt the ciphertext in message 123a, where the ciphertext can include the initiator authentication value 404a. Responder 101x can also internally calculate a value for the

initiator authentication value 404a and the received value 404a, and if the two values are equal then responder 101x can consider the initiator 102' to be authorized. The responder 101x can take the roll of an enrollee and process a configuration attribute 330. The responder 101x can generate a random number comprising an enrollee nonce E-nonce 332a. Responder 101x can then conduct an encryption step 322c, where encryption step 322c can comprise (i) an encryption step 322 using key ke 327a, and (ii) the plaintext to be encrypted can be the E-nonce 332a and the configuration attribute 330. The output of an encryption step 322c can be a ciphertext 322d. Responder 101x can then send initiator 102' a DPP Configuration Request message 123b, where the configuration request 123b can include the ciphertext 322d, where the plaintext in ciphertext 322d can include E-nonce 332a and the configuration object 330. DPP Configuration Request message 123b can be processed by responder 101x according to section 6.3.3 of DPP specification version 1.0.

Initiator 102' can receive DPP Configuration Request message 123b via connection 192 and using DPP application 102g. As contemplated herein, a DPP application 102g can perform the steps for initiator 102' depicted in Figure 5a, such as transmitting/receiving DPP 191 messages using a first MAC.I 102n for connection 192 and transmitting/receiving messages with DPP server 103 using MAC.N 102q via access network 112. A DPP application 102g in initiator 102' can also perform the depicted steps for an initiator 102' in Figure 5a such as, but not limited to, 504, 505, 313, 314, etc. Initiator 102' can receive message 123b and process the message. Initiator 102' can conduct a decryption step 328c using decryption step 328 depicted in Figure 4c above. A difference between decryption step 328 and decryption step 328c is that decryption step 328c uses the ciphertext 332d as input into the symmetric decryption algorithm 405b. Initiator 102' can use the previously calculated key ke 327a from a step 402c in Figure 5a in order to read the plaintext within ciphertext 332d. Initiator 102' can record the plaintext E-nonce 332a and configuration attribute 330.

Initiator 102 can send DPP server 103 a message 513 via secure connection 302b from a step 301, where message 513 can include an identity for device 101 such as ID-token.device 206 and the plaintext value for the configuration attribute 330. Message 513 can be sent via secure session 302b, so the "plaintext" would be encrypted when transmitted over secure session 302b. In step 335, DPP server 103 can process the received configuration attribute 330. For the exemplary embodiment depicted in Figure 1a and Figure 5a, configuration attribute 330 can specify that device 101 with responder 101x may have a role of WiFi client. In exemplary embodiments, DPP server 103 can previously select a primary access network 308aa in a step 308a within a step 301 depicted in Figure 3a above, using the received networks available list 304c from initiator 102'. Also, in a step 308b in a step 301 in Figure 3a above, DPP server 103 could receive or record the set of network credentials 109 associated with the selected primary access network 308aa for device 101 using device identity ID-token.device 206 (or ID.device 101i).

In step 335, DPP server 103 can process the set of network credentials 109 for primary access network 308aa into a configuration object. The configuration object with network credentials 109 can comprise a connector or configuration object as specified in section 6.3.5 and 6.3.6 of DPP specification version 1.0. The configuration object with network credentials 109 can be optionally signed by DPP server 103 using the recorded initiator bootstrap private key bi 104b. In a different embodiment, a DPP server 103 can omit a signature for network credentials 109 in a step 335, and an initiator 102' can sign the network credentials 109 using the derived initiator ephemeral private key pi 102b. The steps to create and verify a signature with ECC PKI keys can be according to FIPS publication 186-4, and other possibilities exist as well. In Figure 5a, a DPP server 103 can also take the other steps described for a step 335 in Figure 3a above.

In a step 336, DPP server 103 can optionally encrypt network credentials 109 using an asymmetric encryption algorithm and a public key from the responder comprising one of (i) the responder bootstrap public key Br 101a or (ii) the derived responder ephemeral public key Pr 101e (received in message 506), or (iii) the public key for a device recorded by a certificate for device 101 in cert.device 101m. In some preferred exemplary embodiments, a step 336 can comprise encrypting network credentials 109 into credentials 109' using (i) the public key in cert.device 101m, and the elliptic curve Elgamal asymmetric encryption algorithm. The encryption algorithm could be from using Elgamal asymmetric encryption as summarized in the Wikipedia article for ElGamal encryption from July 10, 2018, which is herein incorporated by reference. A set of network credentials 109 that may be optionally encrypted using Elgamal asymmetric encryption and a responder public key is depicted as credentials 109' in Figure 5a. The optional Elgamal asymmetric encryption in a step 336 can use a set of parameters 199a in order to specify values such as encoding rules, byte or bit order, padding, etc. An optional encryption step 336 could also comprise (i) deriving a symmetric key and encrypting the symmetric key with Elgamal encryption and a responder public key and then (ii) encrypting the set of credentials 109 using a symmetric ciphering algorithm 405a and the derived symmetric key, in order to determine an encrypted set of credentials 109'.

In exemplary embodiments, an encrypted set of credentials 109' can be sent to initiator 102' instead of plaintext credentials 109, since an initiator 102' can read plaintext credentials 109, but not feasibly read encrypted credentials 109'. Note that encrypted credentials 109' can include a plaintext public key "h" or "g^x" along with the credentials 109' in order for responder 101x to use a private key such as responder bootstrap private key br 101b or responder ephemeral private key pr 101f to decrypt the encrypted credentials 109'. A reason that a step 336 can be performed in Figure 5a is that (i) initiator 102' could otherwise read and operate on a plaintext network credentials 109 (since initiator 102' uses encryption key ke 327a to encrypt plaintext 403c), and (ii) a DPP server 103 or a network 107 may prefer for initiator 102' to not read plaintext network credentials 109

(since initiator 102' may not be secured or under the full control of a network 107). In other embodiments, an encryption step 336 can be optionally omitted and plaintext credentials 109 can be sent to an initiator 102' via the secure session 302b via message 514.

In another embodiment, the asymmetric encryption step 336 can be performed by an external network, such that the external network generates the asymmetrically encrypted credentials 109', and the asymmetrically encrypted credentials 109' could be stored in a device credentials table 340 from Figure 3b. For example, credentials 109' could comprise the exemplary depicted credentials in table 340 of "from ...", where "..." was an external network in a networks available list 304c. In other words, a DPP server 103 may not be able to read the plaintext credentials 109 within an asymmetrically encrypted credentials 109', since an external network could encrypt the credentials using a step 336 before providing the credentials to network 107 and DPP server 103.

DPP server 103 can then send initiator 102' a message 514, where the message 514 can include an identity for device 101 such as ID-token.device 206 and the network credentials 109', where the network credentials 109' can be in the form of a configuration object and optionally encrypted using an Elgamal asymmetric encryption algorithm. Although a single value for network credentials 109' in message 514 are depicted, network credentials 109' could provide credentials for device 101 to connect with both a selected primary network 308aa and a selected backup network 308ab (with different credentials for each). Or, two sets of configuration objects with two sets of network credentials 109' could be sent in a message 514. Other possibilities exist as well without departing from the scope of the present invention.

Initiator 102' can receive message 514 and process the received credentials. At step 322e, initiator 102 can conduct an encryption step 322e using an encryption step 322 depicted and described in connection with Figure 4c above. Encryption step 322e can be performed using key ke 327a for device 101. For encryption step 322e, the plaintext value for the E-nonce 332a and the configuration object comprising network credentials 109' (where network credentials 109' may be optionally encrypted) can be input into a symmetric encryption algorithm 405a in order to generate a ciphertext 322f. Note that the use of asymmetrically encrypted credentials 109' can be optionally omitted, and an initiator 102' in a step 322e can input plaintext credentials 109 into a symmetric ciphering algorithm 405a.

Responder 101x can receive message 124 from initiator 102' with ciphertext 322f. Responder 101x can conduct a decryption step 328d using decryption step 328 depicted in Figure 4c below. A difference between decryption step 328 and decryption step 328d is that decryption step 328d uses the ciphertext 322f as input into the symmetric decryption algorithm 405b. Responder 101x can use the previously calculated key ke 327a in order to read the plaintext within ciphertext 322f. Note that the plaintext within ciphertext 322f can include an encrypted network credentials

109'. Responder 101x can confirm that plaintext in ciphertext 322f includes the previously sent E-nonce 332a.

For embodiments that use an optional encryption step 336 on network credentials 109 by DPP server 103 in step 336 (or an external network) in order to create encrypted network credentials 109', a responder 101x can use an asymmetric decryption step 337 in order to convert the encrypted network credentials 109' into plaintext network credentials 109. Responder 101x could use an Elgamal asymmetric decryption algorithm and either a recorded (i) responder bootstrap private key bi 101b or (ii) responder ephemeral private key pr 101f. Note that device 101 could decrypt credentials 109' using a private key for the public key in a certificate cert.device 101m, and in this case responder 101x could pass the received asymmetrically encrypted credentials 109' to device 101. Note that when using elliptic curve Elgamal asymmetric decryption, the responder 101x or device 101 uses the private key that corresponds to the public key used with asymmetric encryption of the credentials 109 in a step 336. Responder 101x could also use a selected set of cryptographic parameters 199a in order to conduct the Elgamal asymmetric decryption.

Note that encrypted network credentials 109' could include the public key "h" or "g" in order to conduct the Elgamal asymmetric decryption. For embodiments (a) without an optional encryption step 336 on network credentials 109 to create encrypted credentials 109', then (b) responder 101x can read the plaintext network credentials 109 in a step 337. Note that credentials 109 may be previously signed by DPP server 103 or initiator 102' using a private key for the initiator 102'. The responder could also verify a signature for the credentials 109 using a public key for the initiator, such as the initiator ephemeral public key Pi 102a or the initiator bootstrap public key Bi 104a.

After reading the plaintext network credentials 109 and optionally verifying a signature using a recorded public key 102a or 104a, device 101 can complete the configuration step 106 by loading the network credentials 109 and begin operating a WiFi radio 101h with the network credentials 109, as depicted in Figure 1a. In this manner, device 101 can obtain connectivity to network access point 105. In exemplary embodiments, the initiator 102' can conduct the initiator restore step 129 depicted and described in connection with Figure 1d above, such that initiator 102' can be returned to the previous state where WiFi radio 102w operates with a user configuration 131, thus minimizing impacts of the prior initiator configuration step 127 for an initiator user 102u. As depicted in Figure 5a, the collection of steps beginning with step 339 for responder 101x through step 106 for responder 101x can comprise a step 513.

Figure 5b

Figure 5b is a simplified message flow diagram illustrating an exemplary system with exemplary data sent and received by a device database, a DPP server, an initiator proxy, and a responder, in accordance with exemplary embodiments. Before starting steps and message flows depicted in Figure 5b, the initiator, DPP server, and device database depicted for system 501 in Figure 5b may previously complete exemplary message flows and steps depicted for a step 301 in Figure 3a above. System 501 can include a device database 104, DPP server 103, initiator proxy 102', and responder 101x operating in device 101. Initiator proxy 102' can communicate with DPP server 103 via IP network 113 using an access network 112, also as depicted in Figure 3a above. As contemplated herein, an "initiator proxy 102'" depicted in Figure 5b may also be referred to as initiator 102', and an initiator 102' and initiator proxy 102' may be equivalent herein. A difference between system 501 in Figure 5b and system 300 in Figure 3a can be that the exemplary secret shared keys comprising key k2 314c and ke 327a are recorded and operated within an initiator 102'. In other words, a DPP server 103 does not need access to these keys for the embodiment depicted in Figure 5b. In exemplary embodiments depicted in Figure 5b, both (i) the initiator bootstrap private key bi 104b and (ii) the responder bootstrap public key Br 101a can remain recorded in a DPP server 103 and not exposed to initiator 102'. The exemplary embodiment depicted in Figure 5ba can comprise a third mode 306a of operation for initiator 102' and DPP server 103, as depicted in Figure 3b above.

After a step 301 from Figure 3a, DPP server 103 can conduct a step 502 in order to select a responder bootstrap public key Br 101a for device 101 using an identity ID-token.device 206. The value ID-token.device 206 could be received within step 301 in message 305 as depicted in Figure 3a. The responder bootstrap public key Br 101a could be recorded in a device database 104 or received from a device manufacturer for a device 101. Other possibilities exist as well for a network 107 to have access to a responder bootstrap public key 101a without departing from the scope of the invention. In some embodiments in order for a DPP server 103 to record a responder bootstrap public key Br 101a, an initiator 102' could conduct a PKEX key exchange in band via WiFi link 192 using PKEX key 198, where initiator 102' receives the key 198 in a message 311 in a step 301 in Figure 3a. Initiator 102' could (i) conduct the PKEX using key 198 with responder 101x during a step 303b, and then send the received responder bootstrap public key Br 101a to DPP server 103. Other embodiments depicted in Figure 3a and Figure 5a also could optionally use a PKEX with key 198 in order to receive the key Br 101a "in band" via WiFi link 192, such as if device database 104 does not record key Br 101a before a step 301 in Figure 3a. As mentioned for a step 502 in Figure 5a, a responder 101x could record a plurality of responder bootstrap public keys Br 101a, and the key 101a can be selected in a step 502. A MAC address for MAC.R 101d for device 101 using ID-token.device 206 can also be selected from a server database 103a in a step 502.

Initiator 102' can then conduct a step 313 to derive the initiator ephemeral public key P_i 102a and the initiator ephemeral private key pi 102b. A step 313 was previously depicted and described above in Figure 3a. Although DPP server 103 conducted the step 313 in Figure 3a, an initiator 102' can also conduct the step 313 in order to derive the ephemeral PKI key pairs. For example, initiator 102' also derives the initiator ephemeral PKI keys in a step 313 in Figure 5a. Initiator 102' can then send DPP server 103 the derived initiator ephemeral private key pi 102b in a message 521. Message 521 could be securely send to DPP server 103 via secure session 302b. Message 521 can also include device identity ID-token.device 206 (or ID.device 101i) in order for DPP server 103 to track the device 101 and initiator 102' associated with the key pi 102b. DPP server 103 can record the data received in message 521 in a server database 103a, as depicted in Figure 3b.

DPP server 103 for a step 314z can comprise DPP server 103 conducting a first initiator key exchange step 314 using an Elliptic Curve Diffie Hellman (ECDH) key exchange 401a, as depicted and described in connection with Figure 4a below. Data input for a step 314z can comprise the initiator ephemeral private key pi 102b and the recorded responder bootstrap public key Br 101a. Data output from a step 314z can comprise the secret shared key M 410 as described in Figure 4a above. In other words, the difference between a step 314z and a step 314 is the step 314z can terminate before the key derivation function 401a and the output of a step 314z can comprise the secret shared key M 410. In an alternative embodiment for system 501 in Figure 5b, DPP server 103 can also calculate key $k1$ using the full set of steps for a key exchange 314 (but key $k1$ may not be needed by DPP server 103 for the embodiment depicted in Figure 5b).

At step 316, DPP server 103 can calculate secure hash values for initiator bootstrap public key Bi 104a and responder bootstrap public key Br 101a, using a secure hash algorithm 199e. The values are depicted in Figure 3a as " $H(Bi)$ " and " $H(Br)$ ", respectively. DPP server 103 can then send initiator 102' a message 522, where message 522 can include an identity for device 101 ID-token.device 206 (or ID.device 101i), the hash values for the initiator bootstrap public key Bi 104a and the responder bootstrap public key Br 101a, the shared secret key M 410 from a step 401a in key exchange step 314z, and the MAC address for the responder 101x comprising MAC.R 101d. Message 522 can be sent through secure session 302b and received and recorded by initiator 102'.

Initiator 102' can receive message 522 and conduct a series of steps in order to create and process a DPP authentication request message 123. Initiator 102' can input the received value M 410 from a message 522 into a key derivation function 402a in order to derive a secret shared key $k1$ 314a, depicted as a step 402a for initiator 102' in Figure 5b. At step 504 the initiator 102' can determine the initiator capabilities for conducting a DPP 191 with device 101 and responder 101x. In the exemplary embodiment depicted in Figure 5b, initiator 102' can select and use the capabilities of the role as a configurator. In other embodiments, an initiator 102' could select and use the

capabilities of the role an enrollee for a step 504. In addition, an initiator 102' can generate a random number for an initiator nonce I-nonce 317a in a step 504. A step 504 can also take place before a step 402a in some exemplary embodiments.

Initiator 102' in step 315 can then conduct the encryption step 315 depicted in Figure 4a, and also described in Figure 3a and Figure 5a. The plaintext input into a symmetric encryption algorithm 405a can comprise the initiator nonce 317a and initiator capabilities from a step 504 above. The shared secret key k1 314a can be used with symmetric ciphering algorithm 405a in a step 315. The ciphertext output from an encryption step 315 can comprise ciphertext 315a.

Initiator 102' can then conduct a step 316a to send responder 101x a message 122, where message 122 can include the hash values for (i) the initiator bootstrap public key Bi 104a and (ii) the responder bootstrap public key Br 101a, the derived initiator ephemeral public key Pi 102a (from a step 313 above), and ciphertext 315a, where ciphertext 315a is encrypted with key k1 and includes an initiator nonce 317a and the initiator capabilities. Sending DPP authentication request message 122 was also depicted and described in connection with Figure 1b, Figure 1c, Figure 3a, and Figure 5a above.

In Figure 5b, responder 101x can receive message 122 and conduct the series of steps 325. The series of steps 325 for responder 101x were depicted and described in connection with Figure 3a above, and steps 325 can comprise the steps 318, 319, 320, 321, 314b, 322, and 315b. Responder 101x can process the message 122 using steps 325 and generate a DPP authentication response 123. The steps 325 for a responder 101x to process the received message 122 and generate the response message 123 can comprise section 6.2.2 of DPP specification version 1.0. Other possibilities exist as well without departing from the scope of the present invention. In general, responder 101x can use the recorded bootstrap private keys and the derived ephemeral private keys in order to conduct ECDH key exchanges. Responder 101x can also (i) derive a responder nonce 318b, (ii) decrypt ciphertext in message 122 and (iii) encrypt ciphertext in message 123 using derived secret shared keys from the ECDH key exchanges. Responder 101x can send initiator 102' the DPP authentication response 123 after conducting the steps 325.

Initiator 102' can receive the DPP authentication response 123 and process the message. Using broadcast messages over connection 192, initiator 102' can identify the message from responder 101x based on the hash values for the bootstrap public keys in response 123. For embodiments where message 122 and response 123 are sent as unicast messages, initiator 102' can identify the message from responder 101x based on the MAC.R 101d in the response 123. Initiator 102' can then extract and record responder ephemeral public key Pr 101e (depicted for a response 123 in Figure 3a). Initiator 102' can also a key verification step on the received key Pr 101e according to FIPS publication SP 800-56A (revision 2).

Initiator 102' can then send DPP server 103 a message 506 using secure connection 302b via access network 112, where message 506 can include (i) an identity for device 101 such as ID-token.device 206 (or ID.device 101i), and (ii) the responder ephemeral public key Pr 101e. In an exemplary embodiment a message 506 can include the MAC address MAC.R 101d to also identify device 101 and the associated key Pr 101e. DPP server 103 can receive the message 506 and record the responder ephemeral public key Pr 101e along with the device identity in a server database 103a, as depicted in Figure 3b above.

As depicted in Figure 5b, initiator 102', DPP server 103, and responder 101x can then conduct a step 512. A step 512 can comprise the series of steps and message depicted and described in connection with Figure 5a above. In summary, initiator 102' can conduct a key exchange step 319a to derive the key k2 314c and then decrypt the ciphertext from message 123 into plaintext using a step 320b. Key exchange step 319a can also calculate a shared secret key N 411 as depicted in Figure 4b. The plaintext from a step 320b can comprise a second ciphertext 322a encrypted with a key ke 327a. DPP server 103 can (i) conduct a key exchange step 401c in order to derive shared secret key L 412. DPP server 103 can send initiator 102' the shared secret key L 412. Initiator 102' can conduct a key derivation function step 402c from Figure 4c using the values M 410 (from message 522), N 411 (from a step 319a), and L 412 (from DPP server 103 in a message 510 in a step 512) in order to derive key ke 327a, as depicted in Figure 4c. Initiator 102' can decrypt the second ciphertext 322a in message 123 into plaintext with a decryption step 328 using the derived key ke 327a and read the value R-auth 404. Initiator 102' can also use a step 328 to compare the received, decrypted value for R-auth 404 with an internally calculated value for R-auth, and responder 101x can be authenticated if the two values for R-auth 404 are equal. Initiator 102' can calculate a value for I-auth 404a and encrypt the value with key ke 327a in a step 322b. Initiator 102' can then send responder 101x a DPP authentication confirm message 123a in a step 512, as depicted in Figure 5a and Figure 5b.

As depicted in Figure 5b, initiator 102', DPP server 103, and responder 101x can then conduct a step 513. A step 513 can comprise the series of steps and message depicted and described in connection with Figure 5a above. In summary, responder 101x can use a step 339 to receive and process the DPP authentication confirm message 123a. Responder 101x can decrypt the I-auth value 404a in a step 331. Responder 101x can calculate the value for I-auth 404, and if the received and calculated values for I-auth 404a are equal, then initiator 102' can be authenticated with responder 101x. Responder 101x can send a DPP configuration request 123b message, which can include a ciphertext 322d an E-nonce 332a and a configuration attribute 330. Initiator 102' can decrypt the ciphertext 322d and send DPP server 103 the configuration attribute 330. DPP server 103 can send initiator 102' a set of network credentials 109. The set of credentials 109 may optionally be encrypted with an asymmetric encryption algorithm using a public key for the responder, and for

these embodiments the set of credentials 109 can be designated an encrypted set of credentials 109'. The public key for the responder with an optional asymmetric encryption for the credentials 109' could be (i) the responder bootstrap public key Br 101a, or (ii) the responder ephemeral public key, or (iii) a private key for the device public key in cert.device 101m (e.g. step 337 in Figure 5a).
5 Credentials 109 or credentials 109' can also be signed by either (i) the DPP server 103 using the recorded initiator bootstrap private key bi 104b, or (ii) using the derived initiator ephemeral private key pi 102b.

Continuing with a step 513 in Figure 5b, the credentials 109 or credentials 109' could be signed by both the DPP server 103 and the initiator 102 in a step 513. The initiator 102' can receive
10 the signed network credentials 109' or 109 and encrypt the credentials and the E-nonce 332a using the derived key ke 327a. The initiator 102' can send responder 101x a DPP configuration response 124 message with the encrypted and signed credentials for device 101. Responder 101 can verify the signature and decrypt the credentials 109 or asymmetrically encrypted credentials 109'. Device 101 can apply the credentials 109 and connect with another node using the credentials 109, and the
15 other node could be network access point 105. The other node could also be a WiFi peer-to-peer (P2P) connection. In another embodiment, device 101 could operate as an access point 105 for other WiFi clients, and in this embodiment the credentials 109 received in a message 124 could be for device 101 to operate as a WiFi access point using the credentials 109. Other possibilities exist as well for a responder 101x to receive a set of credentials 109 from an initiator 102' without departing
20 from the scope of the present invention.

System 501 as depicted in Figure 5b can provide several benefits for conducting a device provisioning protocol, compared to the conventional technology depicted in Figure 1b and Figure 1c. First, an initiator bootstrap private key bi 104b can remain recorded in a network 107 and does not need to be shared with an initiator 102'. The operating environment and security for an initiator
25 102' or initiator 102* in Figure 1b and Figure 1c may be outside the control of a network 107 or a device 101 with responder 101x. If an initiator bootstrap private key bi 104b is sent to initiator 102' or initiator 102*, then the initiator bootstrap private key bi 104b cannot be securely reused (since initiator 102' or initiator 102* could then share the key bi 104b or attempt to reuse it with other devices 101 recording the same initiator bootstrap public key Bi 104a). Further, recording the same,
30 shared initiator bootstrap public key Bi 104a across a plurality of devices 101 (such as during device manufacturing or before distribution) can provide benefits for scalability and ease of deployment, especially if the initiators 102' or initiators 102* are not known to a network 107 before distribution of devices 101.

In other words, recording a same initiator bootstrap public key Bi 104a across a plurality of
35 devices may not be securely feasible if the initiator bootstrap private key bi 104b needs to be used by multiple, insecure initiators 102' or initiators 102*. In addition, the responder bootstrap public key

Br 101a can remain securely recorded in a network 107 and not exposed to an initiator 102' for a system 501 in Figure 5b (where Br 101a can be recorded in a device database 104 instead of being received by an initiator 102' from a message 503 in Figure 5a). During the lifetime of a device 101 that could be a decade or longer, a device owner or device user may prefer the device 101 to conduct a device provisioning protocol multiple times, such as after a "factory reset" for device 101. Keeping or storing the responder bootstrap public key Br 101a and initiator bootstrap private key bi 102b in a network 107 can support the steps for a device provisioning protocol to be securely reused for the same device 101.

Yet another benefit of a system 300, system 500, and system 501 is that an overall trust level by responder 101x for conducting a device provisioning protocol can be enhanced or increased, since both an initiator 102' and a DPP server 103 must work in conjunction in order to successfully complete a DPP 191. In other words, for a system 300, system 500, and system 501, security risks for an insecure initiator 102' can be compensated by likely a higher overall level of security for a DPP server 103. These are just a few of the exemplary benefits for systems 300, 500, and 501 compared to conventional technology for conducting a device provisioning protocol, and other benefits will be available as well for those of ordinary skill in the art.

Figure 6a

Figure 6a is a simplified message flow diagram illustrating an exemplary system with exemplary data sent and received by a device database, a DPP server, a WiFi access point, and a device, in accordance with exemplary embodiments. Before starting steps and message flows depicted in Figure 6a, a device 101 with responder 101x could have completed the exemplary configuration step 106 as depicted in Figures 1a, 3a, 5a, and 5b above in order for device 101 to receive and load a set of network access credentials 109. System 600 can include a device database 104, a DPP server 103, a WiFi access point 105, and a device 101. WiFi access point 105 can broadcast an SSID with a value of SSID.network-AP 109a, which also could have been observed by an initiator 102' in a step 303a to collect a networks available list 304c. A DPP server 103 (or another server within a network 107) could (i) conduct a step 308a to select WiFi access point 105 from the networks available list 304c and then (ii) obtain the credentials 109 using a step 308b, and then send the credentials 109 through initiator 102' in a DPP message 124.

Configuration data for device 101 to utilize with WiFi access point 105 could be included in data and values for network credentials 109 for device 101. For embodiments where a selected access network 308aa for device 101 is not a WiFi access point 105 (such as possibly a 5G wireless WAN connection), then network credentials 109 could comprise a profile for an embedded universal integrated circuit card (eUICC) as specified in GSMA standards SGP.22 and SGP.02. For

embodiments where credentials 109 comprise an eUICC profile, then the credentials 109 may preferably be asymmetrically encrypted into credentials 109' using a step 336 from Figure 5a by an external network recording the eUICC profile, and then asymmetrically decrypted by responder 101x or device 101 using a step 337 from Figure 5a. DPP server 103 would record the eUICC profile as credentials 109' (or "from ..." in Figure 3b). For embodiments where asymmetrically encrypted credentials 109' comprising an eUICC profile are received by responder 101x in a DPP Configuration Response 124 message, then device 101 can also operate an eUICC as specified in as specified in GSMA standards SGP.22 and SGP.02.

Values or data in credentials 109 could comprise frequencies or channels to utilize, a service set identifier (SSID) or network name, user identities and passwords, and/or public certificates for device 101, public certificates for a certificate authority, etc. As contemplated herein, a set of credentials 109 can include supporting data to use the credentials 109 and/or connect with an access network 112 or network 107, such as configuration data for a WiFi radio 101h in device 101. The supporting data could comprise Config.network-AP 109c as depicted and described in connection with Figure 1a. Data collected for a set of credentials 109 was also described in a step 335 in Figure 3a.

Device 101 and access point 105 can conduct a WiFi session setup 602 using the WiFi radio 101i with credentials 109. WiFi connection setup 602 can utilize standard WiFi security protocols such as, but not limited to, WPA2, WPA3, or subsequent and related standards. In exemplary embodiments, WiFi connection setup 602 utilizes PMK and a PMKID, where PMK and PMKID can be uniquely associated with device 101. Or WiFi connection setup 602 could use a preshared secret key (PSK) in credentials 109. At step 603, device 101 can select the DPP server 103 and a URL for DPP server 103. A URL for DPP server 103 could be included with credentials 109, or otherwise fetched by device 101 from an IP network 113 after session setup 602. A step 603 can also comprise device 101 confirming expected connectivity to an IP network 113 is provided via access point 105, such as performing domain name server (DNS) requests and downloading data via HTTP or HTTPS/TLS from an IP network 113.

Device 101 and DPP server 103 can then conduct a secure session setup 604. Secure session setup 604 could comprise a session using TLS, IPSec, VPN, SSH, or similar secure networking technologies to establish secure sessions. In exemplary embodiments, DPP server 103 uses the responder bootstrap public key Br 101a as a public key for device 101 in secure session 603 and device 101 uses the initiator bootstrap public key Bi 104a as a public key for DPP server 103 in secure session 603. In other words, (i) DPP server 103 uses the initiator bootstrap private key bi 104b as the private key for DPP server 103 in secure session 603, and (ii) device 101 uses the responder bootstrap private key br 101b as the private key for device 101 in secure session 603. In exemplary embodiments, the public keys responder bootstrap public key Br 101a and the initiator

bootstrap public key Bi 104a are not transmitted between the nodes in a secure session setup 603, but rather the public keys are selected for use since they are already recorded by the two nodes. In this manner, security of secure session 603 can be enhanced, because the public keys are not transmitted or received in a plaintext form, but rather are already previously recorded by the two nodes. In an exemplary embodiment, secure session setup 603 could designate or identify the initiator bootstrap public key Bi 104a (and corresponding private key) for device 101, which could be selected by DPP server 103 sending an initiator bootstrap keys sequence number 197 in a secure session setup 603, for embodiments where device 101 can record a plurality of initiator bootstrap PKI keys, as depicted in device table 104d in Figure 2b. Note that a secure session setup 604 could also use different public keys than those depicted in Figure 6a, such as a certificate for DPP server 103 cert.DPPS 103c and a certificate for device 101 cert.device 101m.

At step 605a, device 101 can download and install a configuration package for device 101 using WiFi session 602 in Figure 6a. The configuration package can be downloaded from network 107, and also potentially from DPP server 103 using secure session 604. Configuration package for device 101 can comprise an updated operating configuration for device 101. Configuration package for device 101 could be downloaded from a network 107. Configuration package for device 101 could include updated software or firmware for device 101, different credentials for device 101 to use when connecting with servers within a network 107, such as potentially a reporting server or a monitoring server within a network 107, timers for device 101 to use with servers in a network 107, and other data as well for the ongoing operation of device 101 with a network 107. After internally recording or loading the files for a configuration package in a step 605a, device 101 can perform a reboot, so that device 101 restarts with the new files from the configuration package. Upon a reboot in a step 605a, connections 602 and 603 may temporarily terminate with the reboot, but device 101 can re-establish connections 602 and 603 after reboot. A step 605a can then also comprise device 101 creating a report 605aa, where report 605aa includes a status code with success or errors for each file in a configuration package. In other words, report 605aa can record the success or errors of applying each of the files in a configuration package, which may be useful for network 107 in system 100 to know the configured state of device 101 after a step 605a. Note that step 605a can take place before a secure session setup 604.

At step 605b device 101 can perform an RF frequency scan or sweep all available mobile network operators and network access technologies around device 101 in order to collect a second networks available list 304a. The second networks available list 304c can be useful for DPP server 103 or network 107 to confirm that a selected primary access network 308aa for device 101 is in fact a preferred network. An exemplary first networks available list 304c was depicted in Figure 3a above (for initiator 102'). Although initiator 102' also collected a first networks available list 304c, device 101 can operate with a different antenna and set of radios 101h with different capabilities

than initiator 102'. For example, RSSI or RF signal strength measurements by initiator 102' in a step 303a may be different than for a device 101. As another example, device 101 may operate at a different location (such as an exemplary tens of meters away) from the location of initiator 102' in a step 303a, and thus a preferred access network 112 for device 101 may be different using the second networks available list 304a from device 101 in step 605b than a first networks available list 304a from initiator 102'. In other words, a network selection step 308a may provide an initial estimate for a preferred access network 112 (identified by selected primary network 308aa in Figure 3b), but the actual/updated preferred access network 112 for device 101 using the criteria from a step 308a could be selected in a step 605b using the second networks available list 304c from device 101. A step 605b for device 101 can include the actions for an initiator 102' in a step 303a to collect a first networks available list 304a for initiator 102'.

Device 101 can send DPP server 103 a message 606, where message 606 can include a device identity ID-token.device 206, the second networks available list 304c collected by device 101 in a step 605b, an initiator bootstrap sequence number 197 for the most recent initiator bootstrap public key Bi 104a used by device 101, and a report 605aa, which could comprise the report generated in a step 605a. As depicted in Figure 6a, message 606 can be sent via (i) access point 105 and then (ii) access network 112 for access point 105. Although a DPP server 103 is depicted in Figure 6a as communicating with device 101 and receiving message 606, another server in a network 107 or associated with network 107 could receive the message 606 (using a secure session similar to secure session 604 depicted in Figure 6a). Note that message 606 could also include the most recent initiator bootstrap public key Bi 104a recorded by device 101.

A DPP server 103 could then conduct a step 308a using the second networks available list 304c from device 101, in order to select an updated, selected primary access network 308aa. A step 308a was depicted and described above in connection with Figure 3a. For embodiments where a step 308a by DPP server 103 selects a different, second primary access network 308aa for device 101 in Figure 6a (e.g. different than a first primary access network 308aa in Figure 3a), a DPP server 103 could then conduct a step 308b in order to receive an updated, second set of credentials 109 for the second primary access network 308aa. Note that the credentials 109 received and recorded in a step 308b by DPP server 103 can comprise an asymmetrically encrypted credentials 109', where a server functioning as the source of asymmetrically encrypted credentials 109' could perform a step 336 depicted and described in connection with Figure 5a. In exemplary embodiments, only device 101 could feasibly read the plaintext credentials 109 for an asymmetrically encrypted credentials 109', where device 101 would use a recorded private key (e.g. key 101b, 101f, or the private key for cert.device 101m) in order to perform an elliptic curve Elgamal asymmetric decryption on the credentials 109'.

A DPP server 103 could conduct a step 607, where DPP server 103 can process the received report 605aa in order to confirm a configuration package for device 101 was successfully applied in a step 605a. DPP server 103 could also forward report 605aa to other servers in a network 107 in a step 607. If errors or problems are detected in report 605aa in a step 607, then DPP server 103 can send a command to device 101 to back out the configuration package (e.g. uninstall) or restore a previous state for device 101.

At step 608, DPP server 103 can determine if recorded initiator bootstrap public key Bi 104a for device 101 should be updated. There could be many reasons both device 101 and network 107 may prefer to update initiator bootstrap public key Bi 104a for a responder 101x operating in device 101. A device 101 may have a lifetime of a decade or longer, and increased key security, use of different cryptographic algorithms, or different cryptographic parameters 199 may be preferred over time. Another reason could be that device 101 or network 107, after (a) a first configuration step 106 via a first session of a device provisioning protocol, may prefer (b) to support a second configuration step 106 via a second session of a device provisioning protocol, where the second session could use a different initiator bootstrap PKI key pair. Another reason could be that a first device 101 used the same key (e.g. same numeric value) for an initiator bootstrap public key Bi 104a as second device 101 (or possibly multiple other devices 101), which is also depicted for “shared keys” in a key table 104k in a device database 104 in Figure 2b. Security could be increased for the first device 101 by changing the recorded initiator bootstrap public key Bi 104a after the key has been used. Other reasons for a device 101, DPP server 103, or network 107 may exist for a DPP server 103 to determine key Bi 104a should be updated for device 101, without departing from the scope of the present invention. Figure 6 depicts an embodiment where DPP server 103 selects to update the initiator bootstrap public key Bi 104a for device 101 in a step 608.

At step 609, DPP server 103 can derive a new, updated initiator bootstrap public key Bi 104a and a corresponding new, updated initiator bootstrap private key bi 104b. The DPP server 103 can use a selected set of cryptographic parameters 199a to derive the keys. The DPP server 103 can use a random number generator and a key pair generation algorithm in order to derive the new, update initiator bootstrap PKI keys. The DPP server 103 can use the step 313 depicted and described in Figure 3a to derived the initiator bootstrap PKI keys (e.g. using same or similar steps as deriving initiator ephemeral PKI keys). Although Figure 6a depicts DPP server 103 as conducting a step 609 to derive new initiator bootstrap PKI keys, another server in a network 107 could conduct a step 609, including device database 104. At step 609, DPP server 103 can also update initiator bootstrap keys sequence number 197 recorded in a device database 104 depicted in Figure 2b to reflect or account for the use of new initiator bootstrap PKI keys. The updated number 197 is depicted in Figure 6a an initiator bootstrap keys sequence number 197a. In exemplary embodiments, the initiator bootstrap keys sequence number 197 can be used at a later time by DPP server 103 when conducting a step

302d for device 101 (e.g. querying device database 104 for initiator bootstrap PKI keys in Figure 3a), such as if DPP server 103 supports a device provisioning protocol for device 101 again at a later time.

A DPP server 103 can then send device database 104 a message 609a, where message 609a can include a device identity ID-token.device 206 (or ID.device 101i), the new, updated initiator bootstrap public key Bi 104a and the corresponding new, updated initiator bootstrap private key bi 104b, where the PKI keys were generated in a step 609. Message 609a can also include the updated initiator bootstrap keys sequence number 197a. Note that DPP server 103 and device database 104 can continue to use secure session 302a from a Figure 3a in order to secure message 609a. In some embodiments, initiator bootstrap private key bi 104b can be recorded only within DPP server 103 in a server database 103a, and for these embodiments initiator bootstrap private key bi 104b can be omitted from a message 609a.

At step 610, device database 104 can record the data from a message 609a in a device table 104d for device 101 using ID-token.device 206 (or ID.device 101i). The data can be recorded by adding a row in device table 104d for device 101. The new, updated initiator bootstrap public key Bi 104a and the corresponding new, updated initiator bootstrap private key bi 104b could be recorded in the added row, along with the received value for an initiator bootstrap keys sequence number 197a. In this manner, device database 104 can record and identify the initiator bootstrap PKI keys for use with device 101 at a later time, for reasons such as, but not limited to, the reasons discussed in connection with step 608 above. Database 104 can then send DPP server 103 a message 611, where message 611 can confirm that step 610 was successfully completed and the new initiator bootstrap PKI keys were recorded in a device table 104d.

After receiving message 611, DPP server 103 can send device 101 a message 612 via secure session 604. Message 612 can include the new, updated initiator bootstrap public key Bi 104a, which was previously derived by DPP server 103 in a step 609 and recorded by device database 104 in a step 610. Message 612 can also include (i) a selected set of parameters 199a associated with the new Bi 104a (e.g. key length, ECC curve name, etc.), and (ii) the updated initiator bootstrap keys sequence number 197a. Device 101 can receive message 612 with the new, updated initiator bootstrap public key Bi 104a and updated initiator bootstrap keys sequence number 197a.

At step 613, device 101 could record the data received in message 612 with a responder 101x operating in device 101. In exemplary embodiments, in step 612, device 101 could also use the received parameters 199a to derive an updated responder bootstrap private key br 101b and responder bootstrap public key Br 101a. Device 101 could use the equivalent steps as DPP server 103 in step 313 depicted and described in Figure 3a in order for responder 101x to derive a new, updated bootstrap private key br 101b and responder bootstrap public key Br 101a. A step 613 can also comprise device 101 recording the derived PKI keys and received key Bi 104a in nonvolatile

memory of device 101. Device 101 can also update a responder 101x operating in device 101 to use the new PKI keys for any subsequent, later instance of a device provisioning protocol either with an initiator 102' or an initiator 102*.

5 In message 614, device 101 can send DPP server 103 and "OK" message noting the successful receipt and recording of the new, updated initiator bootstrap public key Bi 104a. Message 614 can also include the new, updated responder bootstrap public key Br 101a. At a step 615, DPP server 103 can receive message 614 and record the data in a server database 103a for a device 101 using device identity ID-token.device 206. In exemplary embodiments, the new, updated responder bootstrap public key Br 101a can be used by DPP server 103 when supporting an initiator 102' to
10 conduct a device provisioning protocol with device 101 and responder 101x at a later time.

DPP server 103 can then send device database 104 a message 615, where message 615 can include a device identity ID-token.device 206 (or ID.device 101i) and a confirmation that device 101 received and recorded the new, updated initiator bootstrap public key Bi 104a. In this manner, device database 104 can confirm that the new key Bi 104a with corresponding private key bi 104b is
15 "ready for use", and can be selected in a subsequent query 309 (in Figure 3a) from a DPP server 103 for device 101 with identity ID-token.device 206. As depicted in Figure 6a, message 615 can also include a new, updated responder bootstrap public key Br 101a. Device database 104 can also record the received new key Br 101a in a key table 104k for device 101 with identity ID-token.device 206 as depicted in Figure 2b. In this manner, device database 104 can select the new,
20 updated responder bootstrap public key Br 101a (along with initiator PKI keys) in a subsequent query 309 from a DPP server 103 for device 101, in order to conduct a later device provisioning protocol 191.

Initiator 102' can conduct a step 617, where step 617 can comprise the DPP application 102g or initiator 102' sending a query 618 to DPP server 103 in order to confirm successful completing of
25 configuration step 106. DPP server 103 could determine if a configuration step 106 is complete based on the successful establishment of secure session 604 and receipt of a report 605aa. DPP server 103 could also determine that a configuration step 106 is complete using a report 605aa and any of: (i) device 101 has wireless connectivity properly established for a selected primary network 308aa and/or selected backup network 308ab (e.g. credentials 109 were received by device 101 and
30 installed), (ii) device 101 has successfully collected a networks available list 304c, (iii) a configuration package was downloaded and installed in a step 605, (iv) device 101 has received an updated initiator bootstrap public key Bi 104a, and/or (v) device 101 has successfully updated responder bootstrap PKI keys.

As depicted in Figure 6a, DPP server 103 can reply to query 618 with a message 619 of
35 "device OK", if configuration step 106 has been confirmed by DPP server 103 as completed, such as through any of (i) through (v) in the paragraph above. Individual components within (i) through (v)

in the paragraph above could be recorded or confirmed in a report 605aa. Not all of (i) through (v) are required for some embodiments, although connectivity for device 101 to an IP network 113 can be required for DPP server 103 to consider a configuration step 106 completed. In exemplary embodiments, upon successful receipt of a message 619 with a status of “device OK” for device 101 or an equivalent message 619, then initiator 102’ can display to initiator user 102u that a configuration step 106 for device 101 has been completed and device 101 can be left in operation, potentially without additional steps or manual changes to be performed by initiator user 102u. For embodiments where message 619 provides a status of “device not OK” or equivalent, then initiator 102’ could display to initiator user 102u an error status and initiator user 102u could work with device 101 and network 107 in order to diagnose and rectify an error code or error condition in order to receive a subsequent message 619 of “device OK”.

Figure 6b

Figure 6b is a graphical illustration of an exemplary system, where a responder records a plurality of initiator bootstrap public keys and the responder selects and uses an initiator bootstrap public key, in accordance with exemplary embodiments. The present invention contemplates that the systems and methods for (i) using an initiator 102’ with a DPP server 103 and a responder 101x for a device provisioning protocol, where (ii) a responder 101x can record a plurality of initiator bootstrap public keys Bi 104a. Benefits can include the plurality of initiator bootstrap public keys Bi 104a can be recorded in a device 101 with responder 101x before device 101 is taken to the physical location of a network access point 105. The plurality of initiator bootstrap public keys can be written to device 101 during manufacturing or distribution, thereby increasing the flexibility of (a) recording at least one compatible or supported initiator bootstrap public key Bi 104a in a device 101 for initiator 102’, before (b) an initiator 102’ is selected or known for conducting the device provisioning protocol steps 191 with a responder 101x. In exemplary embodiments, different values for recorded initiator bootstrap public keys Bi 104a can represent different possible networks 107 for configuration 106 of device 101 with responder 101x. As contemplated herein, (a) pre-recording a plurality of initiator bootstrap public keys Bi 104a in a plurality of devices 101 can still keep the various systems depicted herein secure, because (b) the corresponding initiator bootstrap private keys bi 104b can remain recorded individually and separately within each of a plurality of networks 107 which can conduct the device provisioning protocol 191 with the plurality of devices 101 recording the same values for initiator bootstrap public keys Bi 104a.

System 601 in Figure 6b can include an initiator 102’ and a device 101, where device 101 can operate a responder 101x. Initiator 102’ in Figure 6b can operate as an initiator 102’ in Figure 1a, Figure 3a, Figure 5a, etc. above. Responder 101x can record (i) responder bootstrap PKI keys Br

101a and br 101b, as well as (ii) at least one of a set of selected cryptographic parameters 199a. Although Figure 6b depicts only a single set of PKI keys Br 101a and br 101b, and a set of selected cryptographic parameters 199a, a responder 101x could also record a plurality of different sets of PKI keys Br 101a and br 101b, with different parameters 199a (e.g. a first set of responder bootstrap keys for a first ECC curve P-256, a second set of bootstrap keys for a second ECC curve P-384, etc.). Responder 101x could determine which of the different sets of responder bootstrap PKI keys to utilize for conducting a device provisioning protocol 191 based on the received hash value for a responder bootstrap public key Br 101a received in a message 122. Although not depicted in Figure 6b, a responder 101x could pre-compute the hash values for each of a possible plurality of bootstrap public keys Br 101a using a hash 199e using cryptographic parameters 199a for each different responder bootstrap public key pair. Responder 101x could (i) listen for receiving a hash 199e value for any of the recorded bootstrap public keys Br 101a in an incoming message 122. When responder 101x observes a matching hash 199e value for a recorded key Br 101a, then select the private key br 101b for use in creating a response message 123. In other embodiments, a single recorded set of bootstrap PKI keys Br 101a and br 101b can be sufficient in order for device 101 to securely conduct a device provisioning protocol 191.

In some embodiments of the present invention, a responder 101x could record a plurality of different initiator bootstrap public keys Bi 104a in an initiator bootstrap public keys table 622, as depicted in Figure 6b. Each of the different initiator bootstrap public keys 104a could also be recorded with both (i) a hash 199e value for each individual key Bi 104a, and (ii) a selected set of cryptographic parameters 199a. Each of the different initiator bootstrap public keys 104a could be associated with a different network 107, as depicted by the exemplary values for possible networks 107 (e.g. “Google”, “Amazon”, etc.). The corresponding bootstrap private key bi 104b for each of the different depicted initiator bootstrap public keys Bi 104a could be recorded in each of the different networks 107. The ninth listed exemplary network 107 in a table 622 in responder 101x can comprise the exemplary network 107 depicted in Figure 1a, which can also operate the DPP server 103 in Figure 3a or Figure 5a, etc.

Although a single initiator bootstrap public key Bi 104a is depicted in Figure 6b for each different exemplary network 107, each different exemplary network 107 could also record multiple different initiator bootstrap public keys Bi 104a, such as (i) a first initiator bootstrap public key Bi 104a for a network 107 with a first set of selected cryptographic parameters 199a (e.g. with exemplary curve P-256), and (ii) a second initiator bootstrap public key Bi 104a with a second set of selected cryptographic parameters 199a (e.g. with exemplary curve P-384). Other possibilities exist as well for recording a plurality of keys Bi 104a for the same network 107 in a table 622 without departing from the scope of the present invention.

Further, the exemplary recorded initiator bootstrap public key Bi 104a for multiple networks 107 in a table 622 can either comprise (i) a “shared key” depicted and described in a key table 104k in Figure 2b where the value or number for the key Bi 104a can be recorded across a plurality of different devices 101, or (ii) a “unique key”, where the value or number for the key Bi 104a can be unique for a device 101 (e.g. not recorded in other devices 101). The use of unique initiator bootstrap public keys Bi 104a recorded for responders 101x is also depicted in a key table 104k in Figure 2b. For embodiments where each device 101 records a “unique key” for the initiator bootstrap public key Bi 104a (such as during device manufacturing), then a device database 104 could record both (i) the unique initiator bootstrap public key Bi 104a for device 101 and (ii) the corresponding unique initiator bootstrap private key bi 104b (such as receiving the keys from the device manufacturer). Other possibilities exist as well, where a set of initiator bootstrap public keys 104a recorded in a device 101 for a network 107 can comprise either (i) shared keys or (ii) unique keys, without departing from the scope of the present invention.

As depicted in Figure 6b the responder 101x in device 101 can receive the message 122 from an initiator 102'. Message 122 was depicted and described in connection with Figure 3a above, and can comprise a DPP authentication request message. Responder 101x can process the message 122 and extract the hash 199e value of an initiator bootstrap public key Bi 104a. Responder 101x can (a) query 620 a table 622 with the hash 199e value H(Bi) in order to (b) select and read the value for the initiator bootstrap public key Bi 104a. The results of a query 620 can comprise a result 621 with key Bi 104a and parameters 199a. The result 621 (comprising at least one key Bi 104a as depicted in Figure 6b) can be used with a set of steps 325 depicted and described in connection with Figure 3a and Figure 5a. Steps 325 can generate a DPP authentication response 123 for initiator 102' using the selected key Bi 104a.

Although not depicted in Figure 6b, for embodiments where responder 101x records a plurality of different responder bootstrap PKI keys 101a and 101b, a query 620 can also be performed by responder 101x in order to select one responder bootstrap public key Br 101a and the corresponding responder bootstrap private key br 101b. Responder 101x could record the plurality of different responder bootstrap PKI keys in a table 623, along with a hash value 199e for each responder bootstrap public key Br 101a (not shown). Upon receipt of a message 122, responder 101x could conduct a query 620 using the received hash 199e H(Br) for a key Br 101a in a table 623 with a plurality of different responder bootstrap public keys Br 101a in order to select at least one key Br 101a and the corresponding private key br 101b. In exemplary embodiments, if the hash 199e value received in a message 122 does not match an internally calculated or recorded hash 199e value for a recorded responder bootstrap public key Br 101a, then responder 101x and device 101 remains silent on WiFi link 192, where WiFi link 192 is depicted above in Figure 3a, Figure 5a, etc.

Figure 7

Figure 7 is a graphical illustration of an exemplary system, where an access point conducts a configuration step for a device using a hosted device provisioning protocol, in accordance with exemplary embodiments. The present invention contemplates that the systems and methods for using an initiator 102' with a DPP server 103 can support a device provisioning protocol 191 in order to configure a network access point 105'. System 700 in Figure 7 can include an unconfigured network access point 105', a mobile phone 701, and a network 107. Unconfigured network access point 105' can include a responder 101x, where responder 101x can have the same or equivalent functionality as a responder 101x for a device 101 depicted and described above in Figure 1a, Figure 1b, Figure 1c, Figure 3a, Figure 5a, and Figure 5b. Responder 101x in network access point 105' can include a responder bootstrap public key Br 101a, a responder bootstrap private key br 101b, and an initiator bootstrap public key Bi 104a. PKI keys could be recorded in access point 105' before access point 105' is installed at an operating location, such as during manufacturing or distribution of unconfigured access point 105'. Responder 101x operating in unconfigured access point 105' can have responder capabilities of an enrollee.

Mobile phone 701 can comprise a smart phone or a mobile handset, such as a mobile phone manufactured by Apple® or running an Android based operating system. Mobile phone 701 could also comprise another mobile and networked computing device, such as a laptop or a tablet. Mobile phone 701 could include an initiator 102', where initiator 102' could have the same or equivalent functionality as initiator 102' depicted and described in connection with Figure 1a, Figure 1b, Figure 1c, Figure 1d, Figure 2a, Figure 3a, Figure 5a, Figure 5b, and Figure 6a. In exemplary embodiments, initiator 102' can operate as an application for mobile phone 701 downloaded from a repository of mobile phone applications, such as an "app store" or "play store", or possibly a private application. In other words, embodiments herein contemplate that a mobile phone 701 could comprise the physical entity for an initiator 102' and the software or firmware instructions for operation of initiator 102' could comprise a DPP application 102g. Other physical and/or logical arrangements for the operation of an initiator 102' are possible as well, without departing from the scope of the present invention. Mobile phone 701 can have both a WiFi radio 102w and a WAN radio 102h. Mobile phone 701 can communicate with unconfigured access point 105' via the WiFi radio 102w and a network 107 via the WAN radio 102h and an access network 112. Initiator 102' for mobile phone 701 can have capabilities of a configurator.

Network 107 can operate a DPP server 103, where a DPP server 103 in Figure 7 can have the functionality of a DPP server 103 depicted and described in connection with Figures 1a, Figure 3a, Figure 3b, Figure 5a, Figure 5b, and Figure 6a. Although not depicted in Figure 7, a network 107 can also operate other supporting servers and/or running processes depicted in Figure 1a, Figure 2a,

etc. Network 107 could be connected to mobile phone 107 via the access network 112 and an IP network 113 depicted in Figure 1a.

As depicted in Figure 7, mobile phone 107 using initiator 102' could conduct the series of messages for a device provisioning protocol 191 with responder 101x operating in unconfigured access point 105'. The series of messages for a device provisioning protocol 191 could comprise the messages 122, 123, 123a, 123b, and 124 as depicted and described in connection with Figure 3a and Figure 5a. In order for initiator 102' to send and receive the device provisioning protocol messages 191 depicted in Figure 7, initiator 102 transmit and receive a series of messages with DPP server 103 in network 107, where the series of messages could comprise a hosted device provisioning protocol 702.

For a first mode 306a, representing embodiments depicted in Figure 3a (where initiator bootstrap and ephemeral secret keys remain in DPP server 103), the series of messages for a hosted device provisioning protocol 702 could comprise the messages between initiator 102' and DPP server 103 that are depicted in Figure 3a (e.g. messages 302b, 305, 311, 317, etc.). For a second mode 306a, representing embodiments depicted in Figure 5a where the initiator bootstrap private key bi 104b can remain in DPP server 103, the series of messages for a hosted device provisioning protocol 702 could comprise the messages between initiator 102' and DPP server 103 that are depicted in Figure 5a (e.g. messages 302b, 305, 311, 503, 506, 510, etc.). For a third mode 306a, representing embodiments depicted in Figure 5b where the initiator bootstrap private key bi 104b and responder bootstrap public key Br 101a can remain in DPP server 103, the series of messages for a hosted device provisioning protocol 702 could comprise the messages between initiator 102' and DPP server 103 that are depicted in Figure 5b (e.g. messages 302b, 305, 311, 521, 522, etc.).

Upon completion of a device provisioning protocol 191 and a hosted device provisioning protocol 702, unconfigured network access point 105' can conduct a configuration step 106 to load a set of network credentials 109 and begin operating with the credentials 109, as depicted in Figure 7. The configuration step 106 can convert the unconfigured network access point 105' into an operating network access point 105, such that WiFi clients (or "sta") can connect with network access point 105 using a compatible set of network credentials 109. As depicted in Figure 7, a mobile phone 701 could also operate as an initiator 102' with a roll of a configurator for the access point 105'. Mobile phone 701 could also conduct a device provisioning protocol 191 using a hosted device provisioning protocol 702 with DPP server 103 in order to transfer a set of compatible network credentials 109 to device 101 in Figure 7, such that device 101 could connect with network access point 105'. The compatible network credentials 109 for device 101 can specify that device 101 operates in a WiFi client mode with network access point 105.

Figure 8

Figure 8 is a graphical illustration of an exemplary system, where an access point operates as an initiator for a device with a responder, in order transfer a set of network credentials to the device, in accordance with exemplary embodiments. The present invention contemplates embodiments where a network access point 105 can operate as an initiator 102' for a device 101 operating as a responder 101x, in order for the network access point 105 to transfer network credentials 109 to the device 101. System 800 in Figure 8 can include an unconfigured device 101', a network access point 105, and a network 107.

Network access point 105 in a system 800 can operate with a set of network access credentials 109 and also operate as an initiator 102'. Network access point 105 can connect with a network 107 via an access network 112. Network 107 can operate a DPP server 103, where the function and operation of a DPP server 103 is depicted and described above in connection with Figure 1a, Figure 3a, Figure 5a, etc. Although not depicted in Figure 8, network 107 could also operate a device database 104 and DPP server 103 could include a server database 103a. Network access point 105 could comprise a WiFi router and operate a WiFi radio. Network access point 105 can have the internal electrical components equivalent to an initiator 102' depicted and described above in connection with Figure 1d, including a CPU, RAM, nonvolatile memory, and a system bus. For the embodiment depicted in Figure 8, the initiator 102' in network access point 105 could comprise software, firmware, or a running process within network access point 105. Initiator 102' in network access point 105 could have the same or equivalent functionality as initiator 102' depicted and described in connection with Figure 1a, Figure 1b, Figure 1c, Figure 1d, Figure 2a, Figure 3a, Figure 5a, Figure 5b, and Figure 6a. Network access point 105 could operate with capabilities as a configurator 801, as depicted in Figure 8.

An unconfigured device 101' could be a device with a WiFi radio 101h and operate a responder 101x. The electrical components inside device 101' could be similar to the electrical components depicted for an initiator 102' in Figure 1d, such as including a CPU, RAM, nonvolatile memory, an operating system, and a system bus. Responder 101x in unconfigured device 101' could have the same or equivalent functionality as a responder 101x depicted and described above in Figure 1a, Figure 1b, Figure 1c, Figure 3a, Figure 5a, and Figure 5b. Unconfigured device 101' can have capabilities as an enrollee 802.

As depicted in Figure 8, network access point 105 initiator 102' could conduct the series of messages for a device provisioning protocol 191 with responder 101x operating in unconfigured device 101'. The series of messages for a device provisioning protocol 191 could comprise the messages 122, 123, 123a, 123b, and 124 as depicted and described in connection with Figure 3a and Figure 5a. In order for initiator 102' to send and receive the device provisioning protocol messages 191 depicted in Figure 8, initiator 102' (or AP 105 operating the initiator 102') transmits and

receives a series of messages with DPP server 103 in network 107, where the series of messages could comprise a hosted device provisioning protocol 702. The series of messages for a hosted device provisioning protocol 702 were depicted and described in connection with Figure 7 above. Note that the steps and messages for conducting a hosted device provisioning protocol 702 can depend on a mode 306a, where the mode 306a can be selected by a DPP server 103 in a step 306. The present invention also contemplates that a mode 306a for a hosted device provisioning protocol 702 can be selected by an initiator 102'. For these embodiments, where mode 306a is selected by an initiator 102', a selected mode 306a could be transmitted from initiator 102' to DPP server 103 in a message 305 depicted and described in connection with Figure 3a above.

The combined (i) device provision protocol 191 between network access point 105 and device 101 and (ii) hosted device provisioning protocol 702 between network access point 105 and DPP server 103 can securely transfer a set of compatible network access credentials 109 to device 101'. Upon completion of a device provisioning protocol 191 and a hosted device provisioning protocol 702, unconfigured device 101' can conduct a configuration step 106 to (i) load a compatible set of network credentials 109 for network access point 105 and (ii) begin operating with the credentials 109, as depicted in Figure 8. The configuration step 106 can convert the unconfigured device 101' into an operating device 101, such that device 101 can conduct a WiFi connection setup step 803 using the received set of compatible network credentials 109.

Conclusion

Various exemplary embodiments have been described above. Those skilled in the art will understand, however, that changes and modifications may be made to those examples without departing from the scope of the claims.

CLAIMS

1. A method for supporting a device provisioning protocol (DPP), the method performed by a first computing device, the method comprising:

a) establishing, by the first computing device, a secure session with a server, wherein the secure session comprises at least authentication of the first computing device;

b) receiving, by the first computing device and from a second computing device, a tag value for a responder;

c) sending, by the first computing device and to the server, at least a portion of the received tag value;

d) sending, by the first computing device and to the second computing device, an initiator nonce, wherein the initiator nonce is encrypted using at least a first shared secret derived by the first computing device;

e) receiving, by the first computing device and from the second computing device, a responder ephemeral public key for the second computing device;

f) sending, by the first computing device, the responder ephemeral public key to the server;

g) receiving, by the first computing device and from the server, a second shared secret, wherein the second shared secret is derived using at least a responder bootstrap public key and the responder ephemeral public key;

h) deriving, by the first computing device, an encryption key using at least the first shared secret and the second shared secret;

i) encrypting, by the first computing device, an authorization value using at least the derived encryption key;

j) sending, by the first computing device and to the second computing device, the encrypted authorization value; and

k) sending, by the first computing device and to the second computing device, a ciphertext of a network credential, wherein the ciphertext is encrypted using at least the derived encryption key.

2. The method of claim 1, wherein the first computing device comprises an initiator and the second computing device comprises the responder, and wherein the responder ephemeral public key comprises a responder protocol public key.

3. The method of claim 2, wherein the first computing device comprises a mobile phone operating a software application for the initiator.

4. The method of any one of claims 1 to 3, wherein the responder bootstrap public key is received from the server, and wherein the first shared secret is derived using the responder bootstrap public key and an initiator ephemeral public key.

5. The method of any one of claims 1 to 4, wherein the tag value includes the responder bootstrap public key, wherein the portion of the received tag value comprises the responder bootstrap public key, and wherein the first shared secret is derived using the responder bootstrap public key and an initiator ephemeral public key.

6. The method of any one of claims 1 to 5, wherein the initiator nonce is sent in a DPP authentication request, wherein the responder ephemeral public key is received in a DPP

authentication response, wherein the authorization value is sent in a DPP authentication confirm message, and wherein the ciphertext is sent in a DPP configuration response message.

7. The method of any one of claims 1 to 6, wherein the first shared secret is derived using (i) an initiator ephemeral private key, (ii) the responder bootstrap public key, and (iii) a first elliptic curve Diffie Hellman key exchange, and wherein the second shared secret is derived by the server using an initiator bootstrap private key and a second elliptic curve Diffie Hellman key exchange.

8. A method for supporting a device provisioning protocol (DPP), the method performed by a network, the method comprising:

a) storing, by the network, an initiator bootstrap private key and a responder bootstrap public key;

b) establishing, by the network, a secure session with a computing device, wherein the computing device authenticates with the network, and wherein the network receives a tag value for a responder via the secure session;

c) sending, by the network and to the computing device, (i) an initiator bootstrap public key for the initiator bootstrap private key, and (ii) an initiator configuration;

d) receiving, by the network and from the computing device, a responder ephemeral public key;

e) conducting, by the network, a first elliptic curve Diffie-Hellman (ECDH) key exchange using the responder bootstrap public key, the responder ephemeral public key, and the initiator bootstrap private key in order to derive a shared secret key;

f) sending, by the network and to the computing device, the derived shared secret key via the secure session; and

g) receiving, by the network and from the computing device, a plaintext configuration attribute, wherein a ciphertext configuration attribute is decrypted into the plaintext configuration attribute using at least the derived shared secret key.

9. The method of claim 8, wherein the network includes a database and a DPP server, and wherein the database stores the initiator bootstrap private key and the DPP server conducts the first ECDH key exchange.

10. The method of claim 8 or 9, wherein the computing device reads the tag value for the responder, and wherein the computing device establishes a DPP session with the responder over a WiFi connection.

11. The method of any one of claims 8 to 10, wherein the derived shared secret key comprises a value of “L” according to the device provisioning protocol.

12. The method of any one of claims 8 to 11, wherein the computing device derives an initiator ephemeral private key at least for conducting a second ECDH key exchange, wherein the second ECDH key exchange derives a value of “M” according to the device provisioning protocol, and wherein the computing device derives the value of “M” before the computing device sends a DPP authentication request message.

13. The method of claim 12, further comprising decrypting, by the computing device, the ciphertext configuration attribute using at least (i) the shared secret key received from the network, and (ii) the value of “M”.

14. A mobile device for supporting a device provisioning protocol (DPP), the mobile device comprising:

a camera for reading a tag value associated with a WiFi device;

a WAN radio for establishing a secure session with a server, for sending the tag value to the server, for receiving from the server (i) an initiator configuration (ii) a first hash value of a responder bootstrap public key and (iii) a first ciphertext;

a WiFi radio for operating with a user configuration before the mobile device reads the tag value, for operating with the initiator configuration after the mobile device reads the tag value, for transmitting a DPP authentication request message with the first ciphertext and the first hash value, and for receiving a DPP authentication response message with a second ciphertext and a responder ephemeral public key;

a nonvolatile memory for storing (i) the user configuration while the WiFi radio operates with the initiator configuration, and (ii) a DPP application for conducting the DPP over the WiFi radio;

a processor for validating the responder ephemeral public key, for encrypting the responder ephemeral public key and the second ciphertext for the secure session, and for receiving a third ciphertext from the server, wherein the third ciphertext includes data for a DPP authentication confirm message; and

a data bus for electrically connecting the camera, WAN radio, WiFi radio, nonvolatile memory, and processor, and for transferring a fourth ciphertext with a set of credentials for the WiFi device from the WAN radio to the WiFi radio.

15. The mobile device of claim 14, wherein the mobile device (i) does not store and (ii) does not receive the responder bootstrap public key for the WiFi device.

16. The mobile device of claim 14 or 15, wherein the DPP application applies the user configuration to the WAN radio after the mobile device sends the set of credentials to the WiFi device, and wherein the mobile device comprises at least one of (i) a mobile phone, (ii) a tablet computer, and (iii) a laptop computer.

17. The mobile device of any one of claims 14 to 16, wherein the WiFi device comprises a responder for the DPP, and wherein the DPP authentication request message is transmitted by the mobile device as a broadcast message over a WiFi link.

18. The mobile device of any one of claims 14 to 17, wherein the server records an initiator bootstrap private key corresponding to an initiator bootstrap public key, wherein the mobile device receives a second hash value of the initiator bootstrap public key via the secure session, and wherein the DPP authentication request message includes the second hash value.

19. The mobile device of any one of claims 14 to 18, wherein the third ciphertext is encrypted with a symmetric ciphering key, and wherein the symmetric ciphering key is derived

using an elliptic curve Diffie Hellman (ECDH) key exchange and at least (i) an initiator bootstrap private key, (ii) the responder bootstrap public key, and (iii) the responder ephemeral public key.

20. The mobile device of claim 19, wherein the WiFi radio transmits a DPP authentication confirm message that includes the third ciphertext, wherein the WiFi radio receives a DPP configuration request message after the WiFi radio transmits the DPP authentication confirm message, wherein the WiFi radio transmits a DPP configuration response message with the fourth ciphertext to the WiFi device, and wherein the fourth ciphertext is encrypted with the symmetric ciphering key.

21. A method for supporting a device provisioning protocol, the method performed by a server, the method comprising:

- a) recording, by the server, an initiator bootstrap private key;
- b) receiving, by the server and from an initiator, a responder bootstrap public key;
- c) deriving, by the server, (i) an initiator ephemeral private key, (ii) a corresponding initiator ephemeral public key, and (iii) a pseudo random number comprising an initiator nonce;
- d) conducting, by the server, (x) a first elliptic curve Diffie-Hellman (ECDH) key exchange using the responder bootstrap public key and the derived initiator ephemeral private key in order to (y) derive a first symmetric ciphering key;
- e) encrypting, by the server, the initiator nonce into a first ciphertext using the first symmetric ciphering key;
- f) sending, from the server to the initiator, the first ciphertext and the initiator ephemeral public key;

g) receiving, by the server and from the initiator, a second ciphertext and a responder ephemeral public key;

h) conducting, by the server, (x) a second ECDH key exchange using the responder ephemeral public key and the derived initiator private key in order to (y) derive a second symmetric ciphering key;

i) decrypting, by the server, the second ciphertext into a first plaintext using the second symmetric ciphering key, wherein the first plaintext from the second ciphertext includes a third ciphertext;

j) conducting, by the server, (x) a third ECDH key exchange using the received responder bootstrap public key, the received responder ephemeral public key, and the recorded initiator bootstrap private key in order to (y) derive a third symmetric ciphering key;

k) decrypting, by the server, the third ciphertext into a second plaintext using the third symmetric ciphering key, wherein the second plaintext includes a received responder authentication value;

l) determining, by the server, the received responder authentication value equals a calculated responder authentication value using at least, in part, the initiator nonce;

m) encrypting, by the server, a set of network credentials using the third symmetric ciphering key; and

n) sending, from the server to the initiator, the encrypted set of network credentials.

22. The method of claim 21, further comprising sending, from the server to the initiator, a fourth ciphertext before sending the set of network credentials, wherein the fourth ciphertext includes an initiator authentication value, and wherein the fourth ciphertext is encrypted using the third symmetric ciphering key.

23. The method of claim 21 or 22, further comprising the initiator operating as an initiator proxy, wherein the initiator proxy does not receive and does not operate with the initiator bootstrap private key.

24. The method of any one of claims 21 to 23, further comprising receiving, by a discovery server, a tag value from the initiator, wherein the discovery server uses the tag value to send the initiator a domain name for the server.

25. The method of any one of claims 21 to 24, wherein the server records a set of cryptographic parameters for the initiator bootstrap private key and the initiator ephemeral private key, wherein the server uses the set of cryptographic parameters for conducting the first, second, and third ECDH key exchanges.

26. The method of claim 25, wherein the server uses a key derivation function and the set of cryptographic parameters with the first, second, and third ECDH key exchanges in order to respectively derive the first, second, and third symmetric keys.

27. The method of any one of claims 21 to 26, further comprising establishing, by the server and the initiator, a secure session, wherein the first, second, and third ciphertexts are transmitted through the secure session.

28. A method for supporting a device provisioning protocol (DPP), the method performed by a mobile phone, the method comprising:

a) operating, by the mobile phone, a DPP application, wherein the DPP application comprises an initiator for the device provisioning protocol;

b) establishing, by the mobile phone, a secure session with a server, wherein the secure session comprises at least, in part, authentication of the DPP application;

c) receiving, by the mobile phone, a tag value for a responder;

d) sending, by the mobile phone and to the server, the received tag value;

e) receiving, by the mobile phone and from the server, an initiator configuration, wherein the mobile phone uses the received initiator configuration with a WiFi radio in the mobile phone.

f) receiving, by the mobile phone and from the server, an initiator ephemeral public key and a first ciphertext, wherein the first ciphertext includes an initiator nonce;

g) sending, by the mobile phone and to the device, the initiator ephemeral public key and the first ciphertext, wherein the mobile phone uses the received initiator configuration to send the initiator bootstrap public key and the first ciphertext to the device;

h) receiving, by the mobile phone and from the device, a responder ephemeral public key and a second ciphertext, wherein the second ciphertext includes a responder authentication value;

i) sending, by the mobile phone and to the server, the responder ephemeral public key and the second ciphertext; and

j) receiving, by the mobile phone and from the server, a third ciphertext, wherein the third ciphertext includes a set of network credentials for the device.

29. The method of claim 28, further comprising using a camera for the mobile phone in order to receive a QR code for the device, wherein the received QR code includes the tag value and

a set of cryptographic parameters, and wherein the tag value does not include the responder bootstrap public key.

30. The method of claim 28 or 29, wherein the mobile phone does not receive a responder bootstrap public key from the device, and wherein the server records the responder bootstrap public key before the mobile phone establishes the secure session with the server.

31. The method of any one of claims 28 to 30, further comprising the mobile phone conducting a radio-frequency scan in order to obtain a networks available list, wherein the mobile phone (i) sends the network available list to the server before (ii) receiving the set of network credentials in the third ciphertext.

32. The method of claim 31, further comprising the server using the networks available list in order to select a primary access network for the device, wherein the server uses the selected primary access network to select the set of network credentials from a server database.

33. The method of any one of claims 28 to 32, further comprising the mobile phone using (i) a channel list for the initiator configuration and (ii) the WiFi radio for the mobile phone to cycle through the channel list and send the initiator ephemeral public key and the first ciphertext to a responder on a plurality of channels in the channel list.

34. The method of any one of claims 28 to 33, wherein (a) sending, by the mobile phone and to the device, the initiator ephemeral public key and the first ciphertext comprises (b) a Device Provisioning Protocol (DPP) authentication request message.

35. A system for supporting a device provisioning protocol (DPP), the system comprising:

an authentication server for authenticating an initiator and for sending the authenticated initiator an initiator configuration;

a mobile phone for operating a DPP application, wherein the DPP application comprises at least, in part, the initiator, for receiving a responder bootstrap public key from a responder, wherein a secure hash value for the responder bootstrap public key is associated with a device identity, for using the initiator configuration to communicate over a WiFi link, for receiving a responder ephemeral public key using the WiFi link, for receiving an encrypted set of network credentials from a network, and for sending the encrypted set of network credentials to the responder over the WiFi link;

a device database for recording the device identity, an initiator bootstrap private key, and a set of cryptographic parameters for the bootstrap private key;

a DPP server for receiving the device identity from the initiator, for querying the device database using the device identity in order to receive the initiator bootstrap private key, for receiving the responder bootstrap public key from the mobile phone, for conducting an elliptic curve Diffie-Hellman (ECDH) key exchange using (i) the received responder bootstrap public key (ii) the received responder ephemeral public key. and (iii) the received initiator bootstrap private key, wherein the output of the ECDH key exchange comprises a symmetric ciphering key, for selecting an initiator mode for the initiator, and for sending the initiator mode to the mobile phone;

a processor in the DPP server for encrypting the set of network credentials using the symmetric ciphering key and the set of cryptographic parameters; and

a LAN interface in the DPP server for sending the encrypted set of network credentials to the mobile phone over a secure connection established between the DPP server and the mobile phone.

36. The system of claim 35, wherein the mobile phone operates as an initiator proxy, and wherein the mobile phone does not receive and does not operate with the initiator bootstrap private key.

37. The system of claim 35 or 36, further comprising a responder for recording an initiator bootstrap public key corresponding to the initiator bootstrap private key, and for using the recorded initiator bootstrap public key and to derive the symmetric ciphering key,

38. The system of claim 37, further comprising a network access point for using the set of network access credentials, and for providing connectivity with an IP network to the responder using the set of network access credentials

39. The system of any one of claims 35 to 38, wherein the mobile phone receives the responder ephemeral public key in a DPP authentication response message, wherein the DPP authentication response message includes a ciphertext of a responder authentication value, and wherein the ciphertext is encrypted with the symmetric ciphering key.

40. The system of any one of claims 35 to 39, wherein the DPP server uses the initiator bootstrap private key with a plurality of different responder bootstrap public keys, wherein the

plurality of different responder bootstrap public keys are associated with a plurality of different responders.

41. A method for supporting a device provisioning protocol (DPP), the method performed by a computing device, the method comprising:

a) recording, by the computing device, (i) a plurality of responder bootstrap private keys, and (ii) a plurality of initiator bootstrap public keys, wherein at least the plurality of initiator bootstrap public keys is recorded during one of a manufacturing and a distribution of the computing device;

b) operating, by the computing device, a WiFi radio and a responder, wherein the WiFi radio communicates with an initiator proxy;

c) receiving a DPP authentication request message with (i) a first secure hash value of a responder bootstrap public key and (ii) a second secure hash value of an initiator bootstrap public key;

d) selecting (i) a responder bootstrap private key from the plurality of responder bootstrap private keys using the first secure hash value and (ii) the initiator bootstrap public key from the plurality of initiator bootstrap public keys using the second secure hash value, wherein a DPP server records a corresponding initiator bootstrap private key for the initiator bootstrap public key;

e) generating a DPP authentication response using at least the responder bootstrap private key and the initiator bootstrap public key; and

f) receiving a DPP configuration response with network credentials for an access point.

42. The method of claim 41, wherein the plurality of initiator bootstrap public keys is recorded before the computing device is located with the access point.

43. The method of claim 41 or 42, wherein the computing device is associated with a tag, and wherein the initiator proxy reads the tag in order to receive the DPP authentication request message from the DPP server.

44. The method of any one of claims 41 to 43, further comprising:
recording, by the computing device a corresponding plurality of responder bootstrap public keys;

selecting a responder bootstrap public key from the plurality of responder bootstrap public keys using the first secure hash value;

generating the DPP authentication response using at least the responder bootstrap public key; and

an authentication server for authenticating an initiator and for sending the authenticated initiator an initiator configuration.

45. A method for supporting a device provisioning protocol (DPP), the method performed by a network, the method comprising:

a) recording a responder bootstrap public key, a set of cryptographic parameters, and an identity token for a device in a device database;

b) authenticating a mobile handset, wherein the network receives the identity token for a device from the authenticated mobile handset;

- c) selecting the responder bootstrap public key and the set of cryptographic parameters using the identity token;
- d) deriving, by the network and using the set of cryptographic parameters, (i) an initiator ephemeral private key, (ii) a corresponding initiator ephemeral public key, and (iii) a pseudo random number comprising an initiator nonce;
- e) conducting, by the network, (x) a first elliptic curve Diffie-Hellman (ECDH) key exchange using the responder bootstrap public key and the derived initiator ephemeral private key in order to (y) derive a first symmetric ciphering key k1;
- f) encrypting, by the network, at least the initiator nonce into a first ciphertext using the first symmetric ciphering key k1;
- g) sending, from the network and to the mobile handset, the first ciphertext and the initiator ephemeral public key;
- h) receiving, by the network and from the mobile handset, a second ciphertext and a responder ephemeral public key;
- i) conducting, by the network, (x) a second ECDH key exchange using the responder ephemeral public key and the derived initiator private key in order to (y) derive a second symmetric ciphering key k2;
- j) decrypting, by the network, the second ciphertext into a first plaintext using the second symmetric ciphering key k2, wherein the first plaintext from the second ciphertext includes a third ciphertext;
- k) conducting, by the network, (x) a third ECDH key exchange using the recorded responder bootstrap public key and the received responder ephemeral public key in order to (y) derive a third symmetric ciphering key ke;

l) decrypting, by the network, the third ciphertext into a second plaintext using the third symmetric ciphering key ke , wherein the second plaintext includes a received responder authentication value;

m) determining, by the network, the received responder authentication value equals a calculated responder authentication value using at least, in part, the initiator nonce;

n) encrypting, by the network, a set of network credentials for the device using the third symmetric ciphering key ke ; and

o) sending, from the network and to the mobile handset, the encrypted set of network credentials.

46. The method of claim 45, wherein the network receives the responder bootstrap public key from at least one of (i) a device manufacturer and (ii) a device owner.

47. The method of claim 45 or 46, wherein the identity token comprises a secure hash value of the responder bootstrap public key.

48. The method of claim 45 or 46, wherein the identity token comprises a device identity for the device.

49. The method of any one of claims 45 to 48, wherein the network and the mobile handset establish a secure session after the mobile handset authenticates with the network, and wherein the first, second, and third ciphertexts are transmitted through the secure session.

50. The method of any one of claims 45 to 49, wherein the network comprises at least an authentication server, the device database, and a DPP server.

51. The method of any one of claims 45 to 50, wherein the device operates a responder for the device provisioning protocol, and wherein the device (i) mutually derives the third symmetric ciphering key ke and (ii) decrypts the encrypted set of network credentials using the third symmetric ciphering key ke .

52. The method of any one of claims 45 to 51, wherein the device does not record an initiator bootstrap public key for the network.

53. The method of any one of claims 45 to 52, wherein the network receives the set of network credentials for the device from the authenticated mobile handset.

54. The method of any one of claims 45 to 52, wherein the network receives the set of network credentials from a device owner.

55. The method of any one of claims 45 to 54, wherein the initiator ephemeral public key comprises an initiator protocol public key, and wherein the responder ephemeral public key comprises a responder protocol public key.

Figure 1a

100

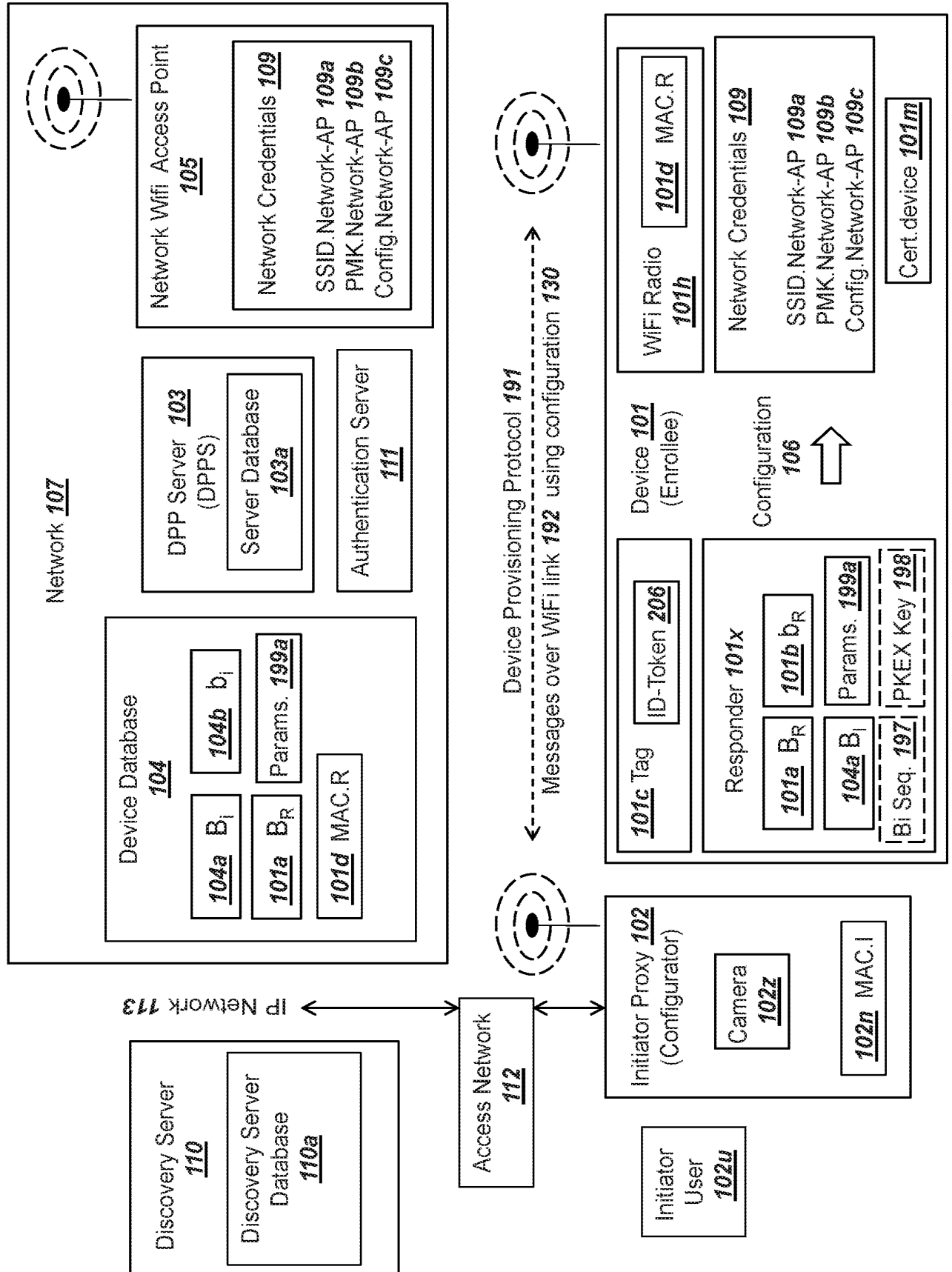


Figure 1b

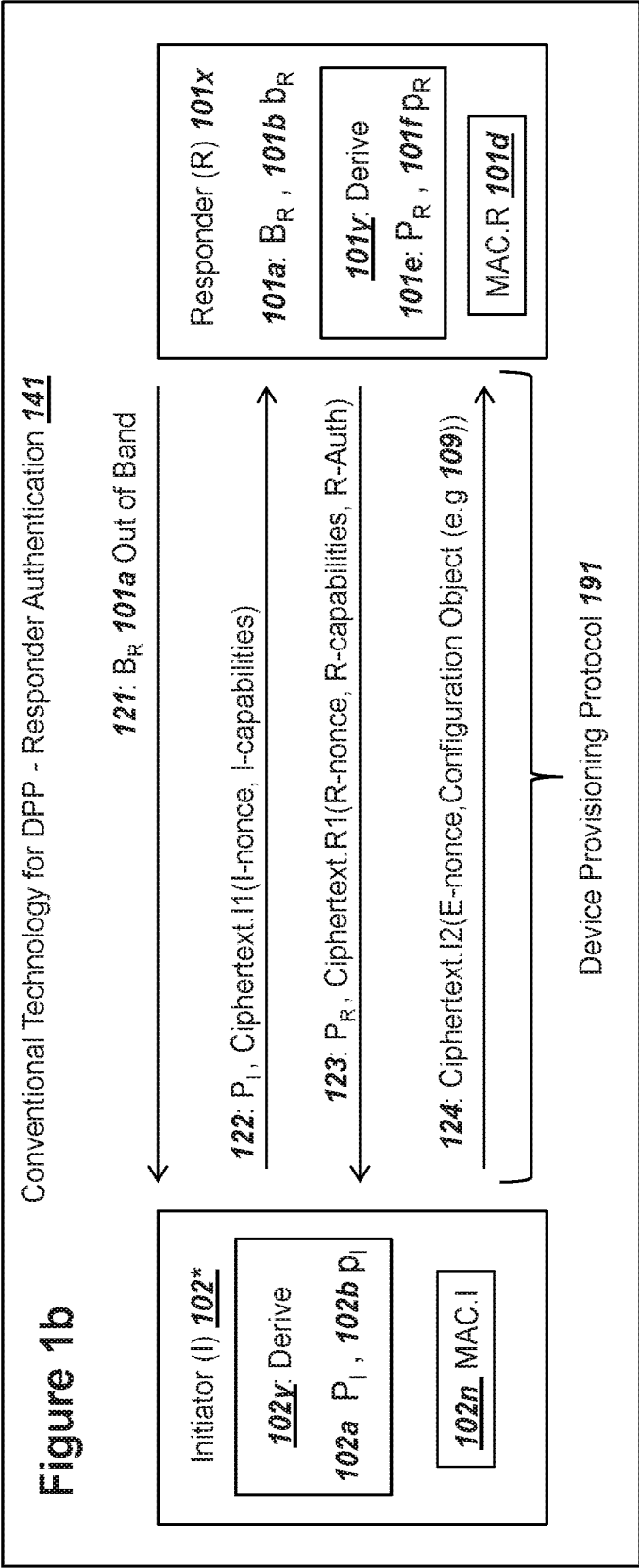


Figure 1c

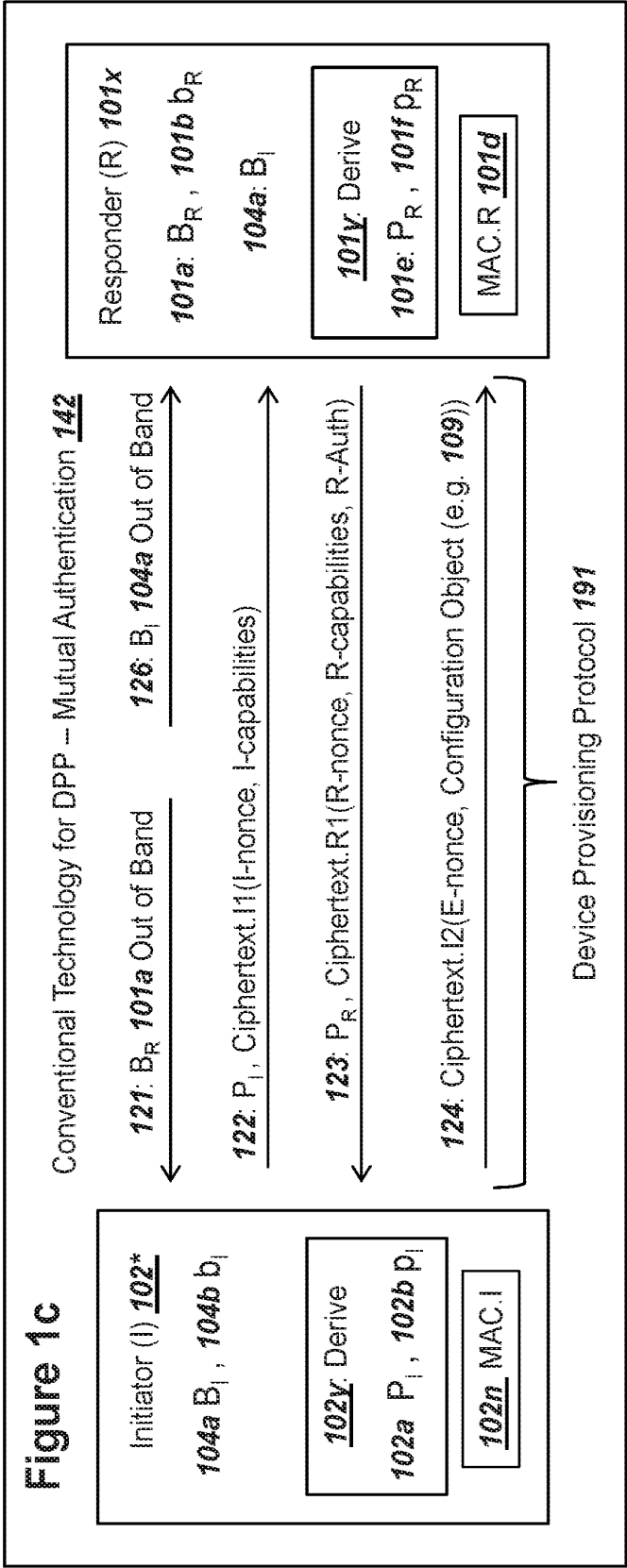
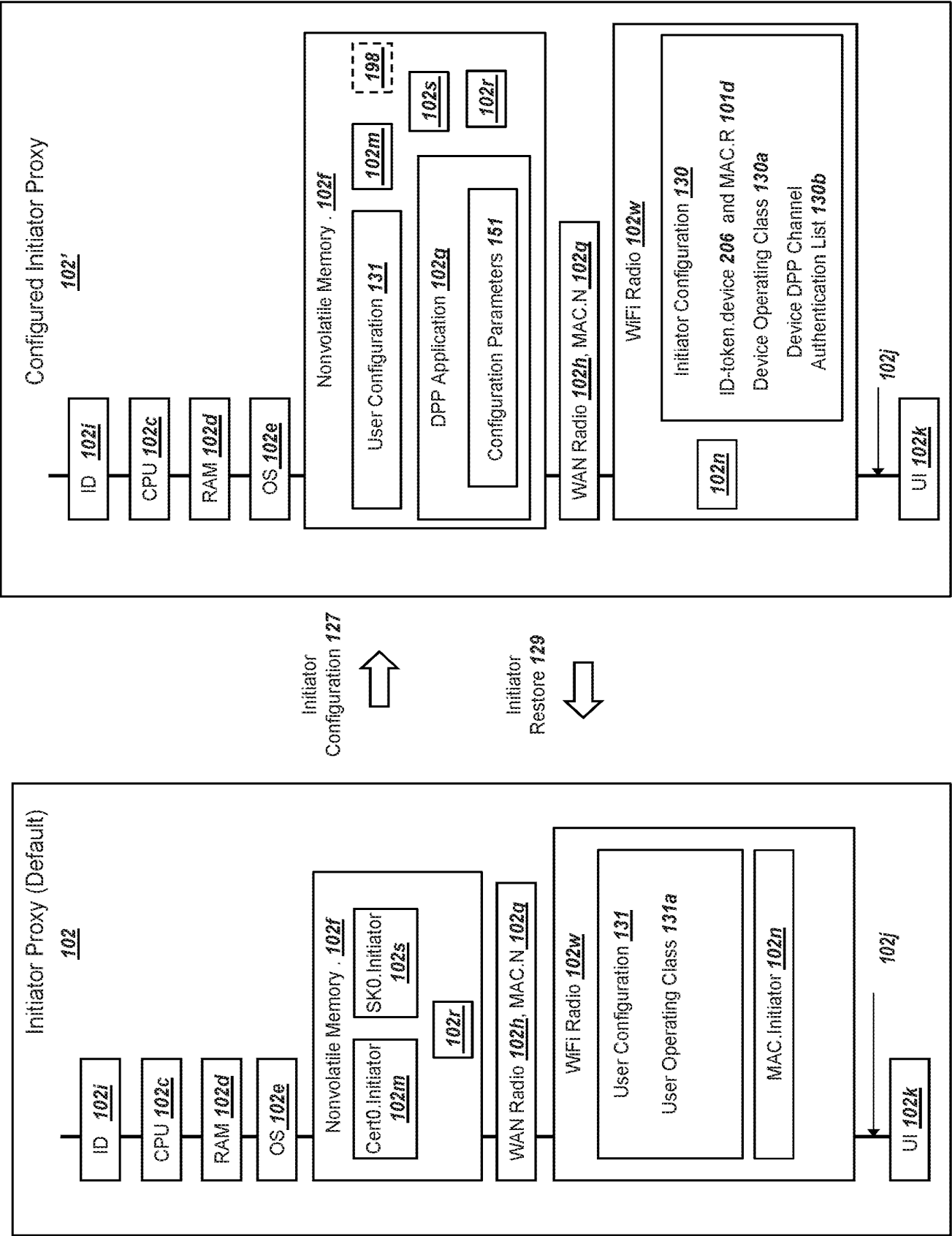


Figure 1d



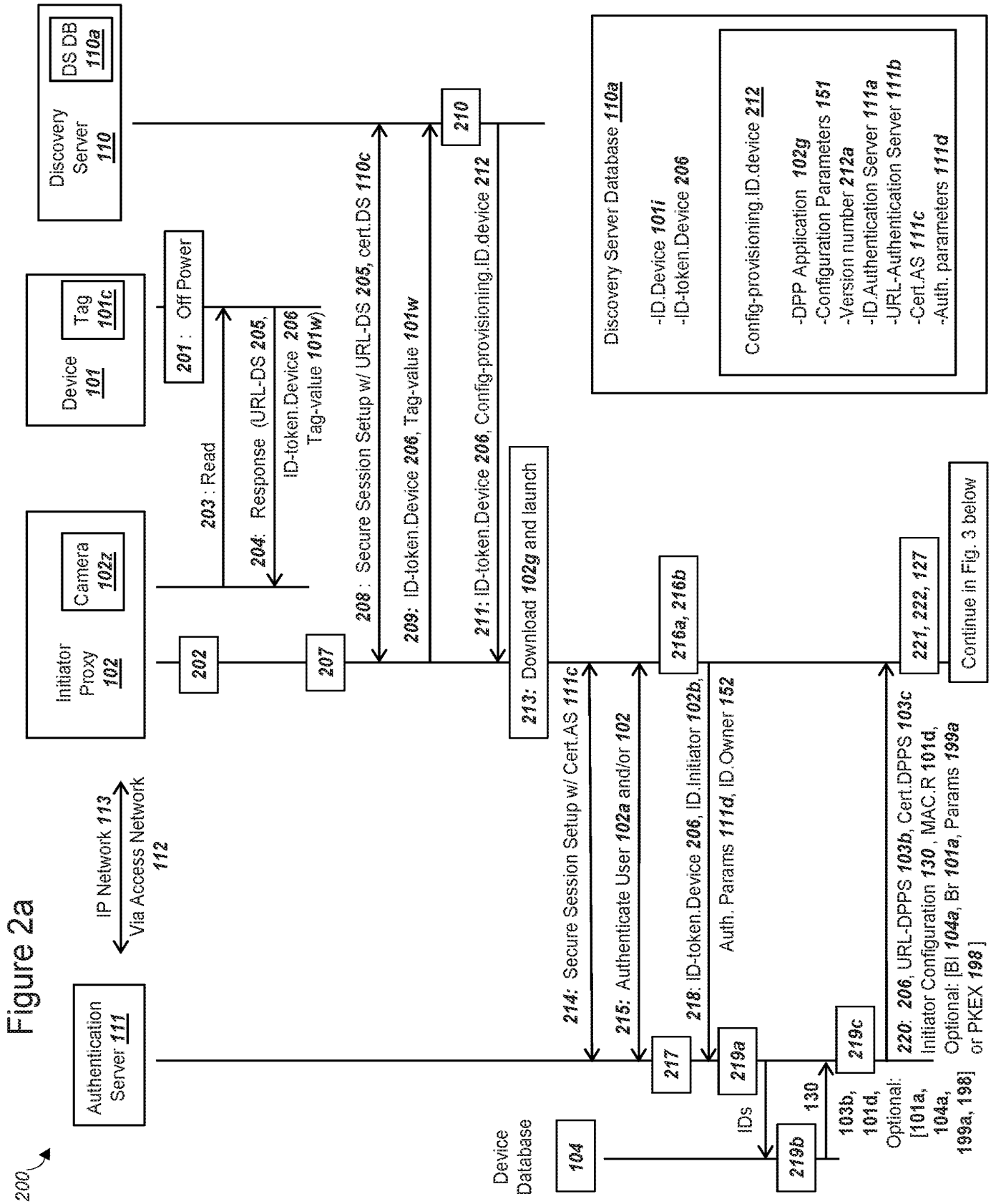


Figure 2b

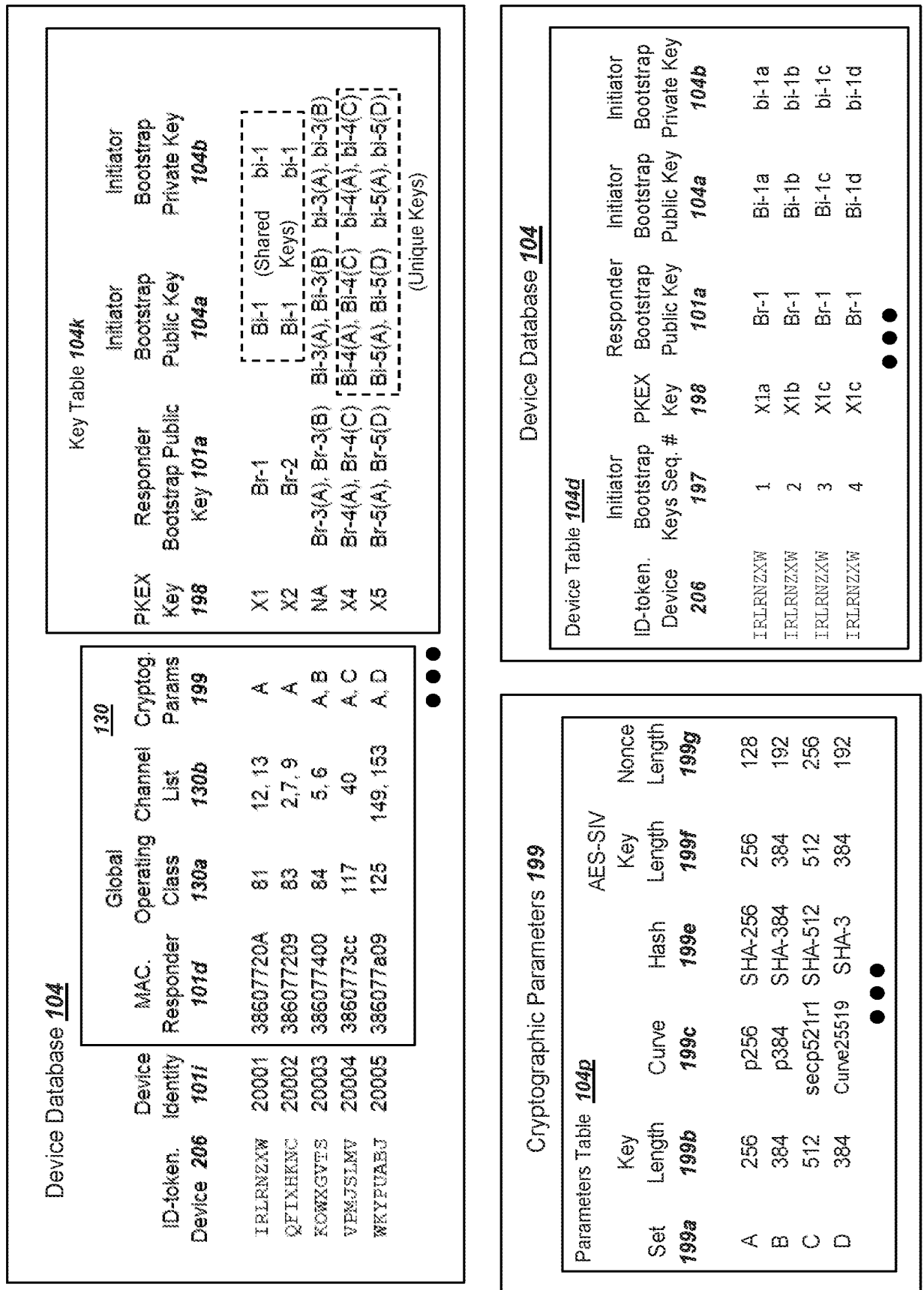


Figure 3b

Server Database 103a												
ID-token. Device 206	MAC. Responder 101d	MAC. Initiator 102q	Initiator Mode 306a	Initiator Cryptog. Params 199a	Initiator		Resp.		Initiator		PKEX	
					Ephem. Public Key 102a	Ephem. Private Key 102b	Ephem. Public Key 101e	Ephem. Private Key 101f	Bootstrap Keys Seq. # 197	Bootstrap keys 399	Shared Secret k1 314a	Shared Secret k2 314c
IRLRNZXW	38607720A	6cf049080	Fig. 3	A	PI-1	pi-1	Pr-1	Pr-1	4	Br-1, Bi-1, bi-1	X1	ke-1
QFIXHKNC	386077209	1c6f654d2	Fig. 5a	A	-	-	Pr-2	Pr-2	1	Br-2, Bi-1, bi-1	NA	-
KOWXGVTS	386077400	7071bce29	Fig. 5b	B	-	pi-3	Pr-3	Pr-3	2	{Br-3, Bi-3, bi-3} B	NA	-
VPMJSLMV	3860773cc	7071bce29	Fig. 3	A	PI-4	pi-4	Pr-4	Pr-4	1	{Br-4, Bi-4, bi-4} A	X4	ke-4
WKYFUABJ	386077a09	7071bce29	102*	D	-	-	-	-	1	{Br-5, Bi-5, bi-5} D	X5	-
● ● ●												

Device Credentials Table 340				
ID- token.Device 206	Selected Network (Primary) 308aa	Selected Network (Backup) 308ab	Primary Network Credentials 109	Backup Network Credentials 109
IRLRNZXW	109a - 1	304a - 3	109 - 1	From xFinity
QFIXHKNC	109a - 1	304a - 6	109 - 1	From LPWAN A
KOWXGVTS	304c - 2	109a - 1	From I-Mobile	109 - 1
VPMJSLMV	304c - 4	109a - 2	From Google	109 - 2
WKYFUABJ	109a - 3	304a - 4	109 - 3	From Google
● ● ●				

Figure 4a

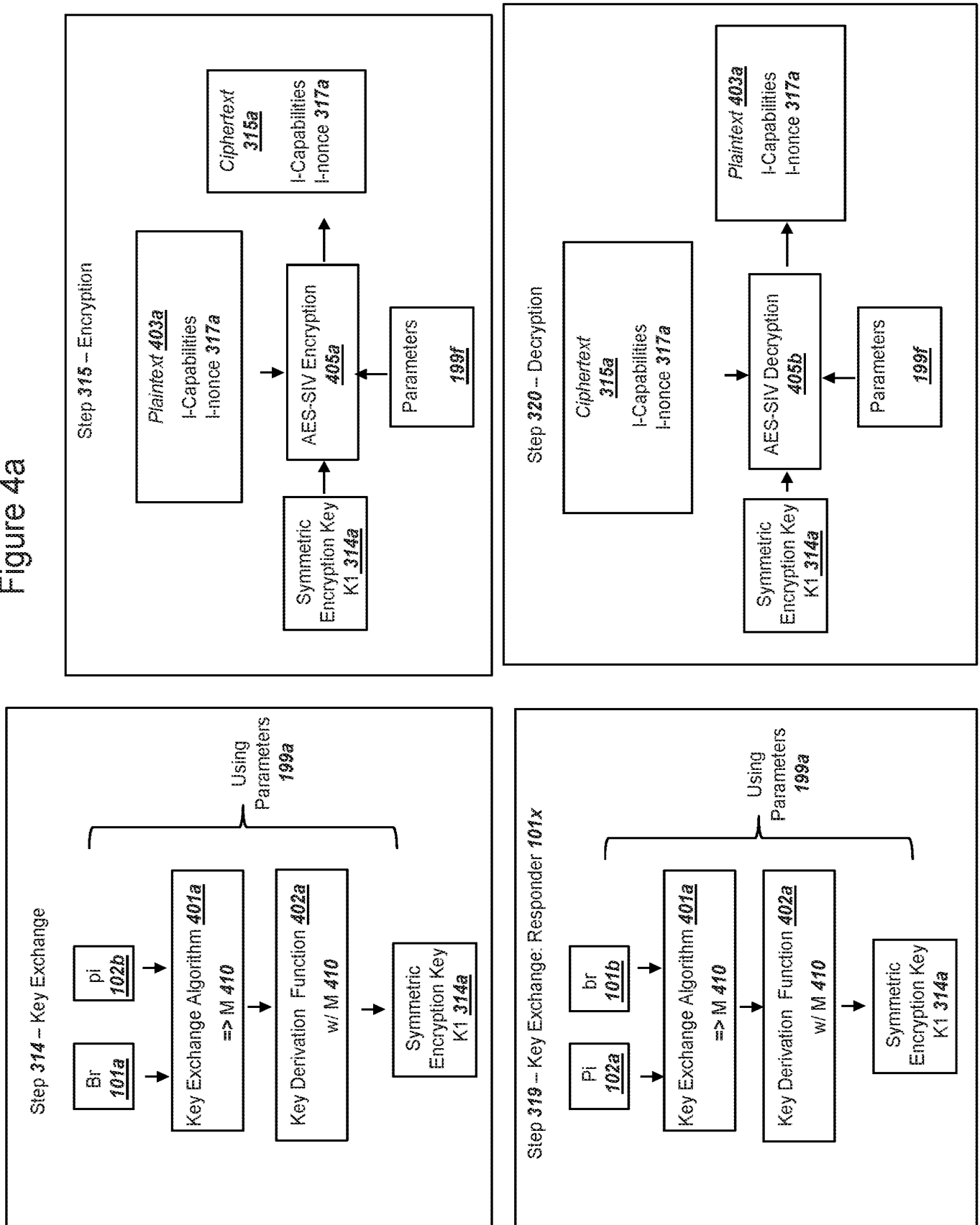


Figure 4b

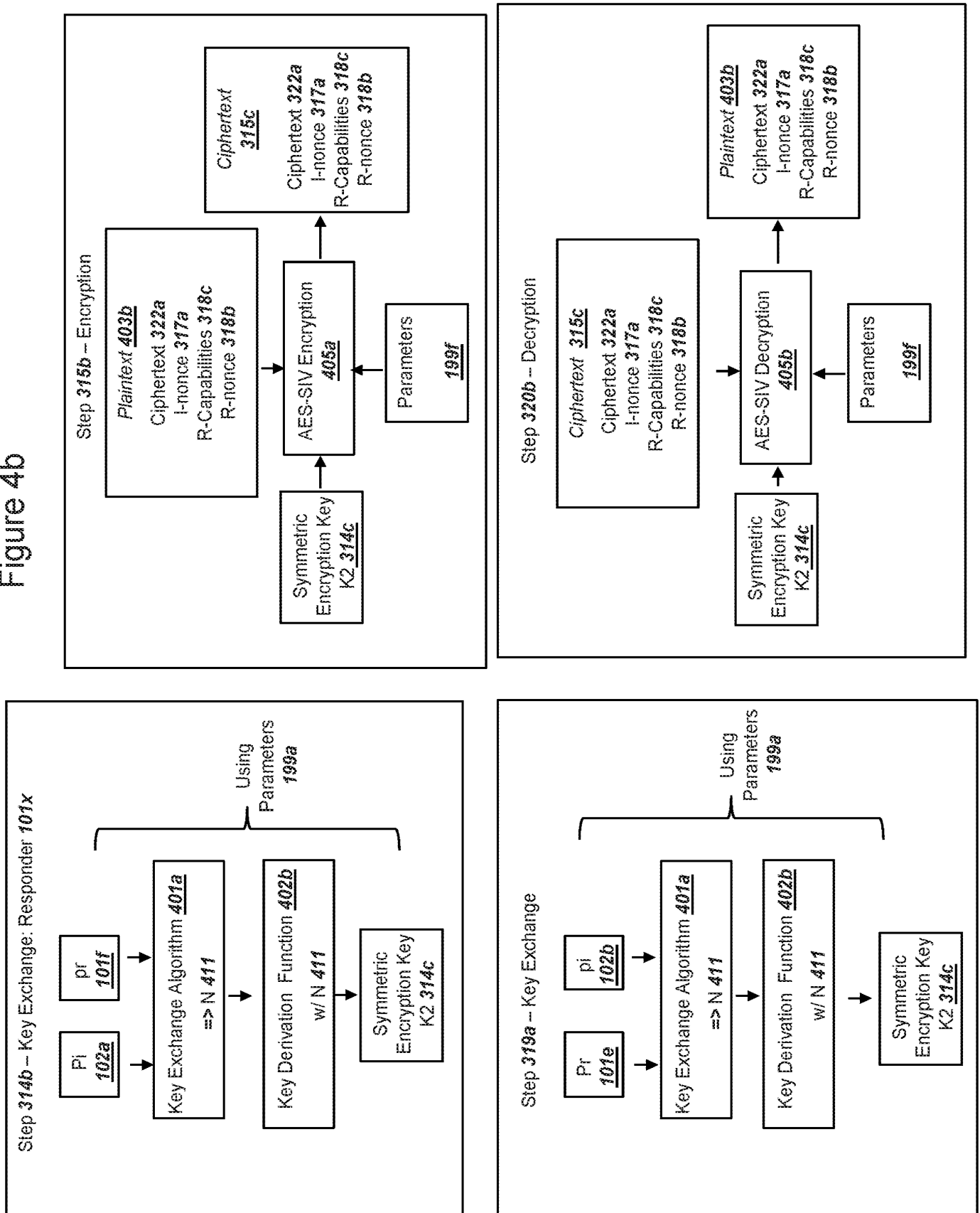
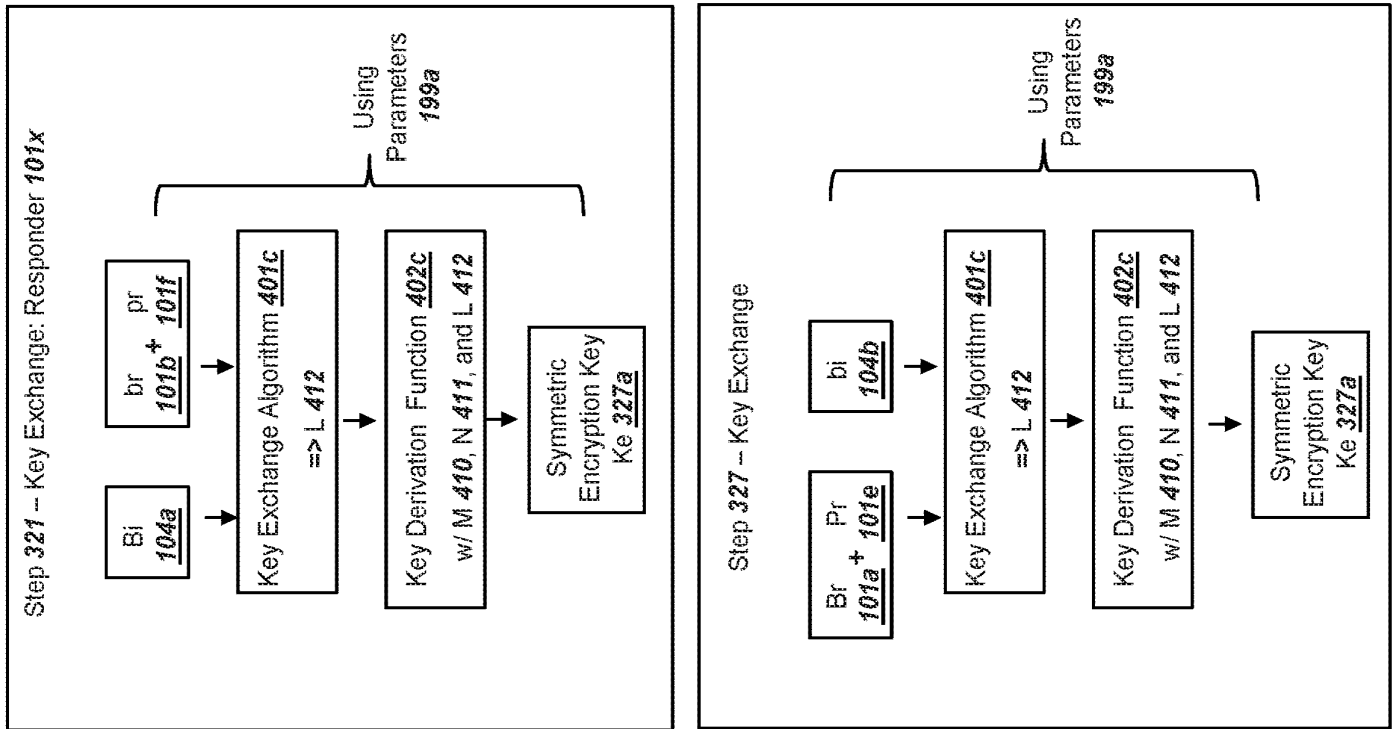


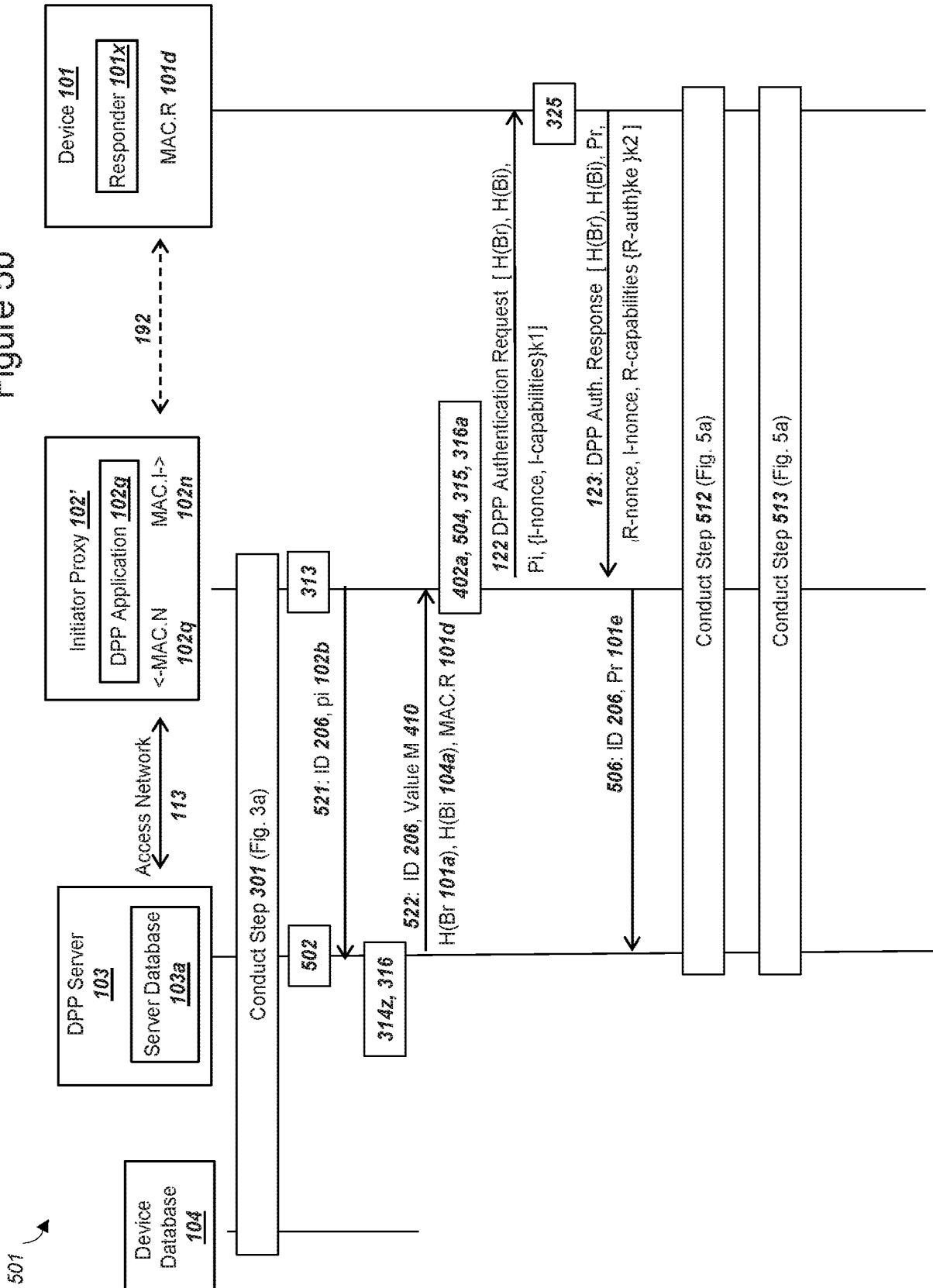
Figure 4c



500



Figure 5b



7.9.6.1

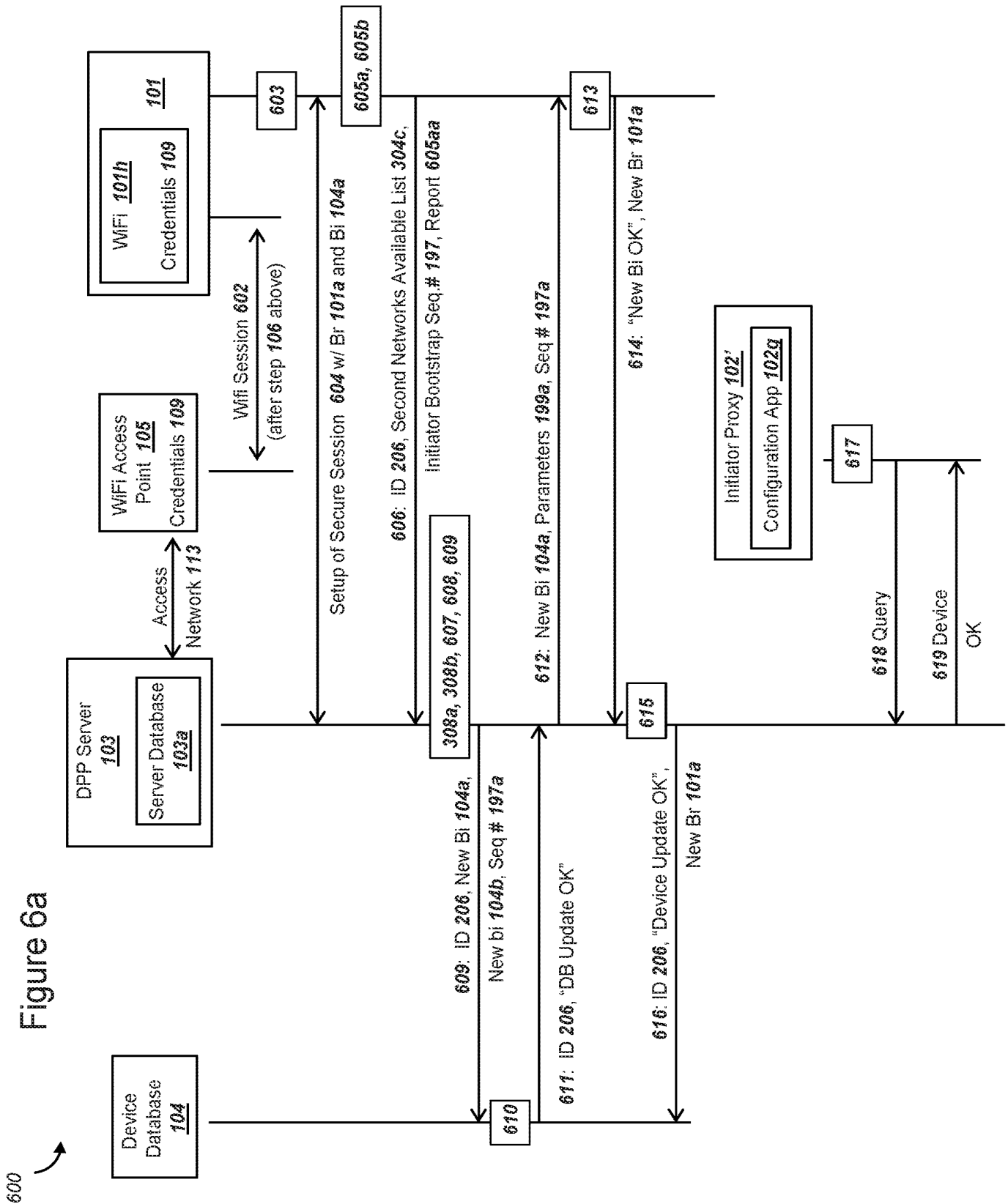


Figure 6b

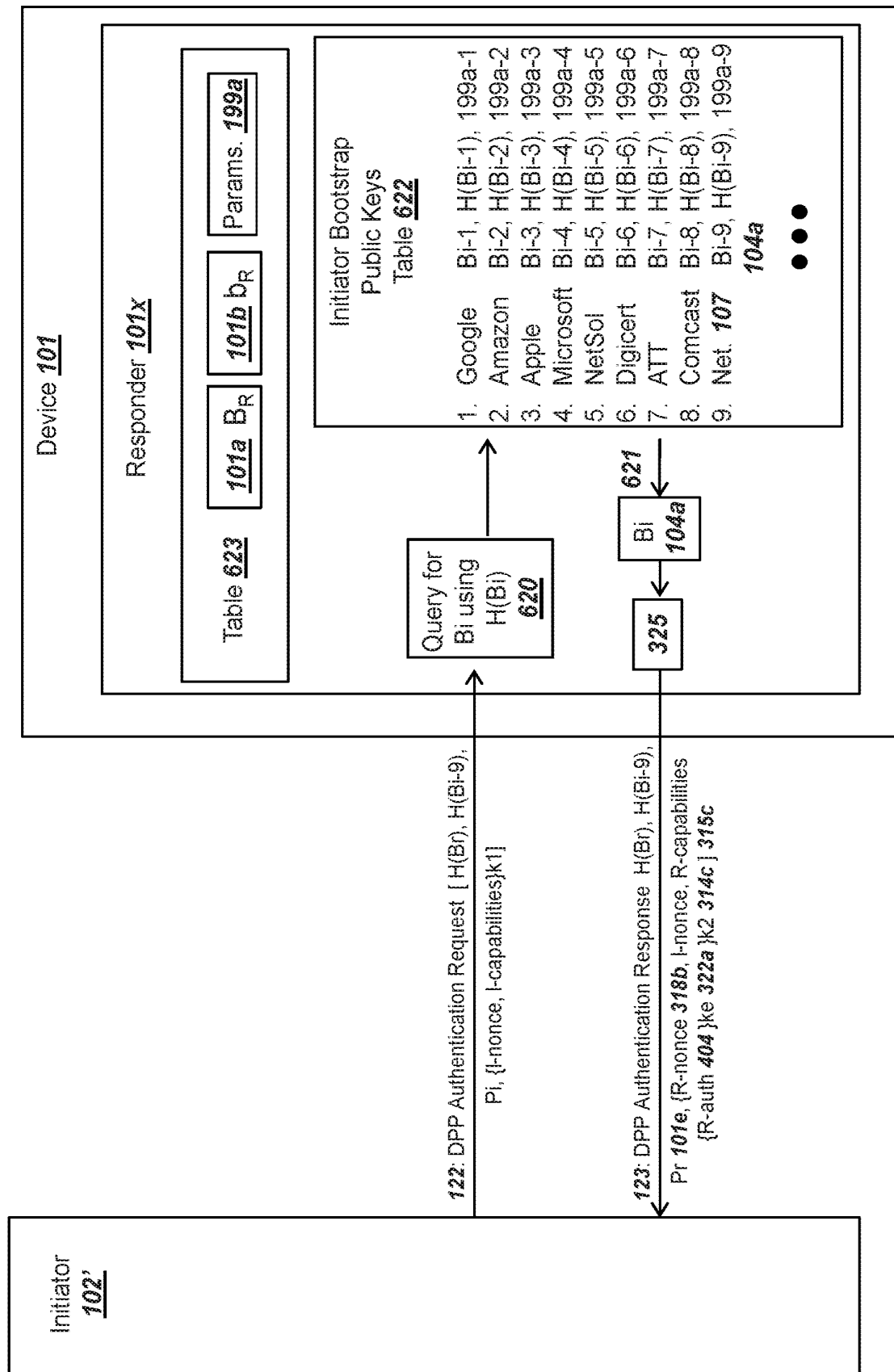


Figure 7

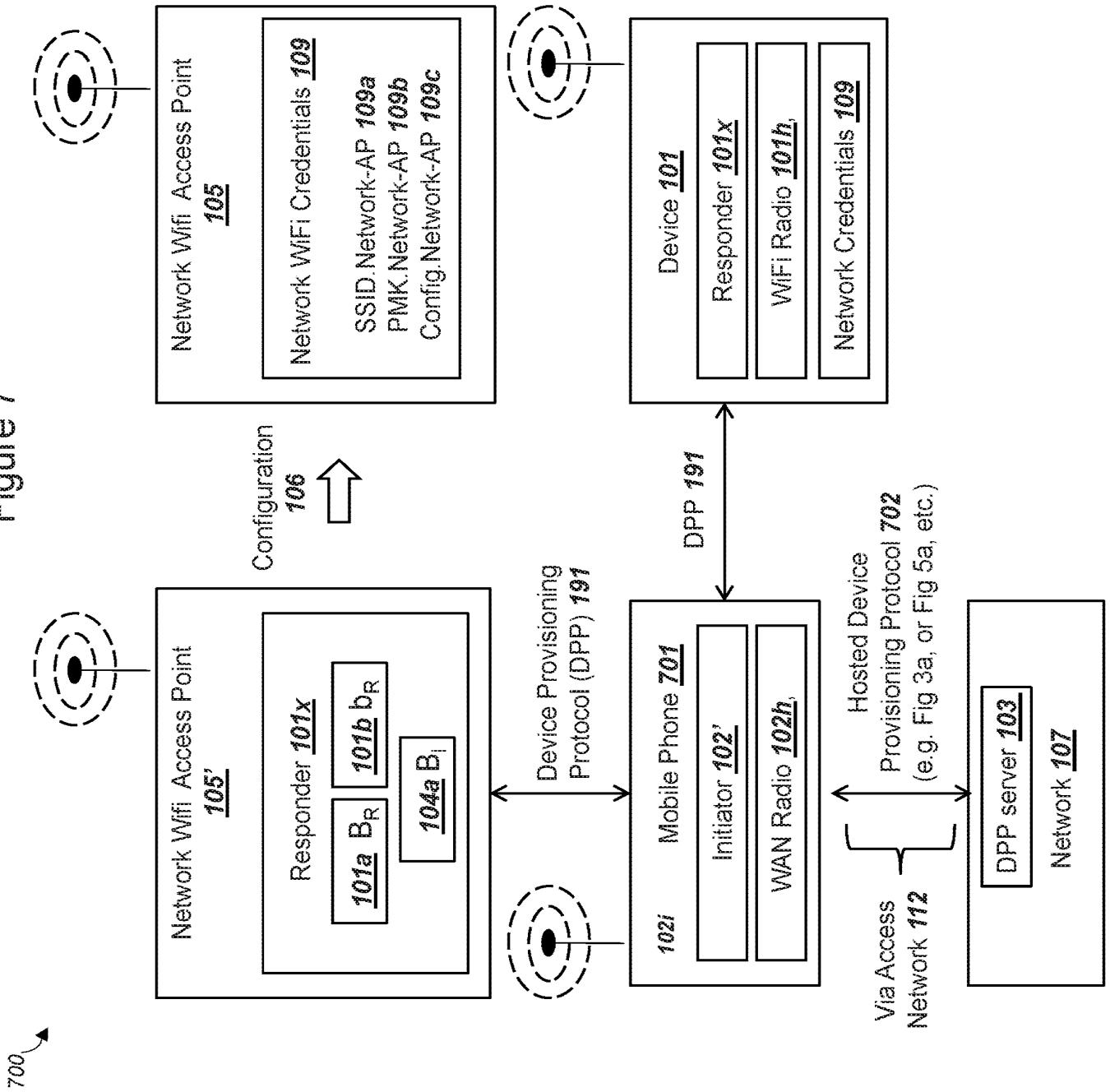
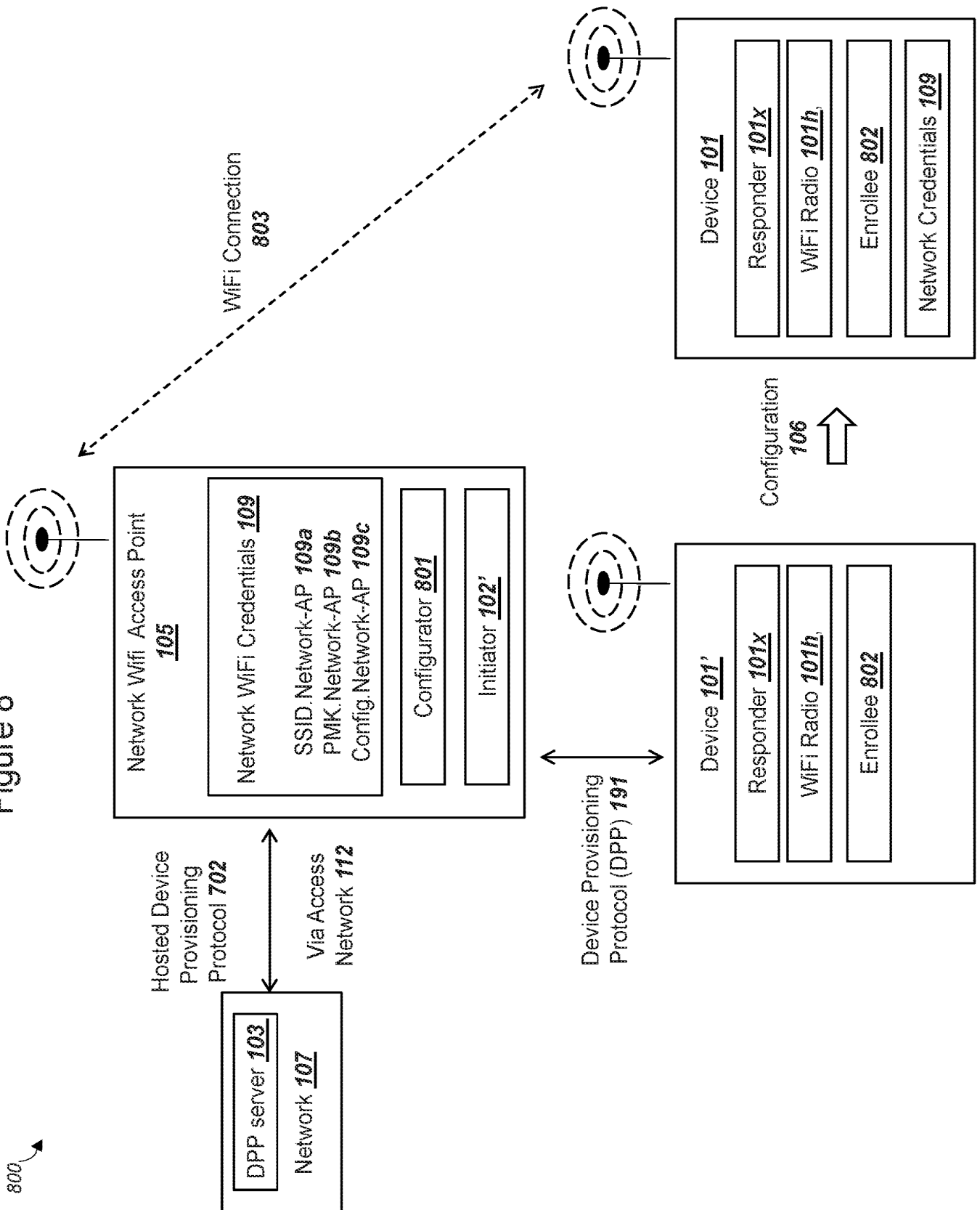
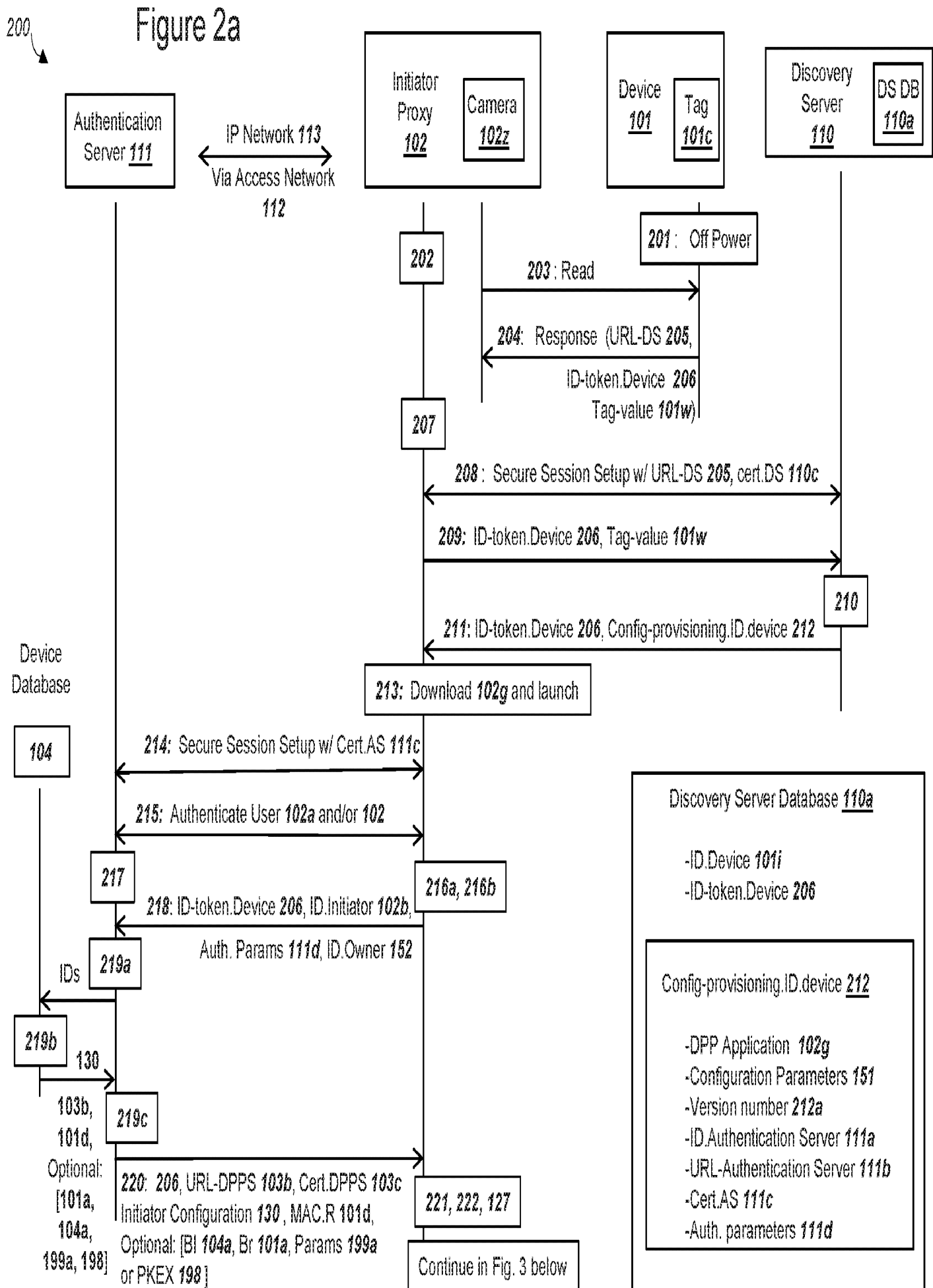


Figure 8







ELECTRONIC ACKNOWLEDGEMENT RECEIPT

APPLICATION #
16/453,682

RECEIPT DATE / TIME
08/05/2021 04:38:23 PM ET

ATTORNEY DOCKET #
63121/0184

Title of Invention

Embedded Universal Integrated Circuit Card Supporting Two-Factor Authentication

Application Information

APPLICATION TYPE Utility - Nonprovisional Application
under 35 USC 111(a)

PATENT # -

CONFIRMATION # 2186

FILED BY Victoria Gilmore

PATENT CENTER # 60272997

FILING DATE 06/26/2019

CUSTOMER # 1912

FIRST NAMED
INVENTOR John A. Nix

CORRESPONDENCE
ADDRESS -

AUTHORIZED BY Albert Boardman

Documents

TOTAL DOCUMENTS: 19

DOCUMENT	PAGES	DESCRIPTION	SIZE (KB)
1-RCE.pdf	3	Request for Continued Examination (RCE)	1318 KB
2-TransmittalLetter.pdf	4	Transmittal Letter	126 KB
3-IDS.pdf	7	Information Disclosure Statement (IDS) Form (SB08)	1011 KB
11-FR10- WO2019178312A1.pdf	137	Foreign Reference	13070 KB
12-FR11- WO2020123959A1.pdf	56	Foreign Reference	5024 KB

13-FR12- WO2020061023A1.pdf	150	Foreign Reference	16324 KB
14-FR13-CA3110641A1.pdf	151	Foreign Reference	16201 KB
15-FR14-CA3110468A1.pdf	147	Foreign Reference	15811 KB
16-FR15-DE19803936A1.pdf	8	Foreign Reference	677 KB
17-FR16- WO2019178312A1.pdf	137	Foreign Reference	13070 KB
18-NPL1- 410824GB2SearchReport.pdf	5	Non Patent Literature	2959 KB
3-FR1-GB2518976B.pdf	66	Foreign Reference	2628 KB
4-FR2-3- WO2021007235A1.pdf	102	Foreign Reference	6382 KB
5-FR4-WO2021087221A1.pdf	69	Foreign Reference	9133 KB
6-FR5-WO2020223319A1.pdf	67	Foreign Reference	7590 KB
7-FR6-WO2020006162A1.pdf	117	Foreign Reference	11304 KB
8-FR7-WO2019246206A1.pdf	83	Foreign Reference	8384 KB
9-FR8-WO2019222319A1.pdf	146	Foreign Reference	15735 KB
10-FR9- WO2019209598A1.pdf	152	Foreign Reference	16134 KB

Digest

DOCUMENT	MESSAGE DIGEST(SHA-512)
----------	-------------------------

1-RCE.pdf	6D365C2942832FD7A73761C772736E4446D1921ED0A621390F0A7C43ADB14429852B315EA5D91BFDE3992E357A1DF18BD028E7FB80FFE68B90F89BC5721EAC65
2-TransmittalLetter.pdf	4E588890A3871F0DC97100A1F7766D110CFA7252D352C06CB E922A558F5BFAF39A56D0B5B3A8143F41D967C99F7099BB0C AAB532673D680A368917B5983BA3A4
3-IDS.pdf	1E9ED4447FEDE8640013A415620355C1F4463231C2C9977D62 A073D7E8E7507493755FBD8A69ECC1B3D2DC21C170FCC7D9 D4A92F4CDD2DB500505D0F54BB22FD
11-FR10- WO2019178312A1.pdf	67B3CACAE9228D13DA26CAFC12618A4764278701C719B49D A1A3AA62BA299CCF853F2A578E64F5E47B67FC24CD58007C8 7A68D34F804E4B909F7913CB0CFD46
12-FR11- WO2020123959A1.pdf	A7931710E66051D7A341E35004DD65842B1C2E983B383CB609 3E28EB73293D48F19F040C9380FB7204F33B056A0960D1D183 094C69071F1DEF4484CCE8C85D52
13-FR12- WO2020061023A1.pdf	3B2861BBFC9F1A0E552B3F36816FD3E2F4B3BA84F241DBB13 BDDD9C04BF3704C37D9407AEE0946858A23D80FB0E72099E3 6E6468EB0D168D53A495B33240AD4C
14-FR13-CA3110641A1.pdf	7B5A682B37470FB66AC94AC129040CE532E2DDCCE0504DAB C74D0AA5BA1B67D0F954E8CE88543505C96EEFD8C8BDF962 03518A0E86E3AED88F9E26BEADB8C560
15-FR14-CA3110468A1.pdf	19B9AB019D2D68392A546C8679C60CA5F5378E539DA6EF996 B9B73F390ED087E049BF1714F7985E200BE82563097041E59F A85426DD329FFB1FFD540BEEAABFE
16-FR15-DE19803936A1.pdf	CEA60D3303C34506C1E0CEAC57AB672023E278B4B6DCA152 9118B1D27888440E9C86D9CCE9DA539EF947B3D5D39102083 3298E0E958263F6B56E713D78702CD5
17-FR16- WO2019178312A1.pdf	CE5DD53F62121F48849D9EC88B805AFA027D39D4304E806CD E7E826C8C51CFAC212B58C6624A01930F05D1FD793ABC9667 D56A065A7F4820452DD2C9D69DCA65

18-NPL1- 410824GB2SearchReport.pdf	ADE352E6209D606739804B15E560FC300062362BDF3B419BD CDD91BA95D523D465940BF653EB394914A9B143EA3B09F622 BC8DDEE28A74C173F76FAE3DF3CB10
3-FR1-GB2518976B.pdf	BDD7439F26A27BB1111BA62C76164834BC888F4B97103B0AC 65805F31E1A971AA6A5DEC2A95862485F92489677ADF747FC7 C2535381E8A48F27EE70B65C9EBE2
4-FR2-3-WO2021007235A1.pdf	F3E2484234084812992A190BBD2BDC2E0D018E597F0D18D9A 493005896A7EB2F8F72F2E1C1273D6B94EBAC8196DAC3C70E B94B36C7B118B96CFEBFE469455B0E
5-FR4-WO2021087221A1.pdf	DFD0ECC4C8F7F1254EE8D93660E29FC3E9B5254EF919886B4 79DC05BDBBC134C2413EBC76723EB1677DEFF34052EB53B6 6DBE1C8A80DB71125F7F17A126D1C10
6-FR5-WO2020223319A1.pdf	B340F4C43BBB0C01C66D8EF2C83ADC1587848BD8CB5FE88C 77BBBC1E44FCFEBA9EC1942F7D8D7821DDAD713FD99786A8 02290E0AEC65DA9266094F7E5FFE5B1E
7-FR6-WO2020006162A1.pdf	1DA7C43830E6BF48EAF992CD378141AD2BBC274AD676C80C CC555DF8116E6A546916E114CFCAF785A4D1C2CBE0E0A9D7 48A8CE1B1DD4DEEC8B5A60337E1F5F6E
8-FR7-WO2019246206A1.pdf	DD13C79EC19A546342D0BA748F0B922E7961EB298E275FFC6 F775EB195942A520D52C76308687CDECC9646AC49D14BBFD6 5825B97F875FDEA3BC43BFF5FBF057
9-FR8-WO2019222319A1.pdf	9EF635008ED3B0B2C722E4767E083FC03A1D7ACBEBE862E7 B9B315883CC5BCEBDFB98829E9F5F0B949F0AFDA69AD2420 DBC4848DD981E765C17B38A27758D8CB
10-FR9-WO2019209598A1.pdf	FB2F8F95B442D9D870CD747F92922AEF0135AA4462F307AA6 E2FA97581F493C91628B746FBB5611482685154BF3CC0F7E59 8D0B1B87E028AF372BE054691EE89

This Acknowledgement Receipt evidences receipt on the noted date by the USPTO of the indicated documents, characterized by the applicant, and including page counts, where applicable. It serves as evidence of receipt similar to a Post Card, as described in MPEP 503.

New Applications Under 35 U.S.C. 111

If a new application is being filed and the application includes the necessary components for filing date (see 37 CFR 1.53(b)-(d) and MPEP 506), a Filing Receipt (37 CFR 1.54) will be issued in due course and the date shown on this Acknowledgement Receipt will establish the filing date of the application.

National Stage of an International Application under 35 U.S.C. 371

If a timely submission to enter the national stage of an international application is compliant with the conditions of 35 U.S.C. 371 and other applicable requirements a Form PCT/DO/EO/903 indicating acceptance of the application as a national stage submission under 35 U.S.C. 371 will be issued in addition to the Filing Receipt, in due course.

New International Application Filed with the USPTO as a Receiving Office

If a new international application is being filed and the international application includes the necessary components for an international filing date (see PCT Article 11 and MPEP 1810), a Notification of the International Application Number and of the International Filing Date (Form PCT/RO/105) will be issued in due course, subject to prescriptions concerning national security, and the date shown on this Acknowledgement Receipt will establish the international filing date of the application.



19 BUNDESREPUBLIK
DEUTSCHLAND



DEUTSCHES
PATENT- UND
MARKENAMT

12 Offenlegungsschrift
10 DE 198 03 936 A 1

51 Int. Cl.⁶:
H 01 L 33/00
H 01 L 31/101
H 01 L 31/0203
H 01 L 31/0232

21 Aktenzeichen: 198 03 936.0
22 Anmeldetag: 30. 1. 98
43 Offenlegungstag: 5. 8. 99

DE 198 03 936 A 1

71 Anmelder:
Patent-Treuhand-Gesellschaft für elektrische
Glühlampen mbH, 81543 München, DE; Siemens
AG, 80333 München, DE

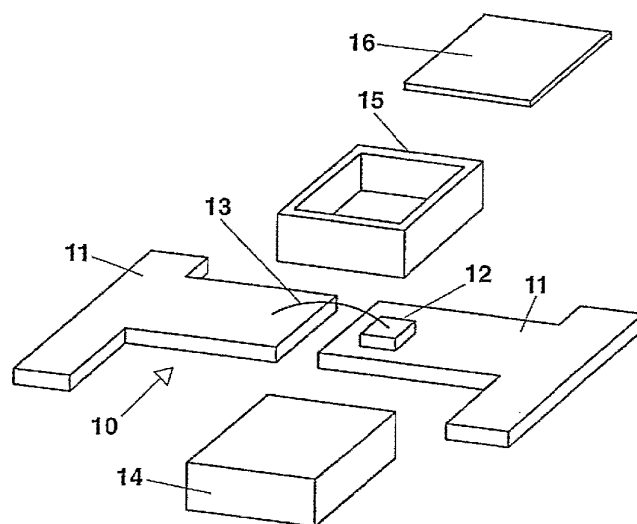
74 Vertreter:
Pokorny, G., Rechtsanwalt., 81543 München

72 Erfinder:
Waitl, Günter, 93049 Regensburg, DE; Langer,
Alfred, 86438 Kissing, DE; Weitzel, Reinhard, Dr.,
82216 Maisach, DE

Die folgenden Angaben sind den vom Anmelder eingereichten Unterlagen entnommen

54 Ausdehnungskompensiertes optoelektronisches Halbleiter-Bauelement, insbesondere UV-emittierende
Leuchtdiode und Verfahren zu seiner Herstellung

57 Bei einem optoelektronischen Halbleiter-Bauelement
ist ein strahlungsemittierender oder -empfangender
Halbleiterkörper (Chip) auf einem elektrisch leitenden Lei-
terrahmen befestigt und von einem Gehäuse umgeben.
Alle eingesetzten Gehäusematerialien und der Leiterrah-
men weisen im Temperaturbereich, der in Herstellung
und Anwendung auftritt, aneinander angepaßte thermi-
sche Ausdehnungskoeffizienten auf.



DE 198 03 936 A 1

Die Erfindung betrifft ein ausdehnungskompensiertes optoelektronisches Halbleiter-Bauelement gemäß dem Oberbegriff des Anspruchs 1 und ein Verfahren zur Herstellung des Halbleiter-Bauelements. Es handelt sich dabei beispielsweise um Leuchtdioden, die im Bereich von 320 bis 1600 nm emittieren, vorzugsweise um UV-emittierende Leuchtdioden mit Lumineszenzkonversion in weißes Licht. Deren Einsatzgebiet ist hauptsächlich die Allgemeinbeleuchtung. Aber auch reine UV-Anwendungen ohne Lumineszenzkonversion sind möglich. Als Halbleiter wird meist ein Nitrid des Galliums u/o Indiums u/o Aluminiums verwendet.

Stand der Technik

Ein optoelektronisches Halbleiter-Bauelement wie beispielsweise eine Leuchtdiode ist in seiner Anwendungsmöglichkeit eingeschränkt, ohne daß dieser Mangel von der Fachwelt bisher so empfunden wird. Scheinbar ist Kunststoff ideal als Gehäusewerkstoff geeignet, da er leicht verarbeitet werden kann. Weitverbreitet ist die Gießharztechnologie mit Epoxid. Die scheinbar ideale Eignung des Kunststoffs als Gehäusewerkstoff verdeckte bisher den Mangel einer fehlenden Abstimmung seines Ausdehnungsverhaltens bei Temperaturwechselbelastung. Durch die derzeit übliche Verwendung eines Kunststoffgehäuses ist die Arbeitstemperatur auf den Bereich von -55°C bis $+110^{\circ}\text{C}$ eingeschränkt. Außerdem scheiterte ein Einsatz bei sehr kurzen Wellenlängen (im UV-Bereich) bisher an der starken Degradation des Gehäuses.

Für die Erzeugung weißen Lichts für Zwecke der Allgemeinbeleuchtung kommen neuerdings weißes Licht emittierende Leuchtdioden in Frage. Sie basieren auf der Erzeugung des weißen Lichts mittels einer blau oder auch UV-emittierenden Leuchtdiode. Übliche Konzepte zur Realisierung basieren auf radialen Bauformen, die für die Durchsteckmontage (beispielsweise sog. LUCOLED-Design) geeignet sind, oder SMT-geeigneten Bauformen, dem sog. Topped-Design für die Oberflächenmontage (SMT-LED). Genaueres hierzu findet sich beispielsweise in dem Artikel "White-light diodes are set to tumble in price" von Philip Hill, OLE Oktober 1997, S. 17 bis 20. Beim LUCOLED-Konzept wird beispielsweise von einem blauen Emittier auf Basis GaN mittels Lumineszenzkonversion weißes Licht erzeugt.

In dem Artikel "High Power UV InGaN/AlGaIn Double Heterostructure LEDs" von Mukai, Morita und Nakamura, ist der Aufbau einer UV-emittierenden Leuchtdiode mit einer Emissionswellenlänge von etwa 370 nm beschrieben. Die Leuchtdiode weist einen Chip auf, der auf einem Leiterahmen montiert ist und in einem Kunststoffgehäuse eingegossen wird.

In dem Artikel "Reliability Behavior of GaN-based Light Emitting Diodes" von D. Steigerwald, werden Leuchtdioden verschiedener Farbe untersucht. Es wurde festgestellt, daß die Degradation der Leuchtdiode mit kürzerer Emissionswellenlänge bis herab zu 470 nm stark zunimmt. Entscheidenden Anteil an der Degradation haben die Faktoren Betriebsstrom und Umgebungstemperatur sowie das Gehäuse aus Kunststoff. Zur Untersuchung wurde das Epoxidharz-Gehäuse vorübergehend entfernt.

Insgesamt erscheint daher der Betrieb einer UV-emittierenden Leuchtdiode bisher wenig erfolgversprechend, da die UV-Strahlung das Gehäuse schädigt. Bei Verwendung

eines blau emittierenden Leuchtdiode als Lichtquelle ist jedoch andererseits die Lichtausbeute und Effizienz relativ niedrig (ca. 5 lm/W).

Frühere optoelektronische Halbleiter-Bauelemente hatten als Gehäuse ein Metall-Glas-System, wobei in einer Metallkappe eine Glaslinse eingepaßt wurde (siehe PCT-A PCT/DE96/01728). Damit konnte zwar höheren Anforderungen an die optischen Eigenschaften entsprochen werden, aber die Herstellkosten für die Bodenplatte und die Linsenkappe sind unverhältnismäßig hoch, die Fertigung ist sehr aufwendig und die hohen Toleranzen in Fertigung und Justierung erlauben nur bedingt den Einsatz dieser Technologie. Die optische Perfektion des Systems ist daher für viele Applikationen nicht ausreichend.

Darstellung der Erfindung

Es ist Aufgabe der vorliegenden Erfindung, ein optoelektronisches Halbleiter-Bauelement bereitzustellen, das sich für einen erweiterten Einsatzbereich hinsichtlich Temperatur und Umgebungsfeuchtigkeit eignet und das in einem Wellenlängenbereich von etwa 300 nm bis 1600 nm arbeiten kann.

Diese Aufgabe wird durch die kennzeichnenden Merkmale des Anspruchs 1 gelöst. Besonders vorteilhafte Ausgestaltungen finden sich in den abhängigen Ansprüchen.

Im einzelnen handelt es sich bei der vorliegenden Erfindung um ein optoelektronisches Halbleiter-Bauelement, bei dem ein strahlungsemitterender oder -empfangender Halbleiterkörper auf einem elektrisch leitenden Leiterahmen befestigt ist und von einem gasdichten Gehäuse umgeben ist. Alle eingesetzten Gehäusematerialien und der Leiterahmen weisen im Temperaturbereich, der in Herstellung und Anwendung auftritt, aneinander angepaßte thermische Ausdehnungskoeffizienten auf.

Vorteilhaft hat das Gehäuse einen Grundkörper mit einer Ausnehmung, wobei der Grundkörper an dem Leiterahmen gasdicht befestigt ist. Der Halbleiter-Chip ist in der Ausnehmung auf dem Leiterahmen befestigt. Die Ausnehmung ist mit einer Abdeckung gasdicht verschlossen.

Der Grundkörper u/o die Abdeckung ist aus Glas oder Quarzglas oder Keramik oder Glaskeramik gefertigt. Die einzelnen Materialien sind im Temperaturbereich von -60°C bis 150°C optimal aufeinander abgestimmt. Damit läßt sich die Junctiontemperatur T_j von derzeit 100°C auf 130°C und mehr erhöhen. Dadurch sind LEDs im Außenbereich bis zu einer Umgebungstemperatur von etwa 100°C einsetzbar, so daß sie sich für Automobilanwendungen oder Outdoor-Informationssysteme eignen.

Das erfindungsgemäße Halbleiter-Bauelement besitzt den Vorteil, daß konventionelle Methoden der Befestigung unter Hochtemperaturbedingungen für den Halbleiterchip auf dem Leiterahmen angewendet werden können. Bei den derzeit verwendeten Kunststoffgehäusen ist dies aufgrund der Temperaturempfindlichkeit von Kunststoff nicht möglich. Statt dessen werden hier Leitleber, die beispielsweise Ag enthalten, verwendet.

Ein weiterer Vorteil des erfindungsgemäßen Bauelements besteht darin, daß eine höhere Stabilität erzielt wird, weil Grenzschichteffekte zwischen Halbleiterchip und Kunststoffumhüllung ausgeschlossen sind. Weder im Langzeitbetrieb noch beim Löten tritt Delamination auf, so daß die Lichtauskopplung stabilisiert ist.

Ein besonderer Vorteil der vorliegenden Erfindung besteht insbesondere darin, daß damit erstmals grundsätzlich die Möglichkeit geboten wird, eine UV-emittierende Leuchtdiode bereitzustellen, die eine hohe Lichtausbeute und Effizienz besitzt und die sich preisgünstig herstellen

läßt. Dies liegt nicht nur an der grundsätzlichen fertigungs-
freundlichen Konzeption, sondern auch daran, daß unter den
jetzt zum Einsatz kommenden Materialien sich auch solche
mit hoher Transparenz im UV und Beständigkeit gegen UV-
Strahlung befinden, beispielsweise Quarzglas. Für Kunst-
stoffe ist diese Anforderung praktisch nicht zu erfüllen.

Des weiteren tritt keine mechanische Zug- und Druckbe-
anspruchung des Halbleiterchips auf. Folglich gibt es keine
streßinjizierte Degradation.

Die heute bei Kunststoff auftretende Vergilbung durch
Temperatur und UV-Einwirkung, die eine Verringerung der
Lichtleistung im sichtbaren Spektrum zur Folge hat, tritt bei
dem erfindungsgemäßen Bauelement nicht auf.

Der Leiterraum besteht vorteilhaft aus an sich bekanntem
Kupfer-Manteldraht oder einem Material mit ähnlichem
Ausdehnungsverhalten, beispielsweise Ni-Fe-Legierungen.
Unter Kupfer-Manteldraht wird ein Draht verstanden, des-
sen Seele aus einer Ni-Fe-Legierung und dessen Mantel aus
Kupfer besteht.

Meist ist das Gehäuse mehrteilig aus Einzelteilen zusam-
mengesetzt, wobei die Einzelteile gasdicht miteinander ver-
bunden sind, insbesondere durch direkte Verschmelzung
oder durch Klebemittel. Dabei eignen sich besonders org.
oder anorg. Kleber, beispielsweise Wasserglas.

Die Abdeckung auf dem Rahmenteil sitzt entweder groß-
flächig über die Oberfläche des Rahmens einschließlich der
Ausnehmung oder sie ist nur im Bereich der Ausnehmung
positioniert.

Vorteilhaft wird als Material für das Gehäuse niedrig-
schmelzendes Glas verwendet, vorteilhaft Weichglas, insbe-
sondere Bleiglas oder Alkaliglas. Es ist typisch ab 400°C
verarbeitbar. Der thermische Ausdehnungskoeffizient liegt
etwa bei $8 \text{ bis } 11 \times 10^{-6}/\text{K}$.

Die Differenz zwischen den thermischen Ausdehnungs-
koeffizienten einzelner Teile des Gehäuses sollte nicht grö-
ßer als $\pm 15\%$ sein. In diesem Sinne sind obige Komponen-
ten gut aufeinander abgestimmt, wenn beispielsweise als
Leiterrahmen bzw. Metalledurchführung eine Einschmelzle-
gierung des Typs VACOVIT oder VACON oder VACODIL
verwendet wird. Diese enthalten Ni, Fe als Hauptbestand-
teile der Legierung, denen Cr oder Co zugesetzt sein kann
(auch für Keramik geeignet).

Die Herstellung des erfindungsgemäßen optoelektrischen
Halbleiter-Bauelements kann auf verschiedene Weise
erfolgen. Ein vorgestanztes Leiterband kann mit Preßglas-
teilen oder Sinterglasteilen beaufschlagt werden, die dann
miteinander verklebt werden. Oder es werden von einem
Glasrohling (Stab) Einzelstücke abgeschnitten, mit dem
Leiterrahmen in Kontakt gebracht, erhitzt und formgepreßt.
Auf diese Weise wird der Leiterraum direkt in das Ge-
häuse eingebettet und die Einzelteile des Gehäuses werden
miteinander verschmolzen.

Ein Sockel/Kappen-Aufbau kann einstückig realisiert
werden. Oft ist es aber fertigungsfreundlicher, diesen Auf-
bau zweistückig zu realisieren., wobei auch hier ein Ver-
schmelzen oder Verkleben (beispielsweise Silikonkleber) in
Frage kommt.

Das erfindungsgemäße Konzept eignet sich hervorragend
zur Schaffung einer LED als Glühlampenersatz. Dabei ist
insbesondere an eine Flächenleuchte gedacht, die aus mehr-
eren LEDs als Einzelpixel eines Arrays aus mehreren Rei-
hen und Spalten zusammengesetzt ist. Vorteilhaft werden
die Einzelpixel gesondert elektronisch angesteuert. Dabei
kann die Flächenleuchte nicht nur als Lichtquelle, sondern
teilweise (beispielsweise eine Reihe des Arrays) oder ganz
als Informationsmedium eingesetzt werden, indem LEDs
ein Piktogramm o. ä. formieren. Eine andere Realisierung
ist eine kugelförmige Leuchte, die innen einen Zuleitungs-

stab besitzt, wobei mehrere Einzelpixel um den Stab herum
gruppiert sind.

Für den Einsatz für Allgemeinbeleuchtungszwecke kann
die Flächenlampe so hergestellt werden, daß eine Anzahl
Grundelemente (Einzelpixel), die zu einer Lampe zusam-
mengefaßt werden, vollkonfektioniert oder teilkonfektio-
niert ohne Abdeckung bzw. ohne Linse auf ein Substrat
(Träger) aufgebracht und verschaltet werden. Sie werden
von einem gemeinsamen Gehäuse umgeben. Eine etwaige
Lumineszenzkonversion kann hier durch eine gemeinsame
Abdeckung realisiert werden. Hierdurch kann eine extrem
flache Bauweise erzielt werden. Die Fläche des Trägers
kann beliebig geformt sein, beispielsweise als Kugel oder
Rundstab (s. o.). Da das Basiselement ein Halbleiter-Bauteil
ist, kann die benötigte Ansteuerlektronik auf dem gleichen
Substrat integriert werden, beispielsweise mittels Siebdruck
oder SMT. Auch Dimmung ist über einen weiten Bereich
möglich. Flächenhafte Lampen können durch direkte Chip-
montage in erfindungsgemäß hergestellten Multichip-Ge-
häusen hergestellt werden. Dabei sind auch Module mög-
lich, die weißes Licht durch drei Einzelkomponenten (Rot,
Grün, Blau) erzeugen. Schließlich ist hiermit auch eine ge-
zielte stufenlos regelbare Einstellung der Farbtemperatur
möglich. Als Beispiel wird auf eine Grundbaueinheit mit
weißem Licht verwiesen, der weitere einzelne Komponen-
ten, die rote oder blaue Anteile zur Senkung oder Erhöhung
der Farbtemperatur beitragen, zugemischt werden.

Figuren

Im folgenden soll die Erfindung anhand mehrerer Aus-
führungsbeispiele näher erläutert werden. Es zeigen:

Fig. 1 ein Halbleiter-Bauelement, im Schnitt

Fig. 2 ein weiteres Ausführungsbeispiel eines Halbleiter-
Bauelements, in perspektivischer Explosionsdarstellung

Fig. 3 ein Ausführungsbeispiel einer Flächenleuchte

Fig. 4 ein Ausführungsbeispiel einer Lampe

Beschreibung der Zeichnungen

In **Fig. 1** ist eine Leuchtdiode **1** gezeigt in Schnitt (**Fig. 1a**) und Draufsicht (**Fig. 1b**). Sie weist einen Halbleiterchip **2** auf, der auf einem Leiterraum **3** befestigt ist. Der Chip **2** ist elektrisch leitend mit zwei durch ein Bodenteil **4** hindurchgeführten Elektrodenanschlüssen **5**, die Bestandteile des Leiterraums sind, verbunden. Ein Kontaktdraht **6** bewirkt die Verbindung des Chips **2** mit dem einen Elektrodenanschluß **5a**, während die elektrische Verbindung mit dem anderen Elektrodenanschluß **5b** durch Bondverbindung der elektrisch leitenden Unterseite des Chips mit der Trägerfläche, die einstückig mit dem Elektrodenanschluß **5b** ausgebildet ist, bewerkstelligt wird. Die zur optischen Abbildung des Chips vorgesehene Linse **7** ist als oberes Abschlußteil einer hohlzylindrischen Kappe **8** ausgebildet. Die Kappe **8** umgibt einen wesentlichen Teil des Bodenteils **4**.

Das Bodenteil wie auch die Kappe sind beispielsweise aus Bleiglas der folgenden Zusammensetzung hergestellt: SiO_2 60–65 Gew.-%; PbO 20–22 Gew.-%; K_2O 4–10 Gew.-%; Na_2O 4–7 Gew.-%. Ein anderes Bleiglas hat die Zusammensetzung SiO_2 46–50 Gew.-%; PbO 37–42 Gew.-%; K_2O 0.5–5 Gew.-%; Na_2O 7–13 Gew.-%; Al_2O_3 0–2 Gew.-%.

Es eignen sich auch bleifreie Gläser als Ersatz für Bleiglas, wie sie beispielsweise in US 5 391 523, EP-A 734 051, EP-A 603 933 und DE-A 195 47 567 beschrieben sind. Dort sind auch weitere geeignete Durchführungen beschrieben.

Fig. 2a zeigt ein weiteres Ausführungsbeispiel einer Leuchtdiode in Topled-Ausführung. Der Leiterraum **10** besteht aus zwei metallischen Bändern **11**, von denen eines

in ans sich bekannter Weise mit dem Chip **12** bestückt ist. Ein Kontaktdraht **13** verbindet den Chip **12** mit dem zweiten Band. Das Gehäuse besteht hier aus einem quaderförmigen Bodenteil **14**, das von unten an den Leiterraum **10** geklebt wird. Ein rechteckiges Rahmenteil **15** wird von oben auf den Leiterraum **10** befestigt. Zusätzlich wird eine Abdeckung **16** benötigt, die das Rahmenteil nach oben abschließt und vorteilhaft optische Eigenschaften besitzt. Die Abdeckung wird dann nicht benötigt, wenn die Leuchtdiode als Element (Pixel) einer Flächenleuchte eingesetzt wird. In **Fig. 2b** ist ein Chipträgerband **18** zur Herstellung eines optoelektronischen Halbleiter-Bauelements gezeigt.

In **Fig. 3** ist ein Ausschnitt aus einer Flächenleuchte **20** gezeigt. Sie besteht aus einem gemeinsamen Träger **21**, auf den ein quaderförmiges Gehäuse **22** aufgeklebt ist. Die Oberseite ist mit einer gemeinsamen Abdeckung **23** versehen. Das quaderförmige Gehäuse besitzt Aussparungen, in denen einzelne Halbleiter-Bauelemente **24** untergebracht sind. Sie sind UV-emittierende Leuchtdioden. Die Umwandlung in weißes Licht erfolgt mittels Konversionsschichten **25**, die auf allen der UV-Strahlung zugänglichen Flächen angebracht ist. Dazu zählen die innen liegenden Oberflächen der Seitenwände des Gehäuses, der Abdeckung und des Bodenteils. Die einzelnen Pixel sind ähnlich wie in **Fig. 2** dargestellt aufgebaut.

Ein Beispiel einer Kompaktlampe **30** auf der Basis von Leuchtdioden ist in **Fig. 4** dargestellt. Sie ähnelt äußerlich bekannten Lampen, indem ein kugelförmiger Außenkolben **31** auf einem Schraubsockel **32** sitzt. Der Sockel **32** trägt einen Rundstab **33**, der sich weit in den Außenkolben **31** hinein erstreckt. Der Rundstab **33** ist auf seiner Oberfläche mit Leuchtdioden **34** bestückt. Im Inneren des Rundstabs **33** ist die Elektronik **35** verborgen. Der Außenkolben **31** ist innen mit einer Konversionsschicht **36** bedeckt. Die Leuchtdioden emittieren beispielsweise UV oder blaues Licht.

Patentansprüche

1. Optoelektronisches Halbleiter-Bauelement, bei dem ein strahlungsemitterender oder -empfangender Halbleiterkörper (Chip) auf einem elektrisch leitenden Leiterraum befestigt ist und von einem Gehäuse umgeben ist, **dadurch gekennzeichnet**, daß alle eingesetzten Gehäusematerialien und der Leiterraum im Temperaturbereich, der in Herstellung und Anwendung auftritt, aneinander angepaßte thermische Ausdehnungskoeffizienten aufweisen.
2. Optoelektronisches Halbleiter-Bauelement nach Anspruch 1, dadurch gekennzeichnet, daß der Leiterraum aus Kupfer-Manteldraht o. ä. besteht.
3. Optoelektronisches Halbleiter-Bauelement nach Anspruch 1, dadurch gekennzeichnet, daß das Gehäuse mehrteilig aus Einzelteilen zusammengesetzt ist, wobei die Einzelteile gasdicht miteinander, insbesondere durch Klebemittel, verbunden sind.
4. Optoelektronisches Halbleiter-Bauelement nach Anspruch 1, dadurch gekennzeichnet, daß das Gehäuse einen Grundkörper mit einer Ausnehmung aufweist, wobei der Grundkörper an dem Leiterraum gasdicht befestigt ist, daß der Halbleiter-Chip in der Ausnehmung auf dem Leiterraum befestigt ist, und daß die Ausnehmung mit einer Abdeckung gasdicht verschlossen ist.
5. Optoelektronisches Halbleiter-Bauelement nach Anspruch 2, dadurch gekennzeichnet, daß der Grundkörper u/o die Abdeckung aus Glas oder Quarzglas oder Keramik oder Glaskeramik gefertigt ist.
6. Optoelektronisches Halbleiter-Bauelement nach

Anspruch 2, dadurch gekennzeichnet, daß der Grundkörper ein Bodenteil und ein Rahmenteil, das den Halbleiter-Chip umgibt, aufweist, wobei der Leiterraum zwischen dem Bodenteil und dem Rahmenteil angeordnet ist und wobei alle diese Komponenten untereinander gasdicht verbunden sind.

7. Optoelektronisches Halbleiter-Bauelement nach Anspruch 4, dadurch gekennzeichnet, daß die Abdeckung auf dem Rahmenteil sitzt, und zwar entweder großflächig oder nur im Bereich der Ausnehmung positioniert.

8. Optoelektronisches Halbleiter-Bauelement nach Anspruch 5, dadurch gekennzeichnet, daß die Abdeckung optische Eigenschaften besitzt, insbesondere eine Fresneloptik, eine bifokale Linse, eine plankonvexe oder plankonkave Linse aufweist.

9. Optoelektronisches Halbleiter-Bauelement nach Anspruch 1, dadurch gekennzeichnet, daß die Innenseite des Rahmenteils u/o die Innenseite des Bodenteils eine Reflexionsschicht aufweist, insbesondere eine metallische oder oxidische Schicht, wie TiO_2 , Ag oder Al oder Au.

10. Optoelektronisches Halbleiter-Bauelement nach Anspruch 2, dadurch gekennzeichnet, daß der Grundkörper einstückig als Preßglaskörper ausgebildet ist, wobei der Leiterraum in den Grundkörper eingebettet ist.

11. Optoelektronisches Halbleiter-Bauelement nach Anspruch 1, dadurch gekennzeichnet, daß das Gehäuse einen Sockel mit mindestens zwei elektrisch und thermisch leitenden Metalleitungen aufweist, daß der Halbleiter-Chip auf einer der Metalleitungen aufgebracht ist, und daß eine Kappe auf den Sockel aufgesetzt und mit diesem gasdicht verbunden ist, wobei der Sockel und die Kappe aus Glas oder Quarzglas oder Keramik oder Glaskeramik gefertigt sind.

12. Optoelektronisches Halbleiter-Bauelement nach Anspruch 9, dadurch gekennzeichnet, daß die Kappe ein Lichtaustrittsfenster besitzt, die ein optisches Element, insbesondere eine Linse oder einen Filter, aufweist.

13. Optoelektronisches Halbleiter-Bauelement nach Anspruch 1, dadurch gekennzeichnet, daß der Halbleiter-Chip UV-Strahlung, insbesondere im Bereich 320 bis 400 nm, emittiert.

14. Optoelektronisches Halbleiter-Bauelement nach Anspruch 11, bestehend aus einem Chipträger mit annähernd ebener Trägerfläche, einem darauf montierten optoelektronischen Halbleiterchip, der optisch ausgerichtet ist, einem dem Chipträger zugeordnetem und dieses abstützende Gehäuse mit Sockelfunktion, wobei mindestens ein Teil des Gehäuses aus UV-transparentem oder -luzentem Material besteht.

Hierzu 3 Seite(n) Zeichnungen

- Leerseite -

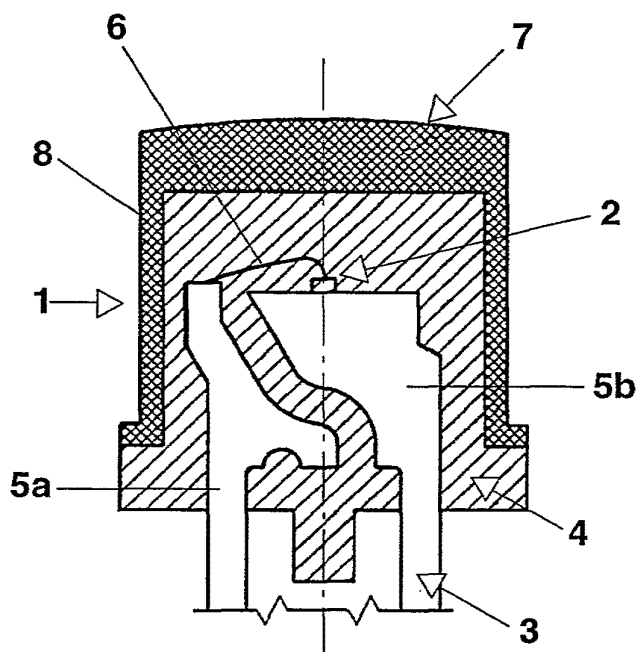


FIG. 1a

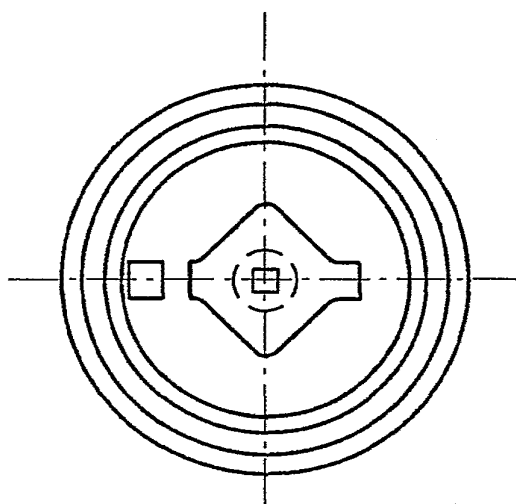


FIG. 1b

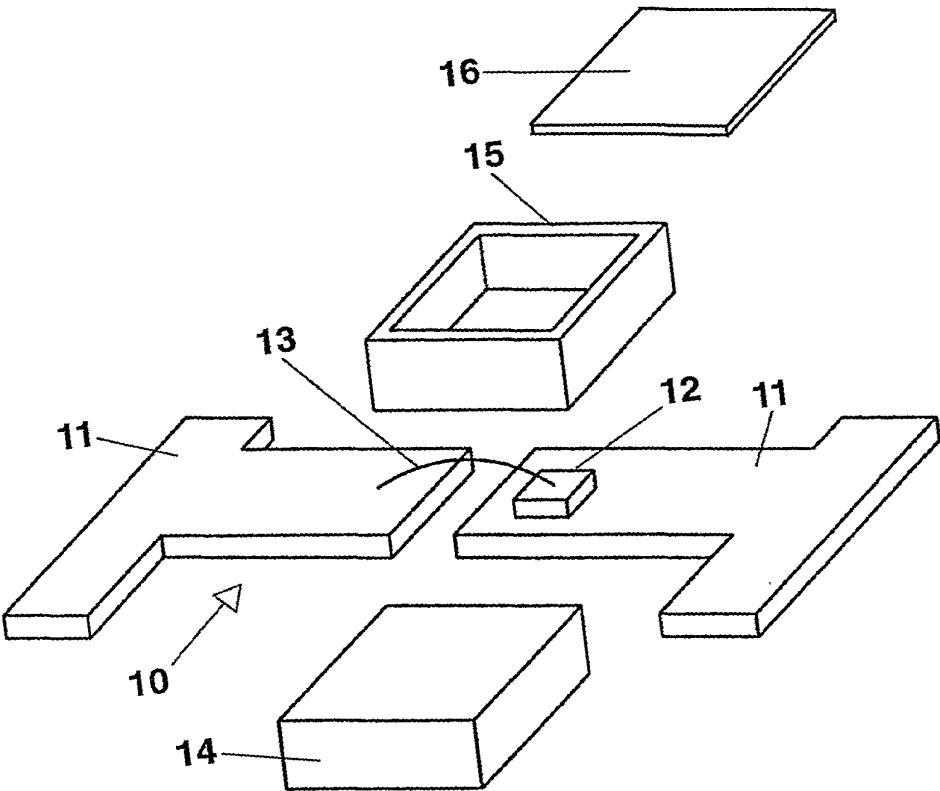


FIG. 2a

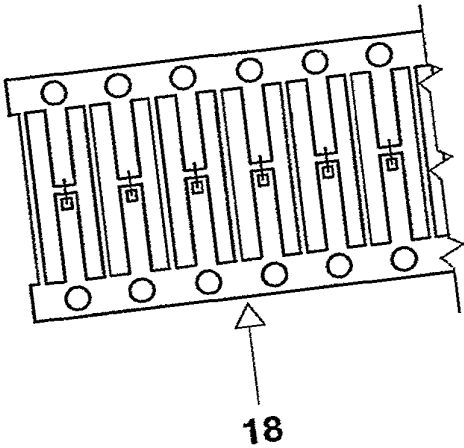


FIG. 2b

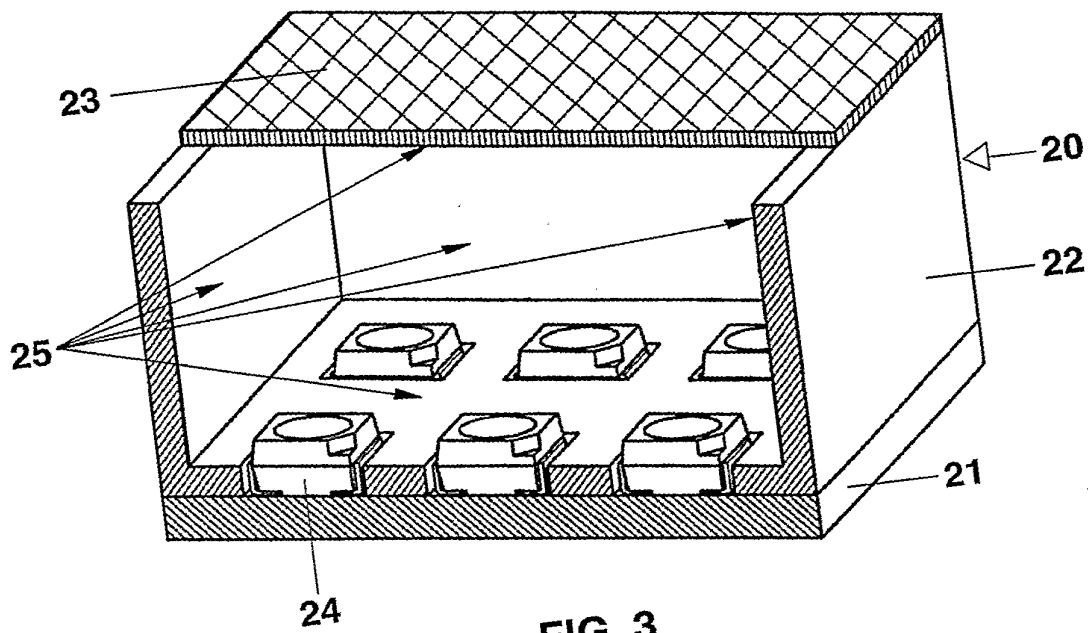


FIG. 3

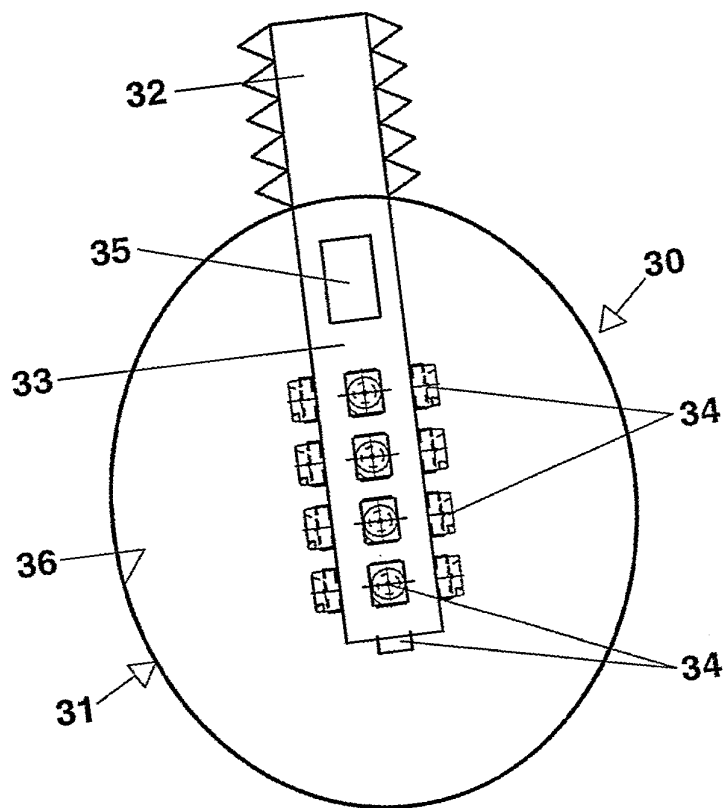


FIG. 4



(51) International Patent Classification:

H04L 29/06 (2006.01) H04W 12/04 (2009.01)
H04L 29/08 (2006.01)

(21) International Application Number:

PCT/US2019/022184

(22) International Filing Date:

14 March 2019 (14.03.2019)

(25) Filing Language:

English

(26) Publication Language:

English

(30) Priority Data:

62/644,195 16 March 2018 (16.03.2018) US

(71) Applicant: IOT AND M2M TECHNOLOGIES, LLC
[US/US]; 807 Davis Street, Unit 2207, Evanston, IL
60201-7104 (US).

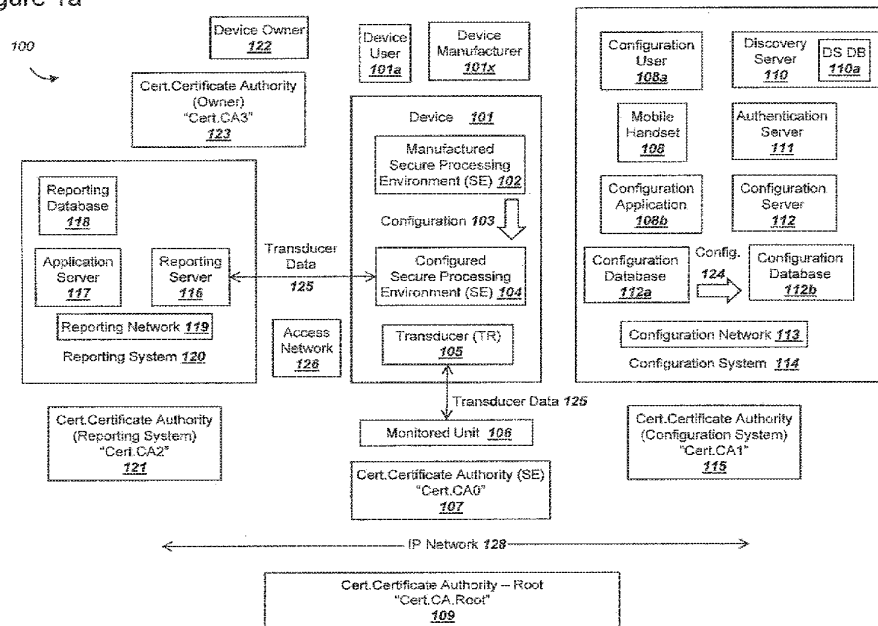
(72) Inventor: NIX, John, A.; c/o IoT And M2M Technologies, LLC, 807 Davis Street, Unit 2207, Evanston, IL 60201-7104 (US).

(74) Agent: THOMSON, Kirsten, L.; McDonnell Boehnen Hulbert & Berghoff LLP, 300 South Wacker Drive, Chicago, IL 60606 (US).

(81) Designated States (unless otherwise indicated, for every kind of national protection available): AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DJ, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IR, IS, JO, JP, KE, KG, KH, KN, KP, KR, KW, KZ, LA, LC, LK, LR, LS, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SA,

(54) Title: CONFIGURATION SYSTEMS AND METHODS FOR SECURE OPERATION OF NETWORKED TRANSDUCERS

Figure 1a



(57) **Abstract:** A device can include an internal secure processing environment (SE) and communicate with a configuration system. The SE may utilize a near field communications (NFC) radio. A mobile handset can connect with the SE in the device using NFC. The mobile handset can communicate with the configuration system and receive configuration data and a software package for the device. The configuration system can derive a PKI key pair and send the derived public key to the device via the mobile handset. The SE and the configuration system can mutually derive an encryption key using the derived PKI key pair. The configuration data can be transmitted over the NFC radio, and the mobile handset can establish a Wi-Fi access point. The software package can be encrypted using the encryption key and transmitted to the device over the established Wi-Fi access point, thereby completing a configuration step for the device.

[Continued on next page]

SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN,
TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(84) Designated States (*unless otherwise indicated, for every kind of regional protection available*): ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, ST, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, RU, TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, KM, ML, MR, NE, SN, TD, TG).

Published:

— *with international search report (Art. 21(3))*

CONFIGURATION SYSTEMS AND METHODS FOR SECURE OPERATION OF NETWORKED TRANSDUCERS

CROSS-REFERENCE TO RELATED APPLICATIONS

5 This international PCT application claims the benefit of the filing date of U.S. Provisional Patent Application Serial No. 62/644,195, filed March 16, 2018, which is hereby incorporated by reference in its entirety.

BACKGROUND

Technical Field

10 The present systems and methods relate to configuration and operation of networked transducers, and more particularly, to systems and methods for securely configuring a secure processing environment within a transducer device, where the transducer device can communicate with a configuration system, a reporting system, and a mobile handset in order to complete configuration.

Description of Related Art

 The ability to connect transducers such as sensors and actuators with a network is a growing field with many economical applications. As the costs for both electronic hardware and bandwidth continues to decrease, the use of networked transducers is expected to continue increasing over the coming decades. Connecting transducers to a network can be referred to as “machine-to-machine (M2M) communications” or “the Internet of Things (IoT).” Among many potential benefits, IoT technologies allow automated monitoring and/or control of assets, equipment, personnel, or a physical location where manual monitoring may not be economical. Many applications for the “Internet of Things” significantly reduce costs by using automated monitoring instead of manual techniques. Prominent examples of IoT applications today include monitoring and control for building heating/air-conditioning, automobiles, alarm systems, and tracking devices. Fast growing markets for IoT applications today include health applications such as the remote monitoring of a person’s fitness activity, heartbeat, or glucose levels, monitoring of industrial equipment deployed remotely in the field, and also security systems.

30 IoT communications can provide remote control over actuators that may be connected to a device, such as turning on or off a power switch, locking or unlocking a door, adjusting a speed of a motor, or similar remote control. A decision to change or adjust an actuator associated with a connected device can utilize one or a series of sensor

measurements. As one example, an IoT device for a truck can periodically report engine status to a remote server, and if the engine is operating outside specifications such as being too hot, including potentially an “alarm” condition, then temperature and an alarm code can be reported to a central server for the IoT device. The server can subsequently instruct the driver and/or a specified mechanic to investigate the engine for potential mechanical malfunctions or other causes. The previous example is just one of many possible applications for IoT technology.

Many IoT applications can leverage wireless networking technologies, in addition to wired technologies such as Ethernet. Wireless technologies such as wireless local area networks and wireless wide area networks have proliferated around the world over the past 20 years, and usage of these wireless networks is also expected to continue to grow. Wireless local area network (LAN) technologies include Wi-Fi and wireless wide area network (WAN) technologies include 3rd Generation Partnership Project’s (3GPP) 3rd Generation (3G) Universal Mobile Telecommunications System (UMTS) and 4th Generation (4G) Long-term Evolution (LTE), LTE Advanced, and Narrow-band Internet of Things (NB-IoT). The many options to connect a transducer device to a network creates opportunities for new products and services, but also creates several classes of problems that need to be solved. Many of these problems or needs in the art pertain to enabling users to securely and efficiently configure the transducer devices. A need exists in the art to allow a user to securely upload to the device a set of network access credentials for the wireless network.

A need exists in the art to support the configuration of transducer devices for installation and operation, including deployment in the field or with monitored units by a user or a technician. A manufactured transducer device can include (i) a secure processing environment which could record a “root of trust” or secret key for the device as well as certificates and cryptographic parameters, and (ii) installed firmware and software. The secure processing environment is often by design isolated in order to enhance security, but device users may need to update the secret key, certificates, and cryptographic parameters. The isolation creates problems for users or device owners to readily access and update the secure processing environment and software for the device. A need exists in the art to allow users or device owners to efficiently change or update for a device both (i) the secret key, certificates, and cryptographic parameters, and (ii) installed firmware and software. A need exists in the art for the users to also efficiently load or transfer network access credentials for a wireless network, in order for the device to obtain network connectivity. A need exists in the art for a configuration system and/or a reporting system to securely record values and identities pertaining to operation of the device, such as a list of

transducers attached, the identity of a monitored unit for the device, as well as the physical location of the device.

Devices with an ARM™-based processor may include a secure processing environment for security operations such as recording keys and certificates. Other processor manufacturers have related security architectures, such as a secure enclave from Apple™ or Intel™. These exemplary processors can be included in a wide variety of devices for machine-to-machine communications, or also applications of “The Internet of Things”. Manufacturers of networked devices with transducers may also sell products with secure elements to increase the secure operation of devices, networks, and servers, where secure operation can comprise confident authentication of users and device, authorization, data confidentiality, and data integrity. Manufacturers of devices may provision a default set of cryptographic values for secure processing environments, secure enclaves, and secure elements, but a need exists in the art to allow device users to update the cryptographic values in order to support the user’s application or environment. In addition, the device and cryptographic values such as keys may have physically been in the hands, possession, or control of third parties before delivery of the device to the user, with the result that the cryptographic values such as keys were also in the hands, possession, or control of the third parties and thus security for a device user could be reduced. A need exists in the art to support a user updating cryptographic values for a secure environment after receipt of the device from a third party.

With conventional technology for configuration and installation of networked devices, a frequent challenge can be that that public key infrastructure (PKI) keys and parameters for default or manufactured configuration of a device may not match the requirements for a configuration system or a reporting system. In other words, the “root of trust” installed by manufacturers of devices and secure enclaves within those devices may have certificates and cryptographic parameters that are not compatible with an operating environment in which the device will be deployed. In other words, the operating environment may use or require (i) different elliptic curve names, (ii) support for different algorithms (ECC, RSA, SIDH, etc.), (iii) more bits of security in the device secret key, etc., than received with a manufactured device. A need exists in the art such that device private keys, cryptographic algorithms, and cryptographic parameters for a user’s application can be securely updated after device manufacturing. A need exists in the art to support the update through a highly automated configuration step.

In addition, devices designed for machine-to-machine communications, or the “Internet of Things”, can frequently be shipped to an end user in a state that is not fully configured for the user. The lack of a full configuration could include incomplete

information for any of the following: (i) current, running firmware or configuration files for the device, (ii) network access credentials, (iii) credentials for access to remote servers, and (iv) proper configuration to support transducers that are actually connected with a device. As one example, hospitals, health clinics, or doctor's offices can receive health monitoring equipment designed to be networked, but without both (i) the networking configured for the equipment, and (ii) updated or patched firmware for the equipment. A need exists in the art to allow general users (e.g. not requiring expert IT skills) to securely configure equipment designed to be networked, including securely updating the device's firmware and credentials for both accessing a network and remote servers.

Many other examples exist as well for needs in the art to support efficient yet secure device configuration by users, and the above are examples are just a few and intended to be illustrative instead of limiting.

SUMMARY

Methods and systems are provided for configuration of networked transducers in order to support secure operation of the networked transducers. The methods and systems can support secure operation of devices for “The Internet of Things (IoT)” which is also known as “Machine to Machine (M2M)” communications. An objective of the invention is to address the challenges noted above for securing the deployment and operation of devices that have radios and transducers, where the devices connect to servers using networks based on Internet protocols.

A device can have a transducer for monitoring a monitored unit, and the device can connect with an access network in order to communicate data for the transducer with a server. The connection with the access network can be via wireless and a radio, or could be through a wired connection such as Ethernet. The device, transducer, and monitored unit can have identifiers such as bar codes, QR codes, MAC addresses, serial numbers, or other identifiers. The device can be produced by a device manufacturer and include a secure processing environment. The secure processing environment can contain a processor and nonvolatile memory, where the manufacturer records a certificate for the device with a PKI public key, cryptographic parameters associated with the certificate, and a private key. The secure processing environment can also include a set of cryptographic algorithms in order to process cryptographic operations such as a key exchange algorithm, a symmetric ciphering algorithm, and also a digital signature algorithm. The secure processing environment can also record a series of certificate authority certificates, such as a certificate authority certificate for the certificate of the device and a root certificate for the certificate authority, plus any intermediate certificates between the certificate authority certificate and the root certificate.

A device owner or device user may prefer the device transition from a manufactured state or delivered state to an operational state, where the transition for the device can comprise a configuration step. Or, after initial operation of the device, a device owner or device user may prefer the device transition from a first configured state to a second configured state, and the transition could also comprise a configuration step. In general, the security of a configuration step can be important because the overall configuration of a system can depend on the security of a configuration step. The configuration step can preferably be completed in a relatively automated manner using secure steps and hardware that is both (i) available for low cost and also (ii) readily easy to obtain or source. A mobile handset such as a smartphone could be used to connect a device

in the manufactured state to an IP network, such that the device in the manufactured state could securely communicate with a reporting system after conduction a configuration step with a configuration system.

5 The configuration step with the configuration system can comprise a first collection of steps for the mobile handset to work in conjunction with the configuration system to prepare for authentication of the device and then configuration with the device. The first collection of steps may be depicted in Figure 2a below. A mobile handset can be operated by a configuration user, which could also be the device user. The first collection of steps can comprise the mobile handset (i) reading a tag or code for the device, where the tag
10 includes a URL and an identity for the device, (ii) sending the tag to a discovery server and (iii) receiving a response from the discovery server with configuration data for provisioning the device with service, credentials, and software. The configuration data can include a URL for an authentication server and the name or location of configuration software (or “configuration app” for the mobile handset operating system).

15 The mobile handset can use the configuration software in order to establish communication between the device and a configuration system. The mobile handset can download the configuration app if not already installed on the mobile handset. The configuration data can also include a certificate for the authentication server and authentication parameters. The mobile handset can conduct a first authentication with the authentication server, where the first authentication comprises the configuration user
20 authenticating with the authentication server. The mobile handset can send device identity information to the authentication server and the authentication server can receive a certificate for the device. The authentication server can send a random number and a signature for the random number to the mobile handset.

25 The configuration step with the configuration system can also comprise a second collection of steps for the device to communicate with the mobile handset and the mobile handset to communicate with the configuration system. The second collection of steps may be depicted in Figure 3 below. The mobile handset and device can establish a NFC peer-to-peer session using the NFC radios in each, and the NFC radio for the device can be
30 dedicated to the secure processing environment for the device. Benefits from using an NFC session include the radio cost can be low and security is higher due to the close physical proximity for the device and mobile handset. The mobile handset can forward the random number and the signature for the random number to the device over the NFC session, as well as a certificate for the authentication server. The device can verify the
35 signature and the certificate. The device can sign the random number received and also generate a second random number.

Continuing with the second collection of steps in a configuration step, the device can send (i) the signature for the random number from the authentication server and (ii) the second random number to the mobile handset, which can forward to the authentication server. The authentication server can sign the second random number, and verify the signature received from the device. The device can verify the signature for the second random number from the authentication server. In this manner, both the authentication server and the device can be mutually authenticated. The authentication server can send the second random number received from the device to a configuration server. The mobile handset can collect a list of identities, which can include an RF sweep analysis to measure all available wireless networks as the location of the device in the form of a networks available list. The mobile handset can send a list of identities to the configuration system pertaining to the device, transducers, and monitored unit, and the configuration system can record the list of identities in a configuration database.

The configuration step with the configuration system can also comprise a third collection of steps for data transferred between the device and the configuration server. The third collection of steps may be depicted in Figure 4a below. A configuration server in the configuration system can also sign the second random number in order to authenticate the configuration server. The configuration server can generate an ephemeral PKI key pair and sign the ephemeral public key, and the data can be sent to the mobile handset in order to forward to the device. The device can verify the signature using a public key in a certificate for the configuration server. The configuration server and the device can use the device PKI key pair plus the ephemeral PKI key pair from the configuration server in a key exchange algorithm in order to mutually derive a symmetric encryption key.

The device can generate a new PKI key pair and send a first ciphertext to the configuration server, where the first ciphertext includes encrypted plaintext of initial configuration information for the device plus the newly derived public key. The device can encrypt the first ciphertext using the derived symmetric encryption key and the configuration server can decrypt the first ciphertext using the derived symmetric encryption key. The configuration server can have a certificate authority sign the newly derived public key for the device. The configuration server can use the identities list to collect a software package for the device. The configuration server can send the new certificate for the device, plus configuration files and network access credentials to the mobile handset in the form of a second ciphertext, and the mobile handset can forward the second ciphertext to the device. The configuration server can send a set of Wi-Fi access point credentials, in an encrypted format, to both the mobile handset and the device via the mobile handset.

The configuration step with the configuration system can also comprise a fourth collection of steps for data transferred between the device and the configuration server. The fourth collection of steps may be depicted in Figure 5 below. The device can receive the new certificate for the device as well as the configuration files from the configuration server via the mobile handset. The device can apply the configuration files and begin using the new certificate and associated derived private key in subsequent cryptographic operations. After receipt of the Wi-Fi access point credentials, the mobile handset can activate a Wi-Fi access point using the credentials, and the Wi-Fi access point may be operated at low power due to the proximity of the device. The device can activate a Wi-Fi client also using the credentials for the Wi-Fi access point received from the configuration server. The configuration server can conduct a key exchange using the new, second, derived public key for the device and the ephemeral private key in order to derive a second symmetric encryption key. The device can conduct the corresponding key exchange in order to derive the second symmetric encryption key.

The configuration server can encrypt the software package using the second symmetric encryption key and transfer to the device via the mobile handset. The device can decrypt the software package and apply the files and reboot, using both the configuration files and software package. The device can connect with the access network using the received network access credentials, and conduct a key exchange with a reporting system. The device can subsequently transfer encrypted transducer data and a report regarding configuration to the reporting system, thereby completing the configuration step for the device. The mobile handset can check with the reporting system that transducer data has been successfully transmitted between the reporting system and the device, and display to the end user that the configuration step is successful and completed.

These as well as other aspects and advantages will become apparent to those of ordinary skill in the art by reading the following detailed description, with reference where appropriate to the accompanying drawings.

BRIEF DESCRIPTION OF THE DRAWINGS

Various exemplary embodiments are described herein with reference to the following drawings, wherein like numerals denote like entities.

5 Figure 1a is a graphical illustration of an exemplary system, where a device with a transducer communicates with a configuration system and a reporting system, in order to conduct a configuration step, in accordance with exemplary embodiments;

 Figure 1b is a graphical illustration of hardware, firmware, and software components for a device, including a secure processing environment, in accordance with
10 exemplary embodiments;

 Figure 2a is a simplified message flow diagram illustrating an exemplary system with exemplary data sent and received by a mobile handset, in accordance with exemplary embodiments;

 Figure 2b is a flow chart illustrating exemplary steps for creating and verifying a
15 digital signature using PKI keys, parameters, and data input, in accordance with exemplary embodiments;

 Figure 3 is a simplified message flow diagram for an exemplary system with exemplary data sent and received by a mobile handset and a device, in accordance with exemplary embodiments;

20 Figure 4a is a simplified message flow diagram for an exemplary system with exemplary data sent and received by a mobile handset and a device, in accordance with exemplary embodiments;

 Figure 4b is a series of flow charts illustrating exemplary steps for a device and a configuration server to derive keys and then use the derived keys to encrypt and decrypt
25 data, in accordance with exemplary embodiments;

 Figure 4c is a series of flow charts illustrating exemplary steps for a device and a configuration server to derive keys and then use the derived keys to encrypt and decrypt data, in accordance with exemplary embodiments;

30 Figure 4d is an illustration of a certificate for a device public key, where the device derived the public key and a corresponding private key, in accordance with exemplary embodiments;

 Figure 5 is a simplified message flow diagram for an exemplary system with exemplary data sent and received by a mobile handset and a device, in accordance with exemplary embodiments;

Figure 6a is a graphical illustration of components for a device, in accordance with exemplary embodiments, and;

Figure 6a is a graphical illustration of components for a device, in accordance with exemplary embodiments.

**DETAILED DESCRIPTION OF EXEMPLARY EMBODIMENTS OF THE
INVENTION**

Figure 1a

5 Figure 1a is a graphical illustration of an exemplary system, where a device with a transducer communicates with a configuration system and a reporting system, in order to conduct a configuration step, in accordance with exemplary embodiments. The system 100 can include a device 101, a configuration system 114, and a reporting system 120. Device 101 can communicate with the configuration system 114 and reporting system 120 via
10 Internet Protocol (IP) network 128. IP network 128 could be a public or private network supporting Internet Engineering Task Force (IETF) standards such as, but not limited to, such as, RFC 786 (User Datagram Protocol), RFC 793 (Transmission Control Protocol), and related protocols including IPv6 or IPv4. A public IP network 128 could utilize globally routable IP addresses, and a private IP network 128 could utilize private IP
15 addresses which could also be referred to as an Intranet. Other possibilities for IP Network 128 exist as well without departing from the scope of the invention. Device 101 could utilize an access network 126 in order to communicate with the reporting system 120 using the IP network 128. After a configuration step 103 described in the present invention and in multiple figures below, device 101 can communicate transducer data 125 with reporting
20 system 120 through access network 126. Note that the security of transducer data 125 for device owner 122, device user 101a, and reporting system 120 during operation of device 101 may generally depend on the security of a configuration step 103.

 Access network 126 could be either a Local Area Network (LAN) or a Wide Area Network (WAN), or potentially a combination of both. Device 101 and access network
25 126 can utilize a variety of wireless technologies to communicate, including IEEE 802.11 (Wi-Fi), Low Power Wide Area (LPWA) technology, 3rd Generation Partnership Project (3GPP) technology such as, but not limited to, 3G, 4G Long-Term Evolution (LTE), or 4G LTE Advanced, NarrowBand – Internet of Things (NB-IoT), LTE Cat M, proposed 5G networks, and other examples exist as well. A wired device 101 can connect to the access
30 network 126 via a wired connection such as, but not limited to, an Ethernet, a fiber optic, or a Universal Serial Bus (USB) connection (not shown). Device 101 could be powered via any of (i) traditional “wall power” potentially with an AD/DC adapter, (ii) a battery which may be periodically recharged, (iii) power over a wired LAN connection such as “power over Ethernet”, and other possibilities exist as well.

Device 101 can include manufactured secure processing environment (SE) 102. The manufactured secure processing environment 102 can also be referred to as a secure enclave or secure element, which will also be designated herein as “SE 102”. Examples of SE 102 as of 2018 can include the secure processing environment in Figure 3 of the ARM
5 ® document “Platform Security Architecture Overview” (v. 1.1) which is incorporated by reference. Other examples for a secure processing environment exist as well, including a “Trusted Execution Environment” according to the standards promoted by Global Platform. SE 102 can comprise functionality of a processor such as an ARM or Intel ® based processor to secure cryptographic key materials including private keys in public key
10 infrastructure (PKI) key pairs, secret shared keys, cryptographic parameters, cryptographic algorithms, a certificate 107 for the secure processing environment certificate authority, a root certificate 109, etc. In other words, although root certificate 109 is depicted as external to SE 102 in Figure 1a, in exemplary embodiments a copy of this root certificate 109 is stored within nonvolatile memory of manufactured SE 102 and configured SE 104.
15 Additional details for components within a SE 102 are provided below in Figure 1b.

As depicted in Figure 1a, the configuration step 103 can convert the manufactured SE 102 into a configured SE 104. In addition, although not depicted in Figure 1a, a configuration step 103 can also be applied at a subsequent time on a previously configured SE 104. In other words, a configuration step 103 may take place multiple times over the
20 life of device 101, such as either (i) when device owner 122 changes or (ii) a certificate for device 101 (e.g. certificate cert0.SE 102k depicted below in Figure 1b) expires. An exemplary embodiment for a first configuration 103 of a device 101 is depicted in Figure 1a.

Device 101 can also include a transducer 105, and may also be referred to herein as
25 a “transducer device 101”. Transducer 105 can be a sensor or an actuator and may be either passive or active. Transducer 105 could be internal within the physical housing of device 101, such as (i) a digital image sensor inside a camera. Transducer 105 could be external to the physical housing of device 101, such as a thermocouple or probe extending from device 101 to a monitored unit 106. Additional details for transducer 105 are also
30 provided in Figure 1b below. Although device 101 is depicted in Figure 1a with a single transducer 105, device 101 could include multiple transducers 105. In addition, although a single device is depicted in Figure 1a, a system 100 can include a plurality of devices 101.

Examples of a monitored unit 106 can include an ATM or vending machine, a truck or automobile, a refrigerated or standard (“dry”) shipping container, or industrial
35 equipment such as, but not limited to, an oil field pump, a transformer connected to an electrical grid or a escalator in a building. Additional examples of a monitored unit 106

include can also include a pallet for shipping or receiving goods, a refrigerator with food, a health monitoring device attached to a person such as, but not limited to, a heart monitor, and a gate or door for opening and closing. Device 101 can utilize a sensor to measure and collect data regarding a parameter of monitored unit 106 such as, but not limited to, 5 temperature, humidity, physical location potentially including geographical coordinates from a Global Positioning System (GPS) receiver, surrounding light levels, surrounding RF signals, vibration and/or shock, voltage, current, and/or similar measurements.

Monitored unit 106 could also be controlled by device 101 via a transducer 105 that is an actuator, where device 101 can change the actuator in order to change a state for 10 monitored unit 106. For example, if monitored unit 106 is a door, then a transducer 105 could include a relay to activate a lock for the door. If monitored unit 106 is a lighting system in a building, then transducer 105 could comprise a switch to turn the lighting level up or down. Other examples exist for monitored unit 106 as well, and the above are intended to be illustrative instead of limiting. In the case where transducer 105 is either an 15 actuator or a sensor, (X) device 101 could connect transducer 105 to the reporting system 120, and (Y) reporting system 120 can both (i) receive transducer data 125 input from sensors with monitored unit 106 and (ii) transmit transducer data 125 output to actuators in order to control monitored unit 106. Thus, in operation and after the configuration step 103, reporting system 120 can remotely monitor and control monitored unit 106 using 20 device 101 and transducer 105.

As depicted in Figure 1a, device 101 can include a device user 101a, a device manufacturer 101x, and a device owner 122. Device user 101a could be a person, group of people, or business entity associated with the operation of device 101. If device 101 monitors a building, then device user 101a could be a building engineer or building 25 manager. If device 101 monitors a home, then device user 101a could be a home owner. Device manufacturer 101x can be the manufacturer or supplier of device 101 to device user 101a. Device manufacturer 101x could purchase the manufactured secure element 102 and integrate it into a housing plus other components in order to produce device 101. In exemplary embodiments, device owner 122 is the owner of device 101. Device owner 122 30 may be the same as device user 101a in some embodiments, but device owner 122 also may be different than device user 101a in other embodiments. For exemplary health care applications in a hospital where device 101 is a medical device to monitor patient health, device owner 122 could be the hospital and device user could be a nurse, doctor, or medical technician. Device owner 122 may have a certificate 123 from a certificate authority 35 associated with the device owner 122.

Each of the different entities depicted for system 100 of device owner, device user, and device manufacturer may control or operate device 101 and SE 102 at different times for the life cycle of device 101. In addition, a device owner or device user may change during the life of device 101 after manufacture and some initial configuration step that could be different than a configuration step 103. This change in control and operation over time, plus potential prior operation in an insecure manner, can create challenges for ensuring secure operation of device 101. A prior party in control of device 101 may not have supported the security procedures or requirements for device owner 122. For example, device owner 122 may require that device 101 is configured after delivery from device manufacturer 101x or possibly after delivery from a previous device owner 122. A configuration step 103 with configuration system 114 can ensure that device 101 is configured to the standards or requirements of device owner 122, without depending on previous security procedures or steps of device manufacturer 101x or device user 101a.

In order to conduct a configuration step 103, system 100 can utilize a configuration system 114. As depicted in Figure 1a, configuration system 114 can include a configuration user 108a, a mobile handset 108, a configuration application 108b, a discovery server 110, an authentication server 111, a configuration server 112, a first configuration database 112a, and a second configuration database 112b. Discovery server 110 can include a discovery server database 110a. A configuration step 124 can convert the first configuration database 112a into the second configuration database 112b, where the second configuration database 112b can record data pertaining to a device 101 and reporting system 120 after a configuration step 103. The configuration step 124 can comprise the server/network-side equivalent step of configuration step 103 depicted for device 103. The elements within a configuration system 114 can be connected via a configuration network 113. Configuration network 113 could be similar to IP network 128 described above.

Configuration system 114 can utilize a certificate 115 for a certificate authority associated with configuration system 114. The certificate authority associated with configuration system 114 may or may not be the same as the certificate authority associated with SE 102, or owner 122, or reporting system 120. Although a certificate authority is not depicted for certificate 115 in Figure 1a, each of the certificates 107, 115, 121, and 123 preferably have a "parent" certificate authority or signed public key associated with the certificates. Certificates 107, 115, 121, and 123 can be formatted similar to SE certificate 419 depicted in Figure 4d below, with different values appropriate for each certificate depicted in Figure 1a. Root certificate 109 may also be similar to SE certificate 419,

except the signature in a root certificate 109 may be self-signed by the public key in root certificated 109.

5 A configuration system 114 can use a mobile handset 108 operated by a configuration user 108a. Details and components for a mobile handset 108 are depicted and described below in connection with Figure 2a. In some exemplary embodiments, a mobile handset 108 can comprise a smart phone, such as based on the Android or IOS operating systems. In other embodiments, instead of using a “mobile handset 108”, a configuration user 108a can operate a “configuration unit 108”, or simply “unit 108”, which can provide the same or equivalent functionality as a “mobile handset 108” as depicted herein. In these other exemplary embodiments, a unit 108 could comprise a portable device, such as laptop, tablet, wearable device such as a smart watch, a USB stick, etc. For embodiments where device 101 is configured in a manufacturing facility or potentially a location away from monitored unit 106, including the location where SE 102 is inserted into device 101, then the element or node depicted as “mobile handset 108” in the present invention could comprise the unit 108 where the unit 108 could operate normally in a fixed location instead of being mobile. For example, unit 108 could be a configuration server to conduct a configuration step 103 for a plurality of devices 101 in sequence. Other possibilities exist as well for a unit 108 that is not a mobile handset to provide the functionality of a “mobile handset 108” herein and below, without departing from the scope of the present invention.

The servers shown for configuration system 114 can be co-located within the same data center or geographically dispersed. A configuration system 114 can also include a plurality of the elements depicted in Figure 1b. Further, some elements of a configuration system 114 could be combined, such (i) as the discovery server 110 could be combined with the authentication server 111 or configuration server 112, or (ii) the configuration server 112 could be combined with the authentication server 111. Other possibilities exist for the arrangement of elements within a configuration system 114 without departing from the scope of the invention. For an alternative exemplary embodiment to that depicted in Figure 1a, discovery server 110 could be operated external to configuration system 114, such as discovery server being associated with device manufacturer 101x, device owner 122, or reporting system 120. Additional details regarding the components and operation of a configuration system 114 will be described below. In some exemplary embodiments, reporting system 120 and configuration system 114 can be combined.

35 A reporting system 120 in system 100 can include a reporting server 116, an application server 117, and a reporting database 118. Although not depicted in Figure 1a, a reporting system 120 and a configuration system 114 can include a firewall to filter packets

received through an IP network 128. The elements within a reporting system 120 can be connected via reporting network 119, which could comprise a network similar to IP network 128. Similar to configuration system 114, the servers for reporting system 120 can be co-located within the same data center or geographically dispersed. The servers and
5 databases shown for both reporting system 120 and configuration system 114 can be either different physical computers such as rack-mounted servers, or different logical or virtual servers or instances operating in a “cloud” configuration.

The elements within reporting system 120 can operate in conjunction to collect data from a plurality of devices 101 through access network 126 and present data or reports to
10 device owner 122, device user 101a, or potentially another user or manager of system 100. Reporting system 120 could also take input from device owner 122 or device user 101a to send control transducer data 125 to device 101 in order to operate a transducer 105 in order to control monitored unit 106. Reporting system 120 could also send control transducer data 125 to device 101 without requiring user input, such as for automated control.
15 Reporting system 120, or elements in reporting system 120, can have a certificate 121 signed by a certificate authority associated with reporting system 120. Certificate 121 may be associated or verified using root certificate 107, although in some exemplary embodiments the root certificate for certificate 121 may be different than root certificate 107. The various servers depicted in Figure 1a could comprise multiple individual physical
20 servers or multiple virtual servers operating in a coordinated manner to provide the functionality shown. In other words, although a single reporting server 116 is depicted in Figure 1a, a reporting server 116 could comprise multiple different physical or virtual servers.

25 **Figure 1b**

Figure 1b is a graphical illustration of hardware, firmware, and software components for a device, including a secure processing environment, in accordance with exemplary embodiments. Figure 1b is illustrated to include several components that can be common within a device 101 and a manufactured secure processing environment 102 or
30 secure enclave 102 (SE 102). Device 101 may consist of multiple components in order to collect and transmit transducer data 125 (shown in Figure 1a) associated with a monitored unit 106. In exemplary embodiments and as depicted in Figure 1b, device 101 can include a device identity 101b, a processor 101c (depicted as “CPU 101c”), random access memory (RAM) 101d, an operating system (OS) 101e, storage memory 101f, a system bus 101g, a

transducer interface 101h, a transducer bus 101i, a radio 101z, and a user interface (UI) 101y.

Device identity 101b could comprise a preferably unique alpha-numeric or hexadecimal identifier for device 101, such as an Ethernet MAC address, an International Mobile Equipment Identity (IMEI), an owner interface identifier in an IPv6 network, a serial number, or other sequence of digits to uniquely identify each of the many different possible units for device 101. Device identity 101b can preferably be recorded in a non-volatile memory or written directly to hardware in device 101 by device manufacturer 101x upon device manufacturing. The CPU 101c can comprise a general purpose processor appropriate for typically low power consumption requirements for a device 101, and may also function as a microcontroller. CPU 101c can comprise a processor for device 101 such as an ARM® based process or an Intel® based processor such as belonging to the Atom or MIPS family of processors, and other possibilities exist as well. CPU 101c can utilize bus 101g to fetch instructions from RAM 101d and operate on the instruction. CPU 101c can include components such as registers, accumulators, and logic elements to add, subtract, multiply, and divide numerical values and record the results in RAM 101d or storage memory 101f, and also write the values to an external interface such as radio 101z.

RAM 101d may comprise a random access memory for device 101. RAM 101d can be a volatile memory providing rapid read/write memory access to CPU 101c. RAM 101d could be located on a separate integrated circuit in device 101 or located within CPU 101c. The RAM 101d can include data recorded in device 101 for the operation when collecting and communicating transducer data 125 regarding monitored unit 106. The system bus 101g may be any of several types of bus structures including a memory bus or memory controller, a peripheral bus, and a local bus using any of a variety of bus architectures including a data bus. System bus 101g connects components within device 101 as illustrated in Figure 1a, such as transferring electrical signals between the components illustrated. System bus 101g can also connect SE 102 to the other elements in device 101 including CPU 101b, storage memory 101f, and transducer interface 101h. Device 101 can include multiple different versions of bus 101g to connect different components, including a first system bus 101g between CPU 101b and RAM 101d (which could be a memory bus), and a second system bus 101b between CPU 101b and transducer interface 101h, which could be an I2C bus, an SPI bus, a USB, Dallas 1 wire, or similar data busses, which could also comprise a transducer bus 101i.

Memory 101f within device 101 can comprise a non-volatile memory for storage of data when device 101 is powered off. Memory 101f may be a NAND flash memory and record firmware for device 101. Memory 101f can record long-term and non-volatile

storage of data or files for device 101. In an exemplary embodiment, OS 101e is recorded in memory 101f when device 101 is powered off, and portions of memory 101f are moved by CPU 101b into RAM 101d when device 101 powers on. Memory 101f (i) can be integrated with CPU 101b into a single integrated circuit (potentially as a “system on a chip” as depicted in Figure 6b below), or (ii) operate as a separate integrated circuit or a removable card or device, such as a removable SD card. Memory 101f may also be referred to as “device storage” and can include exemplary file systems of FAT16, FAT 32, NTFS, ext3, ext4, UDF, or similar file systems. As depicted in Figure 1b, non-volatile memory 101f may also contain SE Memory 102x. SE Memory 102x can be a portion of memory 101f that is allocated to SE 102. In exemplary embodiments, the two sets of memory 101f and 102x can be segmented logically, where memory 102x is encrypted with a symmetric ciphering key 102z within SE 102 by a memory controller for SE 102 (not shown). SE 102 could use a symmetric ciphering algorithm such as the Advanced Encryption Standard (AES-128), AES-192, Triple Data Encryption Algorithm (3DES), or similar algorithms. In other words, CPU 101c would not normally be able to read plaintext data in memory 102y since that memory is encrypted by SE 102.

The operating system (OS) 101e can include Internet protocol stacks such as a User Datagram Protocol (UDP) stack, Transmission Control Protocol (TCP) stack, a domain name system (DNS) stack, etc., and the operating system 101e may include timers and schedulers for managing the access of software to hardware resources within device 101, including SE 102 and transducer interface 101h. The operating system shown of 101e can be appropriate for a low-power device with more limited memory and CPU resources (compared to a server such as reporting server 116 and configuration server 112). Example operating systems 101e for a device 101 includes Linux, Android® from Google®, iOS from Apple®, Windows® 10 IoT Core, or Open AT® from Sierra Wireless®. Additional example operating systems 101e for a transducer device 101 include eCos, uC/OS, LiteOs, Contiki, OpenWRT, Raspbian, and other possibilities exist as well without departing from the scope of the present invention. Although depicted as a separate element within device 101 in Figure 1b, OS 101e may reside in RAM 101d or memory 101f during operation of device 101.

Device 101 can include a radio 101z to communicate wirelessly with networks such as access network 126 depicted and described in Figure 1a above. Radio 101z could connect with an antenna in order to transmit and receive radio frequency signals. Although not depicted in Figure 1b, device 101 could utilize a wired connection such as Ethernet for external communication instead of or in addition to a radio 101z. Device 101 may also optionally include user interface 101y which may include one or more devices for receiving

inputs and/or one or more devices for conveying outputs. User interfaces are known in the art and may be simple for many devices such as a few LED lights or and LCD display, and thus user interfaces are not described in detail here. User interface 101y could comprise a touch screen if device 101 as more sophisticated interaction with user 101a. Device 101
5 can optionally omit a user interface 101y, if no user input or display is required for operating device 101 with monitored unit 106. Although not depicted in Figure 1b, device 101 can include other components to support operation, such as a clock, power source or connection, antennas, etc.

Device 101 can include a manufactured secure processing environment (SE) 102.
10 SE 102 may include separate components of a processor SE CPU 102a, RAM 102b, a Near Field Communications (NFC) radio 102c, an SE external bus controller 102h, a hardware random number generator 102e, an SE flash memory 102d, an SE internal bus 102i, a transducer input/output bus controller 102w, and an electrically erasable programmable read-only memory (EEPROM) 102j. In exemplary embodiments, the NFC radio 102c may
15 support other wireless standards besides NFC (such as Wi-Fi or Bluetooth), and in those embodiments an “NFC radio 102c” can be referenced herein as a “radio 101z”. Components for SE 102 such as CPU 102a, RAM 102b, radio 102c, flash memory 102d could be similar to components described above for device 101 such as CPU 101c, RAM 101d, radio 101z, and storage 101f, respectively, except the components for SE 102 could
20 have lower capacity, bandwidth, functionalities, or costs compared to the equivalent components for device 101. Although SE 102 is depicted as a separate component within device 101, in some embodiments the functionality of SE 102 described below could be implemented in software or firmware, such that a physical SE 102 can be omitted and a device 101 can perform the functions and operations of an SE 102 as described herein.

25 In a first exemplary embodiment, (i) CPU 101c may operate at greater than 200 MHz, while SE CPU 102a may operate at less than 200 MHz, and (ii) RAM 101d may store more than two megabytes while RAM 102b may store a value equal to or less than two megabytes. SE CPU 102a can comprise a processor similar to CPU 101c, but embedded into SE 102 and consequently typically with less resources than a CPU 101c,
30 such as (i) less cache memory, (ii) operating typically at a slower speed, (iii) fewer internal registers, (iv) lower power consumption compared to CPU 101c, etc. In some exemplary embodiments, SE CPU 102a can comprise a processing core within a multi-core processor CPU 101c, and in those embodiments SE CPU 102a could operate with the same or similar clock rates and memory as other cores within CPU 101c.

35 In a second exemplary embodiment, although depicted as separate elements for both (i) CPU 101c, and RAM 101d, and (ii) SE CPU 102a and SE RAM 102b, the two

elements could comprise the same physical hardware but represent time division or time separation of the physical hardware, similar to how the same physical hardware can host multiple logical processing running concurrently. In other words, CPU 101c could switch operations between a “normal” mode functioning as CPU 101c for device 101 and then a “secure” mode as SE 102 for device 101. Or, SE CPU 102a could represent a virtualized process or computer within device 101, where the OS 101e implements virtualization and operates as a host for SE 102. Switching could take place rapidly enough that CPU 101c and SE CPU 102a would appear to other components both inside and outside device 101 as separate processors, while there could be a single physical processor inside device 101.

As depicted in Figure 1b, SE 102 can include a radio 102c, which could support NFC communication. Although illustrated as internal to SE 102, radio 102c could be external to SE 102 in some embodiments. A radio 102c that is external to SE 102 could be connected via a secure bus such as SE bus 102i. In some embodiments, SE bus 102i can extend outside SE 102 in order to connect external components for SE 102 with internal components for SE 102. Radio 102c can comprise a radio for short-distance wireless communication, and be connected with an antenna. Radio 102c, when operating to support NFC communications, could (i) transmit and receive at the globally available frequency for unlicensed use of 13.56 MHz, and (ii) be compatible with the International Standards Organization (ISO) 14443 standards and subsequent and related versions. Radio 102c can operate as any of an NFC reader, NFC writer, and NFC peer-to-peer communication, and preferably supports communication at short range such as within several inches of device 101. Radio 102c can support radio frequency communications with mobile handset 108, when mobile handset 108 is within close physical proximity of device 101, such as less than a few feet in an exemplary embodiment. An exemplary processor with an embedded NFC radio as of early 2018 would be the PN7360AUHN processor from NXP ® Semiconductor. Alternatively, a regular processor (i.e. without a radio 102c) could be combined with a radio controller on a separate chip such as combining an ARM Cortex ® processor with a radio chip like the CLRC663 from NXP ® Semiconductor. Other possibilities for the configuration of a radio 102c with a SE 102 and device 101 are possible in order to support a configuration step 103 without departing from the scope of the present invention.

A benefit of the use of short-range communication for SE 102 is physical security, such that any external device desiring to communicate with SE 102 through radio 102c must be in close physical proximity. In exemplary embodiments radio 102c could support other short-range wireless standards besides NFC. Radio 102c could implement radio frequency protocols that utilize low power and close proximity for the other node for which

SE 102 will communicate with. In exemplary embodiments, radio 102c could be a Bluetooth or Wi-Fi radio, but operate at intentionally reduced in order to require closer physical proximity an external device such as mobile handset 108 to communicate with SE 102. For embodiments where radio 102c operates as a Bluetooth or Wi-Fi radio, radio
5 102c could transmit at a power in an exemplary range of 0.01 – 1.0 watts. Other possibilities for the radio technology and power levels of radio 102c exist without departing from the scope of the present invention.

SE 102 can also include a SE external bus controller 102h, which can manage the sequence and flow of data between SE 102 and device 101. As depicted in Figure 1b, SE
10 external bus controller 102h could electrically connect the SE internal bus 102i with device bus 101g, such that electrical signals, pulses, or waveforms can be transferred between device 101 and SE 102, including the transfer of binary data. SE external bus controller 102h could manage and sequence the flow of data between the two sides, and include a data buffer and logic gates. SE external bus controller 102h can include security
15 components, such that device 101 cannot feasibly read data from EEPROM 102j or SE RAM 102b. SE external bus controller 102h could operate as a “mailbox”, such that (i) device 101 writes values or data to SE external bus controller 102h and then (ii) SE 102 reads the values or data from SE external bus controller 102h.

As illustrated in Figure 1b, device 101 may also contain a random number
20 generator 102e. Random number generator 102e can comprise a physical component that generates random or pseudo random numbers for cryptographic operations of SE 102. A random number generator 102e can also comprise a hardware random number generator. The creation of random numbers with a high degree of entropy is important in the use of subsequent steps such as generating Random1.SE 308 below and also deriving PKI keys
25 such as step 401 below. A seed for random number generator 102e could comprise a plurality of data from within SE 102 appended together in order to accumulate information entropy. To acquire the seed, SE 102 could collect a plurality of transducer 105 measurements or states, radio 102c measurements, clock times or values for CPU 102a, RAM 102b or memory 102x states, etc. In exemplary embodiments, random number
30 generator 102e can include a secure hash algorithm similar to message digest 230 below in Figure 2b operating on the random number seed. In exemplary embodiments, random number generator 102e operates within SE 102 as depicted in Figure 1b, and in this manner the random numbers used for cryptographic operations such as PKI key generation in a step 401 below in Figure 4a can remain reasonably secure and not normally communicated
35 outside SE 102.

SE 102 can include nonvolatile SE flash memory 102d, which can also be referred to as “nonvolatile memory 102d” or “memory 102d”. Nonvolatile memory 102d can comprise a physical memory of NAND or NOR flash memory, such that data recorded in nonvolatile memory 102d continues to be recorded when electrical power is removed from device 101 or SE 102. The data within non-volatile memory 102d can subsequently be read and/or re-written when power is restored to SE 102. Memory 102d for SE 102 can be similar to memory 101f for device 101. Memory 102d can be configured for a file system, such as those listed for memory 101f above.

As depicted in Figure 1b, memory 102d can include config0.SE 102aa. Config0.SE 102aa can comprise a file or set of files with the default configuration of SE 102, such as enabling or disabling radio 102c upon startup, settings for radio 102c and credentials, and parameters for the other elements of SE 102 depicted in Figure 1b. Config0.SE 102aa can also be referred to as an initial device configuration and include values for a device configuration. In an exemplary embodiment, config0.SE 102 comprises the firmware and software installed on device 101 upon manufacturing, similar to software package 104q below in Figure 5, where software package 104q can provide updated software, firmware, and configuration files for device 101 and config0.SE contains the originally installed versions of these files in device 101. In an exemplary embodiment where radio 102c operates as a Wi-Fi radio, config0.SE 102aa can include values such as an SSID, a password or security key, and a user name, or also multiple sets of those and other values.

SE 102 can also include a transducer input/output bus controller 102w (or “TR controller 102w”), which can manage the sequence and flow of data or signals between (a) SE 102 and transducers 105 via (b) transducer bus 101i and transducer interface 101h. As depicted in Figure 1b, TR controller 102w could electrically connect the SE internal bus 102i with transducer bus 101i, such that electrical signals, pulses, or voltages can be transferred between transducers 105 and SE 102, including the transfer of digital data or analog signals. For digital data, TR controller 102w could manage and sequence the flow of data between the two sides, and include a data buffer and logic gates. For analog data, TR controller 102w could include a digital-to-analog converter (DAC) or an analog-to-digital converter (DAC). TR controller 102w can include security measures, such that transducers 105 cannot feasibly read data from EEPROM 102j or SE RAM 102b.

In an alternative exemplary embodiment to that depicted in Figure 1b, SE 102 can use a second, different bus 102i than the depicted bus 102i in order to connect with TR controller 102w. In this alternative exemplary embodiment, external transducers 105 could be physically and electrically separated from secure information such as cryptographic keys or data in memory or the processor. Although TR controller 102w is depicted in Figure 1b

as internal to SE 102 and connected to SE internal bus 102i, in another exemplary embodiment TR controller 102w could be connected to device bus 101g, and SE 102 could communicate with TR controller 102w via SE external bus controller 102h. Other possibilities exist as well for the configuration and connections of TR controller 102w without departing from the scope of the present invention.

Although not depicted in Figure 1b, the various servers shown above in Figure 1a such as configuration server 112 and reporting server 116 and other servers as well can include equivalent internal components as device 101 in order to operate. The servers in Figure 1a could include a processor similar to CPU 101c, with primary differences for the processor server being increased speed, increased memory cache, an increased number and size of registers, the use of a 64 bits for datapath widths, integer sizes, and memory address widths, as opposed to an exemplary 32 or 16 bits for CPU 101c in device 101. Similarly, RAM in a server could be a RAM similar to RAM 101d in device 101, except the RAM in a server could have more memory cells such as supporting exemplary values greater than an exemplary 2 gigabytes, while RAM in device 101 could support fewer memory cells such as less than an exemplary 1 gigabyte. Non-volatile memory for storage in a server in Figure 1a could comprise disks, "solid state drives" (SSDs) or "storage area networks" (SAN) for a server. Instead of a radio 101z, in exemplary embodiments, a server in Figure 1a could use a physical, wired LAN interface such as a high-speed Ethernet or fiber optic connection.

As depicted in Figure 1b, a device 101 or SE 102 can communicate with transducers 105 via a transducer interface 101h. Transducer interface 101h can comprise the physical interface between transducers 105 and device 101, and include electrical connectors or pins for transfer of electrical signals between device 101 and transducers 105. For embodiments where a transducer 105 utilizes a micro universal serial bus (USB), then transducer interface 101h could comprise a micro USB receptacle. Transducer interface 101h can utilize a transducer bus 101i. For digital data acquisition or control of transducers 105, transducer bus 101i could comprise a data bus similar to bus 101g for device 101 or bus 102i for SE 102 as described above. Exemplary buses for digital data include universal serial bus (USB), serial peripheral interface (SPI), inter-integrated circuit (I2C), and Dallas 1-wire. Transducers 105 could also be connected via wireless protocols such as Bluetooth or Wi-Fi, and in this case transducer interface 101h could include a radio similar to radio 101z.

A transducer bus 101i can connect components in order to communicate with transducers 105. Exemplary components for transducer bus 101i include general purpose input/output (GPIO) controller 101k, an analog to digital controller (ADC) 101m, a digital-

to-analog controller (DAC) 101n, and another controller 101j. GPIO 101k could comprise electrical pins that are unused by default but could be programmed via instructions residing in SE RAM 102b or device RAM 101d, where the instructions set the pins on GPIO high or low depending on the transducer 105 connected or configuration of device 101. As one
5 example software or firmware operating on device 101 could program GPIO 101k to operate as a traditional serial or RS-232 interface, and other possibilities exist as well. In an exemplary embodiment, the software package 104q depicted and described below in connection with Figure 5 can include software to program GPIO 101k. ADC 101m could take analog input from transducer 105 and convert it to digital signals for further
10 processing by device 101 or SE 102. As one example, ADC 101m could read a level of a voltage input by measuring the voltage level and outputting a digital value of the measured voltage level, such as an exemplary reading of 0.123 volts.

DAC 101n could take digital input from device 101 or SE 102 and convert it to analog signals for transducer 105. As one example, DAC 101n receive a digital signal
15 from device 101 to output a voltage level, and convert the digital signal to the analog level output, such as an exemplary value of 0.123 volts. Another controller 101j could comprise a different controller than GPIO 101k, ADC 101m, and DAC 101n listed above, such as a controller customized for a particular transducer 105 or application of device 101. Another controller 101j could be a USB controller, military standard 1553 controller, peripheral
20 component interconnect express (PCIe) bus, or an Ethernet networking interface including power over Ethernet. Other possibilities exist as well for another controller 101j operating with a transducer interface 101h without departing from the scope of the present invention.

In exemplary embodiments, transducer interface 101h can connect with several types of transducers, such as (i) a transducers input 105b that could comprise a sensor, or
25 (ii) a transducer output 105c that could comprise an actuator, or (iii) a bi-directional transducer 105a. A bi-directional transducer 105a can transfer data (a) from device 101 to transducer 105a and (b) from transducer 105a to device 101. Exemplary bi-directional transducers 105a could include a radio 105aa or a payment terminal 105ab, and other examples exist as well. Radio 101aa could comprise a radio similar to radio 101z and be
30 connected with an antenna. The transducers 105 shown in Figure 1b are depicted to be illustrative as opposed to limiting. For example, another bi-directional transducer 105a could be a touch screen, where (x) data transmitted to the touch screen from device 101 can provide visual display information such as pixel status or color, and (y) data received from the touch screen going to device 101 is touch information such as location of manual touch
35 on the screen.

Transducer input 105b could support transducers that provide digital or analog data input into transducer bus 101i. Exemplary devices for transducer input 105b depicted in Figure 1a include a probe 105ba, a keypad 105bb, a sensor 105bc, and a switch 105bd. In other words, transducer input 105b could receive data about monitored unit 106 and forward the data to device 101 through transducer interface 101h and transducer bus 101i. Probe 105ba could measure values for monitored unit such as pH, salinity, humidity, voltage, 3-axis orientation, acceleration, or similar physical characteristics of monitored unit 106. Keypad 105bb could be a pad for a device user 101a to enter numbers or letters, including a keyboard or a keypad for entering a PIN code on a payment terminal, or a password. Sensor 105bc could be operated in an active or passive manner. Sensor 105bc can receive an electrical current input from device 101 in an active mode. An exemplary sensor 105 could comprise a thermistor or thermocouple. A sensor 105bc could also comprise a camera for generating a picture or video from the outside of device 101. Switch 105bd could comprise a switch operated by a user 101a or an external automated process associated with monitored unit 106. For example, switch 105bd could forward data to device 101 when electrical contacts associated with switch 105bd are open or closed. Other possibilities exist as well for transducer input 105b without departing from the scope of the present invention.

Transducer output 105c could support transducers that receive digital or analog data output from transducer bus 101i. Exemplary devices for transducer output 105 depicted in Figure 1b include voltage 105ca, a display 105cb, a relay 105cc, and an actuator 105cd. Voltage 105ca could be a voltage output by device 101 for controlling monitored unit 106. For example, monitored unit 106 could include a physical interface similar to transducer interface 101h, which could receive an output voltage 105ca (e.g. output from device 101 but input into monitored unit 106). Voltage 105ca could control a level of a lighting system, a speed of a motor, or other values associated with monitored unit 106. Display 105cb could be a light emitting diode (LED), liquid crystal display (LCD), a computer screen or monitor, or similar systems of displaying information to users 101a. Relay 105cc could be a relay with electrical contacts that are opened or closed via input from device 101, such that the state of monitored unit 106 changes when relay 105cc is opened or closed. An exemplary relay 105cc could be a lock on a door. Actuator 105cd could comprise devices that receive analog or digital input from device 101 in order to change or control the state of monitored unit 106. Exemplary actuators 105cd include electric motors, hydraulic cylinders, piezoelectric actuator, a speaker, solenoids, etc.

Although transducers 105 in Figure 1b are depicted as external to device 101, transducers 105 could be internal to device 101, or device 101 could utilize a mix of

internal and external transducers 105 connected with a transducer interface 101h. Further, a device 101 could implement multiple different transducer interfaces 101h, such as a first transducer interface 101h connects internal transducers 105 and a second transducer interface 101h connects external transducers. Other possibilities exist as well without departing from the scope of the present invention.

As depicted in Figure 1b, SE 102 could also include an EEPROM 102j. EEPROM 102j could comprise long-term, nonvolatile memory storage chipsets or physical units that are designed primarily for writing once or a few times, and reading many times. As contemplated within the present invention, a read-only address within EEPROM 102j could comprise a memory address or another hardware address for read-only operations accessible via bus 102i. Although depicted as using EEPROM 102j, the data and elements depicted within EEPROM 102j in Figure 1b could be alternatively stored in flash memory 102d, where the memory within 102d is tagged or set as being “read only” during most of the life of operation of SE 102. In other words, a physical EEPROM chip or memory cells are not required for some embodiments in order to implement the functionality of EEPROM 102j depicted in Figure 1b.

EEPROM 102j could include a certificate cert0.SE 102k, firmware 102u, and secret key SK0.SE 102y, symmetric encryption key 102z and message authentication code (MAC) key 102za. Cert0.SE 102k for SE 102 can be the default, installed certificate within SE 102 and can take a form similar to SE certificate 419 depicted and described in connection with Figure 4b and Figure 4d below. A device manufacturer 101x could deliver device 101 to user 101a with certificate cert0.SE 102k pre-installed as well as SK0.SE 102y recorded in a nonvolatile memory. Device manufacturer 101x could receive SE 102 from a processor manufacturer with certificate cert0.SE 102k configured within non-volatile memory of SE 102, or device manufacturer 101x could write certificate cert0.SE 102k and corresponding SK0.SE 102y (and/or cert0.SE 102k' and SK0.SE 102y') to SE 102 during a pre-configuration step conducted before the depicted configuration step 103 from Figure 1a. SK0.SE 102y can also comprise two different values or numbers, where a first number is associated with a key exchange such as key exchange step 411a below in Figure 4a and Figure 4b, and a second value or number associated with a digital signature algorithm such as generating signatures 309 and 416 in Figure 2b. Or, the two values or numbers could be represented by separate keys SK0.SE 102y and SK0.SE 102y', respectively.

In exemplary embodiments, SE 102 can also include, record, or store a second secret key SK0.SE 102y'. The first SK0.SE 102y from the above paragraph could be used with key exchange algorithms such as ECDH key exchange step 411a in Figure 4a and

Figure 4b below. The second key SK0.SE 102y' could be used with digital signature algorithms such as those depicted for SE 102 in Figure 2b below. Likewise, SE 102 could also include, record, or store a second certificate cert0.SE 102k', where the second certificate cert0.SE 102k' could be used with digital signature algorithms and include a public key corresponding to the second secret key PK0.SE 102y'. In some embodiments, the second secret key SK0.SE 102y' and the second certificate cert0.SE 102k' could be omitted, and the key SK0.SE 102y could be used with both key exchange step 411a and digital signatures.

In addition, symmetric encryption key 102z and MAC key 102za along with firmware 102u could be written by a manufacturer into non-volatile memory of SE 102 (such as EEPROM 102j) before distribution of device 101 to users 101a. Symmetric encryption key 102z can comprise a key to encrypt and decrypt ciphertext similar to symmetric encryption key 442a depicted in Figure 4b below. Key 102z can be used with a symmetric encryption algorithm 444a depicted and described in connection with Figure 4b below. The ciphertext for key 102z can comprise firmware 102u or components of firmware 102u, such that the firmware 102u or components of firmware 102u could be securely stored outside SE 102 in an encrypted format and loaded into SE 102 after decryption using key 102z. MAC key 102za can be associated with key 102z in order to verify integrity of ciphertext encrypted or decrypted with key 102z.

Certificate cert0.SE 102k can include an identity ID.SE 102m, public key PK0.SE 102n, certificate parameters 102p, and a signature0.CA0 102q. Identity ID.SE 102m may comprise a unique identity for SE 102, in order to identify SE 102 among a plurality of devices 101 that connect with either access network 126, configuration system 114, or reporting system 120. In exemplary embodiments, ID.SE 102m remains the same for a given physical SE 102, even as new or subsequent certificates are issued for SE 102. PK0.SE 102n can comprise a public key for SE 102, and an exemplary public key within an X.509 certificate is depicted in Figure 4d below as PK1.SE 410a. In an exemplary embodiment, the ID.SE 102m is included as the field "serialNumber" (SN) in the subject of an X.509 certificate, or alternatively the ID.SE 102m could be in the "commonName" field (CN).

Certificate parameters 102p can comprise parameters associated with cert0.SE and exemplary parameters for certificate parameters 102p include the parameters 407 depicted in Figure 4b below and also the parameters 407 in Figure 4d below. Certificate parameters 102p can specify algorithms such as RSA or elliptic curve cryptography (ECC) values, ECC curve names, key lengths in bits, signature algorithms, validity dates, encoding rules, etc. Signature signature0.CA0 102q may comprise the signature of the certificate authority

for SE 102, and the public key of the certificate authority for SE 102 could be included in cert.CA0 107. A node of system 100 in Figure 1a can use cert0.SE and cert0.CA 107 to verify that the PK0.SE 102n belongs to SE 102 with identity ID.SE 102m. Additional details regarding the use of public keys, private keys, signatures, and certificates for the components in Figure 1a and Figure 1b will be discussed in figures below, including Figure 2b.

In exemplary embodiments, firmware 102u for SE 102 can include SE bootloader firmware 102t, and SE boot configuration 102v. SE bootloader firmware 102t in EEPROM 102j can provide machine executable code for SE CPU 102a to initiate operations when electrical power is provided to device 101 and SE 102. SE bootloader firmware 102t in EEPROM 102j could also include instructions for SE CPU 102a to load config0.SE recorded in nonvolatile memory 102d, if present, where config0.SE is described with SE nonvolatile memory 101d. SE bootloader firmware 102t could load boot firmware for SE 102, where the boot firmware for SE 102 could be stored in either memory 102d or memory 102x. If boot firmware for SE 102 is stored in memory 102x (where boot firmware for SE 102 is called by bootloader firmware 102t), then in exemplary embodiments boot firmware SE 102 is encrypted with key 102z since the boot firmware would be stored outside SE 102. In exemplary embodiments, bootloader firmware 102t includes a set of cryptographic algorithms 141, where cryptographic algorithms 141 as a minimum supports the use of key 102z and a symmetric decryption algorithm 444b (see Figure 4b) in order to decrypt boot firmware for SE 102 stored outside SE 102, as well as use MAC key 102za to verify integrity of a boot firmware stored in memory 102x.

Bus control libraries 142 could include software or firmware to manage and schedule the input and output of data for SE 102, such as machine code for (i) instructing processor 102a to write data to bus 102i for bus controller 102h or bus controller 102w when data is transmitted outside SE 102, (ii) read data from bus controller 102h or bus controller 102w when data from device 101 or transducer 105 is passed into SE 102, and (iii) reading SK0.SE 102y and/or SK0.SE 102y' from protected memory. Bus control libraries 142 can also include bus read instructions 142a and bus write instructions 142b. Bus read instructions 142a can provide machine executable code for SE CPU 102a to read data from either (i) device 101 using bus controller 102h, or (ii) transducer 105 using bus controller 102w. In this manner, bus read instructions 142a could provide the logical software or firmware interface for SE 102 to receive external data. Bus read instructions 142a could specify memory addresses, registers, or bus addresses where device 101 or transducers 105 can write data in order to be read by SE 102. Bus write instructions 142b

can provide machine executable code for processor 102a to write data output from SE 102 in order to be read by device 101 or transducer 105.

Firmware 102u in SE 102 for device 101 can also include a SE boot configuration 102v, a certificate authority public key recorded in cert.CA0 107 and a root certificate cert.CA.root 109, as depicted in Figure 1b. SE boot configuration 102v can provide values for the configuration or operation of SE 102 when operating with the SE bootload firmware 102t, such as (i) parameters for using bus 102i or controllers 102h and 102w, (ii) the frequency to operate a clock or phase looped logic associated with SE 102, (iii) a firmware version number, (iv) the memory capacity of memory 102d and 102x, (v) the memory addresses, cells, or file sectors to utilize for bus controllers, and (vi) parameters specifying values for hardware within SE 102. In exemplary embodiments, SE boot configuration 102v specifies the activation of radio 102c, such that if (i) a configuration step 103 has not been conducted to configure SE 102, then radio 102c is active, but (ii) if a configuration step 103 has been conducted then SE boot configuration 102v specifies that radio 102c is inactive or turned off. Note that config.SE 102aa could specify the activation of radio 102c instead of SE boot configuration 102v. SE boot configuration 102v may specify the parameters for use of boot firmware 102u, while parameters config0.SE 102aa may specify parameters for operating firmware of SE 102 that is loaded by boot firmware 102t. Cert.CA0 107 and cert.CA.root 109 were also described above in connection with Figure 1a. Note that by recording cert.CA0 107 and cert.CA.root 109 in EEPROM 102j for SE 102, including during manufacturing, these certificates for the certificate authority can be securely distributed along with SE 102 for subsequent use such as verifying signatures as described below.

Figure 2a

Figure 2a is a simplified message flow diagram illustrating an exemplary system with exemplary data sent and received by a mobile handset, in accordance with exemplary embodiments. System 200 can include a mobile handset 108, device 101, a discovery server 110 and authentication server 111. Mobile handset 108 can communicate with discovery server 110 and authentication server 111 via IP network 128. Mobile handset 108 can comprise a smart phone such as a phone based on an Android operating system from Google ® or IOS from Apple ®. Mobile handset 108 can include a battery, radio, and touch screen, as well as a camera 108r. Mobile handset 108 can connect with an access network similar to access network 126 in order to communicate with discovery server 110 and authentication server 111, and the access network used in a system 200 can be a different access network than access network 126 used by device 101 in Figure 1a and

Figure 5. Device 101 in system 200 can comprise a device 101 depicted in Figure 1a and Figure 1b, and device 101 can also include a tag 101s. Tag 101s can comprise a recorded image on device 101 or packaging of device 101 and could be a bar code, QR code, or a serial number for device 101. Tag 101s can provide a unique identifier for device 101. In
5 exemplary embodiments, if device 101 is a medical device, then tag 101s can correspond to a Unique Device Identification (UDI).

Although Figure 2a depicts a mobile handset 108, a configuration unit as described above in Figure 1a could be utilized in system 200 instead of a mobile handset 108. The configuration unit could perform equivalent steps and conduct equivalent message flows as
10 depicted in Figure 2a, Figure 3, and related figures below. In other words, a configuration unit could be a device that is different than a regular mobile handset 108, but perform similar steps and provide similar functionality. In an exemplary embodiment, mobile handset 108 could be a configuration unit customized for operation in systems 100, 200, 300, etc. Mobile handset 108 is depicted for preferred exemplary embodiments because
15 mobile handset 108 could be a relatively ubiquitous piece of equipment for a device user 101a or configuration user 108a, but other equipment could provide equivalent functionality as mobile handset 108.

Either a configuration unit or mobile handset 108 operating in system 200 and related systems below can implement similar elements to those depicted in Figure 1b for a
20 device 101, such as having a processor, memory, data bus, radio, storage, etc. The storage and memory of a configuration unit or mobile handset 108 could record an operating system to manage the hardware devices for mobile handset 108 or a configuration unit, and a software or a firmware application 101b depicted in Figure 1a and Figure 3 could conduct the message flows for unit 108 herein. In other words, for exemplary embodiments using a
25 configuration unit operating instead of the depicted "mobile handset" 108, the configuration unit could also include a configuration application 108b, and an NFC radio 108f, etc. as depicted below for mobile handset 108.

In another embodiment for Figure 2a, tag 101s could comprise an NFC tag such as a tag compatible with the NFC Forum standards for type 1 through type 5 tags (and
30 subsequent or related standards). The NFC technology could also be NFC-A, NFC-B, or NFC-V, or subsequent standards. For these exemplary embodiments where tag 101s comprises an NFC tag, then mobile handset 108 could use an NFC reader instead of camera 108r, and the NFC reader could comprise an NFC radio 108f as depicted and described in connection with Figure 3 below. Mobile handset 101 could utilize an NFC chip and
35 antenna, where the NFC chip operates in read mode with tag 101s. In exemplary embodiments, discovery server 110 can include or be associated with a discover server

database 110a. Database 110a could record information about a plurality of devices 101, such as ID.device 101b, and ID-token.device 206, and a dataset config-provisioning.ID.device 212, which will be described in further detail below for Figure 2a.

At step 201 in Figure 2a, device 101 can be in a powered off state. In alternative
5 embodiments, device 101 could be powered on at step 201, although power for device 101 is not required for embodiments with the steps and message flows outlined in Figure 2a. At step 202, mobile handset 108 could be powered on and placed in proximity of device 101. Mobile handset 108 could be operated by a device user 101a or a configuration user 108a at step 202. At step 203, device user 101a or configuration user 108a could use
10 mobile handset 108 to conduct a “read” step 203, where mobile handset 108 can (i) take a picture of a tag 101s for device 101, or (ii) read a bar code, QR code, or similar code for device 101. As mentioned above, tag 101s could comprise a bar code, QR code, or other encoding of data for device 101, including a serial number. For the alternative embodiment where (i) mobile handset 108 uses an NFC reader and (i) tag 101s comprises an NFC tag,
15 the mobile handset 108 could conduct an NFC read of tag 101s for step 203 and step 204.

For some exemplary embodiments, mobile handset 108 could receive data for the tag 101s (such as URL-DS 205, ID-token.device 206, and tag value 101w) via a packet or message received over a local wireless link between mobile handset 108 and device 101. The local wireless link could comprise any of the example wireless links of Wi-Fi,
20 Bluetooth, and near field communications, and other possibilities exist as well. With Wi-Fi networks, device 101 could periodically transmit data for tag 101s as a broadcast message, and mobile handset 108 could listen for broadcast messages in order to read the tag 101s.

At step 204, mobile handset 108 can receive data from tag 101s, such as a response, where the response can consist of a uniform resource locator (URL) for discovery server
25 110, URL-DS 205, an ID-token.device 206, and a tag value 101w. URL-DS 205 could include a domain name for discovery server 110 and parameters such as the use of HTTP or HTTPS and a destination TCP or UDP port number for sending packets to discovery server 110. ID-token.device 206 could be a token for ID.device 101b, or a value or unique pseudo-random number associated with device 101 and/or ID.device 101b. ID-
30 token.device 206 could be a value that obfuscates ID.device 101b but can preferably uniquely map to ID.device 101b. Through the use of ID-token.device 206 that is readable by mobile handset 108, then the actual value for ID.device 101b can remain confidential at step 204. Tag value 101w in response can include data associated with tag 101s and device 101, such as versions of software supported, product validity or expiration dates,
35 cryptographic parameters supported, or a subset of config-provisioning.ID.device 212. In an exemplary embodiment, tag value 101w could comprise some values of config-

provisioning.ID.device 212 upon manufacturing of device 101, which could be subsequently updated at a later date in discovery server DB 110a before installation or operation of device 101.

At step 207, mobile handset 108 could process the data received in response 204.
5 For embodiments where tag 101s is an image such as a bar code or QR code, at step 207 mobile handset 108 could process the image and extract the data or values for ID-token.device 206, URL-DS 205, and tag value 101w. Continuing at step 207, mobile handset 108 could connect with an access network 126 in order to call the URL in URL-DS 205, which could be an HTTPS/TLS session in exemplary embodiments. In exemplary
10 embodiments, URL-DS 205 is combined with ID-token.device 206, such that the HTTPS session is a unique request to discovery server 110 for each device 101. At step 207, mobile handset 108 could perform additional checks and steps in preparation for communication with device 101, such as verifying data in tag value 101w is compatible with software in mobile handset 108, including configuration software 108b.

15 In message flow 208, mobile handset 108 can establish a secure HTTPS session with discovery server 110 using the URL-DS 205, and the secure session could comprise a transport layer security (TLS) session such as TLS version 1.2, TLS version 1.3, or similar and subsequent standards. As part of the setup of message flow 208, mobile handset 108 can receive a certificate cert.DS 110b from discovery server 110. Cert.DS 110b could
20 comprise an X.509 certificate similar to the certificate depicted in Figure 4d below, and include a signed public key for discovery server 110. In exemplary embodiments, mobile handset 108 verifies the certificate cert.DS 110b using the public key in certificate cert.CA1 115. The verification of signatures using public keys in certificates is similar to the steps for verifying signatures outlined below in step 221 for Figure 2b. Although not
25 depicted in Figure 2a, in exemplary embodiments mobile handset 108 can also authenticate with discovery server 110, such as a configuration user 108a associated with mobile handset 108 entering user credentials in a web site for discovery server 110. Or mobile handset 108 could send discovery server 110 a certificate for mobile handset 108, which the discovery server 110 could verify using a step 221 in Figure 2b.

30 In message 209 and after the secure HTTPS/TLS session setup in message flow 208, mobile handset 108 can send discovery server 110 ID-token.device 206 and tag value 101w. Mobile handset could alternatively send discovery server 110 a subset of the data for ID-token.device 206 and/or tag value 101w. At step 210, discovery server 110 can use data in ID-token.device 206 to query discovery server database 110a for the values or data
35 sets ID.device 101b and config-provisioning.ID.device 212. Discovery server database 110a may comprise a database with tables to track configuration information for a plurality

of devices 101. In exemplary embodiments, Config-provisioning.ID.device 212 contains data used by mobile handset 108 for the configuration of device 101 and SE 102. Exemplary data depicted for config-provisioning.ID.device 212 in Figure 2a includes configuration application “configuration application” 108b, software parameters 108c, version number 108d, ID.Authentication Server 111a, URL-authentication Server 111b, cert.AS 111c, auth. parameters 111d. Additional or other data for mobile handset 108 and device 101 could be included in discovery server database 110a and values for config-provisioning.ID.device 212.

Configuration application 108b can specify the name or URL of a software application for mobile handset 108 to utilize in order to configure device 101 and SE 102. Mobile handset 108 could utilize the name or URL to fetch the software for configuration application 108b. Configuration application 108b could be a public or private application for the operating system of mobile handset 108. For embodiments where mobile handset 108 utilizes an Android-based operating system, configuration application 108b could be downloaded from the Google Play store or a private web site associated with configuration system 114. For embodiments where mobile handset 108 utilizes an Apple ® / IOS –based operating system, configuration application 108 could be distributed through a developer enterprise program or downloaded through iTunes ®. Other possibilities exist as well without departing from the scope of the present invention for a mobile handset 108 to download a configuration application 108b, where the name of configuration application 108b is received by a discovery server 110.

As depicted in Figure 2a, config-provisioning.ID.device 212 sent to mobile handset 108 may also include software parameters 108c and version number 108d. Software parameters 108c may include values and data for mobile handset 108 to utilize configuration application 108b and communicate with device 101 using an NFC radio 108f. Exemplary values include parameters 108c for mobile handset 108 to conduct a configuration step 103, such as, but not limited to, (i) cryptographic algorithms to utilize, (ii) ECC curve names and key lengths and associated data, (iii) selection and parameters for a radio 101z (below) in mobile handset 108 to communicate with device 101, including the selection of one of multiple possible radios in mobile handset 108, (iv) addresses, names, and port numbers to utilize with IP network 128, (iv) timers and retry counts for communication with device 101 or configuration server 112 or authentication server 111, (v) parameters for configuration system 114, (vi) parameters for communicating with device 101, etc. Version number 108d for config-provisioning.ID.device 212 may specify a version number of configuration application 108b to utilize, or a list or dataset of version

numbers of standards supported, such as IP version of IP network 128, NFC type or version for an NFC radio 108f, TLS version supported, etc.

As depicted in Figure 2a, config-provisioning.ID.device 212 sent to mobile handset 108 may also include ID.authentication Server 111a, URL-authentication Server 111b, cert.AS 111c, auth. parameters 111d. ID.Authentication Server 111a could comprise an identity or name for authentication server 111, such as a domain name or multiple domain names for multiple servers. URL-Authentication Server 111b could comprise a URL for configuration application 108b to call in order to authenticate with authentication server 111. In some exemplary embodiments, URL-authentication server 111b includes either ID.Device 101b or ID-token.Device 206, so that when mobile handset 108 using configuration application 108b calls URL-authentication Server 111b, the URL includes a unique string that identifies device 101.

In another exemplary embodiment, mobile handset 108 can (a) call URL-authentication server 111b without an identity for device 101, and subsequently (b) send either ID.Device 101b or ID-token.Device 206. Cert.AS 111c could comprise a certificate for authentication server 111 in a form similar to the exemplary certificate depicted below in Figure 4d, where cert.AS 111c could include a public key for certificate server 111 and a signature of a certificate authority. In exemplary embodiments, a certificate for a certificate authority associated with cert.AS 111c is sent with cert.AS 111c. Authentication parameters 111d can include parameters for configuration application 108b to utilize with authentication server 111, such as, but not limited to, hash or message digest algorithms, digital signature algorithms, padding schemes, key lengths, nonce or challenge lengths, timers, and retry counts, encoding rules, etc.

The above fields and values described for config-provisioning.ID.device 212 and ID.device 101b could be sent from discovery server 110 to mobile handset 108 in a response 211 as depicted in Figure 2a. In an exemplary embodiment, discovery server 110 can send mobile handset the full configuration application 108b with the response 211 instead of a name or URL for configuration application 108b, if the current, updated configuration application 108b is not already downloaded by mobile handset 108. In another exemplary embodiment, mobile handset 108 can download configuration application 108b in a step 213 as depicted in Figure 2a, if mobile handset 108 has not already installed the software for configuration application 108b after receiving response 211. For example, response 211 could include a version number 108d that is higher or different than initially utilized by mobile handset 108 and/or an initial configuration application 108b, and consequently the higher version number 108d could indicate to mobile handset 108 that a different configuration application 108b would be required. In

exemplary embodiments, configuration application 108b can include a hash signature similar that is created using a signature creation 220 step as depicted in Figure 2b below. Step 213 may also include mobile handset 108 launching the configuration application 108b and applying a subset of software parameters 108c or tag-value 101w. If any of the
5 steps depicted in Figure 2a fail, error codes or messages could be displayed through mobile handset 108 to configuration user 108a.

After successful launch of configuration application 108b, mobile handset 108 can connect with authentication server 111. Secure session setup 214 could comprise a secure
10 HTTPS session with authentication server 111 using the URL-authentication server 111b, and the secure session could comprise a transport layer security (TLS) session such as TLS version 1.2, TLS version 1.3, or similar and subsequent standards. Other networking standards besides HTTPS and TLS could be utilized as well for secure session setup 214, such as an IPsec tunnel. As part of the setup of secure session setup 214, mobile handset
15 108 can receive a certificate cert.AS 111c from authentication server 111, if mobile handset 108 had not already received cert.AS 111c from discovery server 110. Mobile handset 108 could verify cert.AS 111c after receiving the certificate, such as verifying a signature from a certificate authority and parent certificates of cert.AS 111c as well.

In message 215, mobile handset 108 transmits authentication information to authentication server 111, and authentication server 111 verifies the authentication
20 information in a step 217. The authentication information transmitted in a message 215 could comprise information to authenticate (i) configuration user 108a, (ii) mobile handset 108, or (iii) both configuration user 108a and mobile handset 108. The authentication information in message 215 may comprise a configuration user 108a entering identification and credential information in a touch screen of mobile handset 108, and the information is
25 passed by mobile handset to authentication server 111. In exemplary embodiments, configuration user 108a provides biometric data such as a fingerprint, an image of the user's face, or similar data, where mobile handset 108 transmits the data through secure session setup 214 to authentication server 111, and the authentication server 111 verifies the biometric data is associated to configuration user 108a. Message 215 may also include
30 the verification of mobile handset 108 with authentication server 111.

In exemplary embodiments, mobile handset 108 records a certificate for mobile handset 108, where the certificate could be similar to the one depicted in Figure 4d below. The certificate for mobile handset 108 includes a public key and a signature by a certificate authority. In an authentication step 217, authentication server can use the information in
35 message 215 to verify the identity of a configuration user 108 and/or mobile handset 108. In embodiments where mobile handset 108 uses a device certificate for mobile handset

108, authentication server 111 could verify that mobile handset 108 also records the private key, by sending a challenge and verifying a signature from mobile handset 108 using the private key. Message 215 and authentication 217 could be conducted using authentication parameters 111d, which could be previously included in Config-provisioning.ID.device
5 212.

In step 216a of Figure 2a, mobile handset 108 may also authenticate authentication server 111, using cert.AS 111c. In step 216b, configuration user 108b can enter device owner 122 information into mobile handset 108, such as an owner name, a DNS name associated with owner 122, or a code or other value to identify owner 122. Device owner
10 122 information could be automatically recorded for configuration application 108b before a configuration step 103 from Figure 1a. The value or information associated with device owner 122 may also be acquired from monitored unit 106, for embodiments where device owner 122 also owns or is responsible for monitored unit 106. The value associate with device owner 122 could be ID.owner 122a. In another exemplary embodiment, device
15 owner 122 information could be recorded in mobile handset 108 earlier than a step 216b, and the information could be fetched from memory in a step 216b. A step 216b can function to associate device 101 with device owner 122 for a configuration system 114 and/or reporting system 120. In exemplary embodiments, millions of devices 101 could be distributed out to thousands or more device owners 122, and configuration system 114 and
20 reporting system 120, as well as device owner 122, may need to know which devices 101 have been configured and/or installed with monitored units 106.

In message 218, mobile handset 108 can send a first list of identities to authentication server 111 through the secure session setup in 214. The list of identities can include an ID.device 101b, ID.MH 108e, and ID.owner 122a, as depicted in Figure 2a. In
25 steps 219, authentication server 111 can process the list of identities received in message 218. A first sub-step 219a could comprise authentication server 111 verifying that mobile handset 108 with configuration user 108a is authorized to configure device 101. In other words, the prior step 217 could be an initial step to verify identities, and then the subsequent step 219a could be verifying that the authenticated identities have privileges or
30 authorization to conduct a configuration step 103. As depicted in Figure 2a, authentication server 111 may need to check or communicate with device owner 122 in order to confirm that mobile handset 108 with configuration user 108a is authorized to conduct configuration step 103 on device 101.

Authentication server 111 can utilize the ID.owner 122a received in message 218 in
35 order to identify and communicate with the correct device owner 122. Although not depicted in Figure 2a, authentication server 111 could utilize a mutually authenticated

secure channel such as TLS v3 with device owner 122 in a step 219a and 219b. In some exemplary embodiments, a step 219a could be optionally omitted, for example if device owner 122 is also device user 101a, such as a home owner installing a device 101 in their home. In these embodiments, an authentication step 217 may be sufficient to allow mobile handset 108 to conduct the configuration step 103. Other possibilities exist as well for an authentication server to authenticate a mobile handset 108 and/or configuration user 108a without departing from the scope of the present invention.

In step 219b, authentication server 111 can query device owner 122 for a certificate for secure processing environment 102 (SE 102), and the certificate could comprise cert0.SE 102k and cert0.SE 102k', where cert0.SE 102k can record PK0.SE 102n and cert0.SE 102k' can record PK0.SE 102n'. Note that in some exemplary embodiments cert0.SE 102k can record two public keys for device 101 of both PK0.SE 102n and PK0.SE 102n'. The query could be made using ID.device 101b, which was previously received in message 218. Cert0.SE 102k could include ID.SE 102m, or ID.SE 102m could be received by authentication server 111 separately from cert0.SE 102k. In a different exemplary embodiment, the query in a step 219b could utilize ID-token.device 206, which could be (i) obtained by mobile handset 108 in read 203, and (ii) sent by mobile handset 108 in message 218. Cert0.SE 102k could also be a certificate for device 101 in some exemplary embodiments, where device 101 does not use separate certificates for the device and secure processing environment.

If owner 122 does not record the certificate cert0.SE 102k, then for step 219a owner 122 could point authentication server 111 to another server or location for the certificate cert0.SE 102k, such as possibly device manufacturer 101x. In some exemplary embodiments, (i) a configuration system 114 could query another server associated with device 101 in order to obtain public key PK0.SE 102n, and (ii) the use of a cert0.SE 102k for recording the public key PK0.SE 102n is not required. Authentication server 111 could receive cert0.SE 102k in a step 219b. In another exemplary embodiment, authentication server 111 could record a plurality of certificates cert0.SE 102k for all devices 101 and in this embodiment then step 219b could represent querying a locally stored cert0.SE 102k with authentication server 111.

In step 220a, authentication server 111 can (i) create a random number random1.AS 223, and (ii) create a digital signature 224. The sub-steps for creating a digital signature are depicted in Figure 2b below for a signature creation step 220. Exemplary details for authentication server 111 to create specifically create digital signature 224 in a step 220a are depicted and described in connection with Figure 2b below.

Authentication server 111 can also conduct a step 220b, which comprises creating a digital signature 225 using the data received in step 219b above, such as the ID.SE 102m, where the digital signature 225 is a signature for the data of ID.SE 102m and a random1.AS 223. Step 220b is depicted in Figure 2b below, and can include the data and steps for a signature creation step 220 equivalent to the step 220a for a signature creation. Although the use of two signatures 224 and 225 are depicted in Figure 2a, where signature 224 can be used primarily by mobile handset 108 and signature 225 can be used primarily by device 101 and SE 102, the use of two separate signatures at step 220a and 220b can be optionally omitted. For example, a step 220a creating signature 224 to authenticate authentication server 111 with mobile handset 108 could be performed with a step 216a above.

As depicted in Figure 2a, authentication server 111 can send mobile handset 108 a message 222 through secure connection 214, where message 222 can include ID.Device 101b, Random1.AS 223, Signature1.AS (ID.Device 101b, Random1.AS 223) 224, and Signature2.AS (ID.SE 102m, Random 1.AS 223) 225. Although not depicted in Figure 2a, a message 222 could also include cert0.SE 102k. Random1.AS 223 may comprise a pseudo-random number generated by authentication server 111 for message 222 and signature 224. Random1.AS 223 and other random numbers utilized by systems 100, 200, 300, and subsequent figures may preferably be unique, or with enough bits and information entropy that collisions or re-use would be sufficiently rare enough to be insignificant for the overall security of a system 100. Signature1.AS 224 is described in Figure 2b below. Signature2.AS 225 was described in the paragraph above and also depicted in Figure 2b below. Authentication server 111 could utilize authentication parameters 111d in the creation of data for message 222. Although not depicted in Figure 2a, in exemplary embodiments, authentication server 111 could also include authentication parameters 111d for the data in signature1.AS 224 and signature2.AS 225, and in this manner authentication parameters 111d would be signed by authentication server 111.

In exemplary embodiments, ID.SE 102m is not sent either in encrypted format or as plaintext to mobile handset 108 from either (i) discovery server 110 or (ii) authentication server 111, or (iii) from other servers in a system 100. By not sending ID.SE 102m to mobile handset 108, ID.SE 102m can remain confidential, such that if mobile handset 108 is insecure and exposed to third parties, including hackers, ID.SE 102m can still remain confidential since it is not passed to mobile handset 108. Signature2.AS 225 (ID.SE 102m, random1.AS 223) is depicted in Figure 2b below, and can comprise a signature by authentication server 111 over random1.AS 223 and ID.SE 102m. ID.SE 102m can comprise a device value, where the device value is a unique number or string associated

with device 101. The device value can be recorded in SE 102 before a configuration step 103 such as during manufacturing of SE 102.

At step 221a in Figure 2a, mobile handset 108 can verify signature1.AS 224, where signature1.AS 224 can be over the values ID.device 101b and random1.AS 223. The sub-steps for verifying a digital signature are depicted in Figure 2b below for a signature verification step 221, and the details for mobile handset 108 to verify digital signature 224 in a step 221a are depicted and described in connection with Figure 2b below. ID.device 101b in message 222 can be useful to track that message 222 is properly received routed by authentication server 111 since both authentication server 111 and mobile handset 108 may connect with multiple different devices 101 over time. In step 221a, mobile handset 108 can take steps to verify that random1.AS 223 is unique and not reused by authentication server 111, such as checking random1.AS 223 with configuration system 114. In exemplary embodiments, authentication server 111 can create a signature 224 over a random number issued by mobile handset 108 instead of a random number generated by authentication server 111, which could be conducted in setup of secure session 214. Or, in preferred exemplary embodiments, authentication server 111 creates a signature 224 for both (i) a random number from mobile handset 108 in a step 216a and (ii) a random1.AS 223 in a step 220b. After conclusion of step 221a depicted in Figure 2a, mobile handset 108, device 101, and the elements depicted in Figure 2a could continue with the message flows and steps depicted in Figure 3 below.

In an alternative exemplary embodiment to that depicted in Figure 2a, discovery server 110 could be optionally omitted, and data within tag 101s or tag value 101w could specify authentication server 111 as well as data for Config-provisioning.ID.device 212. However, the use of a discovery server 110 can be preferred for some exemplary embodiments since a discovery server 110 could provide increased flexibility over the lifetime of device 101, which could be a decade or longer, since parameters Config-provisioning.ID.device 212 and related data could change over time. Including all data for config-provisioning.ID.device 212 in tag value 101w or tag 101s may reduce flexibility for changing or updating the data over time, such as specifying an updated configuration application 108b, or other updated data for configuration system 114 over time.

Figure 2b

Figure 2b is a flow chart illustrating exemplary steps for creating and verifying a digital signature using PKI keys, parameters, and data input, in accordance with exemplary embodiments. The processes and operations, described below with respect to all of the logic flow diagrams and flow charts may include the manipulation of signals by a processor

and the maintenance of these signals within data structures resident in one or more memory storage devices. For the purposes of this discussion, a process can be generally conceived to be a sequence of computer-executed steps leading to a desired result.

These steps usually require physical manipulations of physical quantities. Usually, though not necessarily, these quantities take the form of electrical, magnetic, or optical signals capable of being stored, transferred, combined, compared, or otherwise manipulated. It is convention for those skilled in the art to refer to representations of these signals as bits, bytes, words, information, elements, symbols, characters, numbers, points, data, entries, objects, images, files, or the like. It should be kept in mind, however, that these and similar terms are associated with appropriate physical quantities for computer operations, and that these terms are merely conventional labels applied to physical quantities that exist within and during operation of the computer.

It should also be understood that manipulations within the computer are often referred to in terms such as listing, creating, adding, calculating, comparing, moving, receiving, determining, configuring, identifying, populating, loading, performing, executing, storing etc. that are often associated with manual operations performed by a human operator. The operations described herein can be machine operations performed in conjunction with various input provided by a human operator or user that interacts with the device, wherein one function of the device can be a computer.

In addition, it should be understood that the programs, processes, methods, etc. described herein are not related or limited to any particular computer or apparatus. Rather, various types of general purpose machines may be used with the following process in accordance with the teachings described herein.

The present invention may comprise a computer program or hardware or a combination thereof which embodies the functions described herein and illustrated in the appended flow charts. However, it should be apparent that there could be many different ways of implementing the invention in computer programming or hardware design, and the invention should not be construed as limited to any one set of computer program instructions.

Further, a skilled programmer would be able to write such a computer program or identify the appropriate hardware circuits to implement the disclosed invention without difficulty based on the flow charts and associated description in the application text, for example. Therefore, disclosure of a particular set of program code instructions or detailed hardware devices is not considered necessary for an adequate understanding of how to make and use the invention. The inventive functionality of the claimed computer

implemented processes will be explained in more detail in the following description in conjunction with the remaining Figures illustrating other process flows.

Further, certain steps in the processes or process flow described in all of the logic flow diagrams below must naturally precede others for the present invention to function as described. However, the present invention is not limited to the order of the steps described if such order or sequence does not alter the functionality of the present invention. That is, it is recognized that some steps may be performed before, after, or in parallel other steps without departing from the scope and spirit of the present invention.

The processes, operations, and steps performed by the hardware and software described in this document usually include the manipulation of signals by a CPU or remote server and the maintenance of these signals within data structures resident in one or more of the local or remote memory storage devices. Such data structures impose a physical organization upon the collection of data stored within a memory storage device and represent specific electrical or magnetic elements. These symbolic representations are the means used by those skilled in the art of computer programming and computer construction to most effectively convey teachings and discoveries to others skilled in the art.

In Figure 2b, signature creation 220 can comprise a step using the sub-steps of obtaining a message to sign, calculating a message digest 230, using a private key 226, using a signature algorithm 227, inputting parameters 111d, and calculating a resulting signature 224. The steps and substeps depicted for authentication server creating signature 224 in Figure 2b can also be applied for the subsequent signatures depicted, including signature creation 220b through 220g.

Signature algorithm 227 and signature verification algorithm 228 could comprise an RSA-based digital signature algorithm (DSA) or an ECC based elliptic curve digital signature algorithm (ECDSA), and other possibilities exist as well for signature algorithm 227 and signature verification algorithm 228 without departing from the scope of the present invention. When using a DSA or ECDSA algorithm in non-deterministic mode for a signature creation 220, a value of “k” or “r”, which could comprise a random number can be associated with the digital signature 224. When using a DSA or ECDSA in deterministic mode, such as specified in IETF RFC 6979 and titled “Deterministic Usage of the Digital Signature Algorithm (DSA) and Elliptic Curve Digital Signature Algorithm (ECDSA)”, which are hereby incorporated by reference, then the requirement for a separately transmitted random number with digital signature 224 (such as value “k” or “r”) can be optionally omitted, such that “r” can be deterministically calculated based on the message to sign. In exemplary embodiments, device 101 and servers such as authentication sever 111 can utilize deterministic ECDSA without also sending a random number along with a

digital signature, although the value “r” from the deterministic mode could be sent with the digital signature 224. In other words, a value can be sent with the digital signature 224 that is deterministically derived and associated with the message to sign. In other exemplary embodiments, a random number can be generated a derived value for the random number such as “r” sent with digital signature 224.

For a signature creation 220a step, the exemplary message to sign comprises ID.device 101b and random I.AS 223. The message to sign values can be transmitted to the verifying party, such as shown for message 222 in Figure 2a. The message to sign values can be input into a message digest algorithm 230, which could comprise a standard algorithm such as SHA-256, SHA-3, or similar algorithms. The output of message digest algorithm 230 can be input along with parameters 111d and private key 226 into signature algorithm 227. Parameters 111d can specify encoding rules, padding, key lengths, selected algorithms, curve names, and other values or fields necessary to utilize a signature algorithm 227. Both a signature creation step 220 and a signature verification step 221 use the same or equivalent values for parameters 111d. Private key 226 comprises a private key for authentication server 111. For the additional signature creation 220 instances depicted in Figure 2b, such as signature creation 220b through 220g, the element or node creating the equivalent signature as signature 224 would utilize the private key of the element or node, such as SE 102 using a private key SK0.SE 102y’ in a signature creation instance 220c. Although not depicted in Figure 2b, a random number for values such as “k” and “r” for ECDSA and related signatures could be input as well into a signature algorithm 227. The output of a signature algorithm 227 can be a signature 224, which can be transmitted to another node for verification.

Signature verification 221 can comprise a step using the sub-steps of obtaining a message to verify, calculating a message digest 230, using a public key 228, using a signature verification algorithm 228, inputting parameters 111d and signature 224, and determining a pass/fail 229. If the signature 224 received matches a calculated signature with the public key 228 and message to sign, then the received signature 224 passes or is validated or verified. If the signature 224 does not match the calculated signature in a step 221, then the signature 224 is considered to fail or not be verified. The steps and substeps depicted for mobile handset 108 verifying signature 224 in Figure 2b can also be applied for the subsequent signature verifications depicted, including signature verification 221b through 221g.

Figure 3

Figure 3 is a simplified message flow diagram illustrating an exemplary system with exemplary data sent and received by a mobile handset and a device, in accordance with exemplary embodiments. Before initiating steps and message flows depicted in Figure 3, mobile handset 108, device 101, and the other elements depicted for system 300 in Figure 3 may previously complete exemplary message flows and steps depicted in Figure 2a above. System 300 can include a configuration server 112, authentication server 111, mobile handset 108, and device 101. Mobile handset 108 can communicate with authentication server 111 and configuration server 112 via IP network 128, as depicted in Figure 2a. For system 300, configuration server 112 and authentication server 111 may establish a secure session setup 301, which could comprise establishing a secure communications link between the two servers using protocols such as TLS, IPSec, a virtual private network (VPN), a secure shell (SSH), or similar networking, transport, or application layer technologies in order to establish secure communications between configuration server 112 and authentication server 111. Secure session setup 301 can be equivalent or similar to secure session setup 214 depicted in Figure 2a above.

Note that for some exemplary embodiments, a mobile handset 108 could be omitted, and device 101 and configuration system 114 (with authentication server 111 and configuration server 114) could communicate directly with device 101. For example, device 101 could use a radio such as a 4G or 5G radio or a Wi-Fi radio to communicate with configuration system 114 without using a mobile handset 108, and data session 303 could comprise a data session that includes WAN connectivity. For these embodiments, then the exemplary depicted messages sent or transmitted by configuration system of message 222 (from Fig. 2a), 312, etc. could be transmitted to device 101 instead of mobile handset 108. Further, messages sent or transmitted by device 101 of 307 and 314 could be sent to configuration system instead of mobile handset 108. However, for embodiments where device 101 does not have a configured WAN radio, then a system equivalent to system 200 and 300 could be preferred. Also, note that a different device than a mobile handset 108 could be used for establishing communication between device 101 and configuration system 114, and a Wi-Fi router or another node on a local wireless network could provide the same functionality of mobile handset 108 (where tag 101s could be transmitted via the local wireless network).

Mobile handset 108 in system 300 can operate a running configuration application 108b and an NFC radio 108f. Configuration application 108b may previously be downloaded in a step 213 as depicted in Figure 2a and subsequently launched or activated for operation in a system 300. In exemplary embodiments, configuration application 108b may be (i) included or distributed along with an operating system running on mobile

handset 108b or (ii) acquired at a different time than a step 213, and thus a separate download of configuration application 108b. NFC radio 108f may comprise a radio equivalent or compatible with radio 102c depicted for SE 102 in Figure 1b above. Although radio 108f is depicted in Figure 3 as a “Near Field Communications” (NFC) radio, in alternative embodiments a different radio than NFC could be utilized, such as a radio operating at ISM bands of 900 MHz, 2.4 GHz, 5.8 GHz, or other radio frequencies allowing unlicensed access at low powers such as less than 1 watt of transmitted radio power. As depicted in Figure 3, device 101 can include a secure processing environment (or “secure enclave” or “secure element”) SE 102 which may also have a NFC radio 102c, thereby enabling communication at short range between mobile handset 108 and device 101. In some exemplary embodiments, SE 102 may be connected with NFC radio 102c and NFC radio 102c may be external to SE 102, but in preferred exemplary embodiments NFC radio 102c may be inside device 101.

At step 302 in Figure 3, device 101 may power on from an unpowered or “deep sleep” state. Power could be provided from a battery or converted “wall power”. Device 101 could load portions of OS 101e from storage 101f into RAM 101d. SE 102 could load SE bootload firmware 102t and SE boot configuration 102v, and then SE firmware from SE memory 102x. In exemplary embodiments where SE 102 has sufficient storage memory to record SE firmware and an operating system for SE 102 in SE flash memory 102d, then SE firmware could be loaded from SE flash memory 102d instead of SE memory 102x. In exemplary embodiments, SE memory 102x within device memory 101f is encrypted with symmetric key 102z.

Mobile handset 108 and SE 102 can establish an NFC session setup 303 for peers and “peer-to-peer mode”. Either Mobile handset 108 or SE 102 could initiate the session once the two sides are sufficiently close for the radios to communicate. In exemplary embodiments, a screen on mobile handset 108 instructs configuration user 108a to place the mobile handset near device 101 or next to a target location on device 101 that would cover an antenna for SE radio 102c. In a first exemplary embodiment, NFC radio 108f in mobile handset 108 is the transceiver that initiates polling and in the “initiator” of NFC session setup 303. For this first exemplary embodiment SE radio 102c in SE 102 comprises the target and initially listens for polling from NFC radio 108f. In a second exemplary embodiment, SE radio 102c for SE 102 is the transceiver that initiates polling, and is the “initiator” of NFC session setup 303. After negotiating polling, NFC session setup 303 can utilize protocols for either NFC-A or NFC-F to establish a bi-directional communications channel, or subsequent and related versions of these standards. NFC-A is specified in NFC

Forum-TS Digital Protocol-1.0, which is hereby incorporated by reference and provides a communications channels at baud rates of 106 kbps.

In exemplary embodiments NFC-F is utilized to provide baud rates higher than 106 kbps, such as 212 or 424 kbps. The air interface for NFC session setup 303 could comprise a session using ISO/IEC 18092 / ECMA-340, "Near Field Communication Interface and Protocol-1" and ISO/IEC 21481 / ECMA-352, "Near Field Communication Interface and Protocol-2 (NFCIP-2)". The air interface for NFC session setup 303 could also comprise subsequent or related versions to these standards. Data between mobile handset 108 and SE 102 could be transferred using NFC Data Exchange Format (NDEF) and support a Multipurpose Internet Mail Extensions (MIME) typed object or file transferred from mobile handset 108 to SE 102. In other words, subsequent messages and files transferred using NFC radios could utilize NDEF and MIME objects, although other possibilities exist as well for the NFC standards utilized without departing from the scope of the present invention. NFC session setup 303 could also implement additional air-interface security, such as ECMA-409 2nd edition - NFC-SEC-02: NFC-SEC and related standards, where the air interface is encrypted using AES and Diffie-Hellman key exchanges. In exemplary embodiments, NFC session setup 303 utilizes standard ECMA-352 in order to select ECMA-340, ISO/IEC 14443 and ISO/IEC 15693 as a communication mode between radio 108f and radio 102c.

At step 304 in Figure 3, after successful setup of NFC session setup 303, configuration application 108b in mobile handset 108 could display to configuration user 108a that a communication link with SE 102 has been established. The display could be notification that authentication data can be transferred with SE 102, such as the data through the physical layer and data link layer of NFC radio 108f is enabled. Step 304 could also display error conditions or codes to configuration user 108a, for cases where NFC session setup 303 fails for exemplary reasons such as no NFC radio 102c detected, incompatible radio protocols, too many collisions, too weak an RF signal or RSSI, etc. Note that in an exemplary embodiment, secure session setup 214 may be temporarily disabled or unavailable (such as mobile handset 108 moves outside of range of a mobile network providing connectivity to authentication server 111). In this case, the values subsequently transmitted in message 305 and received in message 307 could be stored in mobile handset 108 until they can be transmitted from mobile handset 108. In this manner, if device 101 is located in a place without an access network 126, such as in a basement of a building where nearby land mobile networks cannot reach, then mobile handset 108 can take subsequent steps such as step 305 while mobile handset 108 is in an offline mode for access network 126.

In message 305, mobile handset 108 can send SE 102 data comprising certificate cert.AS 111c, random number Random 1.AS 223, authentication parameters Auth.Params.111d, and signature Signature2.AS (ID.SE, Auth.Params.111d, Random 1.AS) 225. These values could be previously received and recorded by mobile handset 108 during the prior communication depicted and described in connection with Figure 2a above. Although not depicted in Figure 3, mobile handset 108 could also send other exemplary data in a message 305 than that depicted, including parent certificates for cert.AS 111c, version number 108d, ID.device 101b, and/or ID.mobile-handset 108e. In exemplary embodiments, message 305 is transmitted without receipt of network, transport, or application layer data from SE 102, and SE 102 remains “quiet” or does not communicate data at the network, transport, or application layer with mobile handset 108 until after successfully receiving a message 305. In other words, although SE 102 may communicate at the data-link layer, such as transmitting values related to polling the NFC link, encryption of the radio air-interface for session 303, SE 102 does not respond to mobile handset 108 with application data until after message 305 has been properly received and verified. In this manner, SE 102 would (i) remain silent or idle to a mobile handset 108 or other NFC radios without receiving a verified message 305 and thus (ii) resistant to tampering or probing by radios or handsets that have not established an authenticated session with authentication server 111.

Although not depicted in Figure 3, in an exemplary embodiment a signature2.AS 225 could also be over an initial random number for SE 102 or device 101, which would be different than, in addition to, or in replace of the value random1.AS 223 for a signature 2.AS 225. The initial random number for SE 102 could be (i) recorded in the tag 101s depicted and described in Figure 2a, or (ii) recorded by owner 122 such that authentication server 111 could query owner 122 in a step 219a as depicted in Figure 2a above, or (iii) recorded with date in Config-provisioning.ID.device 212. The initial random number for SE 102 could also be recorded in memory 102d or memory 102j for SE 102 and created by a device manufacturer 101x. In this manner, the initial random number for SE 102 can be generated outside authentication server 111 and included in a signature2.AS 225. Security can be increased since authentication server 111 can be required to sign a number that was both (i) not generated by authentication server 111 and (ii) recorded in device 101 before a configuration step 103 or step 203.

In step 306a, SE 102 can receive the message 305 via the NFC session 303 and record the data in RAM 102b or storage 102d. In step 306a, SE 102 can also verify cert.AS 111c using either (i) cert.CA1 115 directly, and/or (ii) a commonly shared certificate cert.CA.root 109 shared between SE 102 and authentication server 111. In other words, in

exemplary embodiments and before a configuration step 103, SE 102 records in nonvolatile memory either (i) cert.CA1 115 or (ii) cert.CA.root 109, where a parent certificate for cert.AS 111c can be checked with cert.CA.root 109. As described in connection with Figure 2b above, SE 102 could verify the certificate for cert.AS 111c using a signature verification step 221.

After a step 306a, SE 102 can conduct a step 221b depicted in Figure 2b in order to verify signature 225. In exemplary embodiments, signature 225 can be over ID.SE 102m, randomI.AS 223, and authentication parameters 111d, where (i) randomI.AS 223 and authentication parameters 111d are received by SE 102 in a message 305 above and (ii) ID.SE 102m may optionally be not transmitted by mobile handset 108 but rather recorded by SE 102 in nonvolatile memory before configuration step 103. In this manner, SE 102 can receive a signature 225 for data (a) that has not been transmitted by SE 102 (e.g. ID.SE 102m, and/or optionally the initial random number for SE 102 described two paragraphs above), and also (b) mobile handset 108 would need to communicate with configuration system 114 in an authenticated manner in order to receive signature 225.

For embodiments depicted in Figure 3 and Figure 2a, where ID.SE 102m is not transmitted by device 101 before receipt of a message 305 with signature 225, then ID.SE 102m within signature 225 can also function as an authentication token for device 101 and SE 102. The use of an authentication token within signature 225 is also described in the paragraph below. In some embodiments, such as those using lower security levels or where an authentication token is not available, then the use of an authentication token such as ID.SE 102m could be omitted. In another exemplary embodiment, authentication parameters 111d may be optionally omitted from signature 225, such as both (i) not being included in the “message to sign” in signature creation 220 in Figure 2b and (ii) not being included in the “message to verify” in signature verification 221 in Figure 2b. However, including authentication parameters 111d in signature 225 may increase security for SE 102.

Although the use of an ID.SE 102m is depicted as being utilized for signature 225 received from mobile handset 108 and authentication server 111 (in Figure 2a above), other data recorded by SE 102 and not previously transmitted by SE 102 to mobile handset 108 could be utilized as well for the signature 225, such as an authentication token recorded in nonvolatile memory of SE 102. A signature 225 could also be over (i) the authentication token as described in the paragraph above, (ii) the initial random number for SE 102 described four paragraphs above (which could be recorded in the tag 101s), and (iii) randomI.AS 223. In other words, in a different embodiment than that depicted in Figure 3, for a message 305 SE 102 could receive a signature 225 over the authentication token

recorded in nonvolatile memory of SE 102, where (i) SE 102 has not transmitted the authentication token, but (ii) configuration system 114 could receive the authentication token from device manufacturer 101x or device owner 122 or possibly device user 101a. By receiving a signature 225 over data recorded in nonvolatile memory of SE 102 and not transmitted by SE 102, SE 102 can be reasonable assured that mobile handset 108 has authenticated and secured communication with configuration system 114.

In step 306b, SE 102 may use a random number generator 102e from Figure 1b to create a random number random1.SE 308, which could comprise a challenge or nonce in order to authenticate configuration system 114. As depicted in Figure 3, SE 102 may then conduct a step 220c from Figure 2b in order to create a signature 309 for random1.AS 223 using the initial secret key SK0.SE 102y' as the private key for signature 309 creation. Signature 309 could also be over additional data than random1.AS 223, such as random1.SE 308, although this alternative embodiment is not depicted in Figure 3. Further, SE 102 could send separate signatures 309 such as a first signature 309 over random1.AS 223 and a second signature 309 over random1.SE 308. SE 102 may then return a message 307 though NFC session 303 to mobile handset 108, where message 307 can include random1.SE 308 and signature 309, where signature 309 can be over at least random1.AS 223. Although not depicted in Figure 3, message 307 can also optionally include an identity for device 101 and the value of PK0.SE 102n, in order for a configuration system 114 to use with subsequent communications with device 101.

At step 310, mobile handset 108 can receive message 307 and display to configuration user 108a that application-layer data has been successfully transferred with SE 102. In other words, a first display progress indicator could be shown for configuration user 108a when NFC session setup 303 was completed, and a second display progress indicator for configuration user 108a could be shown when message 307 is received. The receipt of message 307 may indicate that message 305 was successfully received and properly processed by SE 102. For embodiments where mobile handset 108 records ID.SE 102m, mobile handset 108 could request for a cert0.SE 102k' from SE 102, owner 122, or configuration system 114 before a step 310, and then authenticate signature 309 in a step 310 using the cert0.SE 102k'. However, authentication of signature 309 by mobile handset 108 is not required in some exemplary embodiments. For embodiments where mobile handset 108 does not have access to access network 126 at the physical location of device 101, then mobile handset 108 could record the data from message 307 in memory and transmit the data to authentication server 111 at a later time when communication with access network 126 was re-established, such as when a mobile network comes back into range for mobile handset 108.

After a step 310, mobile handset 108 can send authentication server 111 a message 311, where message 311 can include ID.device 101b, random1.SE 308, and signature 309 over at least random1.AS 223. Message 311 could be transmitted over a secure session 214 depicted in Figure 2a above. For embodiments where device 101 sends PK0.SE 102n in a message 307, then mobile handset 108 can also send PK0.SE 102n in a message 311. Authentication server 111 can previously maintain a mapping of ID.device 101b and ID.SE 102m based upon the prior steps in Figure 2a above, such as when sending message 222. In addition, authentication server 111 can record authentication parameters 111d for authentication steps with device 101 and SE 102. In other words, a first set of authentication parameters 111d could be used with a first set of devices 101 and a second set of authentication parameters 111d could be use with a second set of devices 101.

Authentication server 111 could also record cert0.SE 102k. Authentication server could conduct a step 221c depicted in Figure 2b in order to verify signature 309 using the PK0.SE 102n' recorded within cert0.SE 102k or cert0SE 102k'. Step 221c may comprise an authentication step for SE 102, such that an identity for SE 102 is verified with configuration system 114 using a "passed" 229 signature 309, since signature 309 was over the random number or "challenge" issued by authentication server 111 in random1.AS 223. In exemplary embodiments, random1.AS 223 is unique and with a sufficient length in order to reduce the possibility of reuse, such as an exemplary length of 32 or 64 bytes, and other possibilities exist as well.

Authentication server 111 may then utilize random1.SE 308 to create signature3.AS (random1.SE 308) 313. Signature3.AS 303 over random1.SE 308 could utilize a signature creation step 220d as depicted in Figure 2b. Signature3.AS 313 can comprise the signature by authentication server 111 for the random number or challenge issued by SE 102 in order to SE 102 to authenticate configuration system 114. Authentication server 111 can send mobile handset 108 a message 312 with signature3.AS 313, where signature3.AS 313 was created in step 220d. Mobile handset 108 can then send SE 102 a message 314, where message 314 can include the signature3.AS 313 over random1.SE 308. SE 102 can receive the message 314 over the radio 102c and verify the signature3.AS 313 using a step 221d as depicted in Figure 2b. SE 102 could input (i) the value for random1.SE 308 issued by SE 102 above into the "message to verify" in a signature verification 221 step and also input (ii) the public key recorded in certificate cert.AS 111c in order to verify signature3.AS 313. When signature3.AS 313 is verified by SE 102, then authentication server 111 and associated configuration network 114 can be authenticated with SE 102, since authentication server 111 was able to properly sign the random challenge issued by SE 102.

Upon successful verification of signature3.AS 313 in a step 221d, SE 102 can send mobile handset 108 an "OK" message in message 316 or other error code. Although not depicted in Figure 3, SE 102 could optionally send along with the "OK" a second random number random2.SE generated by random number generator 102e in SE 102. Random2.SE could be utilized by configuration system 114. In exemplary embodiments, SE 102 could also send additional configuration data for SE 102 along with the OK, such as an updated set of authentication parameters 111d, information about device 101 that is recorded by SE 102, and other possibilities exist as well for data that SE 102 sends after successfully conducting a step 221d. Mobile handset 108 can also forward the "OK" and optional additional information to authentication server 111.

After the mutual authentication depicted in Figure 3, authentication server 111 can then send a message 317 to configuration server 112 the ID.device 101b, ID.mobile-handset 108e, and random1.SE 308. In an alternative exemplary embodiment, message 317 could include random2.SE instead of random1.SE 308, where random2.SE was described above in this paragraph. Configuration server 112 may then perform a step 318 to record that (i) device 101 with ID.device 101b and (ii) mobile handset 108 with ID.mobile-handset 108e are authenticated for configuration system 114. Message 317 could also optionally include encryption or session keys for previously established secure communications with mobile handset 108, such that the encryption or session keys could be reused from a secure session 214.

As depicted in Figure 3, mobile handset 108 using configuration application 108b can conduct a step 319 to collect an identity list 320, where identity list 320 comprises a list of identities for elements of systems 100, 200, and 300 around or nearby to device 101 and monitored unit 106. Identity list 320 can include a networks available list 322. In order to conduct a step 319, mobile handset 108 may need to temporarily disconnect from an access network 126, such as a mobile phone network, in order to perform a full frequency scan of all available mobile network operators and network access technologies around device 101 in order to collect a networks available list 322. The networks available list 322 can be useful for configuration system 114 to select the best or a preferred access network 126 from the networks available list 322. As depicted in Figure 3, a networks available list 322 can also include the access technology available from a plurality of access networks 126 around device 101, such as 2G, 3G, 4G LTE, 5G, Wi-Fi, NB-IoT, LPWAN, etc. In exemplary embodiments, the networks available list 322 of surrounding wireless networks also includes an associated RF signal strength for each, such as RSSI as well as a frequency band, frequency list in MHz, or RF channel for each surrounding wireless network.

A step 319 may also comprise mobile handset 108 collecting a list of identities for monitored units 106 and transducers 105 for device 101, and include the identities in the identity list 320. In exemplary embodiments, a step 319 in Figure 3 may comprise mobile handset 108 scanning for a bar code or QR code on each of the monitored units 106 and transducers 105 in order to collect an identity for each. Mobile handset 108 could also or alternatively take a picture and record the picture for each monitored unit 106 and transducer 105. The picture for monitored unit 106 and transducer 105 could be processed by mobile handset 108 or configuration system 114 in order to identify monitored unit 106 and transducer 105. The identity of owner 122 for device 101 could be entered by the user in a screen or previously acquired by mobile handset 108, such as with a set of configuration parameters config-provisioning.ID.device 212.

Further, in a step 319, mobile handset 108 can preferably obtain (i) geographical coordinates for a data field location.device 325 or (ii) estimated geographical coordinates for the physical location of device 101. In exemplary embodiments, mobile handset 108 can have a GPS or similar receiver for outdoor location, or use Wi-Fi or Bluetooth for approximate indoor locations, and record the location in location.device 325. Although step 319 to collect an identity list 322 for device 101 is depicted as after receiving message 316 (confirming mutual authentication between SE 102 and authentication server 111), step 319 to collect an identity list 322 could be conducted at other times, such as before Figure 3 or after a configuration user 108a takes mobile handset 108 to the approximate physical location for installation or configuration of device 101.

As depicted in Figure 3, configuration server 112 and mobile handset 108 could then establish a secure session setup 321 between the two nodes. Secure session setup 321 could comprise establishing a secure communications link using protocols such as TLS, datagram transport layer security (DTLS), IPSec, a virtual private network (VPN), a secure shell (SSH), or similar networking, transport, or application layer technologies in order to establish secure communications between configuration server 112 and mobile handset 108. Secure session setup 321 could utilize information received by configuration server 112 in message 317 above, such as existing encryption or session keys or certificates for mobile handset 108 from a session 214 (where authentication server 111 and configuration server 112 in turn have a secure session established of session 301). In exemplary embodiments, secure session setup 321 includes mutual authentication for both sides, and may include the transfer of a certificate from each side to the other side. After secure session setup 321, mobile handset 108 can send configuration server 112 a ciphertext.MH 323, where the plaintext inside ciphertext.MH 323 can include the identity list 320 collected by mobile handset in a step 319 above and an exemplary identity list 320 is

depicted in Figure 3. In this manner, a plurality of identifying information regarding device 101 and its operating environment can be securely transferred to configuration server 112.

Configuration server 112 could conduct a step 324 to decrypt and record the identity list 320 received in ciphertext.MH 323. In exemplary embodiments, configuration server 112 records the data from identity list 320 for device 101 in a configuration database 112b as depicted in Figure 1a. The identity list 320 recorded in a step 324 can be utilized in subsequent steps in order to configure device 101 to use transducers 105 with monitored unit 106. Step 324 may also comprise configuration system 114 notifying device owner 122 that SE 102 has been mutually authenticated and identity list 320 has been received. Step 324 could also comprise configuration system 114 (i) processing networks available list 322 in order to select a preferred access network 126 for device 101, and (ii) querying the selected access network 126 for network access credentials 126a for use with the selected access network 126.

Figure 4a

Figure 4a is a simplified message flow diagram illustrating an exemplary system with exemplary data sent and received by a mobile handset and a device, in accordance with exemplary embodiments. Before initiating steps and message flows depicted in Figure 4a, mobile handset 108, device 101, and the other elements depicted for system 300 in Figure 4a may previously complete exemplary message flows and steps depicted in Figure 3 above. System 300 can include a configuration server 112, mobile handset 108, and device 101. An exemplary difference between system 300 depicted in Figure 3 and system 300 depicted in Figure 4a can be the authentication steps with authentication server 111 have been completed and thus no additional message flows with authentication server 111 are depicted in Figure 4a. In other words, authentication server 111 from system 300 in Figure 3 may also be in system 300 in Figure 4a, but without any additional messages flowing to or from authentication server 111. Mobile handset 108 can communicate with configuration server 112 via IP network 128, using secure session 321 from Figure 3 above. Mobile handset 108 and SE 102 in device 101 can communicate using NFC session 303, which was also depicted and described above in connection with Figure 3 above.

At step 401, SE 102 can record authentication server 111 has successfully completed authentication above in a step 221d, in addition to a step 221c where SE 102 has successfully completed an authentication with authentication server 111. A step 401 could also include SE 102 authenticating mobile handset 108 before proceeding to subsequent steps depicted in Figure 4a and Figure 4b. SE 102 could request that mobile handset 108

successfully sign a random number such as random1.SE 308 for a signature creation step 220, where SE 102 could verify the signature of random1.SE 308 (or another random number from SE 102) using a signature verification step 221. For this step 401 where mobile handset 108 is authenticated with SE 102, then SE 102 could request a certificate for mobile handset 108, where the certificate authority for the certificate of mobile handset 108 can be verified using an existing certificate stored in SE 102, such as cert.CA1 115 or cert.CA.root 109. A step 401 could also comprise SE 102 utilizing Domain Name System Security Extensions (DNSEC) in order to verify domain names associated with configuration system 114. SE 102 could send DNS requests with associated security extensions to mobile handset 108, which could forward the requests to access network 126 to resolve the DNS requests and provide associated security data such as certificates for DNS servers.

At step 401, SE 102 can also wait for additional signatures of random1.SE 308 from other servers besides authentication server 111 in order to authenticate the other servers, such waiting for a signature of random1.SE 308 or random2.SE (with message 316) from a configuration server 112. In addition, SE 102 can record all used random numbers in nonvolatile memory such as SE flash memory 102d or storage memory 102y, in order to verify that random numbers are not reused and thus increase resistance to replay attacks. A device 101 with SE 102 could also store or record a second set of parameters 407', potentially different than a first set of parameters 407, where device 101 could use the second set of parameters 407's to derive PKI key pairs below such as in a step 410.

At step 402, configuration server 112 can query a configuration database 112a for a set of parameters 407 supported by SE 102. In a system 100 and 300 which supports thousands or more devices 101 and also potentially dozens or more different models manufactures 101x for device 101, then potentially different sets of parameters 407 could be utilized or compatible for different devices 101. A first set of parameters 407 could include or be associated with a second set of parameters 407', and parameters 407' can be used with subsequent keys derived or selected by device 101 for future communications after completion of the steps depicted in Figure 4a. For example, parameters 407' could be used by device 101 with an access network 126 or reporting server 116 as depicted in Figure 5 below. An exemplary "large" device 101 such for an automobile may support PKI key parameters for RSA algorithms with associated longer keys and greater processor requirements (such as exemplary PKI keys of length 2048 or 4096 bits), while an exemplary "small" device 101 controlling a lock on a door could support PKI key parameters for ECC algorithms with lower processor requirements (such as an exemplary PKI key length of 256 bits).

Configuration server 112 could utilize ID.device 101b or ID.SE 102m in order to query any of (i) configuration database 112a, (ii) device manufacturer 101x, or (iii) device owner 122 in order to obtain a set of parameters 407 compatible for device 101 and SE 102. ID.device 101b could be obtained from a message 211 depicted in Figure 2a. At step 5 402, configuration server 112 could also utilize ID.device 101b or ID.SE 102m in order to query any of (i) configuration database 112a, (ii) device manufacturer 101x, or (iii) device owner 122 for PK0.SE 102n and PK0.SE 102n'. Note that configuration system 114 using authentication server 111 could receive PK0.SE 102n and PK0.SE 102n' in conjunction with a step 219a in Figure 2a, where authentication server 111 could query another server or node in a system 100 or a database with data received from the tag 101s. 10

In some exemplary embodiments, configuration server 112 reads a set of parameters 102p from a certificate cert0.SE 102k and selects parameters 407 that are compatible or identical to parameters 102b. In other words, if certificate cert0.SE 102k has a public key PK0.SE 102n with an exemplary ECC public key based on a secp256r1 curve with length 256 bits, then in a step 402 configuration server 112 can determine that a set of 15 parameters 407 should specify an ECC key pair for the secp256r1 curve with length 256 bits. Or a step 402 could comprise configuration server selecting a subset of authentication parameters 111d from Figure 2a. Preferred exemplary embodiments in the present invention support the use of parameters 407 that are different from parameters 111d or 20 parameters 102p, since (i) parameters 102p may be selected by a device manufacturer 101x, but (ii) device owner 122 may desire a different set of parameters 407 be used for device 101 after a configuration step 103 (e.g. different ECC curve names or key lengths or certificate expiration dates or different certificate authorities to sign public keys, etc.)

Note that configuration server 112 could select a first set of parameters 407 for use 25 with PK0.SE 102n and/or PK0.SE 102n', and a second set of parameters 407' (such as with different curves or key lengths) could be used with a second public key for device 101 such as PK1.SE 410a. Parameters 407' could also specify cryptographic algorithms that are different than classical ECC algorithms, such as algorithms for lattice-based cryptography, code-based cryptography, or supersingular isogeny based cryptography, and other 30 possibilities exist as well. Example possible algorithms for parameters 407' include those described in "Post-Quantum Cryptography Standardization" from Wikipedia dated March 12, 2019, which is herein incorporated by reference.

Continuing at step 402, configuration server 112 can derive an ephemeral public key PKe.CS 402a and an associated secret key SKe.CS 402b, using a random number, a 35 key pair generation function, and the set of parameters 407, where obtaining the set of parameters 407 were discussed in the paragraph above. In addition, although PKe.CS 402a

and SKe.CS 402b are described above as “ephemeral” keys, the keys could comprise long-term keys. Further, PKe.CS 402a and SKe.CS 402b could be derived by another server associated with a configuration system 114 and transferred to configuration server 112 before a step 402. Configuration server 112 or configuration system 114 can utilize a
5 random number generator similar to hardware random number generator 102e in Figure 1b in order to derive the random number and input the random number into the key pair generation function. A key pair generation function could be included in application software running on configuration server 112 or an associated server of configuration system 114, and exemplary software for key pair generation algorithm can comprise
10 OpenSSL, crypto++, or Mozilla libraries from Network Security Services.

Other possibilities exist as well for libraries or software for elements of a system 100 or system 300 to derive a PKI key pair. Parameters 407 may specify algorithms and values associated with the PKI key pairs and their use, such as the exemplary parameters 407 depicted in Figure 4d below. Parameters 407 and 407' can specify elliptic curve
15 names (e.g. NIST P-256, sect283k1, sect283r1, sect409k1, sect409r1, etc.), key lengths, a duration for key validity, uses of the PKI key pair such as for key derivation, signatures, encoding rules, message digest algorithms, etc. Parameters 407 and 407' may also include identifying information associate with the PKI key pair such as a sequence number, a serial number, a domain name of the server or element deriving or using the PKI key pair. Other
20 possibilities exist as well for data in a set of parameters 407 without departing from the scope of the present invention.

At step 220e in Figure 4a, configuration server 112 can use a signature creation 220 algorithm as depicted in Figure 2b above in order to create a signature signature1.CS(random1.SE 308, PKe.CS 402a) 404. Note that a signature 404 could also
25 be over parameters 407 and/or parameters 407' as well. In other words, configuration server 112 can sign both (i) the random challenge from SE 102, where authentication server 111 also previously signed the same random challenge, and (ii) the public key PKe.CS 402a generated above in step 402. The signature1.CS 404 can utilize a long-term private key for CS 112 corresponding to a public key in a certificate cert.CS 112c. Although not
30 depicted in Figure 4a, for embodiments where SE 102 sends a second random number random2.SE with the “OK” message in message 316, signature1.CS 404 could be over random2.SE instead of over random1.SE 308.

For embodiments where authentication server 111 and configuration server 112 are combined into the same server or “cloud”-based service or network such as a configuration
35 network 113 in Figure 1a, then step 220e with the associated signature1.CS 404 could be omitted. A step 220e in Figure 4a could comprise configuration server 112 also including

parameters 407 and/or parameters 407' in the "message to sign" for signature1.CS 404. For embodiments where authentication server 111 and configuration server 112 are combined, the IP address utilized by configuration system 114 for message 312 above and message 403 herein could be the same IP address. The use of separate servers for
5 configuration server 112 and authentication server 111 can increase security and scalability for large networks with tens of thousands or more devices 101 for some embodiments.

Configuration server 112 can then send mobile handset 108 a message 403, where message 403 can include or comprise the signature 1.CS (Random1.SE, PKe.CS) 404, a certificate cert.CS 112c, a certificate authority certificate cert.CA1 115, PKe.CS 402a,
10 parameters 407, parameters 407' and ID.Transaction1 405. Parameters 407' in message 403 could be omitted if device 101 records parameters 407' before a message 403, such as described above with a step 401. In other words, either a configuration system 114 or a device 101 can specify the parameters 407' which could be used in a subsequent step 410 with a SK1.SE and PK1.SE. Although not depicted in Figure 4a, message 403 could also
15 include additional parent certificates for cert.CA1 115, such that SE 102 could verify the certificates with a certificate stored in SE 102 such as a root certificate 109.

In exemplary embodiments, configuration database 112a includes the certificates recorded by SE 102 such that configuration server 112 can determine if any parent certificates for cert.CA1 115 are required in order to be verified up through a root
20 certificate recorded in SE 102. In exemplary embodiments, configuration system 114 can (A) query device manufacturer 101x or device owner 122 in system 100 using ID.SE 102m or ID.device 101b for a list of certificates recorded by SE 102 in order to (B) include certificates in message 403 where the public key for a certificate authority matches the root public key for a root certificate 109 installed and recorded by SE 102. ID.Transaction1 405
25 can comprise a preferably unique transaction identity or session identity, such that configuration server 112, mobile handset 108, and SE 102 can track the flows of data and messages between the elements.

At step 408, mobile handset 108 can receive the message 403 using configuration application 108b. Note that message 403 may be transmitted between mobile handset 108
30 and configuration server 112 using secure session 321 and the data in message 403 may be transmitted as a ciphertext. Mobile handset 108 can optionally verify the signature signature1.CS 404 using a signature verification 221 depicted in Figure 2b, although verification by mobile handset 108 may not be required because mobile handset 108 can utilize secure session 321 for communication with configuration server 112. After step
35 408, mobile handset 108 can transmit message 403 with the data described in the paragraph above to SE 102 using NFC session 303, as depicted in Figure 4a. In exemplary

embodiments, the data within message 403 (such as the data depicted in message 403 from configuration server 112 to mobile handset 102) can be sent over the NFC session 303 link, where the message 403 over the NFC session 303 can be formatted according to the data-link layer of NFC session 303. Further a subset or superset of the data for message 403 received by mobile handset 108 could be in message 403 transmitted by mobile handset 108.

At step 409a SE 102 can receive message 403 and record values from message 403 into both volatile memory RAM 102b and storage 102d. Exemplary data recorded by SE 102 in step 409a includes recording in memory by SE 102 both certificates cert.CS 112 and cert.CA1 115. SE 102 can verify certificate cert.CS 112c using certificate cert.CA1 115, and then either (i) verifying cert.CA1 115 with a certificate already recorded in SE 102 before step 409a, such as cert.CA.root 109 (assuming CA1 has its certificate signed by the root certificate authority shown in Figure 1a), or (ii) requesting mobile handset 108 to fetch a parent certificate for cert.CA1 115 that would link cert.CA1 115 to a certificate already recorded and authenticated within SE 102. SE 102 could use a signature verification 221 step in step 409a to verify certificates. In exemplary embodiments, SE 102 records a plurality of “previously approved” root certificates 109 (in Figure 1b) from different certificate authorities, and step 409a could comprise SE 102 selecting a parent root certificate that matches cert.CA1 115. In exemplary embodiments, the plurality of “previously approved” root certificates 109 for SE 102 could be recorded by a device manufacturer 101x during manufacturing of device 101 or before distribution of device 101 to device user 101a. In exemplary embodiments, SE 102 can also query mobile handset 108 for authenticated certificate revocation data for any certificate received in a message 403. SE 102 can proceed with subsequent steps after SE 102 can confirm that certificates received in a message 403 have not been revoked.

In exemplary embodiments, if SE 102 does not record a parent root certificate 109 (or grandparent, or great-grandparent, etc.) for cert.CA1 115 or cert.CS 112, then in step 409a SE 102 can (i) send an error message to mobile handset 108 and (ii) optionally stop the configuration step 103. The error condition to mobile handset 108 could notify configuration user 108a that procedures may be required in order to load a new or different root certificate 109 or parent certificate in order to proceed. The root certificate 109 can be the foundation of security for SE 102. If hackers or adversaries could load a fake root certificate 109, or have SE 102 accept a certificate authority certificate that is not chained to an authenticated root certificate previously installed in SE 102, then potentially any non “genuinely” authenticated public key could be subsequently loaded, and SE 102 could then be considered no longer secure.

At step 409b, SE 102 can process parameters 407 and parameters 407'. SE 102 can verify that parameters 407 and 407' are compatible with the set of cryptographic algorithms 141. For example, if parameters 407 or 407' specify the use of an elliptic curve that is not supported by SE 102, then SE 102 could send mobile handset 108 and error of the unsupported parameters. In step 409b, SE 102 could also select a subset of parameters 407 that are supported by cryptographic algorithms 141 and send mobile handset 108 and configuration server 112 a report of the selected subset. In exemplary embodiments, where parameters 407 and 407' are compatible with cryptographic algorithms 141 and/or certificate parameters 102p then SE 102 accepts parameters 407 and 407' and records them in nonvolatile memory.

At step 221e, SE 102 can verify the signature1.CS 404 using a step 221e as depicted in Figure 2b. SE 102 could input (i) the value for random1.SE 308 issued by SE 102 above message 307 from Figure 3 into the "message to verify" in a signature verification 221 step and also input (ii) the public key ephemeral PKe.CS 402a received in message 403 into in order to verify signature1.CS 404. In this manner, SE 102 can confirm that PKe.CS 402a is authentically from CS 112. Step 221e in Figure 4a can comprise SE 102 using a public key for configuration server 112 from cert.CS 112c. As mentioned above with message 316, if SE 102 issues a second, different random2.SE with message 316 or related messages, then signature1.CS 404 could be over the different random2.SE instead of random1.SE 308 as depicted in Figure 4a. Although not depicted in Figure 4a, in exemplary embodiments signature1.CS 404 verified in a step 221e can also be over parameters 407 and/or 407', and thus SE 102 can know that parameters 407 and/or 407' are authenticated (since they are signed along with random1.SE 308).

At step 410, SE 102 can derive a public key PK1.SE 410a and an associated secret key SK1.SE 410b, using a random number, a key pair generation function, and data from the set of parameters 407'. The parameters 407' could be received in message 403, or also previously stored by device 101 and SE 102 before the receipt of a message 403, as noted above. For some embodiments, PK1.SE 410a could use parameters 407 instead of parameters 407', and the selection of parameters 407 versus 407' could be based on the receipt of parameters 407' in a message 403 or recording of parameters 407' by SE 102 before receipt of a message 403. SE 102 can utilize a random number generator 102e in Figure 1b in order to derive the random number and input the random number into the key pair generation function. A key pair generation function could (i) be included in cryptographic algorithms 141, and (ii) utilize data from parameters 407. SE 102 can then record both PK1.SE 410a and SK1.SE 410b in a nonvolatile memory, such as memory 102d. In exemplary embodiments, SK1.SE 410b is not transferred outside SE 102

(including SK1.SE 410b not being transferred to device 101 outside SE 102) and thus SK1.SE 410b can remain reasonably secure.

SE 102 can conduct a key exchange 411a step with configuration server 112 using PKe.CS 402a and SK0.SE 102y. Details for key exchange 411a are depicted and described in connection with Figure 4b below. As depicted in Figure 4b below, the output of a key exchange 411a step can be a mutually derived symmetric encryption key 442a and MAC key 442b. SE 102 can then perform a signature creation step 220g in order to create signature2.SE 416, where the “message to sign” data in a step 220g includes PK1.SE 410a, and the private key utilized by SE 102 to create signature2.SE can be SK0.SE 102y’. In this manner, (i) the derived public key for SE 102 can be signed by the previously recorded secret key (SK0.SE 102y’), and (ii) configuration server 112 can have access to corresponding PK0.SE 102n’ in order to verify signature2.SE 416. In preferred exemplary embodiments, the parameters 111d used with a signature creation 220g step are equal or equivalent to parameters 102p recorded in cert0.SE 102k’. Note that PK1.SE 410a that is signed in a step 220g may be associated with a different set of parameters 407 than the parameters 102p, such as PK1.SE 410a using the second set of parameters 407’. For a signature creation step 220g, SE 102 could also include a random number received from configuration system 114, such as random1.AS 223, such that signature2.SE 416 can be over at least PK1.SE 410a and random1.AS 223, thereby ensuring a signature 416 is not replayed.

SE 102 can then create a ciphertext1.SE 415 using an encryption step 412a, where encryption step 412a is depicted and described in connection with Figure 4b below. SE 102 can use the derived symmetric encryption key 442a and MAC key 442b (from key exchange 411a in Figure 4b below) with encryption step 412a. The encrypted data in ciphertext1.SE 415 can include the derived public key PK1.SE 410a and also optionally configuration data for SE 102 such as data config0.SE 102aa. Data for config0.SE 102aa was described above in connection with Figure 1b, and in exemplary embodiments, config0.SE 102aa includes a list of all root certificates recorded by SE 102. Including both (i) ciphertext1.SE 415 of PK1.SE 410a and (ii) signature2.SE 416 may appear redundant to those of ordinary skill in the art, since a subsequent decryption of ciphertext1.SE 415 by configuration server 112 would normally mean that only SE 102 sent the ciphertext1.SE 415 (encrypted using SK0.SE 102y). But, creating a separate signature2.SE 416 for PK1.SE 410a can allow configuration server 112 to send the signature2.SE 416 to other servers, such as a certificate authority 115 as depicted in Figure 4a. Additional details for using signature2.SE 416 are described below in connection with Figure 5.

SE 102 can then send mobile handset 108 a message 413, where message 413 can include ID.Transaction1 405, ciphertext1.SE 415, and signature2.SE 416. As depicted in Figure 4a, ciphertext1.SE 415 comprise plaintext values or data for PK1.SE 410a and config0.SE 102aa, where the plaintext values are encrypted with symmetric encryption key 442a. Signature2.SE 416 can be over derived public key PK1.SE 410a. ID.transaction1 405 can correspond to the transaction ID received in a message 403 and could comprise a session ID or token for SE 102 and configuration server 112 to track the data flow. Although not depicted in Figure 4a, signature2.SE 416 can also be over the transaction ID 405 in addition to PK1.SE 410a. As depicted in Figure 4a, message 413 can be sent over NFC session 303 to mobile handset 108. Mobile handset 108 can then forward the data from message 413 to configuration server 112 using secure session 321. For embodiments where device 101 and/or SE 102 stores parameters 407' before the receipt of a message 403 and uses the parameters 407' to generate SK1.SE 410b and PK1.SE 410b, then message 413 can also include the parameters 407'.

Before or after receipt of message 413, configuration server can conduct a key exchange 411b step, which is depicted and described in connection with Figure 4b below. A key exchange 411b step can allow configuration server 112 to mutually derive a same symmetric encryption key 442a that was derived by SE 102 in a key exchange step 411a. For step 411b, configuration server 112 can use SKe.CS 402b and PK0.SE 102n in order to derive symmetric encryption key 442a. Figure 4a depicted step 411a before receipt of message 413 but step 411a could be conducted after receipt of message 411a. After receipt of message 413, configuration server 112 can then perform a decryption 412b step of received ciphertext1.SE 415 using the derived symmetric encryption key 442a, and the operation of a decryption 412b step is depicted and described in connection with Figure 4b below. After step 412b, configuration server 112 can read the plaintext values of PK1.SE 410a and config0.SE 102aa. Configuration server 112 can then also optionally perform a signature verification step 221g, where signature2.SE 416 over at least PK1.SE 402a can be verified. Note that configuration server 112 can optionally omit signature verification step 221g since successful decryption of ciphertext1.SE 415 could only normally be conducted with an authentic SE 102 (e.g. the entity holding SK0.SE 102y). However, other elements in a system 100 and other systems herein can later use signature2.SE 416 to authenticate that PK1.SE 410a is truly from SE 102.

At step 417, configuration server 112 can send a certificate authority 115 for configuration system 114 a message with ID.SE 102m, PK1.SE 410a and signature2.SE (PK1.SE 410a) 416. Although not depicted in Figure 4a, the message to certificate authority 115 could be over a secure session similar to secure session 321. Although not

depicted in Figure 4a, the message to certificate authority 115 could also include (i) data for (a) parameters 407 or 407' for use with PK1.SE 410a and (b) signature2.SE 416, and (ii) a certificate cert0.SE 102k or cert0.SE 102k'. Certificate authority 115 could then conduct a signature verification step 221g for signature2.SE 416 using PK0.SE 102n from
5 cert0.SE 102k (or a PK0.SE 102n' from a cert0.SE 102k' for digital signatures), such that PK1.SE 410a is authenticated for being from SE 102 using ID.SE 102m (since ID.SE 102m is preferably in certificate cert0.SE 102k'). Once signature2.SE 416 over PK1.SE 410a is verified, certificate authority 115 can conduct a step 418 to create certificate cert1.SE 419, where cert1.SE 419 can be signed by certificate authority 115. Note that for embodiments
10 where signature 416 is also over a random number from configuration system 114, such as random1.AS 223, then (i) the message in a step 417 can include the random number, such that CA 115 can verify signature2.SE 416 is over both PK1.SE 410a and the random number from configuration system 114.

Step 418 can use parameters 407' in order to create certificate cert1.SE 419, and in
15 exemplary embodiments parameters 407' are different than parameter 102q, including specifying (i) different key lengths in bits, (ii) different named ECC curves, and (iii) different validity dates. An exemplary certificate cert1.SE 419 for SE 102 using derived PK1.SE 410a is depicted in Figure 4d below. Certificate authority 115 can send cert1.SE 419 to configuration server 112. Cert1.SE 419 for SE 102 can facilitate SE 102 to begin
20 using PK1.SE 410a and SK1.SE 410b in an authenticated manner with other elements of a system 100, including reporting system 120 or access network 126. The use of cert1.SE 419 can provide additional security for a device owner 122 or device user 101a, since (i) SK1.SE 410 has been derived after receipt of device 101 from device manufacturer 101x, and also with (ii) the security for device 101 no longer depending on SK0.SE 102y and
25 SK0.SE 102y', which have physically been in control of other parties besides device owner 122 or device user 101a before delivery of device 101.

At step 420, configuration server 112 can receive cert1.SE 419 and conduct a signature verification step 221 over the signature created by step 220 depicted in Figure 4d. Configuration server 112 can also query configuration database 112a for parent certificates
30 for cert1.SE 419 and cert.CA1 115 that would link to a root certificate recorded in SE 102. In other words, configuration server 112 in a step 420 finds and records intermediate certificate authority certificates that could be verified by a root certificate listed in config0.SE 102aa. As mentioned above, config0.SE 102aa received by configuration server 112 can contain a list of all root certificates recorded by SE 102. Further, at step
35 420, configuration server 112 can query for intermediate certificates that link cert.CA2 121 to a root certificate listed in config0.SE 102aa, such that SE 102 can verify a certificate

from reporting system 120 with a root certificate already installed in SE 102, such as at least one root certificate 109 burned into EEPROM 102j during manufacturing.

For embodiments where PK1.SE 401a use a different set of cryptographic parameters 407' than parameters 407, then configuration server 112 can conduct a second key derivation step 402' with the parameters 407' in order to generate PKe.CS 402a' and SKe.CS 402b'. At a second key exchange 421a step, configuration server 112 could conduct second a key exchange where (i) the derived, authenticated, and previously signed PK1.SE 410a is used for the key exchange instead of the original PK0.SE 102n, and (ii) the SKe.CS 402b' from a step 402'. Details for the second key exchange 421a step are depicted and described in connection with Figure 4c below. For embodiments where PK1.SE 410a use parameters 407, then configuration server 112 could continue to utilize the ephemeral SKe.CS 402b in a key exchange step 421a. The output of a key exchange step 421a can be a new, second symmetric encryption key 442b and also a new MAC key 442b. In other words, the second key exchange step 421a can be utilized so that configuration server 112 can derive the new symmetric encryption key 442b using PK1.SE 410a and SKe.CS 402b'.

The present invention as depicted in Figure 4a provides an embodiment for security that may also be generally applicable for a server communicating with a node, even outside the use of a configuration 103 step. Specifically, the configuration server has (i) derived an ephemeral key PKI key pair (e.g. PKe.CS 402a and SKe.CS 402b), where the ephemeral key is signed with a static server private key, (ii) conducted a first key exchange 411b with the derived SKe.CS 402b for a first symmetric key 442a, (iii) received a derived PK1.SE 410a from SE 102 where PK1.SE 410a can be either (iii.a) encrypted with the first symmetric key 442a, or (iii.b) verified with PK0.SE 102n', or (iii.c) both iii.a and iii.b, (iv) conducted a second key exchange 421b using the second ephemeral SKe.CS 402b' and the received, derived PK1.SE 410a in order to obtains a second symmetric key 442b, and then (v) encrypting ciphertext with the second symmetric key 442b. In other words, the steps the configuration server 112 herein has taken can be applicable generally to a server communicating with a node and using both static and ephemeral PKI keys for the server to derive a first symmetric encryption key, and then securely receiving a derived public key from the node and using that derived public key with the ephemeral keys to derive a second symmetric encryption key.

At step 422, configuration server 112 with other elements in configuration system 114 can select or obtain network access credentials 126a. Configuration server 112 could query configuration database 112a using the networks available list 322 depicted and described in connection with Figure 3 above, where the networks available list 322 provide

information about wireless networks around device 101. A response to a query to configuration database 112a could provide a selected access network 126 that is preferred based on criteria such as bandwidth cost, expected energy or power requirements (i.e. 3G/4G may use more power than LPWAN), signal strength measurements for networks available list 322, and commercial terms such as agreements between device owner 122 and networks in a networks available list 322. Configuration server 112 could also query other servers in a configuration system 114 or a system 100 in order to obtain a selected access network 126 from networks available list 322 in addition to network access credentials 126a for access network 126.

In an exemplary embodiment, network access credentials 126a received by configuration server 112 or configuration system 114 are encrypted using either PK0.SE 102n or PK1.SE 410a before being received by configuration server 112. A second symmetric encryption key 442 (with a different value than the first symmetric encryption key 442a below in Figure 4b) could be asymmetrically encrypted using PK0.SE 102n or PK1.SE 410a, and the network access credentials 126a could be encrypted with the second symmetric encryption key 442a by an access network 126. In exemplary embodiments, the second symmetric encryption key 442 can be asymmetrically encrypted by a configuration system using PK1.SE 410a and Elgamal asymmetric encryption. Other possibilities exist as well such as another element besides configuration server 112 deriving a second symmetric encryption key 442 using PK0.SE 102n or PK1.SE 410a and then encrypting the network access credentials 126a using the derived second symmetric encryption key 442. In that manner, the encrypted network access credentials 126a can be encrypted uniquely for SE 102 and cannot be read by configuration server 112 or mobile handset 108. In some exemplary embodiments, the network access credentials 126a can include an identity such as an international mobile subscriber identity (IMSI) and a shared secret key "K", and each of the IMSI and the key K can be asymmetrically encrypted with PK1.SE 401a, where device 101 with SE 102 could asymmetrically decrypt the IMSI and key K with SK1.SE 401b.

At step 423, configuration server 112 can collect a software package 104q for device 101 and SE 102. An exemplary software package 104q for device 101 is depicted and described in Figure 5 below. Configuration server 112 could use data from an identity list 320 to query other servers from a system 100 in order to obtain the software. Different components of software package 104q could come from different servers or elements in a system 100. Configuration server 112 can also use data from configuration file config0.SE 102aa in order to select software for software package 104q. The data collected by or queried by configuration server 112 could be stored in a configuration database 112a. An

updated device 101 operating system 101e could be obtained from a device manufacturer 101x. Configuration data config.device 101r could be obtained from servers associated with either device owner 122 or device manufacturer 108x.

At step 423 configuration server 112 can also use ID.transducer from identity list 320 to query a configuration database 112a or configuration database 112b for in order to obtain a configuration file or dataset for device 101 to use with transducer 105, and the dataset could comprise config.TR1 105e. Although not depicted in Figure 1a a system 100 could include a transducer manufacturer similar to device manufacturer 101x, and a configuration server 101 could use ID.transducer to query the transducer manufacturer for data to be included in config.TR1.105e. Data for config.TR 105e could comprise a transducer electronic datasheet (TEDS). At step 423 configuration server 112 can use ID.MU to query a configuration database 112a or configuration database 112b for in order to obtain a configuration file or dataset for device 101 to use with monitored unit 106, and the dataset could comprise config.MU1 106a. The dataset for config.MU1 106a can specify parameters associated with monitored unit 106, such as timer values for device 101 to collect data, a range of maximum or minimum values an actuator for transducer 105 could use with monitored unit 106. In exemplary embodiments, the dataset for config.MU1 contains thresholds for alarm values or conditions, such that if device 101 detects signals outside acceptable levels and within an alarm range in config.MU1, then device 101 immediately reports the alarm condition to reporting server 116 and/or takes other actions as well, including potentially adjusting a transducer output 105c to mitigate the alarm condition. Other possibilities exist as well for data within config.MU 106a without departing from the scope of the present invention.

Further, at step 423 configuration system 112 could request device owner 122 to specify the reporting system 120 for device 101, and subsequently (A) configuration server 112 could send data from identity list 320 or config0.SE 102aa to reporting system 120 and (B) reporting system 120 could send back a dataset to configure device 101 to work with reporting system 120, and the dataset could comprise config.reporting-system 120a. Data within config.reporting-system 120a could specify domain names or IP addresses of reporting server 116, credentials for device 101 to utilize with reporting server 120 such as a user name, password, encryption keys, or a public key, reporting timers, etc. In some exemplary embodiments, network access credentials 126a could be received by configuration server 112 in a step 423 from reporting system 120. In other exemplary embodiments where PKI-based authentication is used for connectivity of device 101 to access network 126, then a step 423 could also comprise configuration server 112 sending access network 126 the certificate cert1.SE 419. Data within config.reporting-system 120a

could also specify cryptographic parameters similar to parameters 407 such as algorithms to utilize, key lengths, curve names, padding schemes, encoding rules, versions of standards to implement (such as TLV v2 versus TLS v3). Configuration server 112 could also query device owner 122 in order to obtain some data for config.reporting-system 120a.

5 In addition, configuration server 112 could query configuration user 108a at mobile handset 108 for additional data or user input if the software package 104q could not be automatically assembled or fully created. Note that software package 104q could include sub-packages, such as a sub-package 104z, where sub-package 104z could depend or be determined based on transducers 105 attached to device 101, or monitored unit 106, or
10 reporting system 120. In an exemplary embodiment, identity list 320 could specify a transducer identity ID.transducer that requires a specific transducer library 105q (or driver) for device 101 with OS 101e to support a selected transducer 105. Configuration server 112 could query a transducer manufacturer for the most current or appropriate transducer library 105q for device 101, and include that transducer library 105q in a software package
15 104q.

In exemplary embodiments a software package 104q can include a set of configuration test vectors 104x, which can comprise software for verifying the functionality of transducers 105 with transducer bus 101i and transducer library 105q. A configuration test vector 104x could provide an instruction to verify signal strength for data
20 received, noise levels for data received, or dynamic range for signals output, etc. Other possibilities exist as well for a configuration server 112 or configuration system 114 to assemble software for device 101 in a software package 104q, without departing from the scope of the present invention. After software package 104q is assembled, configuration server can calculate the size of software package 104q, which could comprise the value
25 package-size.CS 428, where package-size.CS represents the size of the software package 104q in bytes.

At step 424a, configuration server 112 can encrypt data for SE 102 using the derived symmetric encryption key 442b from a previous step 421a. An encryption step 424a is depicted and described in connection with Figure 4c below, and the encryption step
30 424a can create a ciphertext1.CS 427. At step 424a, plaintext input into a symmetric encryption algorithm 444a could include at least one of certificates cert1.SE 419, cert.CA2 121, cert.CA3 123, device 101 configuration data config.TR1 105e, config.MU1 106a, config.device 101r, and config.reporting-system 120a. In other words, plaintext input could include one or more of certificates cert1.SE 419, cert.CA2 121, cert.CA3 123,
35 device 101 configuration data config.TR1 105e, config.MU1 106a, config.device 101r, and config.reporting-system 120a. The data for plaintext in a step 424 can comprise a device

configuration. In addition, plaintext input into a symmetric encryption algorithm 444a at a step 424a could include identity list 320 and network access credentials 126a (or an encrypted network access credentials 126a, where the encrypted network access credentials were encrypted using a public key for SE 102). A step 424a could also include
5 configuration server 112 using plaintext of additional parent certificates for cert.CA1 121 and/or cert CA3 123 that would be signed by an existing cert.CA.root 109 recorded by device 101. The parent certificates would be determined in the prior step 420 as described above.

Configuration server 112 can then send message 425 to mobile handset 108, where
10 message 425 can contain an ID.transaction2 426, PKe.CS 402a', ciphertext1.CS 427, where ciphertext1.CS 427 can include the ciphered data described in the above paragraph and depicted in both Figure 4a and Figure 4c. Other or additional data could be included in a ciphertext1.CS 427 as well, such as the ID.transaction2 426, and identity of configuration server 112 or SE 102 or device 101, or other values as well. Message 427 can also include
15 the value package-size.CS 428, and package-size.CS 428 may also optionally be in ciphertext1.CS 427. In exemplary embodiments, package-size.CS 428 is preferably outside ciphertext1.CS 427 in order for mobile handset 108 to read the value and make decisions or prompt configuration user 108a based on the value of package-size.CS 428. If package-size.CS 428 was only inside ciphertext1.CS 427, then mobile handset 108 would not
20 normally be able to read the plaintext data since mobile handset 108 does not hold or record a symmetric encryption key for ciphertext1.CS 427. Upon receiving message 425, mobile handset 108 can forward data in message 425 to SE 102, as depicted in Figure 4a.

At step 429, mobile handset 108 can process package-size.CS 428 in order to make a decision about subsequent data transfer between mobile handset 108 and device 101 through NFC session 303. Although depicted in Figure 4a as occurring after mobile handset 108 sending message 425, a step 429 could occur before the mobile handset 108 sends message 425 to SE 102. Configuration application 108b can read package-size.CS 428 to determine if the upcoming software package size 104a in bytes should be transmitted to SE 102 via either (A) the existing NFC session 303 or alternatively (B) via a
30 new, different Wi-Fi or Bluetooth session between mobile handset 108 and device 101. For example, if package-size.CS 428 is less than an exemplary one megabyte, then a configuration user 108a for mobile handset 108 could reasonably wait for the future software package 104q to be transmitted over NFC session 303 (such as less than an exemplary 30 seconds). However, if package-size.CS 428 is relatively large, such as
35 greater than an exemplary ~15 megabytes, then a configuration user 108a may prefer to setup a Wi-Fi or Bluetooth session with device 101 in order to transfer the upcoming

software package 104q. A step 429 could also comprise a display on mobile handset 108 prompting configuration user 108a for a decision or threshold for determining the use of Wi-Fi or Bluetooth (instead of using NFC session 303) in order to transfer upcoming software package 104q based on package-size.CS 428.

5 In exemplary embodiments, a step 429 can take place before mobile handset 108 sends message 425, and message 425 to SE 102 can include a signal or command regarding the use of Wi-Fi or Bluetooth to download software package 104q. Note that SE 102 could separately make a decision regarding the use of Wi-Fi or Bluetooth to download software package 104q based on package-size.CS 428, where SE 102 can consider factors that may
10 not be available to configuration application 108b, such as the Wi-Fi and Bluetooth capabilities of device 101, internal security policies of device 101 and SE 102, etc. Figure 4a depicts an embodiment where SE 102 makes the decision regarding physical layer to utilize for transfer of upcoming software package 104q using a step 431 below. However, the present invention also contemplates that mobile handset 108 can make the decision
15 regarding physical layer to utilize for upcoming transfer of software package 104q as well for other embodiments. Further, configuration server 112 could make the decision based on parameters in a configuration database 112a, such as a maximum time allowed for transfer of software package 104q through NFC session 303, such as an exemplary maximum time of 120 seconds before an alternative physical layer to NFC would be
20 attempted such as Wi-Fi or Bluetooth.

After receipt of message 425, at step 421b, SE 102 can derive symmetric encryption key 442b as depicted and described in connection with Figure 4c below. Note that SE 102 utilizes PKe.CS 402a' and SK1.SE 410b as input into the key exchange algorithm 447. In other words, SE 102 can utilize the secret key SK1.SE 410b derived in
25 step 410 with PKe.CS 402a' received in message 425 above in order to obtain symmetric encryption key 442b. In this manner, SE 102 and device 101 can optionally end dependence on a SK0.SE 102y that likely had been physically been outside the control of device owner 122 and device user 101a. In exemplary embodiments, although SK1.SE 410b provides an updated secret key, SE 102 continues to record SK0.SE 102y with
30 corresponding cert0.SE 102k, such that use of SK0.SE 102y could be enabled for a "rollback" or "fallback" scenario such as if nonvolatile memory 102d storing SK1.SE 410b gets corrupted or erased. For embodiments where parameters 407 are used with SK1.SE 401b, then a step 421b can be conducted with PKe.CS 402a from a message 403 and the use of a PKe.CS 402a' from a message 425 could be omitted.

35 SE 102 can then use a decryption 424b step with the derived symmetric encryption key 442b in order to decrypt the ciphertext1.CS 427 and read the plaintext inside

ciphertext1.CS 427. Although not depicted in Figure 4a but depicted in Figure 4c, a message 425 with ciphertext1.CS 427 can also include an initialization vector 445b. As contemplated for all messages depicted in the present invention, a message with a ciphertext can also include in initialization vector 445 and a message authentication code
5 446. At step 430 SE 102 can record the decrypted plaintext data from message 425 in nonvolatile memory 102d, including the new cert1.SE 419, which can comprise the new, primary certificate for SE 102 and cert0.SE 102k can be considered deprecated at step 430. In exemplary embodiments the new cert1.SE 419 received by SE 102 can utilize a different set of parameters 407 than cert0.SE 102k. For example, if (a) a device manufacturer 101x
10 may use a different ECC named curve than configuration system 112, then (b.1) a new cert1.SE 419 would normally be required for file transfers with configuration system 112, and (b.2) different values for parameters 407 can be utilized in cert1.SE 419 than cert0.SE 102k. The selection of a set of parameters 407 by a configuration server 112 were depicted and described in connection with step 402 above.

15 Also at step 430, SE 102 can verify certificates received in message 425, as well as check a certificate revocation list for both cert1.SE 102 and any parent certificates. In exemplary embodiments, (X) all parent certificates through a root certificate are included in message 425, for all certificates received such as cert1.SE 419, cert.CA2 121, cert.CA3 123 in order for (Y) the public key in a root certificate 109 recorded in device 101 or SE 102
20 before configuration step 103 can be used to verify all the certificates received in message 427. Further, in exemplary embodiments contemplated herein, each time a node or component is described as checking verifying a certificate, the node can also check with a certificate revocation list (CRL) recorded in the certificate to verify that the certificate remains valid before an expiration date. A step 430 can also comprise SE 102 sending a
25 request through NFC session 303 to configuration server 112 to check CRL of certificates received in a message 425. Or, in other exemplary embodiments, a message 425 can include a confirmation that CRL for certificates in message 425 have been checked, and in step 430 SE 102 can also verify that message 425 includes the confirmation that CRL for certificates have been checked by configuration system 114. In another embodiment, any
30 certificate received in a ciphertext1.CS 427 is accepted by SE 102 since SE 102 has authenticated configuration server 112 using random1.SE 308 above in Figure 3.

In exemplary embodiments for a step 430 depicted in Figure 4a, (A) message 425 includes a certificate received by SE 102 that cannot be verified by SE 102 using an existing cert.CA0.root 109 inside SE 102 (including possibly due to unsupported
35 parameters 407), but (B) SE 102 accepts the certificate since it was received from an authenticated server (e.g. configuration server 112). In another exemplary embodiment, SE

102 accepts a second, new root certificate cert.CA1.root and/or cert.CA2.root in a ciphertext1.CS 427 since the message 425 and ciphertext is from configuration server 112, which has been authenticated. The inclusion of a second, new root certificate cert.CA1.root and/or cert.CA2.root in a message 425 can be secure for device 101 and SE
5 102 since the certificates are within a ciphertext1.CS 427 and thus cannot feasibly be altered or tampered with by mobile handset 108 or other nodes. Note that Figure 4a and related figures depict mobile handset transmitting ciphertext across a local wireless link such as NFC session 303.

Continuing at step 430, SE 102 can continue processing plaintext data from
10 message 425 while loading the data into memory and also verifying the data. In exemplary embodiments, SE 102 applies network access credentials 126a received in message 427, as well as configuration parameters received, such as config.device 101r. Note that device 101 with radio 101z may utilize network access credentials 126a, and for those embodiments then SE 102 can pass network access credentials 126a and/or config.device
15 101r to device 101 via bus controller 102h and device data bus 101g. Device 101 could store network access credentials 126a and config.device 101r in nonvolatile memory storage 101f. In exemplary embodiments, step 430 also comprises device 101 rebooting in order to apply config.device 101r and network access credentials 126a. In this embodiment, NFC session 303 may temporarily close but reopen upon successful reboot of
20 device 101 and SE 102.

In exemplary embodiments, network access credentials 126a received in a message 425 and includes at least one additional, backup set of network access credentials 126a, such that device 101 can connect with a second access network 126 if the first access network 126 becomes unavailable. The second or backup set of network access credentials
25 126a could be (i) from configuration server 112 and (ii) selected by configuration server 112 using the networks available list 322 in Figure 3. Step 430 may also comprise device 101 using config.reporting-system 120a in order to communicate with reporting system 120 while activating and accessing an access network 126 using the received network access credentials 126a. In this manner, device 101 can perform an “end-to-end” test to verify the
30 supplied configuration values in a message 425 will provide network connectivity to enable the flow of transducer data 125 from device 101 to reporting server 116 as depicted in Figure 1a. For embodiments where device 101 includes a WAN radio such as a 4G, 5G, 6G radio, etc., then network access credentials 126a could be used by device 101 to connect with an access network 126. Device 101 could receive subsequent configuration
35 data such as that depicted in Figure 5 from access network 126.

At step 431, SE 102 can use package-size.CS 428 to determine if NFC session 303 or a separate physical layer should be used for downloading the upcoming software package 104q. An exemplary decision process for SE 102 using package-size.CS 428 in a step 431 was also described above in connection with step 429. Separate wireless physical
5 layers that SE 102 or device 101 could utilize for downloading software package 104q could be any of Wi-Fi, Bluetooth, a land mobile network such as 4G, 5G, etc., or a LPWAN, and other possibilities exist as well without departing from the scope of the invention. The selection of a physical layer by SE 102 for transfer of a software package 104q based on package-size.CS 428 could comprise a networking command 431a. The
10 exemplary embodiment depicted in Figure 4a illustrates the use of Wi-Fi with mobile handset 108 for download of software package 104q and an associated networking command 431a for use of Wi-Fi with mobile handset 108, but other possibilities exist as well.

Device 101 with SE 102 could connect with a Wi-Fi network from an access point
15 in the vicinity of installed device 101, such as the exemplary Wi-Fi networks detected and depicted for a networks available list 322 in Figure 3 (where network access credentials 126a could be for the exemplary Wi-Fi network in the list 322). Alternatively, device 101 with SE 102 could use a set of network credentials 126a in message 425 to connect with a land mobile network as access network 126 and subsequently download software package
20 104q via the land mobile network, and in this case the networking command from device 101 and SE 102 could specify use of the land mobile network for software package 104q. Figure 4a depicts a decision in a step 431 where SE 102 selects activation of a Wi-Fi radio in device 101, such as radio 101z, in order to download software package 104q. As mentioned above, any of configuration server 112, mobile handset 108, configuration user
25 108a, or SE 102 could make the selection for the physical layer to utilize for software package 104q, or they could make the selection in conjunction.

At step 432, SE 102 can generate a configuration report 432a for configuration system 112 regarding the installation and application of certificates, configuration files, and network parameters and credentials received in message 425. For example, report 432a
30 could confirm the success, bandwidth, signal strength, and other values from using both a primary and a backup set of network access credentials 126a with selected access networks 126 from the networks available list 322. Further report 432a can confirm the success, errors, or other states measured from device 101 implementing the configuration parameters for reporting system 120 in the data config.reporting-system 120a, such as the
35 ability to send or receive transducer data 125. Config.reporting-system 120a can include credentials for device 101 to connect with reporting system 120, such a certificate for

reporting server 116. Report 432a could also include data regarding success or errors related to loading and verifying the exemplary certificates in ciphertext1.CS 427 of cert1.SE 419, cert.CA2 121, cert.CA3 123, etc. Report 432a can also include available memory for SE 102 or device 101 (e.g. memory 102b, 102d, 102x, or 101f), such that
5 configuration server 112 can confirm resources are sufficient for the planned transfer of software package 104q. Other possibilities exist as well for a report 432a to contain data pertaining to the status and resources of device 101 or SE 102 without departing from the scope of the present invention.

At step 424c, SE 102 can use an encryption step equivalent to encryption step 424a
10 depicted in Figure 4c in order to create a ciphertext2.SE 434. The symmetric encryption key 442b from a prior step can continue to be utilized for encryption step 424c. The plaintext inside ciphertext2.SE 434 can include a networking command 431a which was determined in step 431 above, and the depicted embodiment shows a network command 431a of “Wi-Fi start”, or for mobile handset 108 to activate a Wi-Fi connection for device
15 101 in order for SE 102 to receive software package 104q. The plaintext inside ciphertext2.SE 434 can also include a report 432a, where an exemplary report 432a was described in the above paragraph. SE 102 can then send mobile handset 108 a message 433, where message 433 can contain ID.transaction2 426 and ciphertext2.SE 434. Although the exemplary embodiment depicted in Figure 4a has SE 102 sending network
20 command 431a in an encrypted format that cannot normally be read by mobile handset 108, the network command 431a could also be sent at plaintext to mobile handset 108.

Mobile handset 108 can forward message 433 to configuration server 112 and then also conduct a step 435. In step 435, mobile handset 108 can store or backup the currently applied network access credentials for Wi-Fi clients when mobile handset 108 operates as a
25 Wi-Fi access point or “hotspot”. In other words, a configuration user 108a may have mobile handset 108 operate as a Wi-Fi access point with specified credentials outside of the purpose of configuring a device 101 (such as when configuration user 108a has mobile handset 108 at their home or office). At step 435 the existing credentials for mobile handset 108 to operate as that Wi-Fi access point are backed up, and step 425 enables (i) a
30 new set of access point credentials to be temporarily utilized for a Wi-Fi access point created by mobile handset 108, where (ii) the “backed up” Wi-Fi access point credentials from this step 435 can be later restored for mobile handset 108.

Configuration server 112 can receive message 433 and conduct a series of steps to process the data in message 433. Configuration server 112 can conduct a decryption step
35 424d in order to convert ciphertext2.SE 434 into plaintext and read networking command 431a and report 432a. The decryption step 424d can correspond to decryption step 424b

depicted and described in connection with Figure 4c below, such as using the same symmetric decryption key 442b that was previously derived by configuration server in a step 421a also depicted in Figure 4c below.

At step 436 configuration server 112 can process networking command 431a and report 432a. Data from report 432a can be recorded in a database 112b. Configuration server 112 can send confirmation data or portions of report 432a to other elements in system 100, such as informing device owner 122 or reporting system 120 or access network 126 of the preliminary configuration of device 101, where the preliminary configuration can be a check that configuration files, certificates and network access credentials delivered in a message 425 were properly received and loaded by device 101 and/or SE 102. In an exemplary embodiment, step 436 could include configuration system 114 notifying reporting system 120 that device 101 should be activated (since the completion of a configuration step 103 is pending). Using report 432a, configuration server 112 can also evaluate or confirm that certificates, configuration data, and network access credentials delivered in the previous message 425 were properly received and applied.

In a step 436 configuration server 112 could use networking command 431a and report 432a to determine that mobile handset 108 should activate a Wi-Fi access point for download of software package 104q to SE 102, instead of continuing to use NFC session 303. In a different exemplary embodiment than that depicted in Figure 4a, configuration server 112 could use networking command 431a and report 432a to determine that SE 102 and device 101 should use a land mobile network such as access network 126 in order to download software package 104q. In yet another exemplary embodiment, configuration server 112 could use networking command 431a and report 432a to determine that SE 102 and device 101 should continue using the existing NFC session 303 in order to download the software package 104q.

At step 437 configuration server 112 can select the access point name (e.g. SSID) and password, as well as other parameters such as security schemes (e.g. WPA2 or WPA3, etc.) for mobile handset 108 to utilize in order to have device 101 connect with mobile handset 108 over Wi-Fi. The data for mobile handset to operate a Wi-Fi access point can be recorded in a dataset Wi-Fi-AP.MH 437a. Configuration server 112 can then conduct an encryption step 424e to create ciphertext2.CS 441, where the plaintext comprises dataset Wi-Fi-AP.MH 437a. Configuration server can utilize previously derived symmetric encryption key 442b to convert the plaintext into the ciphertext2.CS 441. Step 424e could correspond to encryption step 424a depicted in Figure 4c below. Configuration server 112 can then send message 438 to mobile handset 108, where message 408 includes dataset Wi-Fi-AP.MH 437a. Note that dataset Wi-Fi-AP.MH 437a can be encrypted inside secure

channel 321 between configuration server 112 and mobile handset 108, so another layer of encryption can be omitted for message 438. Mobile handset 108 would not normally be able to read ciphertext communicated between configuration server 112 and SE 102 as depicted in Figure 4a. Mobile handset 108 can record and process the data received in message 438.

Configuration server 112 can then send mobile handset 108 another message 440, where message 440 includes an ID.transaction3 439 and the ciphertext2.CS 441, where the creation of ciphertext2.CS 441 was described in the above paragraph. Mobile handset 108 can receive message 440 and subsequently send the data for message 440 to SE 102 and SE 102 can receive message 440. SE 102 can then conduct a decryption step 424f in order to convert ciphertext2.CS 441 into plaintext and read dataset Wi-Fi-AP.MH 437a. Decryption step 424f can correspond to decryption step 424b depicted and described in connection with Figure 4c below, where SE 102 can utilize the derived symmetric encryption key 442b. SE 102 could process the decrypted plaintext Wi-Fi-AP.MH 437a in order to know the credentials to use with a Wi-Fi radio 101z for device 101 and connect with mobile handset 108 using Wi-Fi. Data from the decrypted plaintext Wi-Fi-AP.MH 437a can be stored in nonvolatile memory for SE 102. The successful processing of Wi-Fi-AP.MH 437a by SE 102 could also be a signal for SE 102 to activate Wi-Fi for device 101, and utilize the credentials within Wi-Fi-AP.MH 437a.

At step 441, SE 102 can instruct device 101 to activate Wi-Fi through a radio 101z in order for the hardware and software for device 101 to (i) enable Wi-Fi session 503 depicted below in Figure 5, and (ii) receive the additional data and software package 104q to complete a configuration step 103 as depicted in Figure 1a. In other words, Wi-Fi-AP.MH 437a can provide information (i) for mobile handset 108 to activate a Wi-Fi access point with the credentials for an access point and (ii) for device 101 to activate a Wi-Fi client with the credentials for an access point. Wi-Fi-AP.MH 437a can also provide information such as a maximum power level to utilize in a Wi-Fi session 503, such as a typically lower power level due to the close proximity of the two devices. In exemplary embodiments, Wi-Fi-AP.MH 437a specifies a power level less than 1 milliwatt for either or both nodes maximum transmit power, although other possibilities exist as well.

In an alternative exemplary embodiment, device 101 can use radio 101z to activate a Wi-Fi access point with the credentials for Wi-Fi-AP.MH 437a, and then mobile handset 108 can activate a Wi-Fi client with the credentials for Wi-Fi-AP.MH 437a. In another embodiment for the present invention, either mobile handset 108 or device 101 can activate a Wi-Fi access point and the other side can operate as the Wi-Fi client. Benefits of device 101 activating the Wi-Fi access point for a Wi-Fi session 503 below in Figure 5 is that

device 101 can potentially retain greater control and thus security for the Wi-Fi session 503, but device 101 activating the Wi-Fi access point is not required in order to gain benefits from the present disclosure. Further, although Wi-Fi-AP.MH 437a is depicted in Figure 4a as credentials and parameters for a Wi-Fi network, other local wireless connectivity technologies could be utilized as well, including Bluetooth. For embodiments where NFC session 303 remains open and utilized for the transfer of software package 104q, then the use of Wi-Fi-AP.MH 437a and activation of a Wi-Fi access point and client between mobile handset 108 and device 101 can be optionally omitted.

10 Figure 4b

Figure 4b is a series of flow charts illustrating exemplary steps for nodes to mutually derive keys and then use the mutually derived keys to encrypt and decrypt data, in accordance with exemplary embodiments. Exemplary steps for nodes to mutually derive keys can comprise (i) a key exchange 411a step for SE 102 to derive a symmetric encryption key 442a and (ii) a key exchange 411b step for a configuration server 112 to derive the same symmetric encryption key 442a. Exemplary steps for nodes to encrypt and decrypt data can comprise (i) an encryption 412a step for SE 102 to utilize the symmetric encryption key 442a to convert plaintext to ciphertext, and (ii) a decryption 412b step for configuration server 112 to utilize the symmetric encryption key 442a to convert the ciphertext received from SE 102 into plaintext. The use of all steps a key exchange 411a, a key exchange 411b, encryption 412a, and decryption 412b were also depicted and described in connection with Figure 4a above. Additional detail regarding the use of these steps will be provided herein. Further, although the steps are depicted specifically for the use of particular keys and plaintext/ciphertext combinations in Figure 4b, the steps illustrated can be used with different keys and plaintext/ciphertext combinations. For example, Figure 4c below depicts the same functionality used by the nodes as in Figure 4b, but with (i) different PKI keys input to mutually derive a different symmetric encryption key, and (ii) different plaintext encrypted and different ciphertext decrypted.

A key exchange 411a step for SE 102 to derive a symmetric encryption key 442a can utilize a set of cryptographic algorithms 141 as depicted and described in connection with Figure 1b above. As depicted in Figure 4b, an SE 102 can input both of ephemeral public key from configuration server PKe.CS 402a and initial secret key SK0.SE 102y into a key exchange algorithm 447. The key exchange algorithm 447 could comprise a Diffie Hellman key exchange (DH), an Elliptic Curve Diffie Hellman key exchange (ECDH), and other possibilities exist as well without departing from the scope of the present invention. For some exemplary embodiments, the key exchange algorithm 447 could comprise a

supersingular isogeny Diffie-Hellman key exchange (SIDH). A key exchange algorithm 447 can support either PKI keys based on elliptic curves or RSA algorithms, although support of elliptic curves may be preferred in some exemplary embodiments due to their shorter key lengths and lower processing requirements. A summary of ECDH is included in the Wikipedia article titled "Elliptic Curve Diffie-Hellman" from March 9, 2018, which is herein incorporated by reference.

An exemplary embodiment of key exchange algorithm 447 could comprise a "One-Pass Diffie-Hellman, C(1, 1, ECC CDH)" algorithm as described in section 6.2.2.2 on page 81 of the National Institute of Standards and Technology (NIST) document "NIST SP 800-56A, Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography" from March, 2007 which is hereby incorporated by reference in its entirety. Other key exchange algorithms in NIST SP 800-56A could be utilized as well for a key exchange algorithm 447 in Figure 4b and Figure 4c without departing from the scope of the present invention. For some exemplary embodiments, a key exchange 447 could comprise a key encapsulation mechanism (KEM), where a node such as configuration server 112 encrypts data such as key 442a with a public key such as PK0.SE 102n, and then key 442a can be decrypted with SK0.SE 102y. The ciphertext of key 442a could be sent in message 403.

Other algorithms to derive a shared symmetric encryption key 442a using public keys and private keys may also be utilized in a key exchange algorithm, such as, but not limited to, the American National Standards Institute (ANSI) standard X-9.63. Cryptographic parameters 407 can also include information for using a key exchange algorithm 447 in order to derive a commonly shared symmetric encryption key 442a. Parameters 407 input into a key exchange algorithm 447 can include the key length in bits, an elliptic curve utilized for ECC, a time-to-live for a key 442a that is derived, and similar settings. Additional cryptographic parameters 407 for a PKI keys input into key exchange algorithm 447 can include a supported point formats extension, where the supported point formats extension could comprise uncompressed, compressed prime, or "compressed char2" formats, as specified in ANSI X-9.62. In other words, an ECC keys input into a key exchange algorithm 447 may have several different formats and a set of parameters 407 can be useful to specify the format.

As depicted in Figure 4b, the output of a key derivation function 447 such as an ECDH key exchange can be input into a key derivation function 448. Key derivation function 448 can process the output of a key exchange algorithm 447 in order to further obfuscate and optimize a derived shared secret from the key exchange algorithm 447. For example, the number of bits output from a key exchange algorithm 447 (such as an

exemplary 256 bits) may not match the number of bits utilized for a symmetric encryption key 442a (such as an exemplary 192 bits), and consequently a key derivation function 448 could reduce, process, or truncate the output of a key exchange algorithm 447.

5 In exemplary embodiments, key derivation function 448 could also utilize a secure hash algorithm such as, but not limited to SHA-256, SHA3, or similar algorithms operating on the output of key exchange algorithm 447 in order to make a resulting symmetric encryption key 442a more secure. Further, key derivation function 448 could utilize multiple rounds of XOR logic or processing on the sequence of bits output from the key derivation function 447. Parameter 407 can specify values for the operation of key
10 derivation function 448 such as the number of bits for keys, a hash algorithm to utilize, the number of rounds of XOR on a sequence of bits, and other values as well. As depicted in Figure 4b, the output of a key derivation function 448 can be both a symmetric encryption key 442a and a MAC key 441a, where MAC key 441a can function to verify message integrity of ciphertexts generated with the symmetric encryption key 442a.

15 As depicted in Figure 4b, an encryption 412a step could be utilized by SE 102 with the derived symmetric encryption key 442a in order to convert plaintext 443 into ciphertext 415. An encryption 412 step can utilize a symmetric ciphering algorithm 444a, where symmetric ciphering algorithm 444a could comprise an algorithm according to the Advanced Encryption Standard (AES), Blowfish, and Triple Data Encryption Standard
20 (3DES), and other possibilities exist as well. Parameters 407 could specify the algorithm for a symmetric ciphering algorithm 444a as well as key length and a block cipher mode such as ECB, CBC, OFB, and CFB. Parameters 407 may also specify an authenticated encryption mode such as CCM (NIST SP800-38C), GCM (NIST SP800-38D), and other examples exist as well. Symmetric encryption algorithm 444a can accept input of (i)
25 plaintext 443, (ii) symmetric encryption key 442a, (iii) parameters 407, and (iv) initialization vector (IV) 445a, and output ciphertext 415. As depicted in Figure 4b, and encryption 412a step can output a message authentication code (MAC) of 446a, where the MAC 446a was calculated with the MAC key 441a. MAC 446a allows the other party to verify message integrity of ciphertext 415 by calculating the same MAC 446a value using
30 the MAC key 441a. IV 445a can comprise a random number to introduce randomness into the ciphertext 415, and IV 445a can be transmitted as a plaintext along with the ciphertext 415 in exemplary messages, such as message 413 that contains ciphertext1.SE 415 as depicted in Figure 4a above.

Key derivation 411b step for configuration server 112 depicted in Figure 4b can
35 correspond to key derivation 411a step by SE 102. Key derivation 411b step can comprise the use of ephemeral secret key SK_e.CS 402b and the secure element public key PK₀.SE

102n. Other elements or algorithms within a key derivation 411b step can be equivalent to a key derivation 411a step above, including the use of shared parameters 407. In this manner, configuration server 112 and SE 102 can securely derive the same symmetric encryption key 442a as depicted in Figure 4b. A decryption 412b step for configuration
5 server 112 allows configuration server 112 to convert the ciphertext 415 received into plaintext 443. Decryption 412b step can utilize a symmetric decryption algorithm 444b, which could comprise the same algorithm used in symmetric encryption algorithm 444a except the algorithm being used for decryption instead of encryption. For example, if symmetric encryption algorithm 444a in Figure 4b comprises an AES-192 encryption, and
10 symmetric decryption algorithm 444b can comprise an AES-192 decryption. Note that the same values are input into symmetric decryption algorithm 444b as symmetric encryption algorithm 444a, such as symmetric encryption key 442a, parameters 407, and IV 445a in order to convert ciphertext 415 back into plaintext 443. Configuration server 112 can then read and process plaintext 443 after a decryption 412b step. Configuration server 112 can also verify MAC code 446a using MAC key 441a from a step 411b in order to verify the
15 integrity of ciphertext 415.

Figure 4c

Figure 4c is a series of flow charts illustrating exemplary steps for nodes to
20 mutually derive keys and then use the mutually derived keys to encrypt and decrypt data, in accordance with exemplary embodiments. Exemplary steps for nodes to mutually derive keys can comprise (i) a key exchange 421a step for configuration server 112 to derive a symmetric encryption key 442b and (ii) a key exchange 421b step for SE 102 to derive the same symmetric encryption key 442b. Exemplary steps for nodes to encrypt and decrypt
25 data can comprise (i) an encryption 424a step for configuration server 112 to utilize the symmetric encryption key 442b to convert plaintext to ciphertext 427, and (ii) a decryption 424b step for SE 102 to utilize the symmetric encryption key 442b to convert the ciphertext 427 received from configuration server 112 into plaintext. The use of all steps for a key exchange 421a, a key exchange 421b, encryption 424a, and decryption 424b were also
30 depicted and described in connection with Figure 4a above. Further, although the steps are depicted specifically for the use of particular keys and plaintext/ciphertext combinations in Figure 4c, the steps illustrated can be used with different keys and plaintext/ciphertext combinations. For example, Figure 4b above depicts the same functionality use by the nodes as in Figure 4c, but with (i) different PKI keys input to mutually derive a different
35 symmetric encryption key, and (ii) different plaintext encrypted and different ciphertext decrypted.

As depicted in Figure 4c, a key exchange 421a step can comprise configuration server 112 using ephemeral secret key SKe.CS 402b' and a new, updated SE 102 public key PK1.SE 410a with key exchange algorithm 447. SKe.CS 402b' could be derived in a step 402b' using parameters 407'. As described above in connection with Figure 4a, the output of key exchange algorithm 447 can be input into key derivation function 448. The output of key derivation function 448 can be both a new, second symmetric encryption key 442b and a new, second MAC key 441b. Parameters 407 can be the same for a key exchange 421a and a key exchange 411a in exemplary embodiments, but the parameters 407 could also be different. For embodiments where parameters 407 are the same as a key exchange 411a, then SKe.CSS 402b could be used in step 421a. For embodiments with a different set of parameters 407', then a second SKe.CS 402b' from a step 402' could be used with step 421a. For example, parameters 407' in key exchange 421a could specify longer key length for key 442b compared to key 442a (such as an exemplary 256 bits instead of an exemplary 192 bits), and in this case a key derivation function 448 in key exchange 421a and key exchange 421b could output a longer symmetric encryption key 442b with an exemplary length of 256 bits.

Other possibilities exist as well for parameters 407 to change between a key exchange 421a and a key exchange 411a without departing from the scope of the present invention. A key exchange 421b step can comprise SE 102 using the derived secret or private key SK1.SE 410b with the ephemeral public key from configuration server of PKe.CS 402a' in order to derive the same symmetric encryption key 442b, using the same parameters 407' as key exchange 421a step. The same MAC key 441b can be derived in both a key exchange 421a and a key exchange 421b.

As depicted in Figure 4c, an encryption 424a step for configuration server 112 can utilize the new, second symmetric encryption key 442b for input into the symmetric encryption algorithm 444a. The plaintext may comprise data depicted with message 425 in Figure 4a, and the resulting ciphertext can comprise ciphertext1.CS 427. The plaintext for an encryption step 424a can comprise a device configuration. MAC key 441b can be used to calculate MAC 446b in order to check and confirm validity and integrity of the transmission and proper receipt of ciphertext1.CS 427, such as the presence of any bit errors from the transmission for ciphertext1.CS 427 from configuration server 112 to SE 102. In exemplary embodiments, a new, second initialization vector 445b is created by configuration server 112 and utilized with ciphertext1.CS 427. A decryption 424b step for SE 102 can utilize the new, second symmetric encryption key 442b from step 421b for input into the symmetric decryption algorithm 444b. The ciphertext ciphertext1.CS 427 can be input by SE 102 into the symmetric decryption algorithm 444b along with the

symmetric encryption key 442b in order to output and read the plaintext values. SE 102 can also verify the integrity of ciphertext1.CS 427 by computing a MAC value 446b using the MAC key 441b with the plaintext and a MAC algorithm to confirm the computed MAC value 446b matches the received MAC value 446b, where equal values for the computed and received MAC value 446b means that the integrity of received ciphertext1.CS 427 is confirmed.

Figure 4d

Figure 4d is an illustration of a certificate for a device public key, where the device derived the public key and a corresponding private key, in accordance with exemplary embodiments. Public and private keys in system 100 can utilize PKI techniques based on either RSA or ECC algorithms. A certificate cert1.SE 419 based on an ECC public key for SE 102, such as PK1.SE 410a is illustrated in Figure 4c, although RSA keys or other algorithms supported by post-quantum cryptography could be utilized instead, such as lattice-based cryptography, code-based cryptography, etc. One benefit of using ECC is that an equivalent level of security can be obtained for a much smaller key length. Also, energy may be conserved using ECC algorithms compared to RSA algorithms. Smaller key lengths save bandwidth, memory, processing resources, and power, which are all valuable for a device 101 to conserve a battery and usage of radio-frequency spectrum. For example, an ECC key length of 283 bits can provide security similar to an RSA key length of approximately 2048 bits. Public key PK1.SE 410a can comprise an ECC key in an X.509 certificate, as illustrated in Figure 4c. The values to determine an elliptic curve defining equation could be stored in parameters 407 or 407', and the defining equation or curve name utilized with the key can be disclosed in the certificate. In addition, although the certificate is depicted as cert1.SE 419, the certificate can be used to authenticate device 101 since SE 102 and the identity 102m in cert1.SE 419 can be uniquely and permanently bound to device 101. In other exemplary embodiments, the device identity 101b can be used instead of or in conjunction with ID.SE 102m in a cert1.SE 419, and in those embodiments the device identity 101b could be directly inside a certificate cert1.SE 419.

Certificate 419 could include a signature 220, where signature 220 can be signed using ECC signature techniques, such as the Elliptic Curve Digital Signature Algorithm (ECDSA) for signature algorithm 227 with a secure hash such as SHA256 for a message digest 230. The creation of a signature 220 is depicted and described in connection with Figure 2b above. In order to generate signature 220, the private key associated with certificate authority 115 may also be an ECC-based private key using similar parameters

407. Note that the parameters 407 for a parent public key/certificate do not have to be identical, and the parent key/certificate belonging to CA 115 could have a longer validity or different key lengths, as examples. Note that the public key PK1.SE 410a in a certificate 419 could be based on different algorithm or curve than the private key used for signature 220 in certificate 419. For example, that the public key 410a can be an ECC key, while the signature 220 in certificate 419 could be generated with RSA algorithm and key. Certificate 419 may also include parameters 407', where parameters 407' can specify an elliptic curve utilized with the public key PK1.SE 410a. Parameters 407' could also include the start and end times for the validity of either public key PK1.SE 410a or certificate 419. Other parameters 407' can be utilized in a certificate 419 as well, and parameters 407' may specify values that are not included or external to a certificate 419.

Certificate 419 illustrated in Figure 4c illustrates exemplary embodiments of the present invention. Over the lifetime of a device 101, which could be a decade or longer, multiple device public keys such as PK1.SE 410a may be utilized. In other words, future public keys for device 101 could include a PK2.SE, PK3.SE, etc., and these keys could be signed by a certificate authority 115. The potential use of multiple different module public keys such as key 410a include (i) the expiration of a certificate 419 (including expiration of a public key associated with a certificate authority used in signature 220), (ii) a need to change an elliptic curve specified in a parameters 407', (iii) adding a new public/private key pair for connection with a different reporting system 120, (iv) increasing a key length utilized in a public/private key pair, (v) the transfer of ownership or control of device 101, where a new or different device owner 122 may require or prefer that keys with devices be rotated or renewed, and/or (vi) device 101 connecting to a new server that utilizes a different cryptographic algorithms (i.e. RSA-based instead of ECC-based). Other possibilities exist as well for reasons an SE 102 to derive a new public key PK1.SE 410a. Further, although only one public key PK1.SE 410a is depicted in Figure 4a and Figure 4c, a device 101 may utilize multiple public keys concurrently, and thus required multiple different certificates 419 for each of the different public keys. In exemplary embodiments (i) a first device public key PK1.SE 410a in a first certificate 419 can be utilized with a first reporting system 120, and (ii) a second device public key PK1.SE 410a (possibly derived using a different set of parameters 407' including using a different elliptic curve or different cryptographic algorithm) with a second certificate 419 can be utilized with a second reporting system 120 and/or access network 126.

As illustrated in Figure 4c, an identity associated with device 101 could be included in the "Common Name" (CN) field, which is depicted as ID.SE 102m. The identity associated with device 101 could be ID.device 101b and used instead of ID.SE 102m in a

certificate 419. Either ID.device 101b or ID.SE 102m could be recorded with a “Serial Number” (SN) field in the certificate. Alternatively, ID.SE 102m or ID.device 101b could be recorded in the “Organizational Unit” (OU) field, and other locations for storing or recording identities within a certificate as illustrated in Figure 4c exist as well. In addition, a specific identity or sequence count or sequence number for the public key of SE 102 and device 101 could be recorded in a certificate 419, so that PK0.SE 102n or PK0.SE 102n’ can be differentiated from PK1.SE 410a.

Also, as noted previously herein, the use of a certificate 419 may optionally be omitted, such that device 101 and reporting system 120 or configuration system 114 share public keys without using certificates 419, or the certificates could take a form different than that depicted in Figure 4c. A signature portion 220 of a certificate 419 can also include a value for “r” or similar value used with a non-deterministic use of a digital signature algorithm 227, where the value of “r” was described in Figure 2b above. In other words, an “r” value in a signature 220 of a certificate 419 or other ECDSA certificates described herein can comprise (i) a random number used once in a non-deterministic mode, or (ii) a fixed, deterministic value based on a message digest 230 in Figure 2b which could comprise a deterministic mode. Further, in exemplary embodiments, certificate 419 with a certificate authority signature 220 can comply with IETF RFC 5480 standard “Elliptic Curve Cryptography Subject Public Key Information”.

Figure 5

Figure 5 is a simplified message flow diagram illustrating an exemplary system with exemplary data sent and received by a mobile handset and a device, in accordance with exemplary embodiments. Before initiating steps and message flows depicted in Figure 5, mobile handset 108, device 101, and the other elements depicted for system 500 in Figure 5 may previously complete exemplary message flows and steps depicted in Figure 3 and Figure 4a above. System 500 can include a configuration server 112, mobile handset 108, and device 101. An exemplary difference between system 300 depicted in Figure 4a and system 500 depicted in Figure 5 can be the use of a Wi-Fi access point 501 for mobile handset 108 and a Wi-Fi client 502 for device 101.

In a system 500, mobile handset 108 and configuration server 112 can continue secure session 321 from Figure 4a above, which was also established in Figure 3 above. If a first secure session 321 terminates, such as device 101 is in a location without connectivity to an access network 126 when NFC session 303 is established, then mobile handset 108 could move “back and forth” or between (i) a first location with device 101 and (ii) a second location with communication to access network 126 for communication

with configuration server 112. In this embodiment, secure session 321 may (i) close as mobile handset 108 moves to the location with device 101 and (ii) reopen as mobile handset 108 moves to the location with communication to access network 126 for mobile handset 108.

5 As depicted in Figure 5, mobile handset 108 can conduct a Wi-Fi access point activation step 437b. Configuration data for mobile handset 108 to utilize with Wi-Fi access point 501 could be included in data Wi-Fi-AP.MH 437a from configuration server 112. Values or data in Wi-Fi-AP.MH 437a could comprise frequencies or channels to utilize, a service set identifier (SSID) or network name, user identities and passwords,
10 and/or public certificates for client 502 and access point 501, etc. In exemplary embodiments, Wi-Fi access point 501 does not broadcast the SSID such as in a broadcast frame, and the only client allowed can be device 101 using client 502. In addition, by using an obfuscated and non-broadcasted SSID (possibly a pseudo random string) that has only been shared outside mobile handset 108 with device 101 via ciphertext 441 with Wi-
15 Fi-AP.MH 437a, then only device 101 could reasonably connect with Wi-Fi access point 501. Further, an access list of allowed users for Wi-Fi access point 501 can result in only device 101 being able to connect via Wi-Fi connection setup 503 in Figure 5. Further, Wi-Fi connection setup 503 can utilize standard Wi-Fi security protocols such as, but not limited to, WPA2, WPA3, or subsequent standards. In exemplary embodiments, Wi-Fi
20 connection setup 503 utilizes a preshared secret key (PSK), where the PSK is received by mobile handset 108 in message 438 in Figure 4a above and sent to device 101 in message 440 in Figure 4a. As depicted in Figure 5, in exemplary embodiments, mobile handset 108 and device 101 can utilize network credentials of “user name, password 437aa” to secure connection setup 503.

25 In exemplary embodiments, Wi-Fi-AP.MH 437a can specify a low maximum output power for Wi-Fi access point 501, such as an exemplary 1 milliwatt or less. Wi-Fi-AP.MH 437a can also specify the Wi-Fi standard to utilize within the family of 802.11 standards from IEEE, such as version g, n, ac, etc. During use of a Wi-Fi access point 501 as depicted with system 500, device 101 may normally be in close physical proximity of
30 mobile handset 108, such as less than an exemplary few meters, and consequently relatively lower transmit power for mobile handset 108 could be utilized, compared to the typical maximum power typical for standard Wi-Fi usage in 2018 of approximately 200 milliwatts. In addition, although a Wi-Fi access point 501 is depicted in Figure 5, mobile handset 108 and device 101 could implement a Bluetooth network, where either device 101
35 or mobile handset 108 can function as the master. For this embodiment, parameters and credentials for a Bluetooth network for connection setup 503 could be passed in the data

Wi-Fi-AP.MH 437a. Other short-range and LAN wireless network protocols besides Wi-Fi and Bluetooth could be utilized as well without departing from the scope of the present invention. A configuration database 112a could record if device 101 should utilize Wi-Fi, Bluetooth, or another wireless protocol for connection setup 503. As mentioned above, in
5 exemplary embodiments where the size of software package 104q is deemed to be sufficiently small to be supported by NFC session 303, then a separate connection setup 503 could be omitted and the NFC session 303 could be utilized instead for a system 500 in Figure 5.

For device 101 in system 500, device 101 could use a step 503a in order to activate
10 a Wi-Fi client 502. Parameters and credentials for Wi-Fi client 502 could be received in a message 440 above with Wi-Fi-AP.MH 437a. For a step 503, SE 102 which received data in Wi-Fi-AP.MH 437a could send (i) data to device 101 CPU 102c and radio 101z via (ii) data bus 101d as depicted in Figure 5 in order to activate Wi-Fi client 502. In exemplary
15 embodiments, the radio 101z for Wi-Fi client 502 in device 101 resides outside of SE 102 as depicted in Figure 1b. However, other embodiments and configurations for SE 102 and radio 101z are possible and also depicted and described below in connection with Figure 6a and 6b. Consequently, other configurations of internal elements such as SE 102 and Wi-Fi
client 502 are possible without departing from the scope of the present invention. A step 503 could include the use of a firewall for SE 102 and/or device 101, such that SE 102
20 would not normally respond to packets, queries, data, or probes via connection 503 unless the data successfully includes a properly decrypted ciphertext3.CS 506 below.

At step 504 in Figure 5, configuration server 112 can prepare software package 104q for transfer to device 101 via mobile handset 108. An exemplary software package 104q is depicted in Figure 5, and was also described above in Figure 4a. Software package
25 104q could also be referred to as a second device configuration, where a first device configuration could be the plaintext in a step 424a above. Configuration server 112 could use a configuration database 112a, an identities list 320, data from device owner 122 and device manufacturer 101x, and data from reporting system 120 in order to assemble software package 104q. Configuration server 112 or configuration system 114 could
30 include an updated secure element firmware 102u' in software package 104q. Software package 104q can comprise a compressed format such as using gzip or other compression techniques. Configuration server 112 can also create a full file list 507 for software package 104q in a step 504. The file list 507 can also include a signature 220 from configuration server 112 using a secret key SK.CS, such that SE 102 can use a signature
35 verification step 221 with the PK.CS from cert.CS 112c. In this manner, SE 102 can verify that software package is from previously authenticated configuration server 112.

In preferred exemplary embodiments, configuration server 112 also conducts a signature verification 221 step on individual software components for software package 104q that configuration server 112 receives from other nodes in a system 100. For example, before (a) configuration server 112 signs software package 104q for device 101, which could include files for an updated device OS 101e where the files for the updated device 101e were received by a device manufacturer 101x, (b) configuration server 112 could conduct a signature verification 221 step for a signature by device manufacturer 101x on the files for updated device OS 101e. In other words, in exemplary embodiments, configuration server 112 verifies signatures on files received for software package 104q before configuration server 112 conducts a signature creation step 220 for the software package 104q. In this manner, elements for software package 104 may be sourced from multiple different parties in a system 100, and configuration server 112 could verify authenticity for each element and provide an overall signature or “stamp of approval” of software package 104q for device 101. In this manner, device 101 may not need to check the signatures on individual elements within software package 104q, which could be difficult if device 101 or SE 102 does not have all the intermediate and root certificates for each of the individual element signatures in a software package 104q.

Configuration server 112 can then conduct an encryption step 424g to create ciphertext3.CS 506. The encryption step 424g can be equivalent to encryption step 424a depicted and described in connection with Figure 4c, where the plaintext to be encrypted can comprise both file list 507 and software package 104q (where software package 104q may be previously compressed). Note that second symmetric encryption key 442b can continue to be used with encryption step 424g. In exemplary embodiments, a different, updated value for initialization vector 445b is used with an encryption step 424g. Configuration server 112 can then use connection 321 to send a message 505 to mobile handset 108, where message 505 includes an ID.transaction4 and ciphertext3.CS 506, where ciphertext3.CS was created in step 424g.

Mobile handset 108 can receive message 505 via connection 321 and forward message 505 to device 101 via connection 503. In exemplary embodiments where device 101 is out of range from an access network 126 utilized by mobile handset 108, mobile handset 108 can receive message 505 and store or cache the data contained such as ciphertext3.CS 506. Mobile handset 108 could then temporarily close connection 321 and move to device 101 and re-establish connection 503, and then subsequently send ciphertext3.CS 506 to device 101. In this manner, secure connection 321 and connection 503 as depicted in Figure 5 do not need to be operating concurrently in exemplary embodiments. The present invention allows device 101 to securely conduct a configuration

step 103 even when device 101 is outside the range of an access network 126 (e.g. device 101 is in a physical location where mobile handset 108 cannot reach an access network 126 to connect with configuration server 112 when mobile handset 108 has a connection 503 or 303 established).

5 Device 101 can receive message 505 over connection 503. Device 101 using SE 102 can conduct a decryption step 424h for ciphertext3.CS 506, where decryption 424h can be equivalent to decryption 424b depicted and described in connection with Figure 4c above. The symmetric encryption key 442b in a decryption step 424h for SE 102 can be the same symmetric encryption key 442b for a decryption step 424f in Figure 4a above.
10 Note that in exemplary embodiments, the same symmetric encryption key 442b can be used by SE 102 to (i) decrypt ciphertext2.CS 441 received through NFC session 303 and (ii) decrypt ciphertext3.CS 506 received a different radio link than ciphertext2.CS 441. SE 102 can use decryption 424h to read the plaintext values of file list 507 and software package 104q.

15 SE 102 can then conduct a step 507, where step 507 can include both (i) SE 102 decompressing software package 104q using an exemplary library such as gzip, and (ii) SE 102 conducting a signature verification 221 step over software package 104q, where software package 104q was previously signed by configuration server 112 in a step 504 above. In this manner, SE 102 and device 101 can be reasonably assured that files and
20 components in software package 104q are genuine or authenticated/approved by configuration server 112. SE 102 can (A) report an error or (B) refuse to process software package 104q or elements within software package 104q if (C) a signature verification step 221 fails. Step 507 can also comprise SE 102 verifying that software package 104q contains all the data and files listed in a file list 507

25 At step 508 in Figure 5, SE 102 and device 101 can record the data within software package 104q in either SE storage memory 102d or device storage memory 101f. In exemplary embodiments, files for device OS 101e can be recorded in device storage memory 101f and files for SE firmware can be recorded in SE storage memory 102d, and other possibilities exist as well. A step 508 can also comprise SE 102 and device 101
30 backing up or storing the running configuration for each before loading data in a software package 104q, and in this manner the system can be restored if applying the updated files for software package 104q fails. After internally storing or loading the files for software package 104q, device 101 and SE 102 can perform a reboot, so that both restart with the new files from software package 104q. Upon a reboot in a step 508, connection 503 may
35 temporarily terminate with the reboot, but device 101 can re-establish connection 503 using the radio 101z and Wi-Fi client 502. A step 508 can then also comprise device 101 and/or

SE 102 creating a report 508a, where report 508a includes a status code with success or errors for each file in file list 507 for software package 104q. In other words, report 508a can record the success or errors of applying each of the files in software package 104q, which may be useful for configuration server 112 or other authorized elements in system 100 to know the state of device 101.

SE 102 can then conduct an encryption step 424i, which could correspond to encryption step 424a depicted in Figure 4c. Encryption step 424i could comprise using report 508a as plaintext and the symmetric encryption key 442b in order to create ciphertext3.SE 510. The symmetric encryption key 442b in an encryption step 424i for SE 102 can be the same symmetric encryption key 442b for a decryption step 424f in Figure 4a above. MAC codes 446b and initialization vectors 445b can also be processed with encryption step 424i and also all steps 424 where encryption or decryption takes place. SE 102 can then send mobile handset 108 a message 509 over connection 503, where message 509 includes the transaction identity ID.transaction4 and the ciphertext3.SE 510. As depicted in Figure 5, mobile handset 108 can then forward the data received in a message 509 over connection 321 to the configuration server 112. Configuration server 112 can then conduct a decryption step 424j in order to convert ciphertext3.SE 510 into plaintext, and thereby read the report 508a as plaintext. Decryption 424j can be equivalent to decryption 424b depicted and described in connection with Figure 4c above. The symmetric encryption key 442b in a decryption step 424j for configuration server 112 can be the same symmetric encryption key 442b for an encryption step 424g above in Figure 5.

Upon reading the plaintext report 508a, configuration server 112 can conduct a step 511 as depicted in Figure 5. Configuration server 112 could record in a configuration server database 112b each of the files in a file list 507 for software package 104q that were successfully installed. For embodiments where some non-fatal errors or warnings were recorded in report 508a, then codes or logs of errors in a report 508a could also be recorded in a configuration server database 112b. A step 511 can also comprise configuration server 112 sending messages to other elements in system 100 in Figure 1a regarding the status of device 101, such as sending (i) device owner 122 a notice of successful load of device reporting application 101v from software package 104q, (ii) reporting system 120 a notice that device 101 should begin reporting, and (iii) access network 126 that device 101 will begin connecting with network access credentials 126a, and other possibilities exist as well for configuration server 112 to use or process data received in a report 508a from device 101 and SE 102. If error codes are for the installation of software are received in a report 508a, such as some software in package 104a not being successfully installed, then configuration server 112 could use a step 511 with a configuration database 112a or 112b

to determine next steps, such as potentially re-sending message 505 or resending a portion of the software package 104q with error codes or conditions identified in a report 508a.

After processing step 511, configuration server 112 could perform a step 512, which can comprise a series of steps to close communications for mobile handset 108 and SE 102. Configuration server 112 could generate a command 512a for mobile handset 108, where command 512a instructs mobile handset 108 to close connection 321. Configuration server 112 could generate a command 512b for device 101, where command 512b instructs device 101 to close connection 503 and also being reporting through reporting system 120 using access network 126 in Figure 1a. Configuration server 112 could generate a configuration user report 512c, where report 512c could be for the configuration user 108a operating mobile handset 108. Report 512c could provide a human readable status for display on mobile handset 108 regarding the status of device 101 and a configuration step 103. Report 512c can also provide manual instructions for any modifications of transducers 105 or manual changes for configuration user 108a to perform for device 101, monitored unit 106, or connected equipment in order to complete the configuration step 103. In an exemplary embodiment, a report 512c could request that configuration user 108a attach an external antenna to device 101 in order to improve RF signal strength with access network 126. In another exemplary embodiment, report 512c could instruct configuration user 108a to perform a manual power cycle of device 101 or monitored unit 106. Other possibilities exist as well regarding instructions for a configuration user 108a in a report 512c to complete a configuration step 103 without departing from the scope of the present invention.

After completing a step 512, configuration server 112 can send mobile handset 108 a message 513 through connection 321, where message 512 can include (i) command 512a for mobile handset 108 to close connection 321, (ii) command 512b for device 101 to close connection 503 and begin reporting through reporting system 120 using access network 126 and credentials 126a, and (iii) configuration user report 512c for configuration user 108a. Mobile handset 108 can receive message 512 and forward the portions of the message 513 for (ii) command 512b to SE 102 in device 101. Mobile handset 108 can then conduct a step 514a to close connection 321 with configuration server 112. Mobile handset 108 can then conduct a step 515, where the configuration user report 512c is displayed to configuration user 108a, and a step 515 can also comprise configuration user 108a entering information in a display for mobile handset 108 confirming any manual changes for device 101 or monitored unit 106. In some embodiments, a step 515 can take place before a step 514a, such that mobile handset 108 can send information back to configuration server 112

and a configuration database 112a regarding any manual changes performed as requested in the configuration user report 512c.

At step 516, mobile handset 108 can then restore the previous Wi-Fi access point credentials for Wi-Fi access point 501 recorded in a step 435 above in Figure 4a. In other words, the original Wi-Fi access point credentials used by mobile handset 108 before a configuration step 103 and connection 503 would preferably be restored, to reduce interruption of services for configuration user 108a. For example, configuration user 108a could have a laptop or tablet that periodically connects with mobile handset 108 using a Wi-Fi access point 502 and credentials that were recorded in step 435. If those credentials from step 435 were not restored for mobile handset 108 in a step 516, then credentials in Wi-Fi-AP.MH 437a would continue to be used and the exemplary laptop or tablet would not normally be able to connect with mobile handset 108.

SE 102 and device 101 can receive the portions of message 513 as depicted in Figure 5, where a portion could comprise command 512b for device 101 to close connection 503. SE 102 and device 101 could then perform a step 514b and close connection 503. In exemplary embodiments, device 101 then loads network access credentials 126a for connecting with access network 126. Device 101 could then select an access network 126 for conducting a connection setup 518. Device 101 could select the access network based upon network access credentials 126a or from a networks available list 322 received in an identity list 320 from a message 425. Other possibilities exist as well for device 101 to select an access network for a connection setup 518 without departing from the scope of the present invention.

Device 101 then conducts a connection setup 518 with access network 126. For a connection setup 518, device 101 can use either (i) the network access credentials 126a or (ii) the private key SK1.SE 410b with cert1.SE 419. For example, with 5G networks that support EAP-TLS, then with connection setup 518, device 101 can (i) authenticate with an access network 126 using SK1.SE 410b and cert1.SE 419 and also (ii) mutually derive a symmetric ciphering key with access network 126 using SK1.SE 410b and cert1.SE 419. In some exemplary embodiments, network access credentials 126a could comprise a profile for an embedded universal integrated circuit card (eUICC). In addition, network access credentials 126a could comprise a pointer, URL, or another identifier of an eUICC profile for device 101 to load. The transfer of network access credentials 126a from a message 426 in Figure 4a above could comprise a transfer of the pointer, URL, or another identifier for the eUICC profile.

In addition, although Figure 5 depicts a step 514b of closing connection 503 before connecting with access network 126, device 101 could connect with access network 126

before closing connection 503. In this manner of device connecting with access network 126 before closing connection 503, device 101 can report error conditions or problems for connecting to access network 126 to mobile handset 108 and configuration user 108a. At step 517, device 101 can load the network access credentials 126a in order to connect with access network 126. Note in exemplary embodiments, the access network 126 used by device 101 can be different than the access network 126 used by mobile handset 108, although the two access networks 126 could be the same in other embodiments.

After loading network access credentials 126a in a step 517, device 101 can connect with access network 126 in a step 518. Although an SE 102 is depicted as conducting the connection 518, device 101 may perform the connection 518 such as using a radio 101z for device 101 as depicted in Figure 1b. Connection 518 could be through either a wireless LAN such as Wi-Fi, where network access credentials 126a can comprise data equivalent or similar to data in Wi-Fi-AP.MH 437a, or connection 518 would be through a wireless WAN such as any of a 3G, 4G, or 5G wireless network, including a low power wide area network over non-licensed spectrum such as in the 900 MHz ISM band. As depicted in Figure 5, connection 518 can connect device 101 securely with access network 126 using network access credentials 126a.

At step 519, device 101 can process data in order to connect with reporting server 116, where reporting server 116 can be a server or set of servers in a reporting system 120. Data processed in a step 519 can include collecting data from transducers 105 using device reporting application 101v via a transducer bus 101i, which can include startup or initialization data for transducers 105. In exemplary embodiments, a step 519 also includes a calibration of transducers 105 with assistance of configuration user 108a and mobile handset 108. Mobile handset 108 could display to configuration user 108a instructions or steps related to calibration, such as providing a known reference signal to a transducer 105. An example of providing a known reference signal for transducer 105 could be putting a temperature probe 105ba such as a thermocouple or thermistor in an ice water bath, and other examples exist as well. In other exemplary embodiments, a calibration of transducers in a step 519 can be omitted.

SE 102 in device 101 for a step 519 could also verify a certificate for reporting server 116, reading a DNS name for reporting server 116, a TCP or UDP port number to connect with at reporting server 116, and other steps or data for device 101 or SE 102 could be in the config.reporting-system 120a received in a message 425 depicted in Figure 4a. A certificate for reporting system 120 or reporting server 116 could be included in config.reporting system 120a. In exemplary embodiments, config.reporting-system 120a

contains a list of cryptographic parameters to utilize when communicating with reporting system 120 and reporting server 116 (such as a subset or superset of parameters 407).

Device 101 can then conduct a series of steps in order to establish a secure communications link with reporting server 116. Figure 5 depicts an overview of the connection between device 101 and both access network 126 and reporting server 116, and consequently only a subset of the steps and message flows between those elements are depicted in Figure 5. In exemplary embodiments, the connection between device 101 and reporting server 116 comprises a “datagram TLS” (DTLS) connection as specified in IETF RFC 6347, although other protocols could be used as well. In exemplary embodiments, device 101 sends reporting server 116 the certificate cert1.SE 419 in a message 520, where cert1.SE 419 was received by device 101 and SE 102 above in a message 425 depicted and described in connection with Figure 4a.

In other words, device 101 can begin communicating with reporting server 116 and reporting system 120 using a certificate based on a PKI key pair that was derived by device 101 and SE 102 during a configuration step 103. Cert1.SE 419 can include PK1.SE 410a, and SE 102 can use SK1.410b in with internal, secure, private operations for data in cert1.SE 419. The previously installed certificate cert0.SE and private key SK0.SE 102y can be deprecated or replaced with the new credentials. In this manner, security for device 101 can be increased, since SK0.SE 102y could physically be in the hands or control of other parties besides device owner 122 before installation or operation with monitored unit 106. Further, preferred security practices may require that secret keys and certificates get rotated over time, especially when validity dates for certificates expire, and thus sending a new certificate cert1.SE 419 to reporting server 116 may be periodically required. An exemplary certificate cert1.SE 419 is depicted in Figure 4c. Further, in other exemplary embodiments, message 520 comprises device 101 sending an identity of device 101 such as ID.device 101b to reporting server 116, and reporting server 116 then uses ID.device 101b to fetch the certificate cert1.SE 419 from configuration system 114 or device owner 122.

In addition, over approximately the next decade, devices deployed with cryptographic algorithms and keys supporting ECC cryptography may need to update to support post-quantum cryptography in order to remain reasonably secure. An installed device initially using ECC cryptography could update private keys and algorithms to new post-quantum cryptography algorithms using the steps disclosed herein, including receiving a cert1.SE 419 supporting post-quantum cryptographic algorithms after a device has been initially deployed supporting only ECC and/or RSA based algorithms.

Reporting server 116 can receive the certificate cert1.SE 419 and then use a step 521a to verify the certificate, including using a signature verification step 221 depicted in

Figure 2b. Step 521a could also comprise reporting server 116 and reporting system 120 verifying that device 101 is authorized to use reporting system 120, after device 101 with device identity 101b has been authenticated. Since message 520a could comprise an initial receipt of data from device 101 using ID.device 101b, step 521a could also comprise
5 reporting system fetching or querying configuration system 114 using ID.device 101b for configuration data pertaining to device 101. The device configuration data for reporting server 116 received in a step 521a could be stored in the configuration database 112a or 112b, and can include identity list 320, networks available list 322, software package 104q, files list 507, authentication parameters 111d, and parameters 407. With data from a step
10 521a, reporting server 116 can select subsequent steps or procedures in order to communicate with device 101 and SE 102 inside device 101.

Step 521b for reporting server 116 can comprise reporting server 116 generating an ephemeral public key PKe.RS 522 and an associated ephemeral secret key SKe.RS. For a step 521b, reporting server 116 could use a similar set of steps described in step 402 above
15 for configuration server 112 to derive a PKI key pair, including the use of a random number generator, a random number, a key pair generation function, and a set of parameters 407. Further, although Figure 5 depicts reporting server generating and sending an ephemeral key PKe.RS 522, in exemplary embodiments, SE 102 can (i) generate the ephemeral key pair using similar steps as described above for reporting server 116 or
20 configuration server 112 in a step 402, and (ii) send the signed ephemeral public key to reporting server 116. Step 524a for reporting server 116 can comprise reporting server 116 using a key exchange 421a step as depicted in Figure 4c, except reporting server 116 would utilize PKe.RS 522 as the ephemeral key input into key exchange algorithm 447, along with PK1.SE 410a. The output of step 524a would be a new symmetric encryption key
25 442b for use by reporting server 116 with device 101 and SE 102. Note that a key exchange step 524a could continue to use PK1.SE 410a, which could be received in cert1.SE 419. For the alternative embodiment where the ephemeral public key is derived by SE 102, then a key exchange 524a step for reporting server 116 can comprise reporting server 116 inputting an ephemeral key PKe.SE and a secret key SK.RS into key exchange
30 algorithm 447.

Step 220i for reporting server 116 can comprise reporting server 116 using a signature creation 220i step in order to create signature1.RS 523. The secret key used to create signature1.RS 522 can correspond to the public key in a certificate for reporting server 116. The message to sign for signature1.RS 522 can comprise the ephemeral public
35 key PKe.RS 522, which could be derived in a step 521b above. Reporting server 116 can then send PKe.RS 522 and signature1.RS 523 to SE 102 as depicted in Figure 5. SE 102

can receive the data, and conduct a signature verification step 221i in order to verify signature1.RS 523, using a public key for reporting server 116 in a certificate for reporting server 116. A certificate for reporting server 116 could be received with Config.Reporting System 120a in message 425. At step 524b, SE 102 can conduct a key exchange 524b step which corresponds to the key exchange 524a step conducted by reporting server 116. Step 524b for SE 102 can comprise SE 102 using a key exchange 421b step as depicted in Figure 4c, except SE 102 would utilize PKe.RS 522 as the ephemeral key input into key exchange algorithm 447. The output of step 524b would be a new symmetric encryption key 442b for use by SE 102 and device 101 with reporting system 116.

Device 101 can then conduct at step 525, which can comprise device 101 collecting or sending transducer data 125 for transducers 105. A step 525 could also comprise device 101 running through a set of configuration test vectors 104x and generating a report 525a. Configuration test vectors 104x could comprise device 101 testing dynamic range of transducers 105, having configuration user 108a provide test data for transducers 105, or also having transducers 105 collect initial, "production" data regarding monitored unit 106. Report 525a can comprise a set of data generated by device 101 using transducers 105 with monitored unit 106, including via a configuration test vector 104x. In other exemplary embodiments, a configuration test vector 104x could be omitted and transducers 105 and device 101 could simply begin operation with monitored unit and data could be recorded in a report 525a.

Device 101 using SE 102 can then conduct an encryption step 424k as depicted in Figure 5, and SE 102 can conduct an encryption step 424k to create ciphertext4.SE 527. The encryption step 424k can be equivalent to encryption step 424a depicted and described in connection with Figure 4c, where the plaintext to be encrypted can comprise transducer data 125 and/or report 525a. Note that symmetric encryption key 442b computed in step 524b can be used with encryption step 424k, which can also be equal to the symmetric encryption key 442b computed in step 424m below for reporting server 112. Device 101 can then send reporting server 116 a message 526, where message 526 can include ciphertext4.SE 527, where the plaintext for ciphertext4.SE 527 can comprise transducer data 125 and report 525a.

As depicted in Figure 5, reporting server 116 can receive message 526 and conduct a decryption step 424m. Decryption step 424m can utilize the symmetric encryption key 442b derived in a step 524a above, and decryption step 424m can be equivalent to decryption step 424b depicted in Figure 4c. After decryption step 424m reporting server 116 can read the plaintext values of transducer data 125 and report 525a. At step 528, reporting server 116 can record plaintext data from message 526 in a reporting database

118 or also send data to device owner 122. If expected values for transducer data 125 are received and report 525a meets acceptance criteria, then at step 528 (i) reporting server 116 and reporting system 120 can record that configuration step 103 has been successfully completed, and (ii) reporting system 120 could inform device owner 122 and/or configuration system 114 in Figure 1a that a configuration step 103 of device 101 has been successfully completed. In exemplary embodiments, upon successful completion of a configuration step 103 of SE 102 then SE 102 can be depicted as a secure processing environment 104 or SE 104 as shown in Figure 1a. If error conditions are noted for transducer data 125 or report 525a, then configuration user 108a, device owner 122, and configuration system 114 could be notified by reporting server 116. Upon conclusion of step 528 in Figure 5, both (i) device 101 with SE 104 and (ii) reporting server 116 can proceed with regular operation of device 101 with monitored unit 106 since the configuration step 103 has been completed.

Mobile handset 108 can conduct a step 529, where step 529 can comprise the configuration application 108b sending a query 530 to reporting server 116 in order to confirm successful completing of configuration step 103. As described in the paragraph above, reporting server 116 could determine that configuration step 103 is complete based on the successful receipt of transducer data 125 and report 525a. Receipt of a report 525a can confirm any of: (i) network access credentials 126a for access network 126 function, (ii) device 101 has wireless or wired connectivity properly established, (iii) transducers 105 are connected to device 101 and functioning, (iv) that transducers 105 are collecting transducer data 125 for monitored unit 106, (v) a secure processing environment (SE) 102 within device 101 has been configured with updated keys such as PK1.SE 410a and SK1.SE 410b, (vi) software package 104q has been downloaded and applied for device 101 and SE 102, (vii) a complete set for an identity list 320 has been received, including physical location of device 101 such as geographical coordinates 325, (viii) an updated certificate cert1.SE has been processed and received by reporting system 116, (ix) reporting system 116 can communicate with device 101 in a secure manner, (x) device 101 can utilize a “backup” or second access network 126 with a second set of network access credentials 126a if the first access network 126 becomes unavailable, and (xi) a configuration step 103 for device 101 properly functions, such that a configuration step 103 could be conducted a second time in the future if required, and/or (xii) device 101 records root and intermediate certificates in order to verify a certificate for reporting server 116, configuration server 112, and authentication server 111.

As depicted in Figure 5, reporting server 116 can reply to query 530 with a message 531 of “device OK”, if configuration step 103 has been confirmed by reporting system 116

as completed, such as through any of (i) through (xii) in the paragraph above. Not all of (i) through (xii) are required for some embodiments. In exemplary embodiments, upon successful receipt of a message 531 with a status of “device OK” for device 101 or an equivalent message 531, then mobile handset 108 can display to configuration user 108a that a configuration step 103 for device 101 has been completed and device 101 can be left in operation, potentially without additional steps or manual changes to be performed by configuration user 108a. For embodiments where message 531 provides a status of “device not OK”, then mobile handset 108 could display to configuration user 108a an error status and configuration user 108a could work with device 101 and configuration system 114 in order to diagnose and rectify the error code in order to receive a subsequent message 531 “device OK”.

Figure 6a

Figure 6a is a graphical illustration of components for a device, in accordance with exemplary embodiments. Device 101 as depicted in the above Figure 1a through Figure 5 may comprise several different physical and logical embodiments in the present invention, and Figure 6a depicts a different physical configuration for device 101 than that depicted in Figure 1b. Device 101 as depicted in Figure 6a herein and Figure 6b below can still utilize the components and sub-steps for a configuration step 103 as contemplated herein. As depicted in Figure 6a, a device 101 could include a central processing unit (CPU) 101c, a radio 101z, a transducer bus TR bus 101i, an NFC radio 102c, and a nonvolatile memory storage 101f. The elements for device 101 in Figure 6a can provide equivalent functionality for the elements depicted in Figure 1b above.

As depicted in Figure 6a, a secure processing environment (SE) 102 can be included within CPU 101c in exemplary embodiments. In other words, a separate, dedicated processor for SE 102 is not required in order to obtain the benefits from a configuration step 103 in the present invention, and the functionality of SE 102 could be provided by CPU 101c. In exemplary embodiments, the functionality of SE 102 could either be (i) physically separate hardware within CPU 101c, such as a dedicated processing core or memory registers and transistors that are dedicated to the functionality of SE 102, or (ii) logically separate functions within CPU 101c, such as dedicated “time slices” during the operation of CPU 101c. For a logically separate function of SE 102 within CPU 101c, CPU 101c can switch from a first mode to a second mode, where operation in the first mode can comprise function as a CPU 101c for device 101, and the second mode can comprise a secure processing environment 102. For example, a memory register within SE 102 in Figure 6a could (X) record secret key SK0.SE 102y or SK1.SE 410b in order to

process cryptographic algorithms 141 such as encryption step 412a or 424b when (Y) CPU 101c in Figure 6a operates in the second mode described in the sentence above. At a later time, CPU 101c could switch to the first mode where CPU 101c does not operate as SE 102 and in that case the memory register recording a secret key could be “invisible” or “not available” to CPU 101c. Other possibilities exist as well for a CPU 101c and an SE 102 to be combined without departing from the scope of the present invention.

A radio 101z for device 101 could operate in any mode supporting 2G, 3G, 4G, 5G, or subsequent wireless standards in order to connect with an access network 126. Radio 101z could also support other wireless standards such as the 802.11 “Wi-Fi” standards, or low power wide area networking such as Sigfox, LoRa, or NB-IoT. A device 101 could also use multiple radios 101z in order to connect with different wireless networks. A second radio in device 101 in Figure 6a can comprise NFC 102c, which could utilize standards supporting “near field communications”. The radio NFC 102c can include components similar to radio 101z, but supporting operating at a frequency near 13 MHz and also at low power. Either radio 101z or radio NFC 102c can include standard radio components such as RF filters, RF amplifiers, a clock, and phased loop logic (PLL), and may be connected with an antenna. As described above, although radio 102c is depicted as “NFC 102c”, in other exemplary embodiments, radio 102c can operate using wireless standards other than “near field communications”. In exemplary embodiments radio 102c could operate as a Bluetooth radio (e.g. IEEE 802.15.1 or subsequent), or a Wi-Fi radio supporting standards within the family of standards IEEE 802.11. In addition, although Figure 1b depicted radio 102c as operating within SE 102, a device 101 can also support a radio 102c operating outside SE 102 and connected to CPU 101c via bus 101g.

Figure 6b

Figure 6b is a graphical illustration of components for a device, in accordance with exemplary embodiments. Device 101 as depicted in the above Figure 1a through Figure 6a may comprise several different physical and logical embodiments in the present invention. Figure 6b depicts a different physical configuration for device 101 than that depicted in Figure 1b and 6a, while device 101 can still utilize the components and sub-steps for a configuration step 103 as described herein. As depicted in Figure 6b, a device 101 could include a “system on a chip” (SOC) 601, a transducer 105, and a Radio RF 602b. SOC 601 could comprise a package soldered onto a circuit board within device 101 which integrates the various components depicted for SOC 601. Radio RF 602b can comprise a radio front end for a radio 101z, where radio RF 602b is coupled to radio baseband 602a, where radio baseband 602a comprises a baseband processor for a radio RF 602b. In exemplary

embodiments, radio RF 602b and radio baseband 602a can operate in conjunction to provide the functionality of a radio 101z as depicted and described in connection with Figure 1b. The transducer 105 in a device 101 could comprise a transducer 105 as depicted and described in connection with Figure 1b above, including any of transducer 105 that
5 comprises a bi-directional transducer 105a, transducer input 105b, or transducer output 105c.

A system on a chip (SOC) 601 in Figure 6b could include a processor CPU 101c, a secure processing environment 102, a transducer interface 102h, nonvolatile memory storage 101f, random access memory 101d, and radio baseband 602a, where the
10 components for SOC 601 can be linked via bus 601a. Bus 601a could comprise a data bus similar to bus 101g or bus 102i as depicted and described in connection with Figure 1b above, and bus 601a can have bandwidth and signal connections necessary to support the internal operation of the different components within SOC 601. The processor CPU 101c, SE 102, transducer interface 102h, and random access memory 101d can operate and
15 provide the functionality as depicted and described in connection with Figure 1b above for the same elements shown. Although CPU 101c and SE 102 are depicted as separate elements in Figure 6b, in exemplary embodiments CPU 101c and SE 102 within a SOC 601 could be combined into a single processing environment, such as the combined CPU 101c and SE 102 as depicted and described in connection with Figure 6a. Either (X) CPU
20 101c and SE 102 could operate with separate physical transistors, logic gates, and memory registers in exemplary embodiments, or (Y) SE 102 and CPU 101c could share physical resources within SOC 601 and operate in a logically separate manner. In other exemplary embodiments for a device 101 in Figure 6a and also other figures herein, a separate SE 102 could be optionally omitted, and the functionality of SE 102 could be provided by a CPU
25 101c. As one example, instead of a set of cryptographic algorithms 141 within SE 102 as depicted in Figure 1b being loaded into an EEPROM 102j, the set of cryptographic algorithms 141 could be performed by CPU 101c and stored in a nonvolatile memory 101f for device 101. Other possibilities exist as well for the configuration of a CPU 101c and SE 102 without departing from the scope of the present invention.

30 In exemplary embodiments such as that depicted in Figure 6b, the function of an NFC radio 102c could operate within separate elements within a device 101. The operation of a radio 102c could be divided into a RF baseband 102ca processor and an RF front end 102cb. RF baseband 102ca can provide baseband processing functionality for radio 102c, and RF front end 102cb can provide RF front end functionality such as RF filters and RF
35 amplifiers. RF front end 102cb can be connected to an antenna, which could also be located inside the physical housing of device 101, or an antenna can be external to device

101 and electrically connected to radio RF 602b. The divided NFC radio 102c could also support other relatively short-range wireless communication protocols appropriate for a device 101 communicating with a mobile handset 108. Other protocols besides NFC could comprise Bluetooth, Bluetooth Low Energy (BLE), ZigBee, Z-Wave, 6LoWPAN, Thread, 5 Wi-Fi, Wi-Fi-ah (HaLow), LoRA, and ANT (from ANT Wireless). Other possibilities exist as well for a short range wireless communications protocol supported by radio 102c (or RF baseband 102ca and RF front end 102cb) without departing from the scope of the present invention.

10 Conclusion

Various exemplary embodiments have been described above. Those skilled in the art will understand, however, that changes and modifications may be made to those examples without departing from the scope of the claims.

CLAIMS

1. A method for a device to securely receive configuration data, the method performed by the device, the method comprising:
- a) recording in a protected memory, a first device private key for an elliptic curve Diffie-Hellman (ECDH) key exchange algorithm, a second device private key for a digital signature algorithm, a certificate authority public key, and a first set of cryptographic parameters, wherein the first device private key corresponds to a first device public key, and wherein the second device private key corresponds to a second device public key;
 - b) establishing a wireless connection, wherein the wireless connection is encrypted;
 - c) receiving via the wireless connection, (i) a first server public key for a server (ii) a second set of cryptographic parameters, and (iii) a random number;
 - d) authenticating (i) the server using at least the certificate authority public key, and (ii) the device using at least the received random number and the second device private key;
 - e) deriving a first symmetric ciphering key using at least (i) the first device private key and the first server public key, (ii) the ECDH key exchange algorithm, and (iii) the first set of cryptographic parameters;
 - f) deriving a third device private key and a third device public key using the second set of cryptographic parameters;
 - g) generating a device digital signature over at least the third device public key using the second device private key;
 - h) encrypting at least the derived third device public key into a first ciphertext using the derived first symmetric ciphering key;

- i) sending, via the wireless connection, the first ciphertext and the device digital signature to the server; and
- j) receiving, via the wireless connection, a second ciphertext of a first device configuration, wherein the device decrypts the second ciphertext using at least the first symmetric ciphering key.

2. The method of claim 1, wherein the device includes a secure element, and wherein the secure element for the device conducts steps a), d), e), f), g), and h).

3. The method of claim 2, wherein the protected memory comprises a nonvolatile memory within the secure element with a memory address, and wherein a processor for the device is prevented from reading from the memory address.

4. The method of any one of claims 1 to 3, wherein the first device configuration includes (i) an updated configuration for the secure element and (ii) a device certificate for the derived third device public key.

5. The method of claim 4, wherein a boot firmware for the secure element uses the first device private key for the ECDH key exchange algorithm and the second device private key for the digital signature algorithm, wherein the secure element uses the updated configuration with the device certificate.

6. The method of any one of claims 1 to 5, wherein, after a step j), the device authenticates by using the digital signature algorithm with the first device configuration and the derived third device private key.

7. The method of any one of claims 2, 3, 5, and 6, wherein the first device configuration includes a device certificate for the derived third device public key.

8. The method of claim 7, further comprising: verifying, via the device, the device certificate using at least the certificate authority public key.

9. The method of claim 7 or 8, further comprising: storing, via the device, the device certificate in the protected memory, and using, via the device, the device certificate and the derived third device private key in order to authenticate with at least one of (i) an access network, (ii) a reporting system, and (iii) an authentication server.

10. The method of any one of claims 1 to 9, further comprising:

k) receiving via the wireless connection, a second server public key for the second set of cryptographic parameters;

l) deriving a second symmetric ciphering key using (i) the derived third device private key the second server public key, (ii) the ECDH key exchange algorithm, and (iii) the second set of cryptographic parameters; and

m) receiving a third ciphertext of a second device configuration, wherein the second ciphertext is decrypted by the device using the derived second symmetric ciphering key.

11. The method of claim 10, wherein the second device configuration includes a set of network access credentials, and wherein the device authenticates with a wireless access network using the set of network access credentials.

12. The method of claim 10 or 11, wherein the second device configuration includes at least one file for the device, wherein the at least one file is used by the device with at least one of a device operating system, a device reporting application, a secure element firmware, a secure element operating system, a transducer library, and a configuration test vector.

13. The method of claim 12, wherein a secure element boot firmware uses the first device private key for the ECDH key exchange algorithm and the second device private key for the digital signature algorithm, and wherein the device uses the second device configuration with the secure element firmware.

14. The method of any one of claims 2 to 5, wherein the secure element sends and receives through the device using an external bus controller for the secure element, wherein the device and the secure element are connected via a data bus, and wherein the device sends and receives through a radio.

15. The method of any one of claims 1 to 14, wherein the first set of cryptographic parameters specifies at least a first elliptic curve defining equation, and wherein the second set of cryptographic parameters specifies at least a second elliptic curve defining equation.

16. The method of any one of claims 1 to 15, wherein the device establishes the wireless connection with one of (i) a mobile handset through a local wireless connection and (ii) the server through a wide area wireless connection.

17. The method of claim 16, wherein the local wireless connection comprises one of (i) a Wi-Fi connection, (ii) a Bluetooth connection, and (iii) a Near Field Communications (NFC) connection.

18. The method of claim 16, wherein the wide area wireless connection comprises one of (i) a 4G network connection, and (ii) a 5G network connection.

19. The method of any one of claims 1 to 18, wherein the device includes a tag, wherein the tag specifies a uniform resource locator (URL), wherein a configuration system uses the URL in order to receive (i) a device identity from the tag, and (ii) at least one of the first device public key and the second device public key, and wherein the configuration system includes at least the server.

20. The method of claim 19, wherein a mobile handset reads the tag using at least one of a camera and a NFC reader and sends a tag value from the tag to the configuration system.

21. The method of claim 19 or claim 20, wherein the configuration system uses the device identity to query a database for at least the first device public key and the second device public key.

22. The method of any one of claims 1 to 21, further comprising: conducting, via the device, step a) before distributing the device to a user.

23. The method of any one of claims 1 to 22, wherein the device generates the device digital signature over at least the third device public key and the random number using the second device private key.

24. The method of any one of claims 1 to 23, wherein the server derives the first symmetric ciphering key using (i) a first server private key for the first server public key and the first device public key, (ii) the ECDH key exchange algorithm, and (iii) the first set of cryptographic parameters, and wherein the server uses the first symmetric ciphering key to decrypt the first ciphertext and encrypt the second ciphertext.

25. The method of any one of claims 2 to 5, wherein the secure element includes a hardware random number generator, and wherein the hardware random number generator uses at least a transducer measurement in order to generate a device random number, and wherein the secure element uses the device random number to derive the third private key.

26. The method of any one of claims 1 to 25, wherein the device does not send the first and second device public keys.

27. The method of any one of claims 1 to 26, wherein the digital signature algorithm comprises an elliptic curve digital signature algorithm (ECDSA).

28. The method of any one of claims 1 to 27, wherein the device receives via the wireless connection, (i) the first server public key for the server (ii) the second set of cryptographic parameters, and (iii) the random number with a server digital signature, and wherein the device authenticates the server digital signature using at least the certificate authority public key.

29. The method of any one of claims 1 to 28, wherein the first device configuration includes least one of a certificate authority certificate, a transducer configuration, a monitored unit configuration, an identity list, a device configuration, and a reporting system configuration.

30. The method of claim 29, wherein the identity list includes at least one of a networks-available list, a transducer identity, a device owner identity, a device location value, and a monitored unit identity.

31. The method of claim 9, wherein the access network comprises a 5G network supporting an EAP-TLS authentication, and the method further comprising: conducting, via the device, a connection setup with the device certificate and the third device private key.

32. The method of any one of claims 1 to 31, wherein the first device configuration includes a set of network access credentials, and wherein the device authenticates with a wireless access network using the set of network access credentials.

33. The method of any one of claims 1 to 32, wherein the device authenticates the device (i) using the received random number and the second device private key, and (ii) sending the second device public key with an authenticating digital signature.

34. The method of any one of claims 1 to 32, wherein the device authenticates the device (i) using the received random number and the second device private key, and (ii) sending an authenticating digital signature.

35. The method of any one of claims 1 to 34, wherein the device does not use any of the first device private key, the second device private key, and the certificate authority public key in order to establish the encrypted wireless connection.

36. The method of any one of claims 1 to 35, wherein the first set of cryptographic parameters and the second set of cryptographic parameters are the same.

37. The method of any one of claims 1 to 36, wherein the device receives at least one of (i) the second set of cryptographic parameters and (ii) the first server public key, after the device authenticates the device.

38. The method of any one of claims 1 to 37, wherein the first server public key comprises an ephemeral key.

39. The method of any one of claims 1 to 38, wherein the device receives a server certificate in step c), and wherein the device authenticates the server using at least the certificate authority public key by (i) verifying a server digital signature over at least one of the first server public key and the random number using the server certificate, and (ii) verifying the server certificate using the certificate authority public key.

40. The method of any one of claims 1 to 39, wherein the device derives the first device private key after establishing the wireless connection, and wherein the device records the first device private key after deriving the first device private key.

41. The method of any one of claims 1 to 40, wherein the second set of cryptographic parameters specifies values for algorithms associated with a key encapsulation mechanism, and wherein the algorithms comprise one of lattice-based cryptography, code-based cryptography, and supersingular elliptic curve isogeny cryptography.

42. The method of any one of claims 1 to 41, wherein the third device public key supports at least one of lattice-based cryptography, code-based cryptography, supersingular elliptic curve isogeny cryptography, and elliptic curve cryptography.

43. The method of claim 4 or 5, wherein the updated configuration includes at least one file for the secure element, wherein the at least one file is used by the secure element with at least one of a secure element firmware, a secure element operating system, a transducer library, and a configuration test vector.

44. The method of any one of claims 1 to 43, wherein the device receives via the wireless connection, (i) the first server public key for the server (ii) the second set of cryptographic parameters, and (iii) the random number and an authentication token with a server digital signature, wherein the device authenticates the server digital signature using at least the certificate authority public key, wherein the device records the authentication token in a step a), and wherein the device does not send the authentication token via the wireless connection.

45. The method of claim 44, wherein the configuration system receives an identity token for the device before conducting step c), and wherein the configuration system uses the identity token to query a database for the authentication token, and wherein the configuration system generates the server digital signature over at least the authentication token.

46. The method of claim 45, wherein the authentication token comprises at least one of a device identity and an initial random number.

47. The method of any one of claims 1 to 46, wherein the wireless connection is encrypted before the device receives the first server public key and the second set of cryptographic parameters.

48. The method of any one of claims 1 to 47, wherein the device derives the third public key before the device authenticates the server.

49. The method of claim 48, wherein the device generates the device digital signature before the device authenticates the server.

50. The method of any one of claims 1 to 49, wherein the device generates the device digital signature over at least the third device public key using the second device private key and the first set of cryptographic parameters.

51. The method of any one of claims 1 to 49, wherein the device generates the device digital signature over at least the third device public key using the second device private key and the second set of cryptographic parameters.

52. A method for a device to securely receive configuration data, the method performed by the device, the method comprising:

- a) recording in a protected memory, a first device private key for a key exchange algorithm, a second device private key for a digital signature algorithm, a certificate authority public key, and a first set of cryptographic parameters, wherein the first device private key corresponds to a first device public key, and wherein the second device private key corresponds to a second device public key;
- b) establishing a wireless connection, wherein the wireless connection is encrypted;
- c) receiving via the wireless connection, (i) a first server public key for a server (ii) a second set of cryptographic parameters, and (iii) a random number;
- d) authenticating (i) the server using at least the certificate authority public key, and (ii) the device using at least the received random number and the second device private key;
- e) deriving a first symmetric ciphering key using at least (i) the first device private key and the first server public key, (ii) the key exchange algorithm, and (iii) the first set of cryptographic parameters;

- f) deriving a third device private key and a third device public key using the second set of cryptographic parameters;
- g) generating a device digital signature over at least the third device public key using the second device private key;
- h) encrypting at least the derived third device public key into a first ciphertext using the derived first symmetric ciphering key;
- i) sending, via the wireless connection, the first ciphertext and the device digital signature to the server; and
- j) receiving, via the wireless connection, a second ciphertext of a first device configuration, wherein the device decrypts the second ciphertext using at least the first symmetric ciphering key.

53. The method of claim 52 and any one of claims 2 to 4.

54. The method of claim 52, wherein a boot firmware for the secure element uses the first device private key for the key exchange algorithm and the second device private key for the digital signature algorithm, wherein the secure element uses the updated configuration with the device certificate.

55. The method of claim 52, and any one of claims 53 and 54.

56. The method of any one of claims 52 to 54, further comprising:

- k) receiving via the wireless connection, a second server public key for the second set of cryptographic parameters;
- l) deriving a second symmetric ciphering key using (i) the derived third device private key the second server public key, (ii) the key exchange algorithm, and (iii) the second set of cryptographic parameters; and,

m) receiving a third ciphertext of a second device configuration, wherein the second ciphertext is decrypted by the device using the derived second symmetric ciphering key.

57. The method of claim 52, and any one of claims 11 to 23.

58. The method of any one of claims 52 to 57, wherein the server derives the first symmetric ciphering key using (i) a first server private key and the first device public key, (ii) the key exchange algorithm, and (iii) the first set of cryptographic parameters, and wherein the server uses the first symmetric ciphering key to decrypt the first ciphertext and encrypt the second ciphertext;

59. The method of claim 52, and any one of claims 25 to 51.

60. A method for a configuration system to securely send configuration data to a device, the method performed by the configuration system, the method comprising:

- a) storing a first server private key for an elliptic curve Diffie-Hellman (ECDH) key exchange algorithm, a second server private key for a digital signature algorithm, a first set of cryptographic parameters, wherein the first server private key corresponds to a first server public key, and wherein the second server private key corresponds to a second server public key;
- b) receiving an identity token for a device, a first device public key for the ECDH key exchange algorithm, and a second device public key for the digital signature algorithm;
- c) sending to the device (i) the first server public key for a server (ii) a second set of cryptographic parameters, and (iii) a random number;
- d) authenticating (i) the server with the device using at least the second server private key, and (ii) the device with the server using at least the second device public key;

- e) deriving a first symmetric ciphering key using (i) the first device public key and the first server private key, (ii) the ECDH key exchange algorithm, and (iii) the first set of cryptographic parameters;
- f) receiving, from the device, (i) a first ciphertext of a third device public key and (ii) a device digital signature for at least the third device public key, wherein the third device public key supports the second set of cryptographic parameters;
- g) decrypting the first ciphertext using the derived first symmetric ciphering key in order to read the plaintext third device public key;
- h) verifying the device digital signature using at least the second device public key;
- i) selecting first configuration data for the device using at least the identity token;
- j) encrypting a second ciphertext of at least the first configuration data using the derived first symmetric ciphering key; and
- k) sending, to the device, the second ciphertext.

61. The method of claim 60, further comprising:

- l) generating, before encrypting the second ciphertext, a device certificate for the third device public key, wherein the device certificate includes a certificate authority digital signature, and wherein the first configuration data includes the device certificate.

62. The method of claim 61, wherein, after a step l), the configuration system authenticates the device using at least the device certificate.

63. The method of claim 60, further comprising:

- l) deriving a third server private key and a third server public key for the second set of cryptographic parameters;
- m) sending, to the device, the third server public key;
- n) deriving a second symmetric ciphering key using (i) the derived third server private key and the third device public key, (ii) the ECDH key exchange algorithm, and (iii) the second set of cryptographic parameters;
- o) encrypting a third ciphertext of second configuration data, wherein the second ciphertext is encrypted using the derived second symmetric ciphering key; and
- p) sending, to the device, the third ciphertext.

64. The method of claim 63, further comprising:

- q) generating, before encrypting the third ciphertext, a device certificate for the third device public key, wherein the device certificate includes a certificate authority digital signature, and wherein the second configuration data includes the device certificate.

65. The method of claim 63 or 64, wherein the second configuration data includes a set of network access credentials, and wherein, after a step p), an access network authenticates the device using the network access credentials.

66. The method of any one of claims 63 to 65, wherein the second configuration data includes at least one file for the device, wherein the at least one file is used by the device with at least one of a device operating system, a device reporting application, a secure element firmware, a secure element operating system, a transducer library, and a configuration test vector.

67. The method of any one of claims 60 to 66, wherein the first set of cryptographic parameters specifies at least a first elliptic curve defining equation, and wherein the second set of cryptographic parameters specifies at least a second elliptic curve defining equation.

68. The method of any one of claims 60 to 67, wherein the device includes a tag, wherein the tag specifies a uniform resource locator (URL), wherein the configuration system uses the URL in order to receive the identity token for the device.

69. The method of claim 68, wherein a mobile handset reads the tag using at least one of a camera and a NFC reader and the configuration system receives from the mobile handset a tag value from the tag.

70. The method of any one of claims 60 to 69, wherein the configuration system receives the identity token from the device.

71. The method of claim 70, wherein the configuration system uses the identity token to query a database for the first device public key and the second device public key, wherein the configuration system receives the first device public key and the second device public key from the database, and wherein the device does not send first device public key and the second device public key.

72. The method of claim 71, wherein the configuration system includes the database, and wherein the database stores the first device public key and the second device public key before the configuration system receives the identity token for the device.

73. The method of claim 70, wherein the configuration system receives from the device at least one of the first device public key, the second device public key, and the first set of cryptographic parameters.

74. The method of any one of claims 60 to 73, wherein the identity token for the device comprises one of (i) a device identity, or (ii) an obfuscated device identity.

75. The method of any one of claims 60 to 74, wherein the configuration system, after receiving the identity token, (i) derives the first server public key and first server private key, and (ii) stores the first server public key and first server private key, wherein the first server private key comprises an ephemeral key, and wherein the first server public key comprises an ephemeral key.

76. The method of any one of claims 60 to 75, wherein the configuration system includes (1) a discovery server for at least receiving the identity token for the device, an authentication server for authenticating (i) the server with the device and (ii) the device with the server, and (2) the server for at least (i) deriving the first symmetric ciphering key, (ii) receiving the first ciphertext, and (iii) sending the second ciphertext.

77. The method of any one of claims 60 to 76, wherein authenticating the server with the device comprises (1) using the second server private key and the digital signature algorithm to generate a server digital signature over at least one of (i) the first server public key, (ii) identity token, and (iii) the random number, and (2) sending to the device the server digital signature.

78. The method of any one of claims 60 to 77, wherein, before deriving the first symmetric ciphering key, the configuration system sends to the device the second server public key in a server certificate, and wherein the device verifies the server certificate with a certificate authority public key recorded in the device.

79. The method of any one of claims 60 to 77, wherein, before the device communicates with the configuration system, the device records the second server public key.

80. The method of any one of claims 60 to 79, wherein authenticating the device with the server comprises (i) receiving from the device an authentication digital signature over at least the random number, and (ii) verifying the authentication digital signature using at least the second device public key.

81. The method of any one of claims 60 to 80, wherein the device derives the third device public key after receiving the second set of cryptographic parameters.

82. The method of any one of claims 60 to 81, wherein the configuration system receives from the device the device digital signature over at least the third device public key and the random number.

83. The method of any one of claims 60 to 82, wherein the first ciphertext includes the device digital signature.

84. The method of any one of claims 60 to 83, wherein the device derives the first symmetric ciphering key using (i) the first server public key and a first device private key, (ii) the ECDH key exchange algorithm, and (iii) the first set of cryptographic parameters, and wherein the device uses the first symmetric ciphering key to encrypt the first ciphertext and decrypt the second ciphertext.

85. The method of any one of claims 60 to 84, wherein the digital signature algorithm comprises an elliptic curve digital signature algorithm (ECDSA).

86. The method of any one of claims 60 to 85, wherein the configuration system queries at least one database using the identity token in order to select the first configuration data.

87. The method of any one of claims 60 to 86, wherein the first configuration data includes least one of a certificate authority certificate, a transducer configuration, a monitored unit configuration, an identity list, a device configuration, and a reporting system configuration.

88. The method of claim 87, wherein the identity list includes at least one of a networks available list, a transducer identity, a device owner identity, a device location value, and a monitored unit identity.

89. The method of claim 61 or 64, wherein an access network associated with the configuration system supports an EAP-TLS authentication, and wherein the access network conducts a connection setup with the device using the device certificate for the third device public key.

90. The method of claim 89, wherein the access network comprises a 5G wireless network.

91. The method of any one of claims 60 to 88, wherein the first configuration data includes a set of network access credentials, and wherein an access network associated with the configuration system authenticates the device using the network access credentials.

92. The method of any one of claims 60 to 91, wherein the first set of cryptographic parameters and the second set of cryptographic parameters are the same.

93. The method of any one of claims 60 to 92, wherein the configuration system sends to the device at least one of (i) the second set of cryptographic parameters and (ii) the first server public key after the configuration system authenticates the device with the server.

94. The method of claim 60, further comprising:

- l) deriving a third server private key and a third server public key for the first set of cryptographic parameters;
 - m) sending, to the device, the third server public key;
 - n) deriving a second symmetric ciphering key using (i) the derived third server private key and the first device public key, (ii) the ECDH key exchange algorithm, and (iii) the first set of cryptographic parameters;
 - o) encrypting a third ciphertext of second configuration data, wherein the second ciphertext is encrypted using the derived second symmetric ciphering key;
- and

p) sending, to the device, the third ciphertext.

95. The method of claim 94, wherein the second configuration data includes a set of network access credentials, and wherein, after a step p), an access network authenticates the device using the set of network access credentials.

96. The method of claim 94 or 95, wherein the second configuration data includes at least one file for the device, wherein the at least one file is used by the device with at least one of a device operating system, a device reporting application, a secure element firmware.

97. The method of claim 94 and any one of claims 64 to 93.

98. The method of any one of claims 60 to 97, wherein the configuration system sends the third device public key to an access network, wherein the access network (i) encrypts a second symmetric ciphering key with the third device public key, (ii) encrypts a set of network access credentials with the second symmetric ciphering key, and (iii) the configuration system receives the encrypted set of network access credentials, wherein the configuration system includes the encrypted set of network access credentials in the second ciphertext, wherein the device decrypts the encrypted set of network access credentials with a private key corresponding to the third device public key, and wherein the access network authenticates the device using the set of network access credentials.

99. The method of any one of claims 60 to 98, wherein the configuration system sends the first server public key after authenticating the device with the server.

100. The method of any one of claims 60 to 99, wherein the configuration system receives the second device public key before storing the first server private key.

101. The method of any one of claims 60 to 100, wherein the configuration system selects the first configuration data before sending to the device the first server public key.

102. The method of any one of claims 60 to 101, wherein the configuration system receives the random number from the device, and wherein the configuration system receives the random number with the identity token.

103. The method of any one of claims 60 to 101, wherein the configuration system generates the random number.

104. The method of any one of claims 60 to 103, wherein the configuration system derives the first server private key and stores the first server private key before the configuration system authenticates the server with the device.

105. The method of any one of claims 60 to 104, wherein the configuration system derives the first server private key and stores the first server private key after the configuration system authenticates the server with the device.

106. The method of any one of claims 60 to 105, wherein the configuration system receives at least one of the first device public key and the second device public key after sending to the device at least one of (i) the first server public key, (ii) the second set of cryptographic parameters, and (iii) the random number.

107. A method for a mobile handset to configure a device, the method performed by the mobile handset, the method comprising:

- a) reading a tag value for the device, wherein the tag value includes an identity token for the device;
- b) establishing a secure session with a configuration system using at least the tag value;
- c) sending the identity token to the configuration system;
- c) establishing a local wireless connection with the device;

- d) receiving from the configuration system and sending to the device: a first random number, a first certificate for an authentication server, and a first digital signature over at least the first random number and an authentication token;
- e) receiving from the device and sending to the configuration system: a second random number and a second digital signature over at least the first random number;
- f) receiving from the configuration system and sending to the device a first public key of the configuration system, a set of cryptographic parameters, a second certificate for the configuration system, and a third digital signature over at least the second random number;
- g) receiving from the device and sending to the configuration system, a first ciphertext, wherein the first ciphertext is encrypted using at least the first public key; and
- h) receiving from the configuration system and sending to the device, a second ciphertext of a device configuration for the device.

108. The method of claim 107, wherein the mobile handset uses at least one of a camera and a NFC reader to read the tag value, and wherein the tag value includes at least a uniform resource locator (URL) for the device, and wherein the mobile handset establishes the secure session with the configuration system using at least the URL.

109. The method of claim 107, wherein the mobile handset reads the tag value from input by a user of the mobile handset.

110. The method of any one of claims 107 to 109, wherein the local wireless connection comprises one of (i) a Wi-Fi connection, (ii) a Bluetooth connection, and (iii) a Near Field Communications (NFC) connection.

111. The method of claim 107, wherein the mobile handset establishes the local wireless connection before reading the tag value, and wherein the mobile handset receives the tag value through the local wireless connection, and wherein the mobile handset reads the tag value from a data packet received through the local wireless connection.

112. The method of claim 107, wherein the mobile handset establishes the local wireless connection before the mobile handset establishes the secure session.

113. The method of any one of claims 107 to 112, wherein the mobile handset establishes the secure session over one of (i) a wide area wireless connection, and where the wide area wireless connection comprises at least one of a 4G network connection and a 5G network connection, and (ii) a Wi-Fi connection with connectivity to the Internet.

114. The method of claim 113, wherein the secure session uses at least one of (i) transport layer security (TLS), datagram transport layer security (DTLS), Internet Protocol Security (IPSec), a virtual private network (VPN), and a secure shell (SSH).

115. The method of any one of claims 107 to 114, wherein a tag includes the tag value, and wherein the tag includes the identity token for the device, and wherein the mobile handset reads the tag.

116. The method of any one of claims 107 to 115, wherein the mobile handset sends the identity token to the configuration system through the secure session.

117. The method of any one of claims 107 to 116, wherein the mobile handset receives the first random number from the device and sends the first random number to the configuration system, before the mobile handset receives the first digital signature from the configuration system.

118. The method of any one of claims 107 to 117, wherein the device verifies the first digital signature using at least the first certificate, and wherein the device records the authentication token before the local wireless connection is established.

119. The method of any one of claims 107 to 118, wherein the mobile handset does not receive the first random number from the device before receiving the second digital signature from the device, wherein the device verifies the first digital signature using at least the first certificate before the mobile handset receives the second digital signature, and wherein the device records the authentication token.

120. The method of any one of claims 107 to 119, wherein the configuration system uses the identity token for the device to query a database for a first device public key and the authentication token, and wherein the configuration system uses the first device public key to verify the second digital signature.

121. The method of claim 120, wherein the first device public key is recorded in a first device certificate.

122. The method of claim 120, wherein the mobile handset does not receive the first device public key from the device.

123. The method of any one of claims 1 to 118, wherein the mobile handset receives a first device public key from the device via the local wireless connection, and wherein the mobile handset sends the first device public key to the configuration system before receiving the second ciphertext.

124. The method of claim 123, wherein the configuration system uses the first device public key to verify the second digital signature.

125. The method of any one of claims 107 to 124, wherein the identity token for the device and the authentication token are different, and wherein the authentication token is recorded by the device before step b), and wherein the configuration system uses the identity token to query a database for the device value.

126. The method of any one of claims 107 to 124, wherein the identity token for the device and the authentication token are the same.

127. The method of any one of claims 107 to 126, wherein third digital signature is over the second random number and at least one of the first public key and the set of cryptographic parameters.

128. The method of any one of claims 107 to 127, wherein the device verifies the third digital signature using at least the second certificate for the configuration system.

129. The method of any one of claims 107 to 128, wherein the first ciphertext includes a second public key derived by the device, and wherein the device configuration in the second ciphertext includes a device certificate for the second public key.

130. The method of claim 120 or 123, wherein the mobile handset receives from the device the first ciphertext with a fourth digital signature, and wherein the configuration system verifies the fourth digital signature using at least the first device public key.

131. The method of claim 130, wherein the first ciphertext includes a second public key derived by the device, and wherein the device configuration in the second ciphertext includes a device certificate for the second public key.

132. The method of claim 131, wherein the device authenticates with a reporting system using the device certificate.

133. The method of claim 131, wherein the device authenticates with an access network using the device certificate

134. The method of any one of claims 107 to 133, wherein the device mutually derives a symmetric ciphering key with the configuration system using the first public key and a device private key, and wherein the configuration system mutually derives the symmetric ciphering key and decrypts the first ciphertext using the symmetric ciphering key.

135. The method of claim 134, wherein the device encrypts the first ciphertext using the symmetric ciphering key and the configuration system decrypts the first ciphertext using the symmetric ciphering key, and wherein the configuration system encrypts the second ciphertext using the symmetric ciphering key and the device decrypts the second ciphertext using the symmetric ciphering key.

136. The method of claim 134 or 135, wherein the device mutually derives the symmetric ciphering key using an elliptic curve Diffie-Hellman (ECDH) key exchange and the set of cryptographic parameters.

137. The method of any one of claims 107 to 136, wherein the device configuration includes a set of network access credentials, and wherein, after a step h), an access network authenticates the device using the network access credentials.

138. The method of any one of claims 107 to 137, wherein the device configuration includes at least one file for the device, wherein the file is used by the device with at least one of a device operating system, a device reporting application, a secure element firmware, a secure element operating system, a transducer library, and a configuration test vector

139. The method of any one of claims 107 to 138, wherein the device configuration includes least one of a certificate authority certificate, a transducer configuration, a monitored unit configuration, an identity list, a device configuration, and a reporting system configuration.

140. The method of claim 139, wherein the identity list includes at least one of a networks available list, a transducer identity, a device owner identity, a device location value, and a monitored unit identity.

141. The method of any one of claims 107 to 140, wherein the first ciphertext includes an initial device configuration, wherein the device updates the initial device configuration using the device configuration in the second ciphertext.

142. The method of claim 141, wherein the configuration system uses the initial device configuration in order to select the device configuration.

143. The method of any one of claim 107 to 142, wherein the device records the authentication token before a step b).

Figure 1a

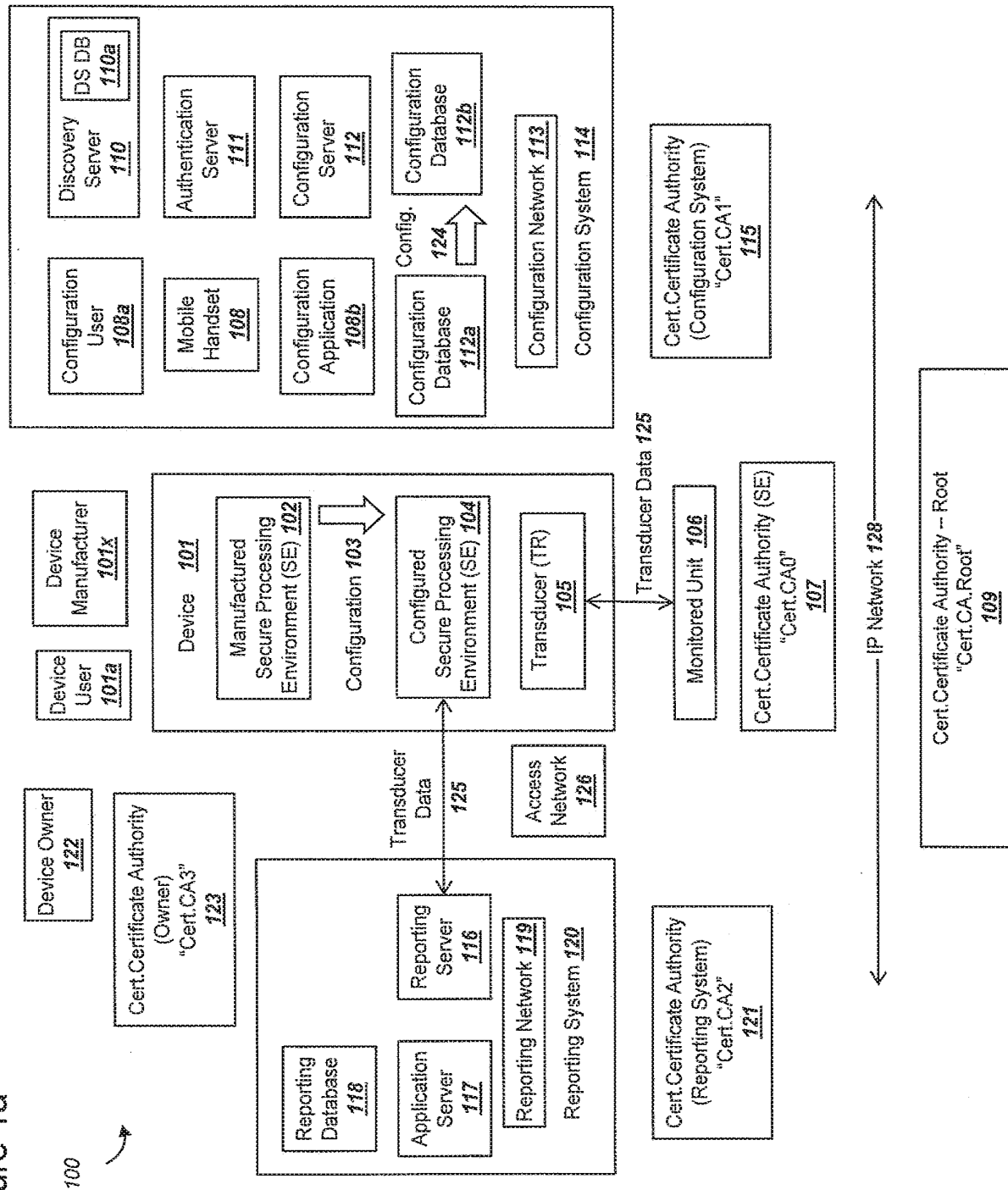
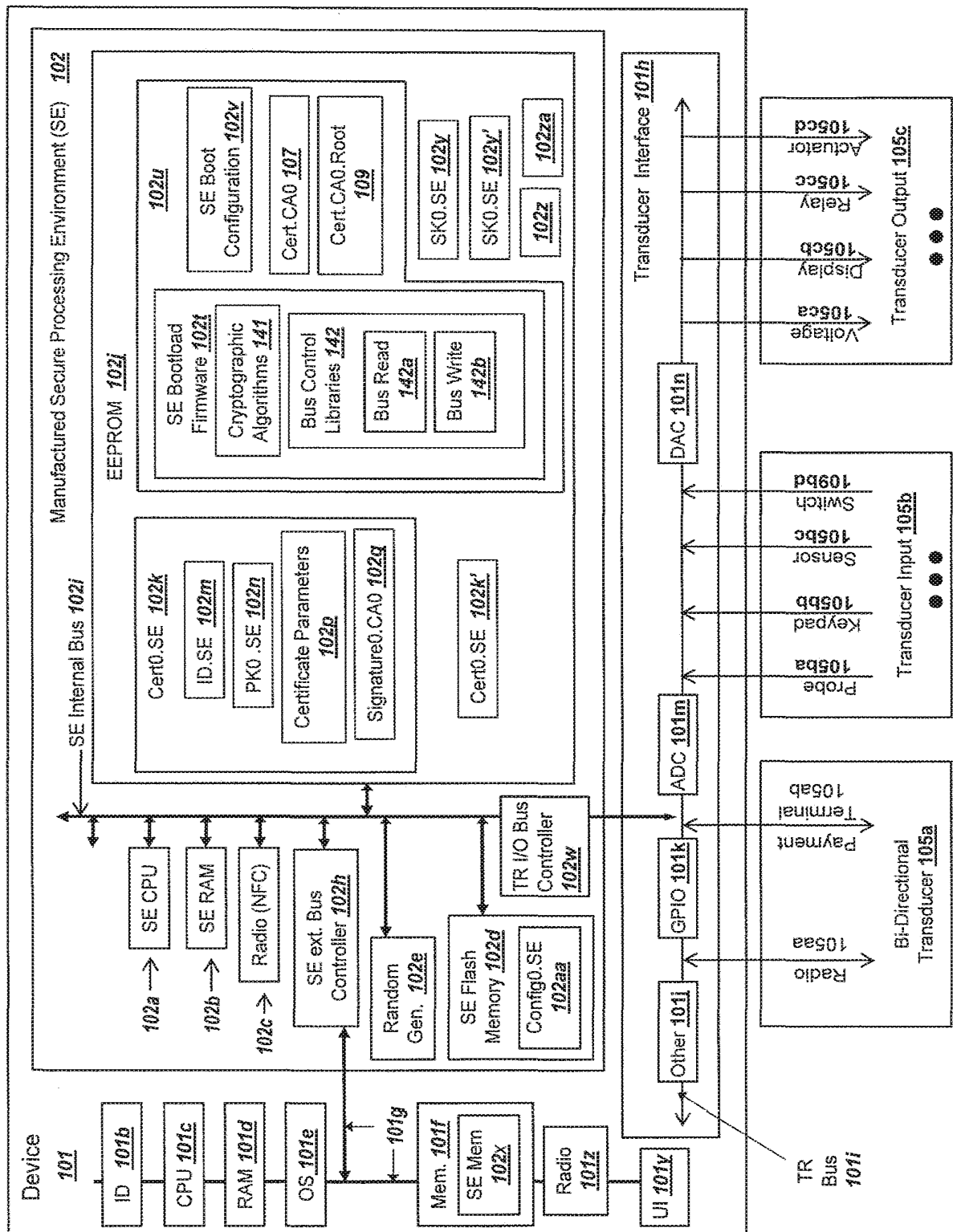


Figure 1b



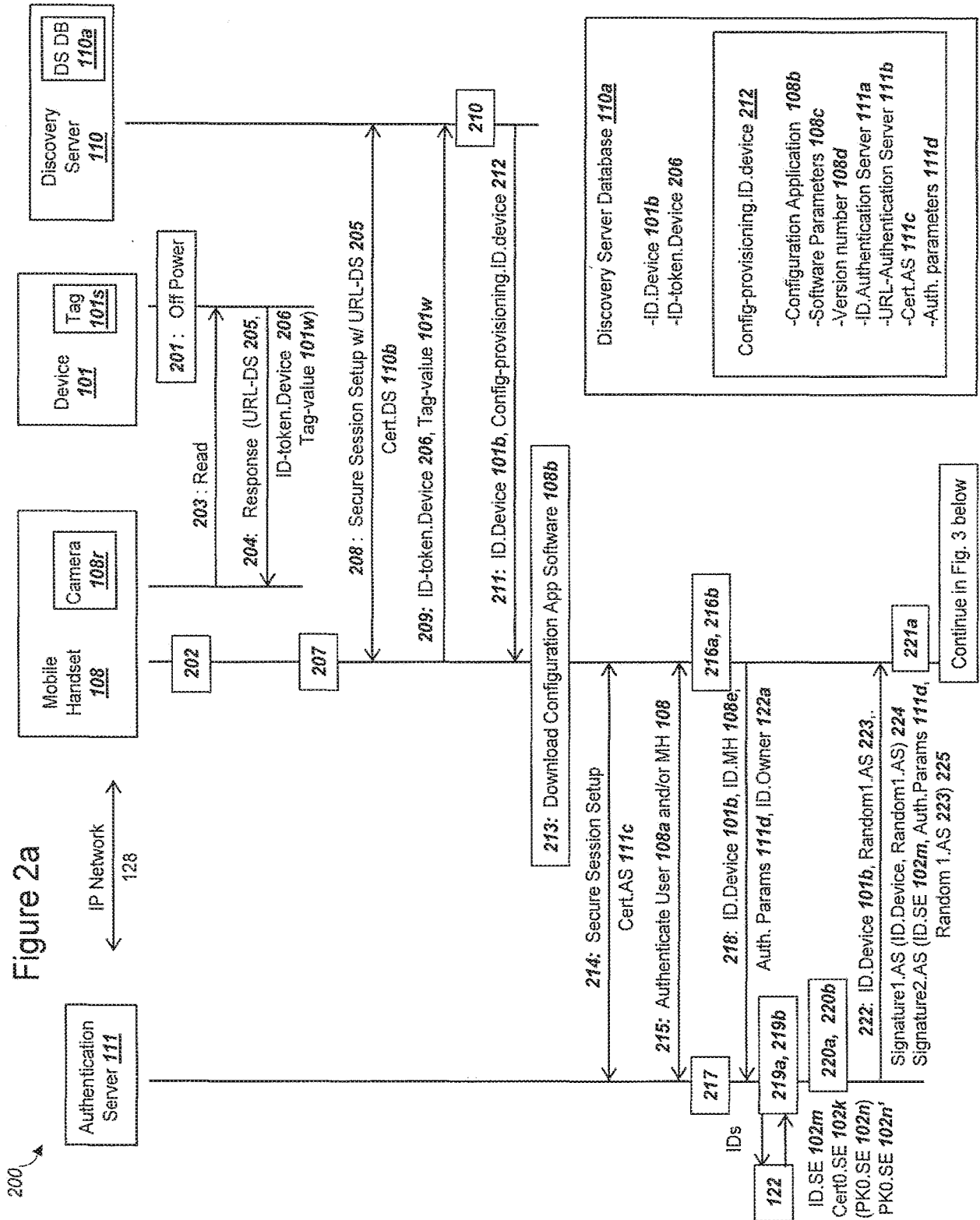


Figure 2b

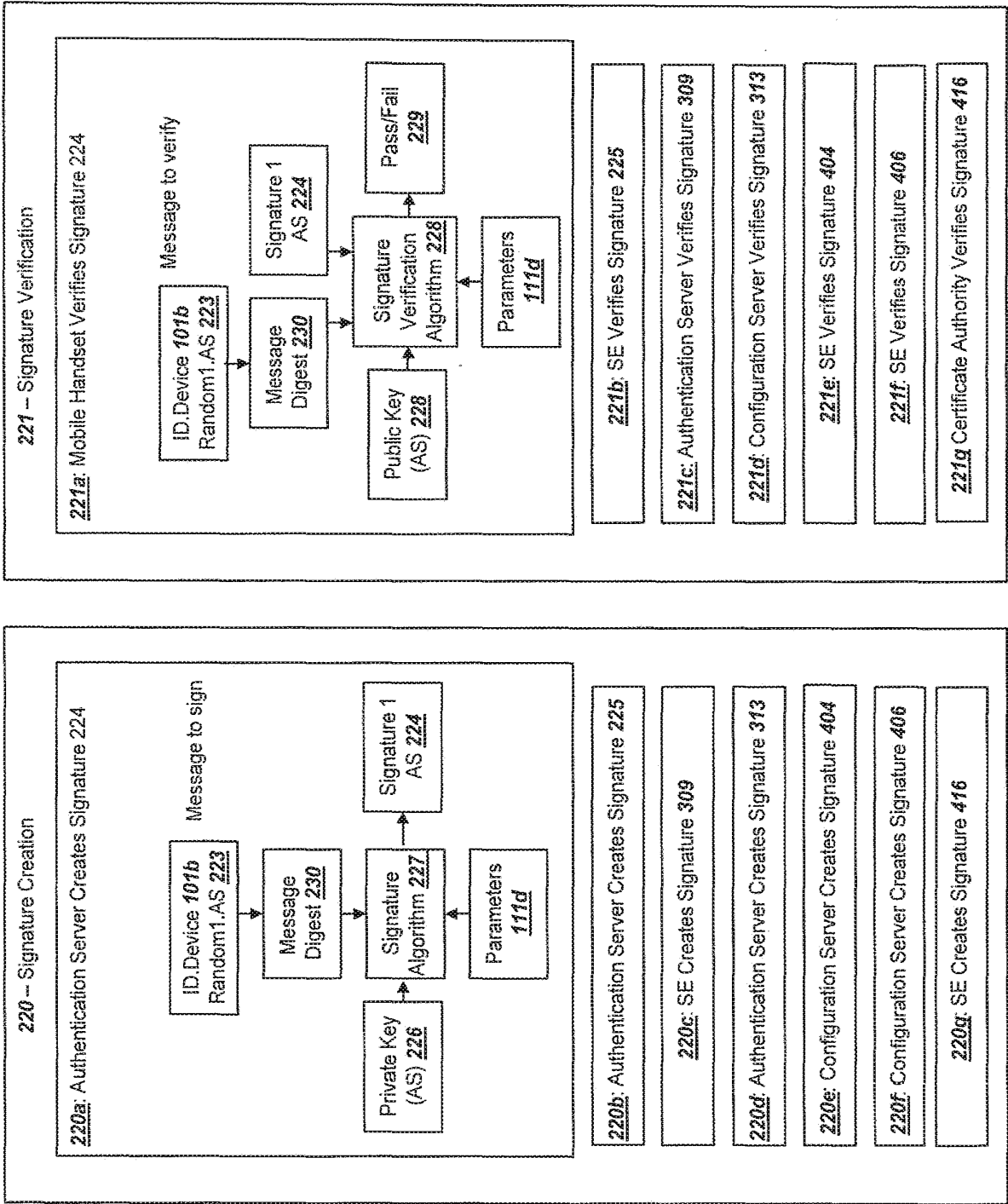
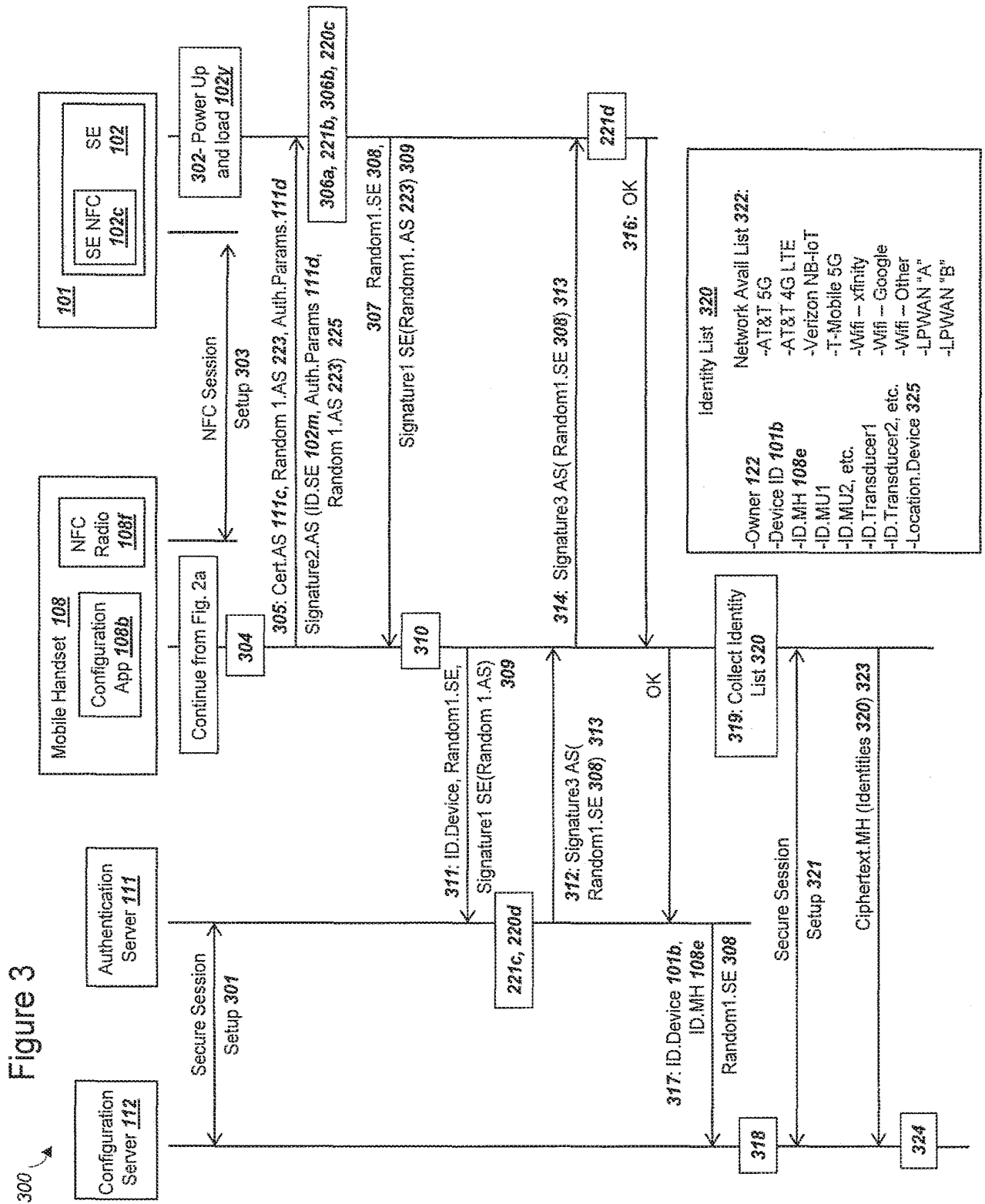


Figure 3



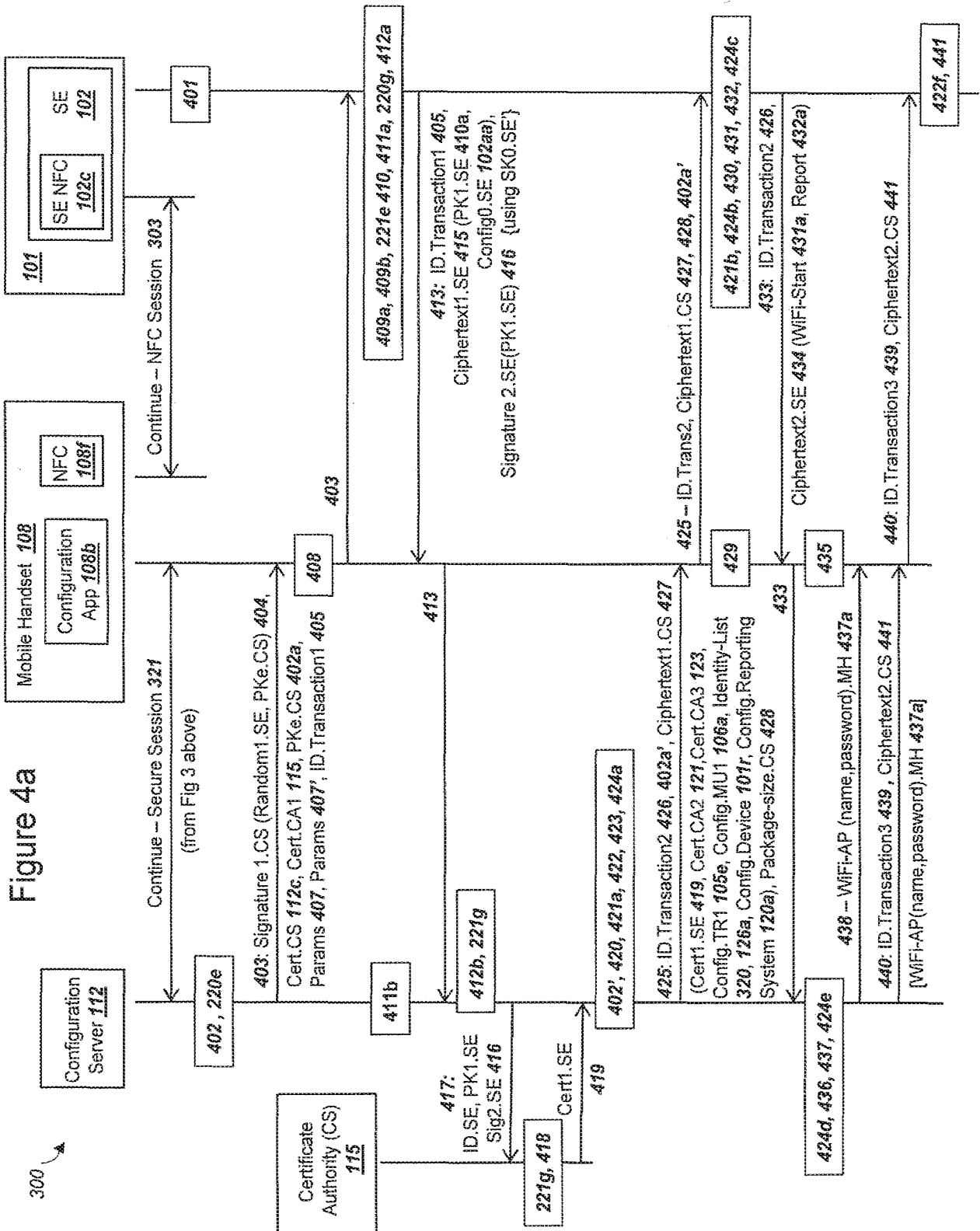


Figure 4b

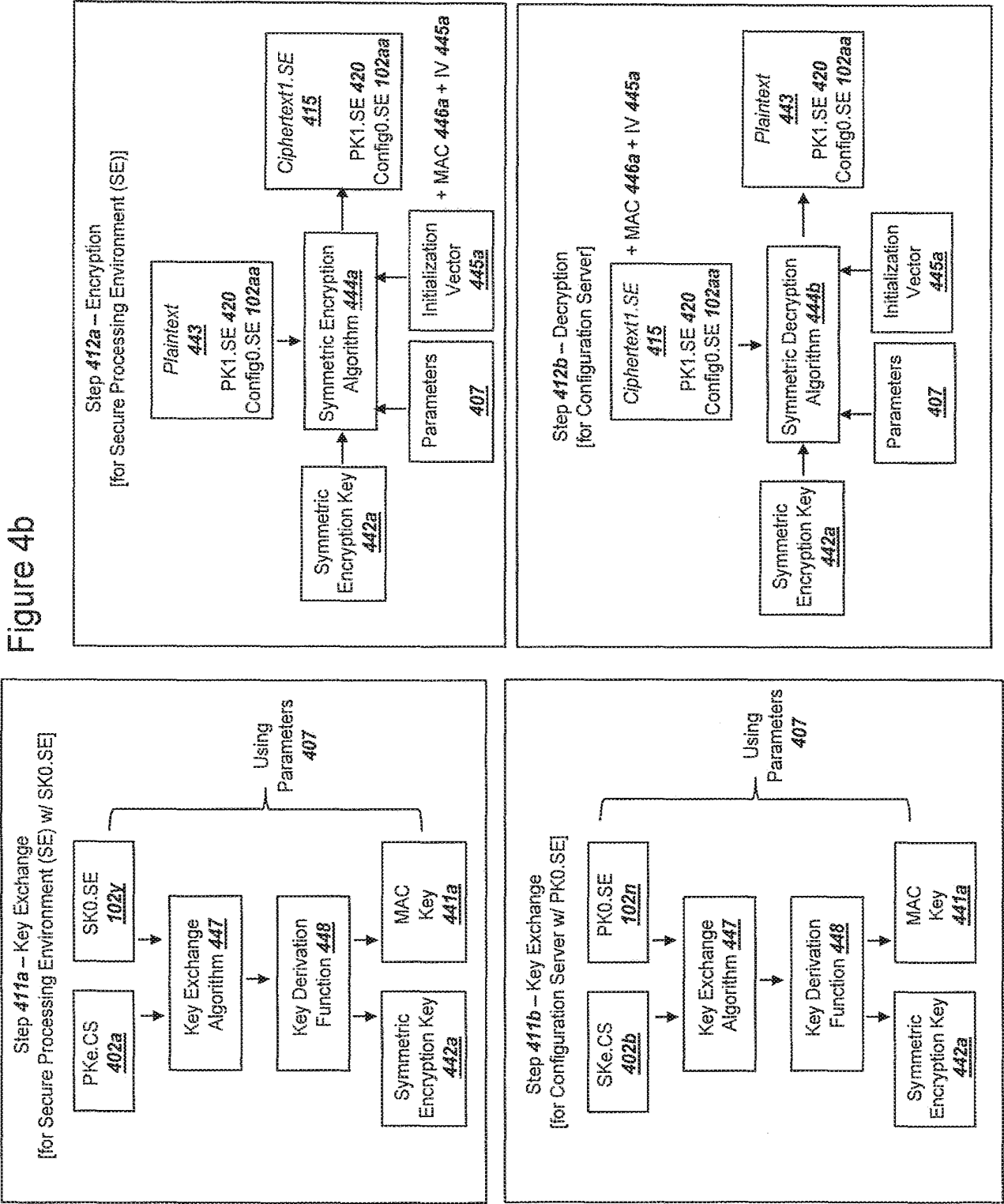


Figure 4c

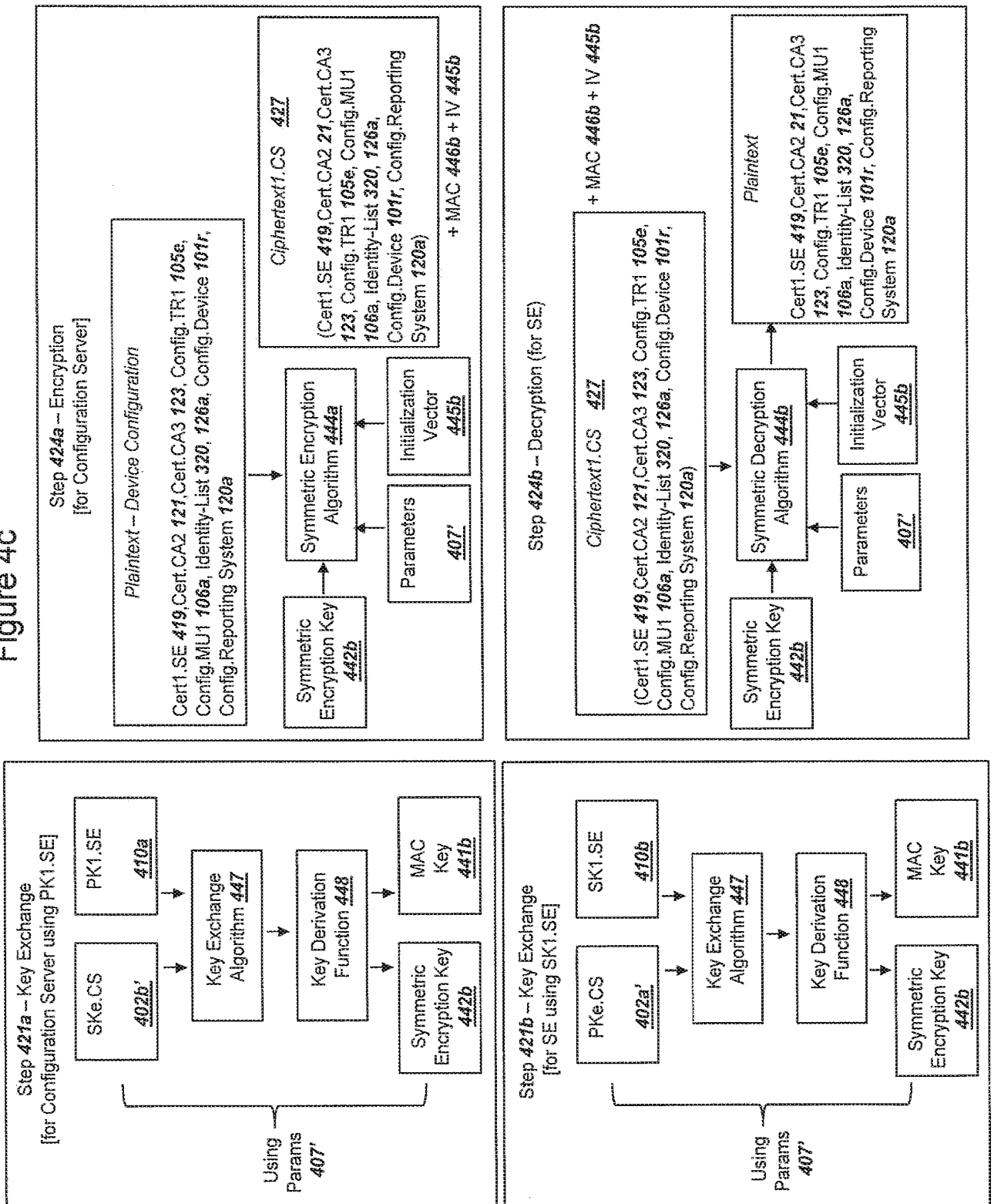


Figure 4d

SE Certificate cert1.SE **419** (with PK1.SE and Signed by CA1 115)

256 bit Elliptic Curve Key with SHA 256 Signature ("... " is long text removed)

```

Version: 3 (0x2)
Serial Number: 20486 (0x5006)
407 → Signature Algorithm: ecdsa-with-SHA256
Issuer: C=US, ST=Illinois, L=Chicago, O=Configuration System 114, ← 114
115 → OU=CA 115, CN=Certificate Authority 115 for Configuration System 114
/emailAddress=ca-115@configuration-system-114.org
Validity
407' → Not Before: Aug 31 21:15:16 2017 GMT
Not After: Aug 30 21:15:16 2022 GMT
Subject: C=US, ST=Illinois, L=Chicago, O=Device Manufacturer 101x,
O=JNIX, OU=Secure Processing Environment Type-Model,
102m → CN=SE ID 102m (Plus Optional Certificate Sequence Number)
Subject Public Key Info:
Public Key Algorithm: id-ecDH
Public Key: (256 bit) ← 407'
410a { 00:c6:f2:1f:b0:ec:78:ed:d5:5a:f9:c2:2b:03:54
...
39:05:82:41:6a:43:95:99:da:19:2e
407' → ASN1 OID: secp256r1
NIST CURVE: P-256
X509v3 extensions:
X509v3 Subject Key Identifier:
F9:03:6E:48:99:CE:04:AB:1C:6B:18:B1:CA:B2:3B:6A:19:4B:81:8D
X509v3 Authority Key Identifier:
keyid:46:AC:7F:83:C9:27: ... :A0:B8:E5:B3:1A:3A:48
DirName:/C=US/ST=Illinois/L=Chicago/O=Configuration System 114
/OU=CA 115/CN=Certificate Authority 115 for Configuration System 114
/emailAddress=ca-115@configuration-system-114.org
serial:02
X509v3 Key Usage:
Digital Signature, Non Repudiation, Key Encipherment, Data Encipherment
X509v3 CRL Distribution Points:
URI:http://www.ca-115-for-configuration-system-114.org/ca/crl/ref_ca.crl
Signature Algorithm: ecdsa-with-SHA256
220 { 26:35:bb:aa:ff:29:7d:ee:3d:8d:e4:a3:b5:14:94:4d:13:ee:
...
c8:ba:f2:38:18:7d:81:0a:66:1e:8c:19:2a:cc:e5:ae:9c:32:
32:c3:5e:4d
-----BEGIN CERTIFICATE-----
MIIGMzCCBRugAwIBAgICUAYwDQYJKoZIhvcNEFBQAwwgaoxCzAJBgNVBAYTAIVT
...
5Ww3GBO1SxbgDihKnSGUAMXWY1laLXJSiKEGZSBmBBYlryOBh9gQpmHowZKszl
rpwyMsNeTQ==
-----END CERTIFICATE-----

```

Figure 5

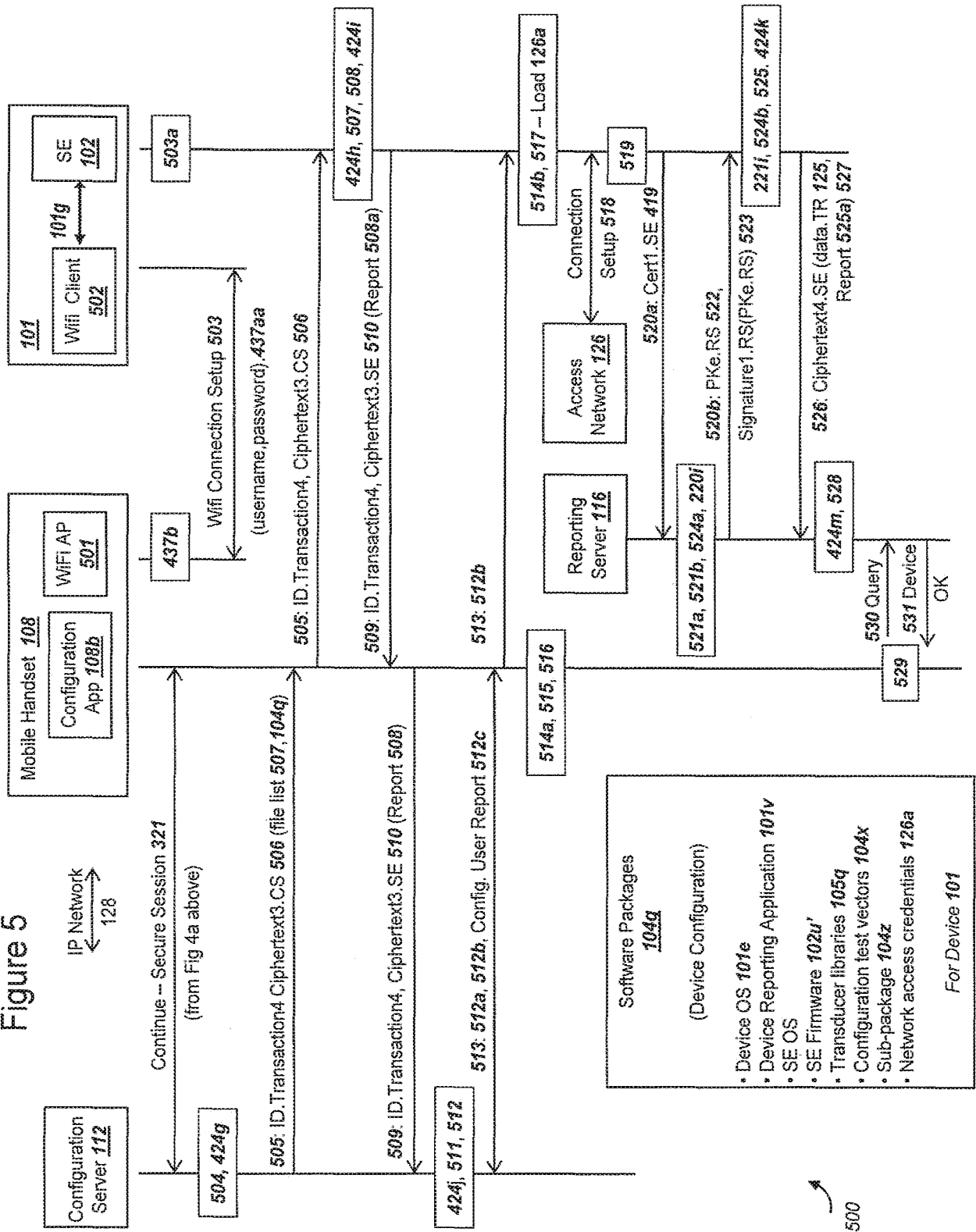


Figure 6a

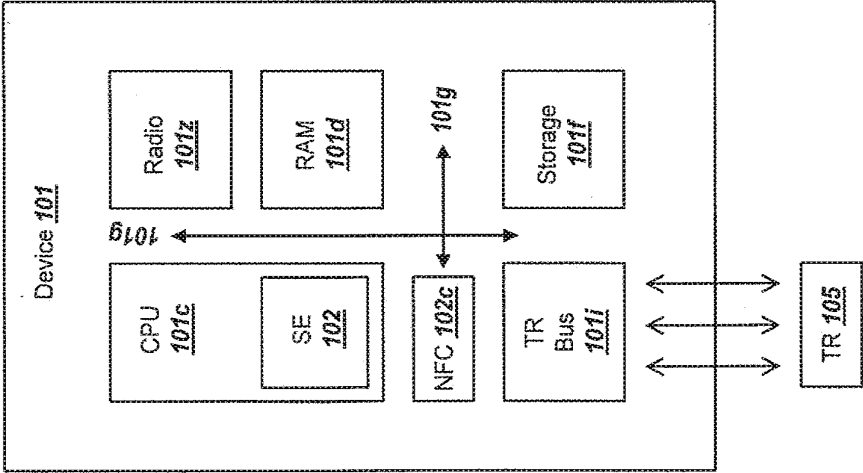
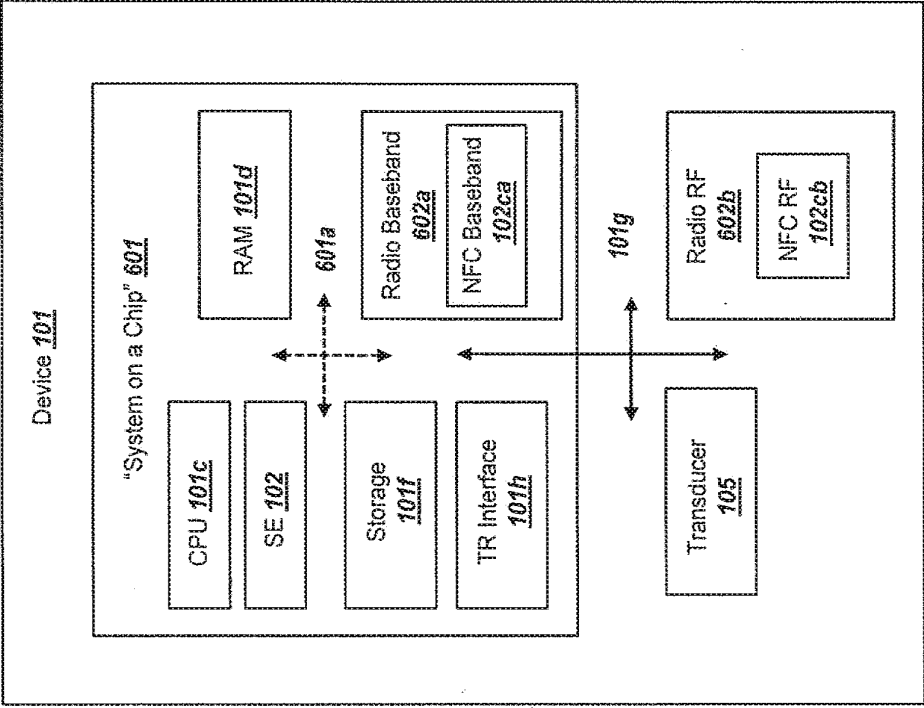


Figure 6b



INTERNATIONAL SEARCH REPORT

International application No

PCT/US2019/022184

A. CLASSIFICATION OF SUBJECT MATTER

INV. H04L29/06 H04L29/08 H04W12/04
ADD.

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

H04L H04W

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

EPO-Internal, COMPENDEX, INSPEC, IBM-TDB, WPI Data

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	US 2017/339120 A1 (ZAKARIA OMAR [US] ET AL) 23 November 2017 (2017-11-23)	107-143
A	abstract	1-106
	paragraph [0035]	
	paragraph [0055] - paragraph [0056]	
	paragraph [0104] - paragraph [0114]	
	figures 1, 11, 12A, 12B	

A	US 2017/373845 A1 (NIX JOHN A [US]) 28 December 2017 (2017-12-28)	1-143
	the whole document	



Further documents are listed in the continuation of Box C.



See patent family annex.

* Special categories of cited documents :

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier application or patent but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art

"&" document member of the same patent family

Date of the actual completion of the international search

16 May 2019

Date of mailing of the international search report

27/05/2019

Name and mailing address of the ISA/

European Patent Office, P.B. 5818 Patentlaan 2
NL - 2280 HV Rijswijk
Tel. (+31-70) 340-2040,
Fax: (+31-70) 340-3016

Authorized officer

Horn, Marc-Philipp

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No

PCT/US2019/022184

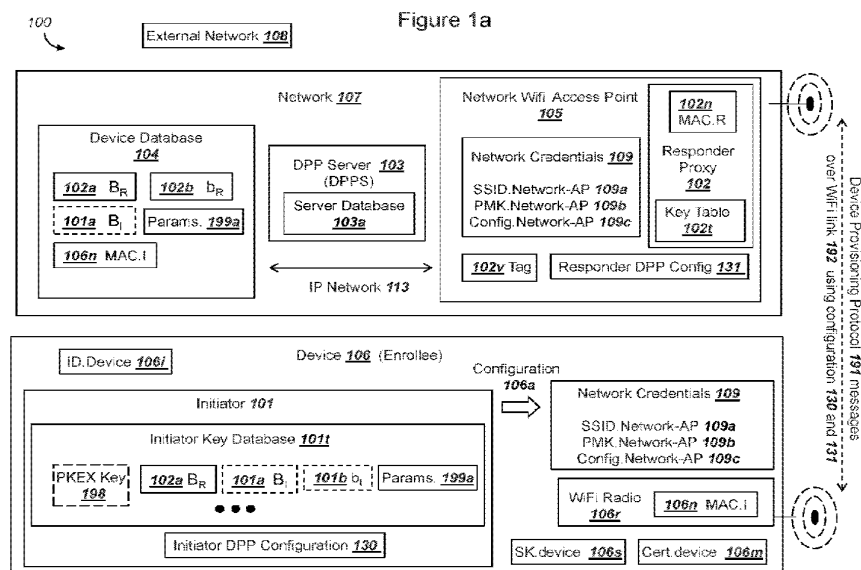
Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 2017339120 A1	23-11-2017	US 2017006003 A1	05-01-2017
		US 2017339120 A1	23-11-2017

US 2017373845 A1	28-12-2017	AU 2014342646 A1	19-05-2016
		CA 2965119 A1	07-05-2015
		GB 2534801 A	03-08-2016
		US 2015071139 A1	12-03-2015
		US 2015095648 A1	02-04-2015
		US 2015106616 A1	16-04-2015
		US 2015121066 A1	30-04-2015
		US 2015143125 A1	21-05-2015
		US 2015180653 A1	25-06-2015
		US 2015304113 A1	22-10-2015
		US 2016149709 A1	26-05-2016
		US 2016164678 A1	09-06-2016
		US 2016234020 A1	11-08-2016
		US 2016270000 A1	15-09-2016
		US 2017188231 A1	29-06-2017
		US 2017237561 A1	17-08-2017
		US 2017302447 A1	19-10-2017
		US 2017373845 A1	28-12-2017
		US 2018254897 A1	06-09-2018
		US 2018262329 A1	13-09-2018
		US 2018270059 A1	20-09-2018
		US 2018343117 A1	29-11-2018
		WO 2015065913 A1	07-05-2015



- (51) International Patent Classification:
G06F 21/30 (2013.01) *G06F 21/62* (2013.01)
- (21) International Application Number:
PCT/US2019/032371
- (22) International Filing Date:
15 May 2019 (15.05.2019)
- (25) Filing Language: English
- (26) Publication Language: English
- (30) Priority Data:
62/672,977 17 May 2018 (17.05.2018) US
- (71) Applicant: **IoT AND M2M TECHNOLOGIES, LLC**
[US/US]; 807 Davis Street, Unit 2207, Evanston, IL
60201-7104 (US).
- (72) Inventor: **NIX, John, A.**; c/o IoT And M2M Technologies, LLC, 807 Davis Street, Unit 2207, Evanston, IL
60201-7104 (US).
- (74) Agent: **THOMSON, Kirsten, L.**; McDonnell Boehnen
Hulbert & Berghoff LLP, 300 South Wacker Drive, Chicago,
IL 60606 (US).
- (81) Designated States (*unless otherwise indicated, for every
kind of national protection available*): AE, AG, AL, AM,
AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ,
CA, CH, CL, CN, CO, CR, CU, CZ, DE, DJ, DK, DM, DO,
DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN,
HR, HU, ID, IL, IN, IR, IS, JO, JP, KE, KG, KH, KN, KP,
KR, KW, KZ, LA, LC, LK, LR, LS, LU, LY, MA, MD, ME,
MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ,
OM, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SA,

(54) Title: A HOSTED DYNAMIC PROVISIONING PROTOCOL WITH SERVERS AND A NETWORKED RESPONDER



(57) **Abstract:** A network can operate a WiFi access point with credentials. An unconfigured device can support a Device Provisioning Protocol (DPP), and record bootstrap public keys and initiator private keys. The network can record bootstrap public and responder private keys and operate a DPP server. A responder proxy can establish a secure and mutually authenticated connection with the network. The network can (i) derive responder ephemeral public and private keys, (ii) record the initiator bootstrap public key, and (iii) select a responder mode for the responder. The network can derive a shared secret with at least the (i) recorded the initiator bootstrap public key and (ii) derived responder ephemeral private key. The network can encrypt credentials using at least the derived shared secret and send the encrypted credentials through the responder proxy to the initiator, which can forward the encrypted credentials to the device, thereby supporting a device configuration.

[Continued on next page]

SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN,
TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(84) Designated States (*unless otherwise indicated, for every kind of regional protection available*): ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, ST, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, RU, TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, KM, ML, MR, NE, SN, TD, TG).

Published:

— *with international search report (Art. 21(3))*

A HOSTED DYNAMIC PROVISIONING PROTOCOL WITH SERVERS AND A NETWORKED RESPONDER

CROSS-REFERENCE TO RELATED APPLICATION

5 This international PCT application claims the benefit of the filing date of U.S. Provisional Patent Application Serial No. 62/672,977, filed May 17, 2018, which is hereby incorporated by reference in its entirety.

BACKGROUND

Technical Field

10 The present systems and methods relate to authentication and configuration of WiFi radios, and more particularly, to using a Device Provisioning Protocol with a networked responder and a set of servers that record PKI keys, in order securely transfer a set of credentials to a device.

Description of Related Art

15 The ability to connect transducers such as sensors and actuators with a wireless network is a growing field with many economical applications. As the costs for both electronic hardware and bandwidth continue to decrease, the use of networked transducers is expected to continue increasing over the coming decades. Connecting transducers to a network can be referred to as “machine-to-machine (M2M) communications” or “the Internet of Things (IoT).” Among many potential benefits,
20 IoT technologies allow automated monitoring and/or control of assets, equipment, personnel, or a physical location where manual monitoring may not be economical. Many applications for the “Internet of Things” significantly reduce costs by using automated monitoring instead of manual techniques. Prominent examples of IoT applications today include monitoring and control for building heating/air-conditioning, automobiles, alarm systems, and tracking devices. Fast growing markets for
25 IoT applications today include health applications such as the remote monitoring of a person’s fitness activity, heartbeat, or glucose levels, monitoring of industrial equipment deployed remotely in the field, and also security systems.

Many IoT applications can leverage wireless networking technologies, in addition to wired technologies such as Ethernet. Wireless technologies such as wireless local area networks and wireless
30 wide area networks have proliferated around the world over the past 30 years, and usage of these wireless networks is also expected to continue to grow. Wireless local area network (LAN) technologies include WiFi based on the series of 802.11 standards from the Institute of Electrical and Electronics Engineers (IEEE). The many options to connect a transducer device to a wireless network

create opportunities for new products and services, but also create several classes of problems that need to be solved. Many of these problems or needs in the art pertain to securely and efficiently configuring the transducer devices. Without configuration including a set of network access credentials, a device cannot normally securely connect with a wireless network. A need exists in the art to allow a user or a network to securely and easily upload to the device a set of network access credentials for a wireless network.

Manually configuring devices to (i) use different security keys, (ii) apply new network access credentials to connect with the owner's selected wireless network can be time consuming and difficult for users. Entering network access credentials is also prone to errors and especially difficult for devices with limited user interfaces, where a limited user interface is common for many devices supporting applications for the "Internet of Things". Loading credentials and network configuration into a device in order to obtain connectivity to a network can also be known as device provisioning or device configuration. In an attempt to address these needs to simplify or automate device provisioning, the WiFi Alliance TM released a Device Provisioning Protocol (DPP) specification on April 9, 2018 as version 1.0 (DPPv1.0), which is herein incorporated by reference in its entirety. Although the DPPv1.0 defines a series of messages between an initiator and a responder in order to support a device configuration, several unmet needs remain in the art in order to securely and easily deploy a Device Provisioning Protocol at wide scale with potentially billions of devices over many years.

To highlight one example unmet need by a DPPv1.0, WiFi capable devices currently being designed and deployed may have an expected life of more than a decade. During that timeframe, the device may need to undergo a configuration or device provisioning step more than one time. Reasons could include either (i) the device undergoes a "factory reset", or (ii) the device needs to be provisioned a second time, such as if the device owner changes or the device is moved to a different network, and other reasons exist as well. But, as stated in DPPv1.0, "threats against this bootstrapping method are possible if public keys and/or codes are reused." A need exists in the art to support a Device Provisioning Protocol in a manner where bootstrapping public keys can be securely reused. A need exists in the art for a responder bootstrap public key to be securely shared among a plurality of devices, while the responder bootstrap private key can remain secured.

In order to securely provision a device, a network may prefer to conduct mutual authentication with an initiator operating in the device that records and operates with initiator PKI keys. The device could be a high value piece of equipment, such as an automobile, health monitoring equipment in a hospital, or industrial equipment, etc. that could be targets for potential hackers seeking to "provision" the devices to a network under their control or a network providing the hackers access to the device. In these cases, an owner or user of a device may prefer that a device provisioning protocol use mutual authentication. The DPPv1.0 supports mutual authentication in order to securely authenticate a device and an initiator before transferring network access credentials to the device, but creates a new and

shifted problem of both (i) securely providing a responder bootstrap public key to an initiator and (ii) keeping the responder bootstrap private key secure. In other words, recording a responder bootstrap public key for a responder that is located nearby the initiator can be difficult if there is no pre-existing relationship or communication between a device with an initiator and a responder before the start of a device provisioning protocol. In addition, an insecure responder may have no pre-existing or sufficiently secured relationship with a network, yet the network may need to securely transfer credentials for the network to the device through the insecure responder. A need exists in the art to securely and readily enable an initiator to have access to an responder bootstrap public key, while also enabling a responder to have access to the corresponding initiator bootstrap public key. A need exists in the art for a network to securely transfer credentials to a device through an insecure responder.

With the widespread deployment of WiFi access points, a device may have access to potentially a plurality of different access points associated with potentially different networks. Although DPPv1.0 supports configuration of a device, DPPv1.0 does not suggest how a device could be pre-configured to support each of the potential plurality of different networks in order to support a device provisioning protocol. Also, DPPv1.0 does not suggest how a pre-configured device could attempt to communicate with the potential plurality of different access points in order to obtain at least one set of network credentials from the potential plurality of different networks. In other words, a need exists in the art for an initiator to use a pre-configuration in order to automatically and securely receive a set of credentials for the device without having to separately conduct an “out of band” receipt of a responder bootstrap public key when the device is taken to the vicinity of a supported network. A need exists in the art for (i) the credentials to be encrypted in a manner such that the responder cannot feasibly read the credentials, where (ii) the device can receive the encrypted credentials and convert the encrypted credentials into plaintext form in order for the device to load and use the plaintext credentials with a network.

A network can use DPPv1.0 in order to connect devices to the network using a responder. The security for conducting a device provisioning protocol according to DPPv1.0 can depend on the security of a responder, since as envisioned by DPPv1.0 the responder (i) records a responder bootstrap private key and (ii) conducts key exchanges with the responder bootstrap private key. However, a responder for conducting a DPP may be outside the control of a network and may also be an insecure device. A responder could be a “rooted” mobile phone operated by a hacker or also a “rouge” access point, as a couple of examples, and many other examples exist as well. Or, the responder could simply be operated by a computing device that is insecure or operates with security policies that do not meet the standards of the network. A need exists in the art for an insecure responder to conduct a device provisioning protocol that is compatible with DPPv1.0, while simultaneously keeping (i) the responder bootstrap private key and (ii) cryptographic operations with the responder bootstrap private key under the control or within the security policies of a network providing credentials to the device.

Many other examples exist as well for needs in the art to support efficient yet secure device configuration or provisioning, and the above are examples are just a few and intended to be illustrative instead of limiting.

SUMMARY

Methods and systems are provided for a networked responder proxy to conduct a device provisioning protocol (DPP) with an initiator operating in a device. The device provisioning protocol can support the Device Provisioning Protocol specification version 1.0 and subsequent versions from the WiFi Alliance™. Conducting a DPP with a device can transfer a set of network credentials to the device, in order for the device to connect with a network using the network credentials. The set of network credentials can also include a network configuration, where the network configuration includes settings to use with a WiFi radio in the device, in order for the WiFi radio to communicate with other nodes with the credentials, including an access point which can provide connectivity to an IP network for the device.

A device can operate an initiator, which periodically automatically transmits DPP authentication request messages until receiving a DPP authentication response from a responder proxy. The initiator can record and operate an initiator configuration, specifying a frequency band to utilize, a wireless networking technology to implement (e.g. 802.11 ac, 802.11ah, 4G LTE, 5G, etc.) and a channel list. The wireless networking technology for an initiator and a responder proxy can support a wide-area network (WAN) as well, such as 4G and 5G networks associated with public land mobile networks. The initiator can record an initiator bootstrap public key, an initiator bootstrap private key, and, when supporting mutual authentication with a responder, a responder bootstrap public key. The PKI keys can be associated with a selected set of cryptographic parameters, where the parameters specify settings to use for cryptographic operations by the initiator. The initiator can also include a random number generator, a PKI key pair generation algorithm, a secure hash algorithm, an elliptic curve Diffie Hellman key exchange algorithm, a symmetric ciphering algorithm, a digital signature algorithm, and an asymmetric ciphering algorithm.

A network can operate an access point, where the access point operates with a set of credentials. The credentials can include an identifier for the network such as an SSID, a pairwise master key (PMK) or a pre-shared key (PSK), a pairwise master key identifier (PMKID), and also a configuration. Credentials for other authentication schemes between a device and an access point could be supported as well, including EAP-TLS, EAP-PSK, EAP-AKA, EAP-AKA', etc. The access point can support wireless networking standards in the series of 802.11 standards. A network can also operate a plurality of access points. The network can operate or be associated with a Device Provisioning Protocol (DPP) server, and a device database. The network can be connected to an IP network, which can comprise the globally routable public Internet. Different servers within a network can be connected using an Intranet, a virtual private network (VPN), Internet Protocol Security (IPSEC), or similar techniques to secure communication between the servers. The access point for a

network can also comprise a enodeb for 4G wireless technology or a gnodeb for 5G wireless networking technology. The credentials with 4G networking technology can include at least a key K and the credentials with 5G networking technology can include at least a key Kausf.

The network can operate or communicate with the device database. The DPP server in the network can also communicate with the device database. The device database can record information for a plurality of devices before initiators in the device conduct the device provisioning protocol with a responder proxy. The device database can record the device identity, an initiator identity, which could be in the form of an initiator media access control (MAC) address, and a key table. The key table in a device database can record for each device an initiator bootstrap public key, a responder bootstrap public key, and a responder bootstrap private key. Recorded data within a device database can be obtained by a network (i) upon manufacturing from a device manufacturer, or (ii) before distribution of a device to the location of the network's access point.

The DPP server for a network can operate a server database, where the server database records information for a plurality of devices connecting with the network. The server database can record the device identity, media access control (MAC) addresses for devices and initiators, a responder mode, a selected set of cryptographic parameters for each device, a responder bootstrap private key, the initiator bootstrap public key, the responder bootstrap public key, a responder ephemeral public key, and responder ephemeral private key, a public key exchange (PKEX) key, a first, second, and third derived shared secret keys for an initiator. Depending on the responder mode, not all PKI keys and secret shared keys from the previous sentence may be recorded in the server database. The DPP server can also include a set cryptographic algorithms, where the cryptographic algorithms can include a random number generator, a PKI key pair generation algorithm, a secure hash algorithm, an elliptic curve Diffie Hellman key exchange algorithm, a symmetric ciphering algorithm, a digital signature algorithm, and an asymmetric ciphering algorithm. The cryptographic algorithms within a DPP server can be compatible with the equivalent set of cryptographic algorithms in the responder and also the initiator.

A responder can be a computing device that includes a WiFi radio and can be associated with a network. The responder can operate the WiFi radio with a responder configuration in order for the responder to send and receive messages with the initiator. The responder can be located in sufficient proximity with the initiator so that the WiFi radio for the initiator and the WiFi radio for the responder can communicate using 802.11 standards. The responder can operate a DPP application, where the DPP application can also include the set of cryptographic algorithms, which are compatible with cryptographic algorithms in both the initiator and the DPP server. The responder can use an IP network in order to communicate with the network via the IP network.

In a first responder mode, the responder can establish a secure, mutually authenticated connection with a DPP server operating in a network. The DPP server can send configuration data to

the responder in preparation for the responder operating as a responder proxy to receive incoming DPP authentication requests from at least one initiator. The responder can conduct a responder configuration step after receiving the responder configuration data from the DPP server. An initiator can receive and/or record a responder bootstrap public key and conduct a series of steps to transmit a DPP authentication request, potentially as a broadcast message on a WiFi channel on which the device operating the responder listens. The initiator can record an initiator key table and an initiator DPP configuration to create and transmit a series of DPP authentication request messages on different channels within the initiator DPP configuration. The responder can receive at least one DPP authentication request message transmitted by the initiator using a channel in a responder DPP configuration. The responder can forward data from the DPP authentication request received from the initiator to the DPP server through the secure session.

Continuing in the first responder mode, the DPP server can then conduct a series of steps in order to process data in the Device Provisioning Protocol (DPP) authorization request received by the responder operating as a responder proxy. The DPP server can derive a responder ephemeral PKI key pair. The DPP server can query a device database for data for the device using either the initiator media access control (MAC) address or the hash value of the initiator bootstrap public key from the DPP authentication request message. The DPP server can conduct a first responder elliptic curve Diffie-Hellman (ECDH) key exchange with the initiator bootstrap public key and the derived responder private key to derive a key k_1 . The DPP server can use the derived key k_1 to decrypt ciphertext from the DPP authentication message and read a plaintext initiator authentication value.

Continuing in the first responder mode, the DPP server can conduct a second responder ECDH key exchange with the received initiator ephemeral public key and the derived responder private key to derive a key k_2 . The DPP server can conduct a third responder ECDH key exchange with the initiator bootstrap public key and a combination of the derived responder private key and the responder bootstrap private key in order to derive a key k_e . The DPP server can derive a responder nonce and calculate a responder authentication value using the received and decrypted initiator authentication value. The DPP server can encrypt the responder authentication value into a first responder ciphertext using the derived key k_e . The DPP server can (i) encrypt the first responder ciphertext and the responder nonce and the initiator nonce using the derived key k_2 into (ii) a second responder ciphertext. The DPP server can send the responder a message with the second responder ciphertext and the derived responder ephemeral public key. The responder operating as a responder proxy can send the initiator a DPP authentication response message, which includes the second responder ciphertext and the responder ephemeral public key. Note that the responder operating as a responder proxy does not need to record the responder bootstrap private key.

Continuing in the first responder mode, the initiator can receive the DPP authentication response from the responder operating as a responder proxy and perform a series of steps to decrypt

the received second responder ciphertext. The initiator can conduct an initiator ECDH key exchange using the responder ephemeral public key and the initiator ephemeral private key in order to derive the key k_2 . The initiator can conduct a second initiator key exchange using (i) a combination of the responder bootstrap public key and the responder ephemeral public key and (ii) the initiator bootstrap private key and in order to derive the key k_e . The initiator can decrypt the second responder ciphertext using the derived key k_2 . The initiator can read the plaintext responder nonce and the first responder ciphertext. The initiator can decrypt the first responder ciphertext using the derived key k_e . The initiator can read the plaintext responder authentication value from the first responder ciphertext. The initiator can calculate the responder authentication value, and if the received responder authentication value matches the calculated responder authentication value, then the responder is authenticated.

The initiator can calculate an initiator authentication value using at least the received and decrypted responder nonce. The initiator can encrypt the initiator authentication value using the key k_e , thereby creating an initiator ciphertext. The initiator can send a DPP authentication confirm message to the responder with the initiator ciphertext. The responder operating as a responder proxy can forward the initiator ciphertext to the DPP server through the secure session. The DPP server can decrypt the initiator ciphertext using the derived key k_e and read the plaintext initiator authentication value. The DPP server can calculate the same initiator authentication value, and if the received, decrypted initiator authentication value matches the calculated authentication value then the initiator is authenticated with the DPP server and the responder proxy.

After the mutual authentication, the responder can begin operating as a configurator and the initiator can begin operating as an enrollee. The initiator can generate an enrollee nonce and also a configuration attribute, where the configuration attribute describes WiFi capabilities for the device. The initiator can create a second initiator ciphertext by encrypting the enrollee nonce and the configuration attribute with the derived key k_e . The initiator can send a DPP configuration request to the responder, where the DPP configuration request can include the second initiator ciphertext. The responder operating as a responder proxy can send the second initiator ciphertext to the DPP server.

The DPP server can receive the second initiator ciphertext from the responder operating as a responder proxy and decrypt the second initiator ciphertext using the derived key k_e , in order to read the plaintext enrollee nonce and the configuration attribute. Using the configuration attribute, the DPP server can select a set of network credentials for the device operating the initiator. The DPP server can encrypt the received enrollee nonce and the network credentials using the derived key k_e , creating a third responder ciphertext. The DPP server can send the third responder ciphertext to the responder operating as a responder proxy. The responder can forward the third responder ciphertext to the initiator in a DPP configuration response message.

The initiator can receive the DPP configuration response message and decrypt the third responder ciphertext using the derived key k_e in order to read the plaintext network credentials. The

initiator can pass the network credentials to the device, which can apply the credentials with the WiFi radio for the device. The device can subsequently connect with a network access point using the received network credentials. The network access point can provide connectivity to an IP network for the device. The DPP server and the device can record that the DPP session is successfully completed.

5 These as well as other aspects and advantages will become apparent to those of ordinary skill in the art by reading the following detailed description, with reference where appropriate to the accompanying drawings.

BRIEF DESCRIPTION OF THE DRAWINGS

Various exemplary embodiments are described herein with reference to the following drawings, wherein like numerals denote like entities.

5 Figure 1a is a graphical illustration of an exemplary system, where a network operates a responder and communicates with an initiator in a device, in order to conduct a device provisioning protocol, in accordance with exemplary embodiments;

Figure 1b is a graphical illustration of a device provisioning protocol for authentication of a responder and configuration of an initiator, in accordance with conventional technology;

10 Figure 1c is a graphical illustration of a device provisioning protocol for mutual authentication of an initiator and a responder, and configuration of an initiator, in accordance with conventional technology;

Figure 1d is an illustration of an exemplary initiator DPP configuration and initiator key database, with tables recording exemplary data for a plurality of bootstrap responder public keys and bootstrap initiator private keys, in accordance with exemplary embodiments;

Figure 1e is an illustration of an exemplary responder bootstrap public keys and initiator bootstrap PKI keys recorded by a plurality of devices, in accordance with exemplary embodiments;

Figure 1f is a graphical illustration of hardware, firmware, and software components for a responder, including a responder configuration step, in accordance with exemplary embodiments;

20 Figure 2a is an illustration of an exemplary device database, with tables for a device database recording exemplary data, in accordance with exemplary embodiments;

Figure 2b is a flow chart illustrating exemplary steps for an initiator to use an initiator DPP configuration and initiator key database in order to automatically connect with a responder proxy, in accordance with exemplary embodiments;

25 Figure 3a is a simplified message flow diagram illustrating an exemplary system with exemplary data sent and received by a device database, a DPP server, a responder proxy, and an initiator, in accordance with exemplary embodiments;

Figure 3b is a simplified message flow diagram illustrating an exemplary system with exemplary data sent and received by a DPP server, a responder proxy, and an initiator, in accordance with exemplary embodiments;

30 Figure 3c is an illustration of an exemplary server database, with tables for a server database recording exemplary data, in accordance with exemplary embodiments;

Figure 3d is a flow chart illustrating exemplary steps for creating and verifying a digital signature using PKI keys, parameters, and data input, in accordance with exemplary embodiments;

Figure 3e is a flow chart illustrating exemplary steps for using asymmetric ciphering in order to encrypt credentials using a public key and decrypting credentials using the corresponding private key, in accordance with exemplary embodiments;

Figure 4a is a flow chart illustrating exemplary steps for conducting a key exchange using PKI keys in order to derive a shared secret key, and for using the derived shared secret key to encrypt and decrypt data, in accordance with exemplary embodiments;

Figure 4b is a flow chart illustrating exemplary steps for conducting a key exchange using PKI keys in order to derive a shared secret key, and for using the derived shared secret key to encrypt and decrypt data, in accordance with exemplary embodiments;

Figure 4c is a flow chart illustrating exemplary steps for conducting a key exchange using PKI keys in order to derive a shared secret key, and for using the derived shared secret key to encrypt and decrypt data, in accordance with exemplary embodiments;

Figure 4d is a flow chart illustrating exemplary steps for conducting key exchanges using PKI keys in order to derive a first and a second shared secret keys, where ECC point addition is performed on output from a key exchange algorithm in order to derive a third shared secret key;

Figure 5a is a simplified message flow diagram illustrating an exemplary system with exemplary data sent and received by a device database, a DPP server, a responder proxy, and an initiator, in accordance with exemplary embodiments;

Figure 5b is a simplified message flow diagram illustrating an exemplary system with exemplary data sent and received by a device database, a DPP server, a responder proxy, and an initiator, in accordance with exemplary embodiments;

Figure 6 is a simplified message flow diagram illustrating an exemplary system with exemplary data sent and received by a DPP server, a responder proxy, and an initiator, in accordance with exemplary embodiments;

Figure 7 is a graphical illustration of an exemplary system, where an access point with an initiator communicates with a responder proxy using a hosted device provisioning protocol, in accordance with exemplary embodiments; and,

Figure 8 is a graphical illustration of an exemplary system, where an access point operates as a responder proxy for a device with an initiator, in order transfer a set of network credentials to the device, in accordance with exemplary embodiments.

DETAILED DESCRIPTION**Figure 1a**

Figure 1a is a graphical illustration of an exemplary system, where a network operates a responder and communicates with an initiator in a device, in order to conduct a device provisioning protocol, in accordance with exemplary embodiments. The system 100 can include a network 107 and a device 106. Network 107 can operate a WiFi access point 105, a DPP server 103, and a device database 104. The nodes within network 107 can communicate via an IP network 113. Device 106 can be a computing device operating an initiator 101 and a WiFi radio 106r. Device 106 can include a device identity 106i for a network 107 to track or identify a device 106 across a plurality of different devices 106. Or, in exemplary embodiments a media access control (MAC) address MAC.I 106n for device 106 can be used as a device identity.

Responder proxy 102 can operate as a responder and send and receive messages with device 106 operating an initiator 101 according to a device provisioning protocol (DPP) 191 as specified in the “Device Provisioning Protocol Specification version 1.0” (DPPv1.0) from the WiFi Alliance™, which is hereby incorporated by reference in its entirety. Subsequent and related versions of a DPP 191 could be supported as well by initiator 101 in device 106 and a responder proxy 102. A summary of the message flows between responder proxy 102 and device 101 according to the DPP 191 are depicted and described in connection with Figure 1b and Figure 1c below, as well as subsequent figures herein. The message flows for a DPP 191 between responder proxy 102 and initiator 101 could be via a WiFi data link 192, where the WiFi data link 192 operated by (i) initiator 101 can use an initiator DPP configuration 130 and (ii) responder proxy 102 can use a responder DPP configuration 131. Responder proxy 102 can be associated with a tag 102v, where tag 102v can record a responder bootstrap public key Br 102a.

Tag 102v can be in the form of a QR code or NFC tag, which could be read by device 106 in order to obtain the responder bootstrap public key Br 102a “out of band”. Or, in some embodiments tag 102v can be optionally omitted, where an initiator 101 can receive or record responder bootstrap public key Br 102a via other means, such as during manufacturing or distribution of device 106 where a device 106 could record a plurality of different responder bootstrap public keys Br 102a in a responder public key table 101ta depicted and described in connection with Figure 1d below.

Responder proxy 102 could operate a WiFi radio as depicted, where the WiFi radio could record a media access control (MAC) address MAC.R 102n. In exemplary embodiments, a wireless device could operate the responder proxy 102, where the wireless device could comprise a mobile handset, a smartphone, a wireless router, a tablet computer, and/or a wireless gateway. Other

possibilities exist as well for the embodiments of a wireless device operating a responder proxy 102 without departing from the scope of the present disclosure. MAC.R 102n for responder 102 could comprise a string of octets to identify responder 102 with any of a WiFi access point 105, device 106, network 107, and DPP server 103. In order to acquire the data to function as a responder according to a DPP 191 and DPPv1.0, responder 102 in Figure 1a can communicate and transmit/receive data from network 107 via IP network 113. Additional details regarding the electrical components within a responder 102 or WiFi access point 105 operating a responder 102 are depicted and described in connection with Figure 1f below. The responder proxy 102 can also include a set cryptographic algorithms, where the cryptographic algorithms can include a random number generator, a PKI key pair generation algorithm, a secure hash algorithm, an elliptic curve Diffie Hellman key exchange algorithm, a symmetric ciphering algorithm, a digital signature algorithm, and an asymmetric ciphering algorithm. The cryptographic algorithms within a responder proxy 102 can be compatible with the equivalent set of cryptographic algorithms in the initiator 101 and also the DPP server 103.

Embodiments of the present disclosure contemplate both (i) some of the functionality of a responder in the DPP specification version 1.0 operates within responder proxy 102 depicted in Figure 1a, and (ii) other functionality of a responder according to the DPP protocol can be performed or conducted by DPP server 103. In other words, (x) from the perspective of device 106, responder proxy 102 can operate as a responder according to the DPP Specification, but (y) in the present disclosure some of the functionality and data for a responder as contemplated by the DPP Specification remains in the DPP server 103. Depictions and descriptions below in the present disclosure may utilize the term “responder 102” or configured “responder 102” , which can also comprise a “responder proxy 102” depicted in Figure 1a. As contemplated herein, a “responder proxy 102” can be equivalent to “responder 102.”

This configuration of operating a responder proxy 102 in a system 100 (along with proper operation of a network 107) can address and solve the needs in the art described above. A responder proxy 102 can communicate with DPP server 103 via IP network 113. Although a responder proxy 102 is depicted as operating in WiFi access point 105 in Figure 1a, other embodiments of the present disclosure contemplate that different WiFi capable devices than access point 105 could operate a responder proxy 102, such as a mobile phone 701 depicted in Figure 7 below. Other possibilities exist as well for a WiFi capable device to operate a responder proxy 102 without departing from the scope of the present disclosure. In addition, although the use of a WiFi radio is depicted in Figure 1a and other figures herein, other wireless networking technologies and radios besides WiFi could be used with responder proxy 102 and device 106 with initiator 101. As one example, (i) a responder proxy could operate with a wireless wide area network (WAN) radio such as a “gnodeb” with a 5G network and (ii) device 106 with initiator 101 could operate a radio module supporting 5G networking standards.

IP network 113 could be either a Local Area Network (LAN) or a Wide Area Network (WAN), or potentially a combination of both. IP network 113 could comprise a packet switched network supporting either IEEE 802.11 (WiFi) standards and/or IETF standards for IP networking such as IPv4 or IPv6. Responder 102 within access point 105 and network 107 could also utilize a variety of WAN
5 wireless technologies to communicate, including Low Power Wide Area (LPWA) technology, 3rd Generation Partnership Project (3GPP) technology such as, but not limited to, 3G, 4G Long-Term Evolution (LTE), or 4G LTE Advanced, NarrowBand – Internet of Things (NB-IoT), LTE Cat M, proposed 5G networks, and other examples exist as well. A wired responder proxy 102 can connect to the IP network 113 via a wired connection such as, but not limited to, an Ethernet, a fiber optic, or
10 a Universal Serial Bus (USB) connection (not shown). In some exemplary embodiments, access to an IP network 113 for responder proxy 102 can be provided by access point 105, where access point 105 may also record and operate with a set of network credentials 109.

IP network 113 could also be a public or private network supporting Internet Engineering Task Force (IETF) standards such as, but not limited to, such as, RFC 786 (User Datagram Protocol), RFC
15 793 (Transmission Control Protocol), and related protocols including IPv6 or IPv4. A public IP network 113 could utilize globally routable IP addresses, and a private IP network 113 could utilize private IP addresses which could also be referred to as an Intranet. Other possibilities for IP Network 113 exist as well without departing from the scope of the disclosure. Although not depicted in Figure 1a, elements within a system 100 could each operate or be associated with a firewall, such that packets
20 from IP network 113 are filtered or restricted in order to secure the communication between the nodes.

Device 106 can comprise a computing device with a WiFi radio for transmitting and receiving data such as via WiFi data link 192. Device 106 can include a WiFi radio 106r. Radio 106r and radio 102i for responder proxy 102 can also support other wireless technologies besides WiFi, including 4G and 5G networks. WiFi radio 106r can operate with a media access control (MAC) address MAC.I
25 106n. Responder 102 can read the value MAC.I 106n from messages transmitted by device 106 over WiFi link 192. Device 106 can operate an initiator 101 and, after a configuration step 106a, also operate a set of network WiFi credentials 109 that are compatible with an access point 105. Device 106 could be powered via any of (i) traditional “wall power” potentially with an AD/DC adapter, (ii) a battery which may be periodically recharged or replaced, (iii) power over a wired LAN connection
30 such as “power over Ethernet”, and other possibilities exist as well. A device 106 can record a certificate cert.device 106m, which can record a device public key, which can be associated with a device private key SK.device 106s also recorded in device 106. For the system depicted in Figure 1a, device 106 operating an initiator 101 can have a roll of an enrollee, and responder proxy 102 can operate with a roll of a configurator.

35 The initiator 101 in device 106 can record an initiator key database 101t, which could comprise a database such as SQLite, or alternatively simply a text file of keys and identities, and other

possibilities exist as well for the form and structure of keys recorded in an initiator 101. In exemplary embodiments, the initiator key database 101t can be stored in a secure processing environment or a secure enclave for a CPU operating in device 106. The initiator key database 101t can optionally include an initiator bootstrap public key Bi 101a, an initiator bootstrap private key bi 101b (for
5 embodiments supporting a mutual authentication 142 depicted in Figure 1c below), a responder bootstrap public key Br 102a, a set of cryptographic parameters 199a for the PKI keys, and optionally a public key exchange protocol (PKEX) shared secret key 198. The initiator 101 in device 106 can comprise a process or running software program operated by a processor and recorded in memory for device 106. Although not depicted in Figure 1a, an initiator 101 can also include a random number
10 generator, a PKI key pair generation algorithm, a secure hash algorithm, an elliptic curve Diffie Hellman key exchange algorithm, and a symmetric ciphering algorithm, a digital signature algorithm, and an asymmetric ciphering algorithm. The cryptographic algorithms within an initiator 101 can be compatible with the equivalent set of cryptographic algorithms in a responder proxy 102 and also DPP server 103.

15 A network 107 can operate an access point 105, where the access point operates with a set of credentials 109. The credentials 109 can include an identifier for the network such as an SSID.network-AP 109a, at least one pairwise master key (PMK) PMK.network-AP 109b or a pre-shared key (PSK), a pairwise master key identifier (PMKID) (not shown), and also a configuration Config.network-AP 109c. The access point can support wireless networking standards in the series of 802.11 standards.
20 Credentials 109 for other authentication schemes between a device and an access point could be supported as well, including EAP-TLS, EAP-PSK, EAP-AKA, EAP-AKA', etc. A network 107 can also operate a plurality of access points 105. In addition, access point 105 can comprise a plurality of different WiFi radios to collectively function as one access point 105, such as with "mesh" WiFi networks. Or, access point 105 can operate as a single WiFi router. The network 107 can include or
25 be associated with a Device Provisioning Protocol (DPP) server 103, and a device database 104. Different servers within the network 107 can be connected using an Intranet, a virtual private network (VPN), Internet Protocol Security (IPSEC), or similar techniques to secure communication between the servers.

The network 107 can operate or communicate with the device database 104. The DPP server
30 103 in the network 107 can also communicate with the device database 104. The device database 104 can record information for a plurality of devices 106 before an initiator 101 for the device 106 conducts the device provisioning protocol 191 with the responder proxy 102. The device database 104 can record an identity for device 106, which could be in the form of either (i) an initiator media access control (MAC) address MAC.I 106n (depicted) or a hash value 101aa for the initiator bootstrap public
35 key Bi 101a (where the hash value 101aa is not shown in Figure 1a but is shown in Figure 1b and other Figures below).

A device database 104 can record at least one responder bootstrap public key 102a with a corresponding responder bootstrap private key 102b, and optionally an initiator bootstrap public key 101a for embodiments supporting a mutual authentication depicted in Figure 1c and other Figures below. Recorded data within a device database 104 can be obtained by a network 107 from either (i) upon manufacturing from a device manufacturer, or (ii) before distribution of a device to the location of the network's access point. Some data in a device database 104 could also be acquired by a responder 102 or a network 107 prior to the start of a DPP 191. For example, the derivation and recording of responder bootstrap private key br 102b can be independent of recording data in a database 104 for a device 106 (such as MAC.I 106n, key Bi 101a, etc.). Device database 104 can also record a set of cryptographic parameters 199a associated with the PKI keys for responder 102 and initiator 101. Additional exemplary data for a device database 104 is depicted and described in connection with Figure 2a below.

The DPP server 103 for a network 107 can operate a server database 103a, where the server database 103a records information for a plurality of devices 106 and a plurality of responders proxies 102 connecting with the network 107. A server database 103a can also record data for initiators 101 in devices 106, as depicted and described in connection with Figure 3c below. Depending on a responder mode 306a in Figures 2a and 3c below, only selected sets of PKI keys and secret shared keys for initiators 101 may be recorded in the server database 103a. The DPP server 103 can also include a set cryptographic algorithms, where the cryptographic algorithms can include a random number generator, a PKI key pair generation algorithm, a secure hash algorithm, an elliptic curve Diffie Hellman key exchange algorithm, a symmetric ciphering algorithm, a digital signature algorithm, and an asymmetric ciphering algorithm. The cryptographic algorithms within a DPP server 103 can be compatible with the equivalent set of cryptographic algorithms in the initiator 101 and also the responder proxy 102.

Note that the security of communications between a device 106 and an access point 105 using credentials 109 may generally depend on the security of a configuration step 106a, which in turn may rely on a device provisioning protocol 191. Further, a manufactured device 106 may not have the credentials 109 or configuration in order to communicate with an access point 105 without successfully completing a DPP 191 and a configuration step 106a (or other manual or cumbersome means) in order for device 106 to securely receive and use network credentials 109. After a DPP 191 and a configuration step 106a described in the present disclosure and in multiple Figures below, device 106 can communicate with access point 105 using credentials 109.

Network 107 can also be associated with an external network 108, which can comprise a network similar to network 107. External network 108 could operate a plurality of access points 105, and in exemplary embodiments although the device provisioning protocol 191 is conducted with network 107, the credentials 109 transferred to device 106 in a configuration step 106a may belong to

the external network 108, and the device 106 could connect with an access point operated by the external network 108 using the credentials 109 instead of connecting with access point 105 operated by network 107. For these embodiments, network 107 could receive the credentials 109 for device 106 as asymmetrically encrypted credentials 109' (described in Figure 3b below) from the external network 108.

Figure 1b

Figure 1b is a graphical illustration of a device provisioning protocol for authentication of a responder and configuration of an initiator, in accordance with conventional technology. Figure 1b depicts a summary of the WiFi Device Provisioning Protocol (DPP) specification, version 1.0 which was published on April 9, 2018. The summary depicted in Figure 1b highlights recorded bootstrap PKI keys, derived ephemeral PKI keys, and messages transmitted and received between an initiator 101 and a responder 102*. The use of keys and messages for a DPP 191 can accomplish both (i) authenticating the responder 102* and (ii) transfer a configuration object to initiator 101 where the configuration object could comprise a set of network access credentials 109 from Figure 1a. Responder 102* is depicted with a "*" and has several differences than a responder proxy 102 depicted in Figure 1a and subsequent Figures below, although some functionality is shared between a responder 102* in Figure 1b and responder proxy 102 herein. Both responder 102* and responder proxy 102 (from Figure 1a and other Figures below) can send and receive the full and complete set of a DPP messages for a Device Provisioning Protocol 191. As contemplated herein, a responder 102* can comprise a responder operating according to the conventional technology depicted in Figure 1b and Figure 1c, while a responder 102 or responder 102' or responder proxy 102' can (i) operate as a proxy for DPP 191, and (ii) record and operate on different PKI keys than those depicted for responder 102* in Figure 1b.

In other words, responder 102* and a responder proxy 102 can send equivalent messages such that a initiator 101 (i) would process the messages in the same manner and (ii) would not normally be able to detect to discern whether a responder proxy 102 or an responder 102* was transmitting and receiving messages for the Device Provisioning Protocol 191. In the present disclosure, responder proxy 102 (not depicted in Figure 1b, but depicted in Figure 1a, Figure 1f, Figure 3a, etc.) can send and receive DPP messages 122, 123, etc. with initiator 101 as an responder according to the DPP Specification 1.0. Responder 102* or a responder proxy 102 can be a process, program, firmware, or software application operating in a computing device with a WiFi compatible radio, in order to communicate with an initiator 101 in a device 106 via a WiFi

data link 192 (in Figure 1a). Exemplary additional configurations for an initiator 101 and a responder proxy 102 within computing devices is depicted and described in connection with Figures 7 and 8 below, and other possibilities exist as well without departing from the scope of the present disclosure.

5 Other differences between a responder 102* and a responder proxy 102 can exist in the present disclosure. For a system 100 with a responder proxy 102 in Figure 1a and Figures below, the recordation of PKI keys, PKI key derivation, and cryptographic processes such as symmetric encryption and decryption can be distributed between a responder proxy 102 and a DPP server 103. However, a responder 102* can internally record the keys necessary for symmetric key
10 exchange, key derivation, and symmetric encryption and decryption, as contemplated by DPP specification version 1.0. The distributed storage and operation with PKI keys for a system that includes a responder proxy 102 is depicted and described in more detail in connection with Figure 3a, Figure 3b, Figure 3c, Figure 5a, Figure 5b, and Figure 6 below. Additional details for internal components, memory structures, values, and operation of a responder proxy 102 in order to
15 provide equivalent functionality as responder 102* for an initiator 101 are described in Figures below.

Initiator 101 can record ephemeral keys and derive symmetric encryption keys in order to authenticate the responder 102* and receive the network access credentials 109 according to a Device Provisioning Protocol 191. For a system with responder authentication 141, an initiator
20 101 can record a media access control (MAC) address MAC.I 106n. MAC.I 106n can (i) be associated with a WiFi radio operating in device 106 that also operates initiator 101 and (ii) comprise a string of octets in order to uniquely identify initiator 101 and/or device 106 within a wireless network. MAC.I 106n can also comprise an identity for device 106 with network 107 in Figure 1a. With current, convention technology, MAC.I 106n can comprise a string of 6 octets,
25 although other possibilities exist as well for a MAC.I 106n. Initiator 101 can also operate a key pair generation function 101y in order to derive ephemeral PKI keys such as an initiator ephemeral public key P_i 101c and an initiator ephemeral private key p_i 101d. As contemplated herein, an ephemeral PKI key can also be referred to as a “protocol” key. In other words, key P_i 101c can be referred to as either (i) an initiator protocol public key P_i 101c or (ii) an initiator ephemeral
30 public key P_i 101c, which can both represent the same numeric value, number, or sequence of digits.

A key pair generation function 101y or 102y could use (i) a random number generator in order to derive the PKI private key and (ii) a selected set of cryptographic parameters 199a (depicted in Figure 1a above and Figure 2a below and other Figures herein) in order to specify an

elliptic curve cryptography (ECC) curve name, key length, key formatting, etc. As contemplated herein, the algorithms used with ECC PKI keys in the present disclosure can be compatible and substantially conform with ECC algorithms and keys as specified in (i) the IETF Request for Comments (RFC) 6090 titled "Fundamental Elliptic Curve Cryptography Algorithms", and (ii) IETF RFC 5915 titled "Elliptic Curve Private Key Structure", and also subsequent and related versions of these standards.

With a random number generated by responder 102* or a responder proxy 102, a key pair generation function 102y, a derived ephemeral responder private key pr 102r can be used to derive a corresponding ephemeral initiator public key Pr 102p. The two derived PKI keys can be used in conjunction for subsequent operations such as an Elliptic Curve Diffie Hellman key exchange and also other cryptographic algorithms such as a key derivation function and a digital signature algorithm. As contemplated in the present disclosure, the use of a capital letter as the first character for a PKI key can represent a public key, the use of a lower case letter as the first character for a PKI key can represent a private key. As contemplated in the present disclosure, the second letter for a PKI key can represent the entity the key is associated with or belongs to (e.g. "r" for responder 102 and "i" for initiator 101). Further, the letter "B" or "b" as the first character for a PKI key can represent a recorded bootstrap key, which could be static or relatively static over the lifetime of a responder or initiator, and the letter "P" or "p" the first character for a PKI key can represent an ephemeral key, which can also be referred to as a protocol key.

Responder 102* in Figure 1b and Figure 1c can include functionality to operate as a responder according to the DPP Specification version 1.0 for the purpose of transmitting and receiving messages according to the DPP protocol 191. For a DPP 191, a responder 102* can record a responder bootstrap public key Br 102a, a responder bootstrap private key 102b, a MAC address MAC.R 102n, and also operate a key pair generation algorithm 102y. A responder bootstrap public key Br 102a and responder bootstrap private key br 102b recorded in responder 102* could comprise an ECC key pair selected and formatted to be compatible with the set of parameters 199a for initiator 101 (in Figure 1a above). Ephemeral bootstrap PKI keys for responder 102* could also be formatted and selected with a compatible set of cryptographic parameters 199a for Pi 101c and pi 101d. Bootstrap PKI keys for responder 102* could be recorded in responder 102* or nonvolatile memory for a WiFi enabled computing device operating a responder 102*. With the conventional technology depicted in Figure 1b, the bootstrap PKI keys could be recorded in responder 102* during manufacturing or a configuration of responder 102* before receiving the DPP authentication request message 122. MAC address MAC.R 102n

can comprise an identity of responder 102* or the computing device operating responder 102* on a WiFi network, and can be similar to MAC.I 106n for an initiator 101 as described above.

An initiator 101 and a responder 102* can conduct a Device Provisioning Protocol (DPP) 191. Again, as contemplated herein, an exemplary Device Provisioning Protocol 191 according to conventional technology could comprise the “Device Provisioning Protocol Specification” version 1.0 as published by the WiFi Alliance™. In message 121, an initiator 101 can receive the responder bootstrap public key Br 102a for responder 102*. The “out of band” transfer of Br 102a in message 121 could comprise several different possible methods for initiator 101 to receive Br 102a, which are also described in the DPP specification 1.0, such as via reading a QR code on the device operating responder 102*, reading a near-field communications tag, using a Bluetooth wireless connection to read Br 102a from responder 102*, and other possibilities exist as well. The data could be read via a tag 102v for responder 102 depicted and described in connection with Figure 1a. A DPP 191 also contemplates that associated information for responder 102* could be transferred in message 121, such as an identity for the WiFi enabled computing device operating responder 102*, MAC.R 102n, in addition to Br 102a and parameters 199a in message 121, as well as a responder DPP configuration 131 as depicted in Figure 1a.

Initiator 101 can receive Br 102a from message 121 and derive PKI keys Pi 101c and pi 101d. Initiator 101 can conduct an Elliptic Curve Diffie Hellman key exchange algorithm 401a as depicted and described below in key exchange 314 step for Figure 4a in order to derive a first shared secret key with responder 102*. Initiator 101 can derive a random number comprising an initiator nonce and also determine a set of capabilities for the initiator, such as a configurator or enrollee as specified in DPP specification version 1.0. Initiator 101 can encrypt the initiator nonce and the initiator capabilities using the derived shared secret key as depicted and described in connection with an encryption step 315 in Figure 4a below. Initiator 101 can send the derived public key Pi 101c and a ciphertext with the encrypted data in a message 122. Although not depicted in Figure 1b, initiator 101 can also send a hashed value of responder bootstrap public key Br 102a in message 122.

As specified in DPP specification version 1.0, message 122 could also include a hashed value for initiator bootstrap public key Bi 101a. But, for a responder authentication 141 depicted in Figure 1b, the hashed value of Bi 101a may optionally not be calculated or sent in message 122 (since responder 102* may not record the corresponding initiator bootstrap public key Bi 101a for a responder authentication 141 as depicted in Figure 1b). The hash algorithm on a bootstrap public key could comprise the SHA-256 hash algorithm according to IETF RFC 5754 titled “Using SHA2 Algorithms with Cryptographic Message Syntax”. Other hash algorithms could be used as

well and specified in a selected set of cryptographic parameters 199 as depicted and described in connection with Figure 2b below, such as SHA-3, SHA-384, etc. Message 122 can comprise a Device Provisioning Protocol Authentication Request according to DPP specification version 1.0. Additional details for a message 122 are provided in section 6.2.2 of DPP specification version 1.0.

Responder 102* can receive message 122 and follow the steps for a responder according to the section 6.2.3 of DPP specification version 1.0, in order to generate a Device Provisioning Protocol Authentication Response, which could comprise message 123. Responder 102* can conduct an Elliptic Curve Diffie Hellman key exchange algorithm 401a as depicted and described below in key exchange 319 step for Figure 4a in order to mutually derive a first shared secret key with initiator 101. Responder 102* can decrypt the initiator nonce and the initiator capabilities using the derived first shared secret key as depicted and described in connection with an decryption step 320 in Figure 4a below. Responder 102* can use the key pair generation algorithm 102y in order to derive an ephemeral responder public key Pr 102p and a corresponding ephemeral private key 102r. Responder 102* can use the derived responder ephemeral private key pr 102r and the received initiator ephemeral public key Pi 101c in order to conduct a key exchange algorithm 314b as depicted below in Figure 4b and derive a second secret shared key. The first mutually derived symmetric key can comprise k1 and the second mutually derived symmetric key can comprise k2 according to the DPP specification version 1.0. Responder 102* can derive a random number comprising a responder nonce and also determine a set of capabilities for the responder, such as a configurator as specified in DPP specification version 1.0.

Responder 102* could also generate a responder authentication value, which could comprise a hash value over at least the initiator nonce received in message 122 and the generated responder nonce. Responder 102* could use the first shared secret key and the second shared secret key in order to derive a third shared secret key. Responder 102* could encrypt the responder authentication value with the third shared secret key and also then include the encrypted responder authentication value with the responder nonce and responder capabilities in a ciphertext. The ciphertext could be encrypted with the second shared key, as depicted and described in connection with an encryption step 315b in Figure 4b below. Responder 102* could send the derived ephemeral responder public key Pr 102p and the ciphertext in a message 123.

Initiator 101 can receive message 123 and record or store the received ephemeral responder public key Pr 102p. Initiator 101 can follow the steps in section 6.2.4 of DPP specification version 1.0 in order to process message 123 and confirm the authentication response from responder 102* in message 123. Initiator 101 can mutually derive the second shared secret

key using the received ephemeral responder public key 102p and the derived initiator ephemeral private key 101d, using a key exchange algorithm 319a as depicted in Figure 4b. Initiator 101 can decrypt the ciphertext in message 123 using the second mutually derived secret shared key and a decryption step 320b in Figure 4b. Initiator 101 can derive the third secret shared key using at least the first mutually derived secret shared key and the second mutually derived secret shared key. Initiator 101 can use the third mutually derived secret shared key to decrypt the encrypted responder authentication value using a decryption step 328 in Figure 4c. Initiator 101 can calculate a value for the responder authentication, and if the calculated value for the responder authentication matches the received value for the responder authentication, then responder 102* can be considered authenticated.

Although not depicted in Figure 1b, after authenticating responder 102*, initiator 101 can send an authentication confirm message, which could comprise a DPP authentication confirm message according to section 6.2.4 of the DPP specification version 1.0. The authentication confirm message can also include an initiator authentication value, which could be encrypted with the third secret shared key. The responder 102* can receive the encrypted authentication confirm value and decrypt the value and also internally calculate the initiator authentication value. If the received initiator authentication value and the calculated authentication confirm values are equal, then the initiator is authenticated with the responder 102*. Note that for a responder authentication 141 in Figure 1b, the authentication of initiator 101 by responder 102* is weak and only comprises verifying that initiator 101 records the bootstrap public key Br 102a for responder 102*. Also, although not depicted in Figure 1b, responder 102* could have the capabilities of a configurator and receive a DPP configuration request message from initiator 101, which could take the role of an enrollee. The DPP configuration request message can be according to section 6.3.2 of the DPP specification version 1.0, and can include an enrollee nonce, which could be a random number generated by the enrollee/initiator.

As depicted in Figure 1b, responder 102* can then send a configuration object in a ciphertext in message 124. Message 124 could comprise a DPP configuration response message according to section 6.3.3 of the DPP specification version 1.0. Responder 102* can record the configuration object before sending message 124. In the present disclosure, the configuration object comprises the network credentials 109 depicted in Figure 1a above. In other words, a configuration object can be configuration information for a device 106 with initiator 101 to use in order to connect with other devices via a wireless network. Configuration object could contain a list of network identifiers, device identifiers, RF band and channel information, configuration parameters, shared keys, PKI keys, names, passwords, group temporal keys, a shared secret 198

for a PKEX key exchange, a pairwise master key (PMK) and PMK identity, etc. An exemplary list of possible values for a configuration object is depicted in table 14 of DPP specification version 1.0. In message 124, the network credentials 109 as a configuration object can be sent in a ciphertext using the third mutually derived secret shared key. As depicted in Figure 1b, the ciphertext can include both the enrollee nonce and the network credentials 109. Initiator 101 can receive message 124, decrypt the ciphertext using the third mutually derived secret shared key, and record the network credentials 109. Device 101 can then apply the network credentials 109 and communicate with access point 105 using the received network credentials 109.

Figure 1c

Figure 1c is a graphical illustration of a device provisioning protocol for mutual authentication of an initiator and a responder, and configuration of an initiator, in accordance with conventional technology. Figure 1c depicts a summary of the WiFi Device Provisioning Protocol (DPP) specification, version 1.0 which was published on April 9, 2018, supporting a mutual authentication 142 by both initiator 101 and responder 102*. The summary depicted in Figure 1c highlights recorded bootstrap PKI keys, derived ephemeral PKI keys, and messages transmitted and received between an initiator 101 and a responder 102*. Many of (i) the PKI keys for initiator 101 and responder 102*, and (ii) the messages transmitted between the nodes are equivalent to those depicted and described in connection with Figure 1b. This description of Figure 1c will focus upon the differences in order for initiator 101 and responder 102* to mutually authenticate.

In order to support a mutual authentication between initiator 101 and responder 102*, initiator 101 can record an initiator bootstrap public key Bi 101a and an initiator bootstrap private key bi 101b. Although initiator bootstrap public key Bi 101a is depicted within initiator 101, another entity besides initiator 101 could record the public key and make it available to other nodes such as a responder 102*. As contemplated by the DPP Specification, the initiator bootstrap keys can be recorded with initiator 101 before conducting a DPP 191 authentication and configuration with responder 102*. For example, the bootstrap PKI keys for the initiator could be recorded during manufacturing or distribution of the initiator, or loaded by an initiator user before conducting the depicted communications with responder 102*. Bootstrap public keys for the initiator can support a selected set of cryptographic parameters 199a, where the selected set of cryptographic parameters 199a can match the parameters used by keys in responder 102*. The DPP specification version 1.0 identifies the minimal set of cryptographic parameters in order to ensure compatibility, which can comprise the “set A” for a set of cryptographic parameters 199 depicted in Figure 2a below.

In order to conduct a mutual authentication 142 in addition to the responder authentication 141 depicted in Figure 1c, a responder 102* should have access to the initiator bootstrap public key Bi 101a. In message 126, a responder 102* can receive the initiator bootstrap public key Bi 101a for initiator 101. The “out of band” transfer of Bi 101a in message 126 could comprise several different possible methods for responder 102* to receive Bi 101a, which are also described in the DPP specification 1.0, such as via reading a QR code, reading a near-field communications tag, using a Bluetooth wireless connection to read Bi 101a from initiator 101, and other possibilities exist as well, including using WiFi direct to first establish a “peer-to-peer” WiFi connection. A DPP 191 also contemplates that associated information for initiator 101 could be transferred in message 126, such as a device identity for initiator 101, MAC.I 106n or hash value 101aa, in addition to Bi 101a. As depicted in Figure 1c, the responder 102* can also record the initiator bootstrap public key Bi 101a, which can be used to authenticate the initiator 101 as described below.

For a mutual authentication 142 between device 101 and responder 102*, the two nodes can exchange the same or equivalent messages as message 122 and message 123, which were also described above in connection with Figure 1b. For mutual authentication, the third shared encryption key ke in DPP specification version 1.0 can be derived by a responder 102* using a key exchange step 321 and the received initiator bootstrap public key Bi 101a as depicted and described in connection with Figure 4c below. The use of initiator bootstrap public key Bi 101a for the key exchange step 321 is described in section 6.2.3 of the DPP Specification version 1.0. The difference between the third mutually derived secret shared key ke for a mutual authentication 142 and a responder authentication 141 is the inclusion of initiator bootstrap public key Bi 101a in the key exchange algorithm 401c and key derivation function 402c in Figure 4c. The first mutually derived secret shared key can comprise k1 and the second mutually derived secret shared key can comprise k2. The third mutually derived secret shared key ke is used by the responder 102* to encrypt the responder authentication value as depicted in an encryption step 322 in Figure 4c. The responder authentication value can include a random number “initiator nonce” or I-nonce from the initiator 101 in message 122. The responder authentication value can comprise the value “R-auth” in step 3 of section 6.2.3 of DPP specification version 1.0.

Initiator 101 can also mutually derive the third secret shared encryption key ke using the initiator bootstrap private key bi 101b and key exchange step 327 depicted and described below in connection with Figure 4c. Initiator 101 can subsequently decrypt the responder authentication value using the mutually derived third secret shared encryption key ke and a decryption step 328 as depicted and described in connection with Figure 4c below. Initiator 101 can calculate the same value for the responder authentication value as described in section 6.2.4 of DPP specification version 1.0. If the received, decrypted responder authentication value matches the calculated responder authentication value, then the responder 102* is authenticated.

The initiator 101 can calculate an initiator authentication value using at least the received responder nonce in message 123 and encrypt the initiator authentication value using the mutually derived third secret shared key ke (from a key exchange step 327) and send the initiator authentication value in a message (not shown in Figure 1c). The responder 102* can (i) decrypt the encrypted initiator authentication value using the third mutually derived secret shared key ke (from a key exchange step 327) and (ii) also calculate the initiator authentication value. If the calculated initiator authentication value by the responder 102* matches the received, decrypted initiator authentication value then the initiator 101 is authenticated with the responder 102* for a mutual authentication. The initiator 101 as enrollee can then receive a configuration object in a message 124. Message 124 can include network credentials 109 encrypted with the third mutually derived secret shared key ke. The initiator 101 can receive message 124 and pass the decrypted network credentials 109 to device 106, which can then apply the network credentials 109 in order to communicate with network access point 105.

Figure 1d

Figure 1d is an illustration of an exemplary initiator DPP configuration and initiator key database, with tables recording exemplary data for a plurality of bootstrap responder public keys and bootstrap initiator private keys, in accordance with exemplary embodiments. Although Figure 1b and Figure 1c above depicted and described conventional technology for an initiator 101, data for an initiator 101 depicted in Figure 1d and used by initiator 101 with a responder proxy 102 is not contemplated by conventional technology. An initiator 101 communicating with a responder proxy 102 can record and operate an initiator key database 101t, where an initiator key database 101t was also depicted and described above in connection with Figure 1a. An initiator key database 101t can include a responder public key table 101ta and an initiator key table 101tb.

Responder public key table 101ta can record a plurality of bootstrap responder public keys Br 102a for a plurality of different networks 107. The present disclosure with the use of a responder proxy 102' supports the recording and operation of the corresponding a bootstrap responder private key br 102b solely within a DPP server 103 and/or a device database 104 as depicted in Figure 1a. This represents a significant difference from the conventional technology depicted in Figure 1b and Figure 1c, which contemplates that a responder 102* would operate and record bootstrap responder private key br 102b. By recording responder bootstrap private key br 102b in secure servers within network 107, the corresponding responder bootstrap public key Br 102a can be "reused" or distributed among a plurality of devices 106 during a pre-configuration step or during manufacturing or distribution, before an end user receives the device 106. The reuse of responder bootstrap public key Br 102a can comprise shared keys 102z as depicted in Figure 2a. In other words, the responder

bootstrap public key Br 102a recorded in a device 106 before the device 106 conducts a DPP 191 can comprise a pre-shared bootstrap responder public key.

In addition, a manufacturer or distributor of device 101 may not know which network 107 among a plurality of possible different networks 107 that an end user for device 106 may wish to connect device 106. Since the responder bootstrap public key Br 102a comprises a public key and can be shared, a network 107 could send the responder bootstrap public key Br 102a to the manufacturer or distributor of device 106, where the responder bootstrap public key Br 102a could be recorded in a plurality of devices 106 in nonvolatile memory. Network 107 could securely record the corresponding responder bootstrap private key br 102b in a server database 103a or a device database 104. A manufacturer or distributor of device 106 could collect the publicly shared responder bootstrap public keys Br 102a from a plurality of different networks 107 and record the resulting plurality of different responder bootstrap public keys Br 102a in a responder public key table 101ta. The responder public key table 101ta could also be recorded in a plurality of devices. In this manner, an (a) exemplary device 106 depicted in Figure 1a could record the appropriate responder bootstrap public key Br 102a for a network 107 depicted in Figure 1a before (b) the device 106 is received by an end user.

The use of a responder public key table 101ta simplifies conducting a DPP 191 between responder 102' and initiator 101, since a separate "out of band" transfer of responder bootstrap public key Br 102a in a step 121 of Figure 1b and Figure 1c could be omitted or is not required by device 106 recording key Br 102a for a responder proxy 102'. A responder proxy 102' can operate with potentially many different keys Br 102a recorded by initiator 101, since responder 102' can forward messages such as DPP 122 to DPP server 103 for processing by DPP server 103 (and DPP server 103 could record and operate on the corresponding private key br 102b).

As depicted in Figure 1d, responder public key table 101ta for an initiator 101 can include data or values for a responder bootstrap public key Br identity 101z, a responder network 107a name or identity, a responder bootstrap public key Br 102a, a secure hash value 102aa over the key Br 102a, and a set of parameters 199 for the public key Br 102a. A responder public key table 101ta could record a plurality of data for a plurality of network 107a names with corresponding key Br 102a. In exemplary embodiments, at least one responder bootstrap public key Br 102a is recorded in a table 101ta for initiator 101. In exemplary embodiments data for an initiator key database 101t can be recorded in protected nonvolatile memory within device 106 and accessible only to initiator 101 or root or privileged processes operating in a device 105.

In exemplary embodiments, an initiator 101 for a device 106 could record a plurality of responder bootstrap public keys Br 102a in a responder public key table 101ta. Reasons could include the use of different keys Br 102a for different purposes, such as a first key Br 102a for a responder authentication 141 and a second key Br 102a for a mutual authentication 142. Or, a third key Br 102a could be recorded or mapped in a tag 102v from responder 102' and received in a message 121.

Responder 101x can designate the use of select keys Br 102a with a higher level of trust, with the following exemplary results (A) and (B). For (A), device 106 could provide additional privileges for a configuration step 106a for an initiator 102' or initiator 102* that uses the second key Br 102a (e.g. a mutual authentication 142 with the second key Br 102a could require responder 102' to have key Bi 101a, and release of key Bi 101a could be under the control of device 106 or recorded only in a device database 104 in a network 107). As one example of additional privileges, a set of credentials 109 delivered to device 106 using the second key Br 102a with mutual authentication 142 using key Bi 101a could allow the installation of a new certificate authority root certificate for device 106. As one example of normal or other privileges, a set of credentials 109 delivered to device 106 using the first key Br 102a could deny the installation of a new certificate authority root certificate for device 106.

As depicted in Figure 1d, initiator key table 101tb in an initiator key database 101t can record values for an initiator bootstrap key identity 101w, a set of parameters 199 for the initiator bootstrap private key bi 101b, the initiator bootstrap public key Bi 101a, a secure hash value 101aa over the key Bi 101a, and the initiator bootstrap private key bi 101b. In exemplary embodiments, an initiator key database 101t can include at least one initiator bootstrap public key Bi 101a in order to support a mutual authentication 142 depicted in Figure 1c. One use of an initiator key table 101tb is also depicted and described in connection with Figure 2b below.

As depicted in Figure 1d, initiator key table 101tb can record data for a plurality of different initiator bootstrap PKI keys. The initiator bootstrap PKI keys could be different according to the set of parameters 199. In other words, a first key bi 101b could be for parameters 199 of "A" with a key length of 256 bits, and a second key bi 101b could be for parameters 199 of "B" with an exemplary length of 384 bits. Different elliptic curves could be specified by parameters 199 for different keys in an initiator key database 101t. In exemplary embodiments where initiator 101 can provide bootstrap public key Bi 101a "in band" to responder 102', such as, but not limited to, initiator 101 using a PKEX key exchange, then table 101tb could record a plurality of different key bi 101b and Bi 101a for the same parameters 199, and initiator 101 could select and deprecate different keys bi 101b and Bi 101a over time. One use of an initiator key table 101tb is also depicted and described on connection with Figure 2b below.

As depicted in Figure 1a, an initiator DPP configuration 130 for initiator 101 is depicted in Figure 1d. An initiator DPP configuration 130 can comprise a collection of data and values recorded in nonvolatile memory for WiFi radio 106r. Initiator DPP configuration 130 can be similar or compatible with a responder DPP configuration 131 as depicted and described in connection with Figure 1f below. Initiator DPP configuration 130 can record values for a device supported configuration 130z, a global operating class 130a and a channel list 130b. Global operating class 130a can specify a WiFi technology and frequency band for operating WiFi radio 106r, and can represent the capabilities of the radio in a manufactured device 106. Global operating class 130a can support

exemplary class values and numbers in Appendix E of the 802.11ac specification, and other possibilities exist as well without departing from the scope of the present disclosure. Channel list 130b can comprise a list of channels for initiator 101 to transmit DPP authentication request messages 122. An initiator 101 in an unconfigured device 106' can rotate through a list of Global operating classes and channel numbers in a channel list 130b and transmit a series of DPP authentication request messages 122 on each combination of global operating class 130a and channel number from a channel list 130b in order to send message 122 to a responder 102' operating with a responder configuration 131.

Figure 1e

Figure 1e is an illustration of an exemplary responder bootstrap public keys and initiator bootstrap PKI keys recorded by a plurality of devices, in accordance with exemplary embodiments. Figure 1e depicts PKI keys recorded for an exemplary three different devices 106, although a system 100 and other systems herein could operate with potentially millions or more devices 106. The data depicted for each device in Figure 1e can comprise exemplary data for a responder public key table 101ta and an initiator key table 101tb within an initiator key database 101t, which were depicted and described in connection with Figure 1d above.

The first two entries for responder bootstrap public keys Br 102a for device 106 (1) and device 106(2) in Figure 1e depicts the same alphanumeric values for base85 binary to text encoding for an exemplary responder bootstrap public keys Br 102a in a first device 106(1) and a second device 106(1). Likewise, the second two entries for responder bootstrap public key Br 102a for device 106 (1) and device 106(2) in Figure 1e depicts the same alphanumeric values for base85 binary to text encoding for an exemplary second responder bootstrap public keys Br 102a in first device 106(1) and a second device 106(1). The depiction of these keys Br 102a illustrates the use of shared keys 102z for a plurality of different devices 106. Although only two devices are depicted with shared keys 102z, many more devices could also record the same shared keys for Br 102a. Each of the shared keys 102z is associated with a different network 107, identified with an exemplary network name 107a. In this manner, a plurality of different devices 106 can record and use the same value for a responder bootstrap public key Br 102a.

The same values for shared keys 102z across different devices 106 could be recorded in device 106 during manufacturing or before distribution to end users of device 106. In this manner, devices 106 could be received by end users in a "partially configured" yet secure state, such that a device 106 could use the recorded keys Br 102a with a responder proxy 102', where a responder proxy 102' does not operate or record the corresponding responder bootstrap private key br 102b. As depicted and described in connection with Figures 3a, 3b, 5a, and 5b below, a DPP server 103 could store and

operate with the corresponding responder bootstrap private key br 102b and thus the key br 102b can remain secured and not distributed out or sent to a responder proxy 102'. In this manner, a DPP 191 can be conducted between initiator 101 and a responder proxy 102' securely, where responder proxy 102' could operate on a device with a lower level of security or trust than DPP server 103. In some exemplary embodiments, responder proxy 102' could operate within an insecure device. Although not depicted in Figure 1e or Figure 1d, an entry for a key Br 102a could also record a "trust level" or "privilege level", such that a configuration object received from a DPP 191 using the key Br 102a with the corresponding privilege level could be authorized by device 106 to perform tasks or operations or record data with a higher level of privilege or trust.

In addition, the use of the same values for shared keys 102z across different devices 106 supports simplified yet secure automatic configuration of devices using a DPP 191, where WiFi access points 105 could operate as responder proxies 102'. As depicted and described in connection with Figure 1f below, a WiFi router operating as a responder proxy 102' could potentially connect with at least one network 107 and DPP server 103 (or perhaps a plurality of DPP servers 103 for different networks 107). Responder proxy 102' could use the hash value 102aa for the responder bootstrap public key Br 102a to determine a network 107 identified by the hash value 102aa (see responder network tables 102t below in Figure 1f). In this manner, responder proxy 102' could route data from a DPP authentication request message 122 to the correct DPP server 103 which operates and records (or can obtain from a device database 104 for network 107) the corresponding responder bootstrap private key br 102b in order to conduct the full series of messages for a device provisioning protocol 191.

In exemplary embodiments, responder bootstrap public keys Br 102a can also comprise a unique key such as a unique key 102vv recorded in a tag 102v for responder 102. Thus exemplary embodiment also support the use of a responder bootstrap public key Br 102a that is not shared across multiple initiators 101. In exemplary embodiments, and as depicted in Figure 1e, the initiator bootstrap private key bi 101b and corresponding initiator bootstrap public key Bi 101a can be different for different devices 106. In this manner, a device 106 can also be identified by the hash value 101aa (depicted in Figure 1d) over the initiator bootstrap public key Bi 101a by responder 102' or network 107. As depicted in Figure 1e, an initiator 101 can record a different initiator bootstrap private key bi 101b and corresponding initiator bootstrap public key Bi 101a for each set of supported parameters 199a. In exemplary embodiments, the initiator bootstrap private key bi 101b and corresponding initiator bootstrap public key Bi 101a can be recorded in a secure nonvolatile memory within device 106 for initiator 101 upon device manufacturing or during device distribution.

Figure 1f

Figure 1f is a graphical illustration of hardware, firmware, and software components for a responder, including a responder configuration step, in accordance with exemplary embodiments.

Figure 1f is illustrated to include several components that can be common within a responder 102. Responder 102 may consist of multiple electrical components in order to both (i) communicate with a network 107 and (ii) transmit and receive messages with an initiator 101 according to a device provisioning protocol 191. Responder 102 can undergo a responder configuration step 127 in order to configure the components in a responder 102 to function as the enhanced responder or responder proxy as contemplated in the present disclosure, such that a responder bootstrap private key br 102b can remain recorded with a network 107 as depicted in Figure 1a above. In addition, although a responder 102 in Figure 1a is depicted as a separate component in a network access point 105, a network access point 105 could function and operate with the components of a responder 102 or responder 102' in Figure 1f. In other words, a network access point 105 or another wireless computing device with a WiFi radio 102x could comprise a responder 102 or responder 102' as contemplated herein. Responder 102 and configured responder 102' could also operate with a radio 102x that supports wireless networking technology other than WiFi, such as radio 102x comprising a radio for 4G LTE or 5G networks, and other possibilities exist as well.

In exemplary embodiments and as depicted in Figure 1f, responder 102 can include a responder identity 102i, a processor 102c (depicted as "CPU 102c"), random access memory (RAM) 102d, an operating system (OS) 102e, storage memory 102f (depicted as "nonvolatile memory 102f"), a WiFi radio 102x, a system bus 102j, and a wireless wide area network (WAN) interface 102h. Figure 1d depicts responder 102 as both a default responder 102 and a configured responder 102'. The default responder 102 can comprise the state of responder 102 before a configuration step 127 and responder 102' can comprise the state of responder 102' after the configuration step 127. As depicted, both a default responder 102 and a configured responder 102' can contain the same hardware components such as CPU 102c, RAM 102d, etc., where a difference between responder 102 and 102' can be in both (i) data recorded in storage memory 102f and a change in the operation of WiFi radio 102x to listen for incoming DPP messages 122.

Responder identity 102i could comprise a preferably unique alpha-numeric or hexadecimal identifier for responder 102, such as an Ethernet or WiFi MAC address, an International Mobile Equipment Identity (IMEI), an owner interface identifier in an IPv6 network, a serial number, or other sequence of digits to uniquely identify each of the many different possible units for responder 102 for a network 107. A system 100 depicted in Figure 1a could operate with a plurality of different responders 102. Responder identity 102i can preferably be recorded in a non-volatile memory or written directly to hardware in responder 102 by (i) a responder manufacturer upon device

manufacturing, or (ii) a pre-configuration step for responder 102 before starting the sequence of steps and message flows depicted in Figure 3a below.

The CPU 102c can comprise a general purpose processor appropriate for typically low power consumption requirements for a responder 102, and may also function as a microcontroller. CPU 102c can comprise a processor for responder 102 such as an ARM® based processor or an Intel® based processor such as belonging to the Atom or MIPS family of processors, and other possibilities exist as well. CPU 102c can utilize bus 102j to fetch instructions from RAM 102d and operate on the instruction. CPU 102c can include components such as registers, accumulators, and logic elements to add, subtract, multiply, and divide numerical values and record the results in RAM 102d or storage memory 102f, and also write the values to an external interface such as WiFi radio 102x or WAN interface 102h.

RAM 102d may comprise a random access memory for responder 102. RAM 102d can be a volatile memory providing rapid read/write memory access to CPU 102c. RAM 102d could be located on a separate integrated circuit in responder 102 or located within CPU 102c. The RAM 102d can include data recorded in responder 102 for the operation when collecting and communicating DPP messages 122 through 124 with initiator 101 depicted in Figures 1b, 1c, 3a, etc. The system bus 102j may be any of several types of bus structures including a memory bus or memory controller, a peripheral bus, and a local bus using any of a variety of bus architectures including a data bus. System bus 102j connects components within responder 102 as illustrated in Figure 1f, such as transferring electrical signals between the components illustrated. Responder 102 can include multiple different versions of bus 102j to connect different components, including a first memory bus 102j between CPU 102c and RAM 102d, and a second system bus 102j between CPU 102c and WiFi radio 102x, which could be an I2C bus, an SPI bus, PCIe bus, or similar data busses.

The operating system (OS) 102e can include Internet protocol stacks such as a User Datagram Protocol (UDP) stack, Transmission Control Protocol (TCP) stack, a domain name system (DNS) stack, a TLS stack, a datagram transport layer security (DTLS) stack, etc. The operating system 102e may include timers and schedulers for managing the access of software/firmware to hardware resources within responder 102, including CPU 102c and WiFi radio 102x. The operating system shown of 102e can be appropriate for a low-power device with more limited memory and CPU resources (compared to a server such as DPP server 103, device database 104, etc.). Example operating systems 102e for a responder 102 includes Linux, Android® from Google®, IOS from Apple®, Windows® 10 IoT Core, or Open AT® from Sierra Wireless®. Additional example operating systems 102e for a responder 102 include eCos, uC/OS, LiteOs, Contiki, OpenWRT, Raspbian, and other possibilities exist as well without departing from the scope of the present disclosure. Although depicted as a separate element within responder 102 in Figure 1f, OS 102e may reside in RAM 102d and/or storage memory 102f during operation of responder 102.

Storage memory 102f (or “nonvolatile memory 102f”) within responder 102 can comprise a nonvolatile memory for storage of data when responder 102 is powered off. Memory 102f may be a NAND flash memory or a NOR flash memory, and other possibilities exist as well without departing from the scope of the present disclosure. Memory 102f can record firmware and/or software for responder 102. Memory 102f can record long-term and non-volatile storage of data or files for responder 102. In an exemplary embodiment, OS 102e is recorded in memory 102f when responder 102 is powered off, and portions of memory 102f are moved by CPU 102c into RAM 102d when responder 102 powers on. Memory 102f (i) can be integrated with CPU 102b into a single integrated circuit (potentially as a “system on a chip”), or (ii) operate as a separate integrated circuit or a removable card or device, such as a removable SD card. Memory 102f may also be referred to as “device storage” and can include exemplary file systems of FAT16, FAT 32, NTFS, ext3, ext4, UDF, or similar file systems.

As depicted in Figure 1f, nonvolatile memory 102f for a default responder 102 may also contain a secret key SK0.responder 102s, a certificate cert.responder 102m, and a certificate cert.CA.root 102w. A secret key SK0.responder 102s in a nonvolatile memory 102f could comprise the private key for a PKI key pair, where the corresponding public key could be recorded in a certificate cert.responder 102m. A secret key SK0.responder 102s and the corresponding public key in a certificate 102m could comprise values to use with asymmetric public key infrastructure (PKI) cryptography, and could utilize exemplary algorithms based on either Rivest Shamir Adleman (RSA) or elliptic curve cryptography (ECC). For use of ECC algorithms, parameters within certificate 102m (and other certificates herein) can specify elliptic curve names (e.g. NIST P-256, sect283k1, sect283r1, sect409k1, sect409r1, etc.). The same ECC parameters can be used with bootstrap PKI keys in the form of parameters 199a. Further, elliptic curves that do not depend on parameters specified by NIST could be utilized as well, such as Curve22519 or FourQ. For use of RSA algorithms, parameters within certificate 102m can specify a modulus and other associated values for using an RSA PKI key pair. For either asymmetric algorithms RSA or ECC for a PKI key pair herein, parameters in a certificate such as cert.responder 102m can specify key lengths, a duration for key validity, uses of the PKI key pair such as for key derivation or signatures, encoding rules, message digest algorithms, etc. Parameters in certificate 102m (and other certificates herein) may also include identifying information associated with the PKI key pair such as a sequence number, a serial number, a certificate revocation list or URL, and a domain name of the server associated with the PKI key pair.

Certificate cert.CA.root 102w can comprise a root certificate for responder 102 to authenticate a server certificate received from DPP server 103 or other servers or nodes in a network 107, where these other nodes for network 107 are depicted and described in connection with Figure 1a, Figure 3a, and other Figures below. After authenticating a server certificate received by responder 102 or 102' with cert.CA.root 102w, responder 102 can conduct additional authentication steps contemplated by

secure protocol standards such as Transport Layer Security (TLS), Datagram Transport Layer Security (DTLS), IPSec, etc. Root certificate cert.CA root 102w can comprise the list or a subset of the list of included root certificates from the Mozilla Foundation with Mozilla projects, where the aggregate list of community approved root certificates and associated parameters is in the widely distributed file certdata.txt from the Mozilla web site.

Nonvolatile memory 102f in a configured responder proxy 102' can include a responder network table 102t. Responder network table 102t could be received from a network 107 for a configuration step 127 in a message 305 depicted below in Figure 3a. Or, table 102t could be loaded by a user of responder 102' or potentially recorded during a pre-configuration of responder 102 or responder 102' before the sequence of message flows and steps in Figure 3a below. Responder network table 102t can record and map a DPP server 103 that is associated with a responder bootstrap public key hash value 102aa. The responder bootstrap public key hash value 102aa can be associated with a network 102ab for responder bootstrap public key Br 102a (e.g. the network equivalent to network 107 that records the responder bootstrap private key br 102b to process an incoming DPP authentication request message 122). The configured responder proxy 102' can use the responder network table 102t to route data from incoming DPP messages 122 with the responder bootstrap public key hash value 102aa to a selected DPP server 103. Different DPP servers 103 could be associated with different networks 107, where a network 107 can comprise a network for H(Br) 102aa, as depicted for a responder network table 102t in Figure 1f. Although not depicted in Figure 1f, a responder network table 102t could also include a set of cryptographic parameters 199a associated with the responder bootstrap public key hash value 102aa, including the hash algorithm 199e associated with the responder bootstrap public key hash value 102aa.

In some exemplary embodiments, the responder network table 102t with a plurality of responder bootstrap public key hash values 102aa could be omitted from a memory of responder 102, and a DPP server 103 or network 107 could record the values. A network 107 or DPP server 103 could use a hash value 102aa received in a message 309 (depicted and described below in connection with Figure 3a) in order to select a DPP server 103. In other exemplary embodiments, a responder network table 102t could include at least one value for a responder bootstrap public key hash value 102aa.

In other words, a plurality of different devices 106 with initiators 101 may use different responder bootstrap public keys Br 102a, where the hash value 102aa would be received in an incoming DPP message 122. The devices 106 could be manufactured or pre-configured with at least one key Br 102a, and the manufacturer could operate or be associated with a DPP server 103 in order to support and conduct a DPP 191 with device 106 and send a configuration object in a DPP configuration response message 124 from Figure 1c. Note that a configuration object in a message 124 could include data other than credentials 109 (and may optionally omit credentials 109), where

the configuration object could include configuration parameters for device 106. Responder 102' can use the received hash value 102aa to select a DPP server 103 in order to further process the message and support a DPP 191, where the network 102ab could be different than a network 107 operating a network access point 105. In exemplary embodiments a network 102ab operating a DPP server with responder bootstrap private key br 102b can be selected using the hash value 102aa over key responder bootstrap public key Br 102a.

A device 106 with initiator 101 could use (i) a first DPP 191 with a first Br 102a to obtain network credentials 109 with network 107 for AP 105, and (ii) a second DPP 191 with a second Br 102a (with the same responder proxy 102') to obtain configuration parameters with network 102ab for device 106. The first DPP 191 could be with a first DPP server 103 and first network 102ab, and the second DPP 191 with a second DPP server 103 and a second network 102ab. The use of responder network table 102t by responder 102' can determine which incoming DPP message 122 is associated with which network 102ab and DPP server 103. In other words, a responder 102' can conduct multiple sessions of DPP 191 with different keys Br 102a, and use a responder network table 102t to determine which DPP messages 122 are associated with which DPP server 103 (via hash value 102aa). Other possibilities exist as well for a responder 102' to use a responder network table 102t in order to process different incoming DPP authentication request messages 122 without departing from the scope of the present disclosure. In exemplary embodiments, a selected DPP server 103 from a responder network table 102t can provide credentials 109 for a DPP configuration response 124, using the steps below such as those depicted in Figure 3a, 3b, Figure 6, etc.

Responder 102 and 102' can include a WiFi radio 102x to communicate wirelessly with networks such as device 101 depicted and described in Figure 1a above, including via WiFi data link 192. WiFi radio 102x could connect with an antenna in order to transmit and receive radio frequency signals. Responder 102 and 102' can operate a WAN interface 102h in order to communicate with network 107, and WAN interface can include a WAN MAC address 102q. WAN interface 102h can comprise either a wireless connection, such as with a 4G LTE network, a 5G network, or can comprise a wired connection such as digital subscriber line (DSL), coaxial cable connection, or Ethernet connections, and other possibilities exist as well without departing from the scope of the present disclosure.

As depicted in Figure 1f, a WiFi radio 102x in a configured responder 102' can record or operate with a responder configuration 131. The values for responder DPP configuration 131 could also specify a version of the 802.11 standards to utilize, such as, but not limited to, 802.11n, 802.11ac, 802.11ah, 802.11ax, or related and subsequent versions of these standards. The responder DPP configuration 131 can be compatible with the initiator DPP configuration 130 of a target WiFi device 101 with responder 102'. The responder configuration 131 could be obtained during an authentication step for responder 102, such as depicted below in Figure 3a (in a message 305), although responder

configuration 131 could be obtained at other times as well, such as possibly being recorded in a WiFi radio 102x before responder 102 begins sending and receiving data through a WiFi link 192.

The responder DPP configuration 131 can include a responder operating class 131a which can be selected from the plurality of potential global WiFi operating classes in table E-4 of IEEE standard specification 802.11ac, which is incorporated by reference herein. The present disclosure contemplates that a responder 102' communicates with multiple initiators 101 either concurrently, or over time. The responder configuration 131 can also include Responder DPP Channel Authentication List 131b, which can comprise a list of channels for responder 102' to receive DPP authentication request 122 messages in order to establish initial connectivity with device 106 and initiator 101. In exemplary embodiments, responder configuration 131 for responder 102' provides data for a responder 102 to receive data such as message 122 from initiator 101, where initiator 101 may operate with different possible values for operating classes or transmitting channels.

A responder configuration 131 can include other data for responder 102' to receive a DPP message from an initiator 101 as well, such as specifying the use of broadcast or unicast for receiving DPP messages, timer values for retry of sending DPP 191 messages, timer values for timeouts for sending and receiving messages, as well as counters for a maximum number of retries on sending messages. In other words, without responder configuration 131, a responder 102 may not normally be able to (i) select the proper WiFi operating channel, and (ii) receive broadcast data from initiator 101. For embodiments where a Device DPP Channel Authentication List 131b is unknown or not available, then responder 102' can periodically listen for incoming DPP 122 messages over time as broadcast messages on all available channels for a radio 102x.

WiFi radio 102x can include standard radio components such as RF filters, RF amplifiers, a clock, and phased loop logic (PLL). WiFi radio 102x can support protocols and specifications according to the IEEE 802.11 family of standards. For example, if WiFi radio 102x operates according to 802.11n standards, WiFi radio 102i could operate at a radio frequency of 2.4 or 5 GHz. In other embodiments where WiFi radio 102i supports 802.11ac standards, then WiFi radio 102x could also support radio frequencies of 0.054 through 0.79 GHz in order to operate in "white space" spectrum. Other possibilities exist as well for WiFi radio to support wireless LAN standards or 802.11 standards and radio frequencies, without departing from the scope of the present disclosure. WiFi radio 102x can access a nonvolatile memory 102f in order to record the depicted configuration values of responder configuration 131 where the nonvolatile memory could be within WiFi radio 102x, or WiFi radio 102x could access memory 102f via bus 102j in order to record and read the values. Device 106 can operate a WiFi radio 106r that is compatible with WiFi radio 102x.

As depicted in Figure 1f, a WiFi configured responder 102' can include configuration data resulting from a configuration step 127, in order for responder 102' to conduct a device provisioning protocol 191 with a device 101. A nonvolatile memory 102f for a WiFi configured responder 102'

can include data recorded for a default responder 102, with the addition of recording a DPP application program 102g. DPP application program 102g (or “DPP application 102g”) could comprise a software package for OS 102e in order for responder 102’ to conduct the series of steps and messages with DPP server 103 and initiator 101 depicted below in Figure 3a, Figure 3b, Figure 4a, etc. DPP application 102g can include cryptographic algorithms in order to support key exchanges, symmetric encryption and decryption, as well as signature verification and asymmetric encryption and decryption. As one example, a DPP application 102g could perform the function of a key derivation 102y from Figure 1b, and other possibilities exist as well.

Responder 102 may also optionally include user interface 102k which may include one or more devices for receiving inputs and/or one or more devices for conveying outputs. User interfaces are known in the art and may be simple for many devices such as a few LED lights or and LCD display, and thus user interfaces are not described in detail here. User interface 102k could comprise a touch screen if responder 102 has sophisticated interaction with user 102a. Responder 102 can optionally omit a user interface 102k, if no user input or display is required for conducting a device provisioning protocol 191 between responder 102 and device 106 with initiator 101. Although not depicted in Figure 1f, responder 102 can include other components to support operation, such as a clock, power source or connection, antennas, etc. User interface 102k could also comprise a web server operating in responder 102 and responder 102’, such that a user could access the user interface 102k with a web browser and view or control depicted data stored in memory for a responder 102 or configured responder 102’. Other possibilities exist as well for hardware and electrical components operating in responder 102 or responder 102’ without departing from the scope of the present disclosure.

Although not depicted in Figure 1f, the various servers shown above in Figure 1a such as DPP server 103 and device database 104 can include equivalent internal electrical components depicted for a responder 102 in order to operate as servers. The servers in Figure 1a could include a processor similar to CPU 102c, with primary differences for the processor in a server being increased speed, increased memory cache, an increased number and size of registers, with the use of a 64 bits for datapath widths, integer sizes, and memory address widths, etc., In contrast, an exemplary 32 bit datapaths could be used for CPU 102c in responder 102 (although CPU 102c could also use 64 bit wide datapath widths if responder 102 comprises a mobile phone such as a smart phone). For embodiments where responder 102 comprises a home WiFi router, then a CPU 102c in responder 102 could comprise an exemplary 32 bit processor, although other possibilities exist as well.

Similarly, RAM in a server could be a RAM similar to RAM 102d in responder 102, except the RAM in a server could have more memory cells such as supporting exemplary values greater than an exemplary 8 gigabytes, while RAM 102d in responder 102 could support fewer memory cells such as less than an exemplary 4 gigabytes. Non-volatile memory for storage in a server in Figure 1a could comprise disks, “solid state drives” (SSDs) or “storage area networks” (SAN) for a server. For a WAN

interface in a server equivalent to WAN interface 102h for responder 102', in exemplary embodiments, a server in Figure 1a could use a physical, wired LAN interface such as a high-speed Ethernet or fiber optic connection. In exemplary embodiments, a device 106 with an initiator 101 can include and operate with hardware and electrical components similar or equivalent to the components for a responder 102' depicted in Figure 1f.

Figure 2a

Figure 2a is an illustration of an exemplary device database, with tables for a device database recording exemplary data, in accordance with exemplary embodiments. A device database 104 depicted and described above in connection with Figure 1a can record data for (i) an unconfigured responder 102 or (ii) a configured responder 102' and a DPP server 103 to conduct a device provisioning protocol 191 with a device 106 operating an initiator 101. A database 104 could record a plurality of tables for a plurality of devices 106 and/or responders 102', such as the depicted values for a key table 104k, and a set of cryptographic parameters 199. Other possibilities exist as well for the organization, tables, and recorded data within a device database 104 without departing from the scope of the present disclosure. Data within device database 104 could be encrypted using a symmetric key. As depicted in Figure 1a, a network 107 could operate and record a device database 104, although another entity besides network 107 could also operate and record the device database 104, such as a device manufacturer or a device owner. For embodiments where a plurality of different networks 107 may communicate with a device 106 through responder 102' over time, each of the different networks 107 can operate a device database 104.

Device database 104 can record sets of ID.device 106i, MAC.I 106n for the WiFi MAC address of the device 106 operating the initiator 101, responder identity 102i, a responder mode 306a, cryptographic parameters 199, and a secret shared key 198 for a public key exchange protocol (PKEX). These values within a device database could be recorded in a device 106 and/or initiator 101 upon or after manufacturing of a device 106 or distribution of device 106 and before device 106 conducts a DPP 191 with responder 102'. The exemplary data in device database 104 could be obtained from the manufacturer or device distributor. For embodiments where some data is not previously recorded in a device database 104 before an initiator 101 sends a message 122 as depicted below in Figure 3a, then a network 107 operating device database 104 could query the device 106 manufacturer or a device 106 distributor in order to receive and record values for a device database 104, and other possibilities exist as well for the source of device 106 data within a device database 104. Note that key 198 for PKEX can be optionally omitted for some devices, with the depiction of "NA".

ID.device 106i can comprise an identity for device 106i, and could be a serial number or identification number or string for device 106. The two identity values shown for a device 101 in a

device database 104 allows a DPP server 103 or network 107 to map or convert between ID.device 106i and MAC.I 106n, and also convert one value into another. MAC.I 106n can comprise the MAC address used by an initiator 101 for conducting a device provisioning protocol 191. In other words, MAC.I 106n can be used by a responder 102' or network 107 in order to identify that an incoming DPP authentication request message 122 is from device 106 (and not a fraudulent or imposter device 106). For embodiments where key Br 102a comprises a shared key 102z, MAC.I 106n can also be used by network 107, database 104, DPP server 103, or responder 102' to verify the message 122 is from device 106 (since only device 106 with MAC.I 106n would normally send the message 122). A message 122 would have a source MAC address, which could be checked against MAC.I 106n that was recorded in a device database 104. Further, the hash value 101aa from a message 122 could be checked against a hash value calculated by device database 104 or DPP server 103 over the recorded initiator bootstrap public key Bi 101a. Additional information for verifying an identity for device 106 is provided for a step 312 below in Figure 3a.

In some exemplary embodiments, MAC.I 106n can be omitted from a device database 104 or otherwise not available to a network 107, and in this case the MAC.I 106n can be recorded in a database 104 after successful receipt of a DPP authentication confirm message 123a depicted in Figure 3b below. In other words, some devices 101 with initiators 106 may change or randomize the MAC.I 106n over time, and the MAC.I 106n stored in a device database 104 can comprise the value of MAC.I 106n used by device 106 to send a DPP authentication request message 122. A network 107 with a device database 104 could operate a plurality of different responders 102'. The responder identity 102i of a responder 102' communicating with a device 106 could be recorded in a device database 104, although in some exemplary embodiments the responder identity 102i could be optionally omitted. Responder identity 102i could be omitted or blank in a device database 104 until after a message 122 is received by a responder 102' and forwarded to device database 104 via network 107.

A device database 104 could also specify a responder mode 306a (in Figure 3a and Figure 3c below) for responder 102' to operate with initiator 101, where different responder modes could comprise a first responder mode as depicted in Figure 3a, where DPP server 103 records and operates on (a) responder bootstrap private key br 102b and (b) ephemeral PKI keys for responder 102' and (c) all derived secret shared keys (as depicted in the first row of Figure 3c). A second responder mode 306a could comprise DPP server 103 recording and operating on responder bootstrap private key br 102b, but not derived responder ephemerals private key pr 102r for responder 102' (e.g. as depicted in Figure 5a). A third responder mode 306a could comprise DPP server 103 recording and operating on responder bootstrap private key br 102b and derived responder ephemeral private key pr 102r, but not derived secret shared keys for encryption such as keys k1, k2, and ke (e.g. as depicted in Figure 5b).

In some exemplary embodiments, a device 106 could comprise a higher value device 106 or require a greater level of security, so the security level of a DPP 191 could be selected using a responder mode 306a. In addition, the responder mode 306a could be selected based on the security level of a responder 102', such that a first responder mode 306a such as in Figure 3a and Figure 3b is selected by network 107 for a responder 102' with a lower level of security. A second responder mode 306a such as depicted in Figure 5b is selected by a network for a responder 102' with a higher level of security, such as a gnodeb in a 5G network or a secured enterprise WiFi router. Other reasons for the selection of a responder mode exist as well without departing from the scope of the present disclosure.

Sets of cryptographic parameters 199 in device database 104 can represent different sets of cryptographic parameters supported by responder 102' and/or initiator 106 in order to conduct a device provisioning protocol 191. For example, some devices 106 may use a first set of cryptographic parameters with PKI keys such as bootstrap and ephemeral keys, and other devices may use a second set of cryptographic parameters, which could be specified by cryptographic parameters 199. Note that devices 101 can optionally support more than one set of cryptographic parameters, as depicted for device with exemplary identity ID.device 106i of "20003", where the device can operate with either a set "A" of cryptographic parameters 199 or a set "B" of cryptographic parameters.

PKEX key 198 could be used with a public key exchange protocol as specified in IETF draft document "Public Key Exchange", which is hereby incorporated by reference in its entirety. In some exemplary embodiments, a PKEX key 198 can be omitted, such as for the example data depicted with an "NA" in Figure 2b. The values depicted of X1, X2, etc. for a PKEX key 198 could comprise the secret key for each device 106 identified by MAC.I 106n. For the exemplary database 104 depicted in Figure 2a and other tables herein, a numeric value following a constant name can depict a subsequent or different instance or numeric value for the constant. In other words, "X1" and "X2" for a PKEX key 198 can represent different values of "X" for the PKEX. PKEX key 198 could be used as depicted and described in Figure 3a with a PKEX session 312e in order to query device 106 for the initiator bootstrap public key Bi 101a if a database 104 does not record the key Bi 101a (or if network 107 cannot obtain key Bi 101a via other means via an IP network 113).

Device database 104 can record a key table 104k, which could comprise exemplary keys as depicted. A key table 104k can include responder bootstrap public key Br 102a, responder bootstrap private key br 102b, initiator bootstrap public key Bi 101a, a secure hash value for responder bootstrap public key 102aa, and a secure hash value for initiator bootstrap public key 101aa. Similarly, values of "Br-1" and "Br-2" depict different values for a responder bootstrap public key 102a, but exemplary values of "Br-1" and then a second "Br-1" can depict that the same value or number may be recorded as the responder bootstrap public key Br 102a in different devices 106 each with an initiator 101. The use of the same value for a responder bootstrap public key Br 102a across different devices 106 can

comprise a shared responder bootstrap public key 102z, which is also depicted and described in connection with Figure 1e above.

Responder bootstrap public key Br 102a in a device database 104 can represent the bootstrap public key for responder 102'. Br 102a was also depicted and described above in connection with Figure 1a, Figure 1b, and Figure 1e above. In exemplary embodiments, a device database 104 can record a plurality of different initiator bootstrap public keys Bi 101a, where each key for a set of parameters 199 can be associated with different initiators 101. As depicted in Figure 2b, for some embodiments different initiators 101 could also share the same numeric value for an initiator bootstrap public key Bi 101a (e.g. depicted devices "20001" and "20002" can both use an initiator bootstrap public key Bi 101a of "Bi-1"). Also note that a DPP 191 can be conducted between a responder 102' and initiator 101 without recording an initiator bootstrap public key Bi 101a in a device database 104 or DPP server 103 or responder 102', which could comprise a responder only authentication 141 as depicted in Figure 1b. For these embodiments, the depicted value for Bi 101a can be "NA" as shown in Figure 2a.

In a key table 104k in device 104, different initiator bootstrap public keys Bi 101a can be associated with a different selected set of parameters 199a. The manufacturer or entity configuring initiator 101 in device 106 may not know which of a plurality of different possible sets of cryptographic parameters 199 may be utilized by an initiator 101 for conducting a DPP 191 in order to be compatible with a responder 102'. For example, DPP specification version 1.0 identifies several possible sets of cryptographic parameters 199 for bootstrap keys that could be utilized in a DPP 191, such as the different values for cryptographic parameters shown in Appendix A of DPP specification version 1.0. As depicted in Figure 2a, an initiator bootstrap public key Bi 101a of "Bi-5 (A)" can represent an instance or number for Bi 101a which uses the selected set "A" of cryptographic parameters 199 (e.g. curve p256 with SHA-256), while initiator bootstrap public key Bi 101a of "Bi-5 (D)" can represent an instance of Bi 101a which uses the selected set "D" of cryptographic parameters 199 (e.g. curve 25519 with SHA-3). Similar designations and uses of different cryptographic parameters 199 with different responder PKI keys can be recorded in a device database 104 as well. In some exemplary embodiments, the values depicted for responder keys could be omitted from a device database 104 and recorded in another server such as DPP server 103.

Responder bootstrap public key Br 102a and corresponding responder bootstrap private key br 102b in a device database 104 can represent the bootstrap public key and the bootstrap private key for a responder proxy 102'. Br 102a and br 102b were depicted and described above in connection with Figures 1a through Figure 1e above. In exemplary embodiments, a device provisioning protocol 191 using a responder 102' and a DPP server 103 can utilize a plurality of different responder bootstrap PKI keys, and different values for the responder bootstrap PKI keys Br 102a and br 102b can be associated with different initiators 101. As discussed above in connection with Figure 1a and 1b, an

initiator 101 could record or obtain a responder bootstrap public key Br 102a, and subsequently the value Br 102 recorded for the initiator 101 could also be stored in a device database 104.

The responder bootstrap public key Br 102a and corresponding responder bootstrap private key br 102b could be stored by a device database 104 concurrently with the initiator bootstrap public key Bi 101a, where the initiator bootstrap public key Bi 101a could be (i) received from a device manufacturer or device distributor, or (ii) obtained by a PKEX session 312e (in Figure 3a), or (iii) responder 102' scanning a QR code or bar code for initiator 101 (and sending the key Bi 101a to device database 104), and other possibilities exist as well for the source of key Bi 101a in a database 104 for device 106 with MAC.I 106n. Or, in exemplary embodiments a device manufacturer or a device distributor could operate a device database 104 and subsequently record the exemplary keys and values depicted of (i) Bi 101a for initiator 101 in device 106 and (ii) recorded Br 102a in device 106, but not (iii) br 102b, which could be recorded by a device database 104 for a network 107 operating or associated with responder 102'. As depicted in Figure 2a, some responders 102' could share a common responder bootstrap public key Br 102a, which could be associated with the same responder bootstrap private key br 102b.

The sharing of responder bootstrap public key Br 102a among multiple devices 106 with initiators 101 (e.g. "shared key 102z") can simplify the manufacturing and distribution of devices by using a common responder bootstrap public key Br 102a for some embodiments. For the present disclosure, the potential tradeoff of greater simplicity for reduced security (by reusing a responder bootstrap public key Br 102a in multiple devices 106) can be avoided, since the responder bootstrap private key br 102b may (i) preferably not be sent to the responder 102', and (ii) remain recorded in and operated by a secured DPP server 103.

Yet, with the embodiments contemplated herein, the full set of device provisioning protocol messages 191 can be sent and received with an initiator 101 by responder proxy 102'. The full set of DPP 191 messages can be transmitted and received by a responder 102' with initiator 101 using either responder authentication 141 or mutual authentication 142. Embodiments for the present disclosure depict and describe the responder bootstrap private key br 102b as residing within a DPP server 103 when a responder 102' conducts a DPP 191 (as depicted and described in connection with Figure 3a, Figure 3b, Figure 5a, etc. below). Consequently in exemplary embodiments and a difference with conventional technology depicted in Figure 1b and Figure 1c, (i) the responder bootstrap private key br 102a does not need to be recorded by a responder 102' (but br 102a is recorded for a conventional responder 102* in Figure 1b and Figure 1c), and (ii) a responder bootstrap public key Br 102a can be commonly shared between multiple initiators 101 in different devices 106 (e.g. "shared key 102z").

Thus, the present disclosure resolves the potential tradeoff of security versus simplification by supporting manufacturing and distribution of devices 106 with initiators 101 with commonly shared responder bootstrap public key Br 102a (also depicted in Figure 1e as "shared key 102z"). Additional

details will be provided in Figures below. The responder bootstrap private key br 102b can be recorded in a device database 104 and subsequently sent to a DPP server 103, but preferably not sent to responder 102', and responder 102' can still conduct a device provisioning protocol 191 for initiator 101 using mutual authentication 142 as depicted in Figure 1c. In exemplary embodiments, a responder bootstrap public key Br 102a is recorded inside an initiator 102, while a device database 104 can record both the responder bootstrap public key Br 102a and the responder bootstrap private key br 102b, as depicted in Figure 2a.

Cryptographic parameters 199 can specify sets of cryptographic parameters that are supported by responder 102' when using bootstrap PKI keys. Cryptographic parameters 199 can be recorded in a device database 104. Cryptographic parameters 199 can record a collection of cryptographic algorithms or specifications such as a set identifier 199a, a key length 199b, an ECC curve name 199c, a hash algorithm 199e, a key length for a symmetric ciphering algorithm 199f, and a nonce length 199g. Asymmetric encryption parameters 199h can specify values and parameters to use with an asymmetric encryption algorithm 351a depicted and described in connection with Figure 3e below. Parameters 199i can specify encoding rules, padding, key lengths, selected algorithms, curve names, use of deterministic mode, and other values or fields necessary to utilize a signature algorithm 338b depicted and described in connection with Figure 3d below. In exemplary embodiments, parameters 199i can specify ASN.1 syntax notation (abstract syntax notation) with distinguished encoding rules (DER). Parameters 199i can also specify the use of either (i) ephemeral PKI keys or (ii) bootstrap PKI keys for use with a digital signature algorithm.

As contemplated herein, when a selected set of cryptographic parameters such as using the words or description "parameters 199a" or "cryptographic parameters 199a" can specify a row of parameters or values in a set of cryptographic parameters 199, such that the collection of values in the row can be used with a device provisioning protocol 191. Set identifier 199a can be an identity for a row or set of values for cryptographic parameters 199. For example, set "A" can comprise cryptographic suite 1 as specified in section 3.2.3 of DPP specification version 1.0. Key length 199b can be the length of keys in bits for PKI keys used with a device provisioning protocol with an initiator 101 in a device 106. ECC Curve name 199b can be a name for an ECC curve used with PKI keys for a device provisioning protocol with an initiator 101 in a device 106.

Hash algorithm 199e in cryptographic parameters 199 can be the name of a secure hash algorithm, such as the exemplary SHA-256 algorithm depicted, which may also be referred to as "SHA-2". Hash algorithm 199e can be used in a key derivation function (e.g. 402a, 402b, etc. below) and also to generate a signature based on a received nonce. A key length for a symmetric ciphering algorithm 199f can specify the length of a symmetric key in bits to use with a symmetric ciphering algorithm. The depicted algorithm in Figure 2a for algorithm 199f is "AES-SIV", although other symmetric ciphering algorithms could be specified in a set of cryptographic parameters as well. In

other words, an additional, different column in cryptographic parameters 199, such as an exemplary “symmetric algorithm” 199z (not depicted) could specify the symmetric ciphering algorithm to utilize with a responder for a device provisioning protocol. Nonce length 199g can specify the length in bits for nonces or random numbers generated by both initiator 102’ and responder 101x, including the length in bits of authentication values, where the nonces can be used for generating signatures and encryption keys.

Figure 2b

Figure 2b is a flow chart illustrating exemplary steps for an initiator to use an initiator DPP configuration and initiator key database in order to automatically connect with a responder proxy, in accordance with exemplary embodiments. An initiator 101 can use the exemplary data recorded in an initiator DPP configuration 130 and initiator key database 101t in order to automatically connect with a responder proxy 102’. The steps depicted in Figure 2b allow a device 106 with initiator 101 to locate or find a compatible responder proxy 102’ in an automated manner, such that no manual intervention is required by a device user to conduct a DPP 191 with subsequent configuration step 106a in order to load a device with credentials 109. This provides benefits over a conventional DPP 191, where additional manual intervention is required for operating a device with initiator 101 as a configurator and obtaining a bootstrap responder public key Br 102a via an “out of band” method, or loading a device 106 with a PKEX key 198.

The processes and operations, described below with respect to all of the logic flow diagrams and flow charts may include the manipulation of signals by a processor and the maintenance of these signals within data structures resident in one or more memory storage devices. For the purposes of this discussion, a process can be generally conceived to be a sequence of computer-executed steps leading to a desired result.

These steps usually require physical manipulations of physical quantities. Usually, though not necessarily, these quantities take the form of electrical, magnetic, or optical signals capable of being stored, transferred, combined, compared, or otherwise manipulated. It is convention for those skilled in the art to refer to representations of these signals as bits, bytes, words, information, elements, symbols, characters, numbers, points, data, entries, objects, images, files, or the like. It should be kept in mind, however, that these and similar terms are associated with appropriate physical quantities for computer operations, and that these terms are merely conventional labels applied to physical quantities that exist within and during operation of the computer.

It should also be understood that manipulations within the computer are often referred to in terms such as listing, creating, adding, calculating, comparing, moving, receiving, determining, configuring, identifying, populating, loading, performing, executing, storing etc. that are often

associated with manual operations performed by a human operator. The operations described herein can be machine operations performed in conjunction with various input provided by a human operator or user that interacts with the device, wherein one function of the device can be a computer.

In addition, it should be understood that the programs, processes, methods, etc. described herein are not related or limited to any particular computer or apparatus. Rather, various types of general purpose machines may be used with the following process in accordance with the teachings described herein.

The present disclosure may comprise a computer program or hardware or a combination thereof which embodies the functions described herein and illustrated in the appended flow charts. However, it should be apparent that there could be many different ways of implementing the disclosure in computer programming or hardware design, and the disclosure should not be construed as limited to any one set of computer program instructions.

Further, a skilled programmer would be able to write such a computer program or identify the appropriate hardware circuits to implement the disclosed disclosure without difficulty based on the flow charts and associated description in the application text, for example. Therefore, disclosure of a particular set of program code instructions or detailed hardware devices is not considered necessary for an adequate understanding of how to make and use the disclosure. The inventive functionality of the claimed computer implemented processes will be explained in more detail in the following description in conjunction with the remaining Figures illustrating other process flows.

Further, certain steps in the processes or process flow described in all of the logic flow diagrams below must naturally precede others for the present disclosure to function as described. However, the present disclosure is not limited to the order of the steps described if such order or sequence does not alter the functionality of the present disclosure. That is, it is recognized that some steps may be performed before, after, or in parallel other steps without departing from the scope and spirit of the present disclosure.

The processes, operations, and steps performed by the hardware and software described in this document usually include the manipulation of signals by a CPU or remote server and the maintenance of these signals within data structures resident in one or more of the local or remote memory storage devices. Such data structures impose a physical organization upon the collection of data stored within a memory storage device and represent specific electrical or magnetic elements. These symbolic representations are the means used by those skilled in the art of computer programming and computer construction to most effectively convey teachings and discoveries to others skilled in the art.

At step 201, device 106 can be powered on and activate a WiFi radio 106r. A step 201 can include device 106 loading a bootloader program and an operating system for device 106, as well as starting a process or operating firmware to perform the steps of an initiator 101. At step 202, device 106 can load initiator DPP configuration 130 and initiator key database 101t from nonvolatile memory,

where DPP configuration 130 and initiator key database 101t were depicted and described above in connection with Figure 1d.

At step 203, device 106 can select a set of parameters 199 and then all recorded responder bootstrap public keys 102a and at least one initiator bootstrap private key bi 101b for the parameters
5 199 from an initiator key database 101t. A hash value 101aa for the corresponding initiator bootstrap public keys Bi 101a can be selected or processed in a step 203 as well. In addition, the initiator bootstrap public key Bi 101a can be selected in a step 203.

At step 204, for each set of parameters 199, the initiator 101 in device 106 can generate an ephemeral PKI key pair comprising initiator ephemeral private key pi 101d and corresponding initiator
10 ephemeral public key Pi 101c. Initiator 101 in device 106 can then process or generate a DPP authentication request message 122 for each responder bootstrap public key Br 102a selected in step 203. As depicted in Figure 3a below, the series of steps for processing a message 122 in a step 204 can comprise steps 313 through 316.

At step 205, initiator 101 operating in device 101 can send (i) a DPP authentication request
15 message 122 on the current operating WiFi channel 130b within an operating class 130a for (ii) each responder bootstrap public key Br 102a selected in a step 203. For the first instance of step 205, the first WiFi channel 130b and associated operating class 130a can be the default channel 130b and class 130a selected by device 106 for WiFi radio 106r upon power up in step 201. A step 205 can also comprise sending each DPP message 122 for each key Br 102a multiple times with retry timers, such
20 as waiting a few seconds between sending each DPP message 122. In exemplary embodiments, each DPP message 122 in a step 205 is sent as a broadcast message on the current channel 130b using MAC.I 106n. In exemplary embodiments, for a step 205 each DPP message 122 for each key Br 102a is sent an exemplary 3 to 5 times. Note that for a step 205, multiple different DPP messages 122 for different keys Br 102a can be sent concurrently in order to speed the process for responder 102'
25 operating with responder DPP configuration 131 and a compatible network 107 for key Br 102a to receive the message 122 via WiFi link 192.

At step 206, initiator 101 or device 106 can determine if a DPP authentication response message 123 for a transmitted DPP authentication request message 122 has been received. If a DPP authentication response message 123 has been received, then initiator 101 and device 106 can conclude
30 the step 200 in order to establish communications with a responder 102 or responder proxy 102' (by receiving a DPP authentication response message 123). If a DPP authentication response message 123 has not been received, then initiator 101 and device 106 can proceed to step 207.

At step 207, initiator 101 or device 106 can determine if device 106 has cycled through all available channels 130b on all available operating classes 130a for WiFi radio 106r operating in device
35 106. If device 106 has not cycled through all available channels 130b on all available operating classes 130a, then device 106 can (i) increment a channel number 130b from the recorded DPP configuration

130 or (ii) if device 106 is already operating at the highest channel number 130b in an operating class 130a then change the operating class 130a for radio 106r and implement the lowest channel number in the new operating class 130a, and then return to step 205 to retransmit the collection of DPP messages 122 for each key Br 102a on the new channel number.

5 At step 207, if device 106 has cycled through all available channels 130b on all available operating classes 130a, then device 106 can (i) store data from a step 204 in nonvolatile memory for access at a later step 200, and (ii) conclude step 200 and enter a period of sleep or dormancy. After a period of time, device 106 could then wake and repeat the series of steps in a step 200 in order to receive a DPP authentication response 123 message from a responder proxy 102'. Note that upon
10 conducting a step 200 again at a later time, for a step 204 device 106 or initiator 101 could read (a) the data previously calculated or derived from a prior step 204 in (b) nonvolatile memory to access data for sending DPP messages 122 again, instead of re-calculating all the steps in 204.

Note that a step 200 can also be conducted for each set of parameters 199 supported by device 106, such as using a first step 200 with a first bootstrap private key bi 101b with a first set of parameters
15 199, and then a second step 200 concurrently with a second bootstrap private key bi 101b with a second set of parameters 199. Using the step 200 depicted for a device 106 operating an initiator 101 can allow device 106 to automatically receive a DPP authentication response message 123 from a responder proxy 102' when device 106 is powered on and with range of a responder proxy 102' for a responder bootstrap public key Br 102a (where responder 102' communicates with a DPP server 103
20 that securely records the responder bootstrap private key br 102b). A step 200 could be conducted with a responder 102* from Figure 1b or Figure 1c as well, such that the use of a responder proxy 102' is not required to use a step 200.

Figure 3a

25 Figure 3a is a simplified message flow diagram illustrating an exemplary system with exemplary data sent and received by a device database, a DPP server, a responder proxy, and an initiator, in accordance with exemplary embodiments. System 300a can include a device database 104, DPP server 103, responder proxy 102', and initiator 101 operating in device 106. Responder proxy 102' can communicate with DPP server 103 via IP network 113 as depicted in Figure 1a. As
30 contemplated herein, a responder proxy 102' depicted in Figure 3a and subsequent Figures below may also be referred to herein as responder 102', and a responder 102' and responder proxy 102' may be equivalent. In other words, a responder 102' may operate as a responder with some functions and operations for DPP 191 by proxy with DPP server 103.

35 Before starting steps and message flows depicted in Figure 3a, the device 106 with initiator 101 can record responder bootstrap public keys Br 102a depicted for an initiator key database 101t

from Figure 1d, where initiator 101 records at least one key Br 102a. The key Br 102a for a initiator 101 could be received in an “out of band” message 121 from Figure 1b and Figure 1c, or the key Br 102a could be recorded in a device 106 during manufacturing or distribution. Initiator key database 101t for device 106 could also record a plurality of responder bootstrap public keys Br 102a in a table 101ta from Figure 1d. For embodiments supporting mutual authentication, then device 106 can also record at least one initiator bootstrap public key Bi 101a and a corresponding initiator bootstrap private key bi 101b, such as in a table 101tb from Figure 1d. A device 106 with initiator 101 can also operate with at least one device supported configuration 130z from a DPP configuration 130 in Figure 1d. Exemplary PKI keys for initiator 101 in device 106 could also comprise the exemplary keys depicted in Figure 1e. Device database 104 can record data depicted and described in connection with Figure 2a.

Responder 102' can be different than responder 102* in Figure 1b and Figure 1c, since (a) some PKI and cryptographic computations (plus recording of PKI keys) that would normally be performed by a responder 102* as contemplated by DPP specification version 1.0 can (b) be performed by DPP server 103 and thus responder 102' is different than an responder 102*. For system 300a, DPP server 103 and device database 104 may establish a secure session setup 302a, which could comprise establishing a secure communications link between the two servers using protocols such as TLS, IPsec, a virtual private network (VPN), a secure shell (SSH), or similar networking, transport, or application layer technologies in order to establish secure communications between DPP server 103 and database 104. Secure session setup 302a can utilize a certificate cert.DPPS 103c for DPP server 103 and a certificate for database 104 in order to provide mutual authentication and mutual key derivation for a symmetric encryption key in secure session 302a.

At step 303, responder 102 can be powered on and begin operating a WiFi radio 102x with a responder configuration 131 as depicted in Figure 1f. At step 303, responder 102 can operate as an unconfigured responder 102 in Figure 1f, where the subsequent responder configuration step 127 depicted in Figure 3a can convert responder 102 into a configured responder 102'. In other words, responder 102' can comprise a responder 102 that has undergone a configuration step 127. At step 303, responder 102 can use a first MAC address MAC.R 102n to transmit or send and receive wireless data via WiFi link 192. At step 303, responder 102 can use a second, network MAC address MAC.N 102q in order to send and receive data with an IP network 113 and network 107 including DPP server 103. Or, in some exemplary embodiments, the same physical interface and MAC address could be used for both MAC.R 102n and MAC.N 102q.

DPP server 103 and responder 102 can establish a secure session 302b, where secure session 302b could comprise a session using TLS, DTLS, IPsec, a VPN, and other possibilities exist as well. Responder 102 can previously record a URL or domain name to reach DPP server 103. Responder 102 can also use a certificate for DPP server 103 comprising cert.DPPS 103c recorded in responder

102 in order to locate and connect with DPP server 103. As depicted in Figure 3a, secure session 302b could also comprise DPP server 103 receiving and using a responder certificate cert.responder 102m in order to authenticate responder 102. Alternatively, DPP server 103 could authenticate a user of responder 102 via a user ID and password entered by user of responder 102 into a user interface 102k for responder 102. In some exemplary embodiments, authentication of a responder 102 or a user of responder 102 for a secure session 302b could be omitted, such as if responder 102 does not record a cert.responder 102m. Secure session 302b could be established using cert.DPPS 103c, similar to a web site providing HTTPS security based on a certificate for the web site (i.e. the session 302b can authenticate the server and secure the session based on the certificate from the server). In exemplary
10 embodiments, responder 102 can authenticate the cert.DPPS 103c using parent certificates up to a recorded root certificate in cert.root 102w from Figure 1f.

After secure session 302b has been established between DPP server 103 and responder 102, DPP server 103 can conduct a step 304. Note that a responder identity 102i for responder 102 could be received by DPP server 103 during setup of secure session 302b. At step 304, DPP server 103 can
15 select a DPP application 102g for responder 102', where DPP application 102g can comprise software or firmware for an unconfigured responder 102 to load and operate in order to function as a configured responder 102'. DPP application 102g was depicted and described in Figure 1f. DPP server 103 could select DPP application 102g based on the OS 102e for responder 102, which could be communicated and received by DPP server 103 in setup of secure session 302b. DPP server 103 could obtain DPP
20 application 102g from another server in network 107 or external networks. At step 304, DPP server 103 could also select or process a responder network table 102t for responder 102, where responder network table 102t was depicted and described in connection with Figure 1f. Responder network table 102t can select which DPP server 103 a configured responder 102' should forward an incoming DPP authentication request message 122 based on the responder bootstrap public key hash value 102aa.
25 Although a DPP server 103 is depicted in Figure 3a as conducting secure session 302b and a step 304 and then a step 306, another server associated with network 107 could perform those functions and steps for a DPP server 103.

At step 306, DPP server 103 can select the responder mode 306a of operation for responder 102' to conduct a DPP 191 for receipt and processing of incoming DPP authentication request
30 messages 122 from an initiator 101. The selected mode of operation for responder 102' can comprise a mode 306a or a responder mode 306a, which could be recorded in a server database 103a as depicted and described in connection with Figure 3c below. The selected mode 306a in a step 306 can specify the distribution of PKI keys and related processing between DPP server 103 and responder 102'. The selected mode 306a in a step 306 can determine the subsequent series of steps and message flows
35 between DPP server 103 and responder 102'. The selected mode 306a in a step 306 could be based on the security established or mutual authentication or server authentication in secure session 302b.

As one example, if secure session 302b used a certificate cert.DPPs 103c but not a certificate cert.responder 102m (e.g. secure session 302b is less secure or not mutually authenticated or responder 102 is not securely authenticated), then selected mode 306a can comprise a first mode depicted in Figure 3a and Figure 3b, where DPP server 103 records and operates on responder ephemeral private key pr 102r and also all secret shared keys k1 314a, k2 314c, and ke 327a. As a second example, if secure session 302b is mutually authenticated (e.g. used a certificate cert.DPPs 103c and a certificate cert.responder 102m), then a selected mode 306a in a step 306 could comprise a second mode 306a for responder 102' to operate as depicted in Figure 5a, where responder 102' records and operates on responder ephemeral private key pr 102r (and DPP server 103 does not record or operate on the key pr 102r).

As another example, a selected mode 306a in a step 306 could also depend on an identity for device 106, such as MAC.I 106n. For embodiments where the selected mode 306a of operation of a responder 102' depends on the identity of a device 106, then a step 306 to select the mode could be conducted after receipt of a DPP message 122 which would include MAC.I 106n in order to identify the device 106. In other words, in some exemplary embodiments, a preferred responder mode 306a for a device 106 could depend on the device 106, such as if device 106 was a higher-value piece of equipment such as an automobile compared to a potentially lower value networked sensor. For these embodiments, the responder mode 306a for responder 102' could be specified in a device database 104, and for these embodiments the selected mode 306a for responder 102' to operate could be conducted by a step 306 after the receipt of message 312a with data from device database 104.

A first mode 306a could (a) specify that DPP server 103 records all public and private bootstrap and ephemerals keys for responder 102', plus the public keys Bi 101a and Pi 101e for initiator 101 such that (b) resulting cryptographic operations and ECDH key exchanges and symmetric ciphering reside within DPP server 103 and results are transmitted to responder 102'. This first mode 306a could represent the message flows and steps in Figure 3a and Figure 3b, with the recording of keys depicted in Figure 3c below. A second mode 306a could specify that DPP server records and operates on the responder bootstrap private key br 102b, but responder 102' records and operates on the responder ephemeral private key pr 102r, and the second mode 306a is depicted in Figure 5a below. A third responder mode 306a could specify that both responder 102' and DPP server 103 record and operate on responder ephemeral private key pr 102r, such that (b) resulting cryptographic operations and key exchanges are shared between DPP server 103 and responder 102'. This third responder mode 306a could represent the message flows and steps depicted in Figure 5b, with the recording of keys depicted in Figure 3c below.

In exemplary embodiments, all the first, second, and third responder modes 306a specify responder bootstrap private key br 102b residing within DPP server 103 and not shared with responder 102'. By not sharing responder bootstrap private key br 102b with responder 102' (which could be a

device with much less security than DPP server 103), the responder bootstrap private key br 102b can remain secured within network 107. These embodiments where responder bootstrap private key br 102b remains in network 107 (e.g. in DPP server 103 and device database 104) support securely sharing the same responder bootstrap public key Br 102a potentially across a plurality of devices 106 with initiators 101 (as well as responder 102' recording key Br 102a). Sharing the same responder bootstrap public key Br 102a across multiple devices supports recording the key Br 102a in a device 106 either (i) during manufacturing or distribution of device 106, or (ii) at a time before any association between device 106 and network access point 105 is known. This simplifies and enables the use of a device provisioning protocol 191 across multiple devices 106 with mutual authentication, where the responder bootstrap public key Br 102a can comprise a shared key 102z. Or, in other embodiments the responder bootstrap public key Br 102a may not be a shared key 102z, but rather unique for a responder 102' (such as a unique responder bootstrap public key Br 102a recorded in a tag 102v), and security is still enhanced because responder 102' may not record or operate on responder bootstrap private key br 102b.

In other words, without the technology described in the present disclosure, a device provisioning protocol with mutual authentication can essentially shift the problem of (a) securely mutually authenticating a device and providing credentials 109 to a device 106 to a new, but related problem of (b) securely providing a responder bootstrap private key br 102b to an responder 102*, where an initiator should record the responder bootstrap public key Br 102a. The first, second, and third modes 306a can solve this shifted/new problem since (i) the responder bootstrap private key 102b can permanently reside in a network 107 and does not need to be sent to the responder 102', and (ii) an initiator can be configured to record key Br 102a before receipt of device 106 by a user (since the same key Br 102a can be written to many different devices). However, with the technology in the present disclosure, the responder 102' can still transmit and receive the full set of a device provisioning protocol 191 messages with an initiator 101 as specified in DPP specification version 1.0.

A fourth mode 306a could specify that responder 102' records all of the responder bootstrap and ephemeral PKI keys, as well as the initiator 101 public keys, such that resulting cryptographic operations and key exchanges reside in responder 102'. This fourth mode 306a could represent the message flows and steps depicted in Figure 1b or Figure 1c, and in this case responder 102' could comprise an initiator 102*. Other possibilities exist as well without departing from the scope of the present disclosure. Additional details for the different responder modes 306a selected in a step 306, with the resulting different allocation or recording of PKI keys and derived symmetric ciphering keys are described below in Figure 3b for a server database 103a.

After a step 306, DPP server 103 can send responder 102 a message 305, where message 305 can include a responder identity 102i, responder network table 102t, responder DPP application 102g, and a responder configuration 131. Note that the responder configuration 131 in message 305 can

represent an updated responder configuration to replace or update a previous responder configuration 131 recorded by responder 102. DPP application 102g could comprise a link or URL for responder 102 to download and install DPP application 102g (comprising such as less than an exemplary 100 bytes), or the full software DPP application 102g for responder 102' to operate, which could comprise
5 more than 1 megabyte of data. Responder identity 102i can be useful for a device such as access point 105 operating a responder 102', since in exemplary embodiments an access point 105 could operate more than one responder 102', and the different responders 102' operating in an access point 105 or other device could be identified using the responder identity 102i. In some exemplary embodiments, the responder mode 306a could also be sent in a message 305 from DPP server 103. In other exemplary
10 embodiments, responder mode 306a for responder 102' can also be sent below in a message 312b, after an identity for device 106 such as MAC.I 106n is obtained from a subsequent message 122.

Responder 102 using responder identity 102i in system 300a can receive message 305 and process the data. Responder 102 can download and install the DPP application 102g from a message 305. Responder 102 can load and apply both responder network table 102t and the received responder
15 configuration 131, which could specify radio frequencies or channels to utilize, such as class 131a and channels 131b. Responder 102 can conduct the responder configuration step 127 depicted and described in connection with Figure 1f in order to begin operating as a configured responder 102'. At step 307, responder 102' can begin listening for incoming DPP authentication request messages 122 using WiFi radio 102x and operating with responder configuration 131 in order to communicate with
20 a WiFi radio 106r operating in device 106 with initiator 101, where the communications can use a WiFi data link 192. In exemplary embodiments, DPP application 102g may be (i) included or distributed along with an operating system 102e running on responder 102' or (ii) acquired at a different time than a message 305, and thus from a separate download of DPP application 102g. In exemplary embodiments, WiFi radio 102x operating in responder 102' can listen for message 122
25 either broadcast or unicast messages or datagrams from device 106 with initiator 101 when using responder configuration 131. In exemplary embodiments, initiator 101 transmits DPP messages 122 using the same or compatible configuration as responder configuration 131.

Device 106 operating initiator 101 can conduct the steps 200 depicted and described in connection with Figure 2b above. In summary, at step 200 in Figure 3a, device 106 may power on
30 from an unpowered or "deep sleep" state. Power could be provided from a battery or converted "wall power". Device 106 could (i) load the equivalent or initiator version of information contained in responder configuration 131 using a bootloader program and (ii) power on WiFi radio 106r. Device 106 can load database 101t for initiator 101 with at least one responder bootstrap public key Br 102a. As a portion of a step 200, device 106 can conduct a step 204 as depicted in Figure 2b. Step 204 can
35 comprise steps 313, 314, 315, and 316. Note that step 200 can comprise a cycle or loop through responder bootstrap public keys Br 102a recorded in database 101t so that initiator 101 send a DPP

authentication request message 122 for each responder bootstrap public key Br 102a until initiator 101 successfully receives a DPP authentication response message 123.

A step 313 can comprise initiator 101 deriving an initiator ephemeral private key pi 101d and an initiator ephemeral public key Pi 101c. Step 313 can use a random number generator to produce a random number, which could be input into a key pair generation algorithm using the set of cryptographic parameters 199a. The resulting PKI key values from conducting a step 313 can comprise the initiator ephemeral private key pi 101d and the initiator ephemeral public key Pi 101c. The data for a step 313 comprising derived ephemeral PKI keys for initiator 101 can be recorded in memory of device 106, such that the ephemeral PKI keys can be derived one time until a DPP authentication response message 123 is received from a responder such as responder 102'.

Step 314 can comprise initiator 101 conducting a first initiator key exchange step 314 using an Elliptic Curve Diffie Hellman (ECDH) key exchange 401a, as depicted and described in connection with Figure 4a below. Data input for a step 314 can comprise the initiator ephemeral private key pi 101d and the selected responder bootstrap public key Br 102a from a step 203. Data output from a step 314 can comprise a key k1 314a as described in Figure 4a below. Step 314 can comprise the key exchange and symmetric key derivation step in DPP specification version 1.0 section 6.2.2 in order for an initiator to obtain key k1 314a. Note that step 314 by initiator 101 can be conducted for each recorded responder bootstrap public key Br 102 in a database 101t. The data for a step 314 can be stored in memory for device 106 such that keys k1 314a for each key Br 102a can be stored and reused for a subsequent "loop" in a step 200 until initiator 101 receives a DPP authentication response message 123, thereby reducing the number of computations and energy required by device 106.

At step 315, initiator 101 can generate a random number for an initiator nonce 317a using parameters 199a. As used herein and throughout the present disclosure, the words "random number" can also mean a pseudo random number, since random numbers generated by nodes may not be truly random. However, their "randomness" may be sufficient in order to securely conduct the cryptographic operations contemplated herein. Initiator 101 can also determine the initiator capabilities, which could comprise an enrollee for the initiator as depicted in Figure 3a. In other embodiments, the responder 102' could have initiator capabilities as an enrollee (where the initiator 101 could be the configurator). A step 315 can comprise encrypting the initiator nonce 317a and initiator capabilities using the derived key k1 from step 314. An encryption step 315 step is further depicted and described in connection with Figure 4a below.

Note that step 315 by initiator 101 can be conducted for (i) each recorded responder bootstrap public key Br 102a in a database 101t with (ii) a corresponding key k1 314a recorded from a step 314. A single initiator nonce 317a can be reused across all encryption steps 315 and the output of all encryption steps 315 comprising ciphertext 315a can be recorded in memory of device 106, such that an encryption step 315 needs only to be conducted once by initiator 101 for each key Br 102a in a

database 101t (by reusing initiator ephemeral private key pi 101d and key $k1$ 314a). In other words, initiator 101 can record one ciphertext 315a for each responder bootstrap public key Br 102a in a database 101t, and a subsequent “loop” or generation of a DPP authentication request message 122 (such as transmitting on a different channel 130b) can reuse the ciphertext 315a recorded for the key Br 102a, key pi 101d, and initiator nonce 317a. In this manner, the number of computations and energy use by device 106 can be reduced, by reusing a previously recorded ciphertext 315a when re-transmitting a DPP authentication request message 122 for the same key Br 102a at a later time.

At step 316, initiator 101 can calculate secure hash value 101aa for initiator bootstrap public key Bi 101a and secure hash value 102aa for responder bootstrap public key Br 102a, using a secure hash algorithm 199e. The values are depicted in Figure 3a as “ $H(Bi)$ ” and “ $H(Br)$ ”, respectively. Initiator 101 can then transmit using radio 106r and channel 130b a DPP authentication request message 122, which can include the hash value 101aa for the initiator bootstrap public key Bi 101a and the hash value 102aa the responder bootstrap public key Br 102a, the derived initiator ephemeral public key Pi 101c, and ciphertext 315a, where ciphertext 315a is encrypted with key $k1$ 314a and includes an initiator nonce 317a and the initiator capabilities.

In exemplary embodiments and as depicted and described in connection with Figure 2b in a step 205 for a step 200 depicted in Figure 3a, a DPP authentication request message 122 for each recorded responder bootstrap public key Br 102a can be transmitted by initiator 101 on channel 130b. In exemplary embodiments, each DPP message 122 can be transmitted as a broadcast message on channel 130b, since initiator 101 may not know a responder 102’ MAC address. As depicted for a step 200 above in Figure 2b, initiator 101 may need to repeat messages 122 for each key Br 102a multiple times on several different channels in a channel list 130b, since (i) responder 102’ may be listening on a different channel in a responder channel list 131b, and (ii) responder 102’ may be using only one of the bootstrap responder public keys Br 102a from a database 101t which corresponds to the key Br 102a recorded and operated by responder 102’.

Although a message 122, message 123, etc. as depicted herein shows key values of “ $k1$ ”, “ ke ”, etc., the depiction of the keys is to illustrate that ciphertext is encrypted with the corresponding key by the use of brackets. In other words, although the key values are shown with the messages, the actual key values are not transmitted inside or with the message in exemplary embodiments. The equivalent syntax is used in DPP specification version 1.0.

Responder 102’ can receive a DPP authentication request message 122 in a step 308a and process the message. For a step 308a, responder 102’ can compare (i) hash values 102aa for $H(Br$ 102a) received in message 122 with (ii) the recorded hash values 102aa in a responder network table 102t. Or, responder 102’ can record the responder bootstrap public key Br 102a and calculate a hash value and compare with the hash value received in a message 122. If (a) the hash value 102aa for $H(Br$ 102a) in message 122 matches or are equal to a recorded or calculated hash value 102aa, then

(b) for a step 308b responder 102' can conduct a key validation step on received initiator ephemeral public key P_i 101c described two paragraphs below. Otherwise, responder 102' can continue to monitor or listen for new, different incoming messages 122 where the hash value 102aa would equal a recorded or calculated hash value 102aa for a responder bootstrap public key Br 102a in responder 102.

An identity for device 106 and initiator 101 could also comprise the secure hash value over the initiator bootstrap public key Bi 101a, which can be value 101aa as depicted for message 122 in Figure 3a. In other words, as contemplated throughout the present disclosure, an identity for device 106 and/or initiator 101 can comprise the hash value 101aa over the initiator bootstrap public key Bi 101a. For example, a device database 104 (Fig 2a), a server database 103a (Fig 3c), or responder 102' could record and use hash value 101aa as an identity for device 106 or initiator 101 in order to select keys for conducting key exchange and encryption/decryption steps. The hash value 101aa could be either (i) instead of or (ii) in addition to $MAC.I$ 106n or device identity $ID.device$ 106i in order to identify and track device 106 with initiator 101 across a potential plurality of devices 106 in systems 100, 300, 500, etc.

A step 308b could comprise responder 102' performing the steps for an ECC Full Public-Key Validation Routine in section 5.6.2.3.2 of FIPS publication SP 800-56A (revision 2) for the received initiator ephemeral public key P_i 101c. Alternatively, step 308b can comprise responder 102' performing the steps ECC Partial Public-Key Validation Routine in section 5.6.2.3.3 of the same FIPS publication. Other example steps within a public key validation step 308 can comprise (i) verifying the public key is not at the "point of infinity", and (ii) verifying the coordinates of the point for the public key are in the range $[0, p-1]$, where p is the prime defining the finite field. Other possibilities exist as well for evaluating and validating a received public key is cryptographically secure in a public key validation step 308b, without departing from the scope of the present disclosure. As contemplated in the present disclosure, a responder 102', a DPP server 103, or an initiator 101 can conduct a public key validation step 308b each time a public key is received. In addition, a responder network table 102t can include a set of parameters 199a for each entry corresponding to a key Br 102a, and the set of parameters 199a could be used for key validation steps in a step 308b.

A step 308c can comprise responder 102' selecting a DPP server 103 based on the received hash value 102aa for responder bootstrap public key Br 102a in message 122. A responder 102' could communicate with a plurality of different DPP servers 103 or networks 107, and a key Br 102a could be associated with different DPP servers 103. For example, different networks 107 or DPP servers 103 could record and operate with the responder bootstrap private key br 102b in order to conduct or support a DPP 191 with initiator 101. The responder 102' could use the hash value 102aa and a responder network table 102t as depicted in Figure 1f in order to determine which DPP server 103 should receive data from the incoming DPP authentication request message 122. Figure 3a depicts

hash value 102aa for DPP server 103 in network 107 from Figure 1a, although hash value 102aa could be for a different network 107 in exemplary embodiments. In other words, DPP server 103 could comprise a depicted “DPP 103 -1” for responder network table 102t in Figure 1f, and a different hash value 102aa for Br 102a in message 122 could be for a different DPP server 103 comprising a depicted value of “DPP 103 -3” in responder network table 102t in Figure 1f. A step 308c can also include responder 102’ starting a timer in order to count the seconds between receiving message 122 and sending upcoming message 123. Additional details regarding the timer from a step 308c are described in a step 324 below.

For some exemplary embodiments, a responder network table 102t as depicted in Figure 1f above could (i) omit the storing or recording of a plurality of responder bootstrap public key hash values 102aa and (ii) include at least one value or identity or domain name for a DPP server 103. Thus, although “DPP server 103” is depicted for a responder network table 102t in Figure 1f, the value recorded could be for an identity or domain name for a DPP server 103, such that responder 102’ could forward data received in a message 122 to the DPP server 103 in a subsequent message 309. For a step 308c, a responder 102’ could select the DPP server 103 and an identity or domain name for DPP server 103 from the responder network table 102t for sending a subsequent message 309 with data received in DPP authentication request 122. For these embodiments where responder network table 102t omits the storing or recording of a plurality of responder bootstrap public key hash values 102aa, then a DPP server 103 could record a table that stores a plurality of responder bootstrap public key hash values 102aa associated with a plurality of DPP servers 103. DPP server 103 could take actions after receiving a message 309 in order to select another DPP server 103 associated with the responder bootstrap public key hash values 102aa received in a message 309 (for embodiments where the received responder bootstrap public key hash value 102aa from a message 309 does not match the DPP server 103 that receives the message 309).

After steps 308a, 308b, and 308c, responder 102’ can send DPP server 103 a message 309 via the secure session 302b. Message 309 can include a responder identity 102i, an identity for initiator 101 and/or device 101 comprising MAC.I 106n, and the data contained in DPP authentication request message 122 received by responder 102’. Note that MAC.I 106n could be observed by responder 102’ as the source MAC address for message 122 received via WiFi link 192. Data from message 122 in message 309 can include secure hash value 101aa for initiator bootstrap public key Bi 101a and secure hash value 102aa for responder bootstrap public key Br 102a, the initiator ephemeral public key Pi 101c, and ciphertext 315a. Although not depicted in Figure 3a, a message 309 could also include the responder bootstrap public key Br 102a, for embodiments where responder 102’ also records key Br 102a. DPP server 103 can receive message 309 and process the data, including recording the received initiator ephemeral public key Pi 101c in a server database 103a, which is also depicted in Figure 3c

below. Before taking steps to process the received ciphertext 315a, DPP server 103 could require additional PKI keys, if they are not already recorded in a server database 103a.

For some embodiments, DPP server 103 can store the equivalent of a responder network table 102t depicted in Figure 1f (and responder 102' may not record the plurality of responder bootstrap public key hash values 102aa for different DPP servers 103). For these embodiments, after receipt of a message 309, DPP server 103 could use the received value of responder bootstrap public key hash value 102aa from a message 309 to query or obtain a different, second DPP server 103 associated with the responder bootstrap public key hash value 102aa. DPP server could forward the data from the message 309 to the different, second DPP server 103. Note that the different, second DPP server 103 could be associated with a different network 107. The different, second DPP server 103 could received the message 309, and subsequently the different, second DPP server 103 could conduct the subsequent steps for a DPP server 103 depicted in Figure 3a and Figure 3b, such as sending a message 312b, conducting the depicted steps of 318 through 315b, sending a message 313 to responder 102', etc.

For embodiments where DPP server 103 does not already record PKI keys for responder 102' and initiator 101, DPP server 103 can send device database 104 a message 310 in order to query for keys and information in order to support a DPP 191 between responder 102' and initiator 101. DPP server 103 can send message 310 through secure session 302a. Message 310 can include an identity for device 101, which could be MAC.I 106n and secure hash value 101aa for initiator bootstrap public key Bi 101a and secure hash value 102aa for responder bootstrap public key Br 102a.

Device database 104 can receive message 310. Device database 104 in step 312 can conduct a query of device database 104 for data associated with the received data from an initiator comprising and identity of MAC.I 106n, hash value 102aa and hash value 101aa. Exemplary data for device database 104 was depicted and described in connection with Figure 2a above. Not all data in the depicted message 310 is required to conduct a query in a step 312, and a hash value 101aa for Bi 101a could be omitted as one example. Or, in exemplary embodiments, the MAC.I 106n may not be recorded in a device database 104, but data for a device 106 could be found using hash value 101aa for Bi 101a. Thus, in exemplary embodiments at least one of MAC.I 106n and hash value 101aa for Bi 101a is received by device database 104 in a message 310, and the at least one MAC.I 106n and hash value 101aa for Bi 101a is recorded in a device database 104.

In exemplary embodiments, a query in a step 312 can select a responder bootstrap private key br 102b, responder bootstrap public key Br 102a, optionally either (i) initiator bootstrap public key Bi 101a or (ii) PKEX key 198 for device 106 from a device database 104. For embodiments where initiator bootstrap PKI keys for device 106 are updated over time, then the most recently updated initiator bootstrap public key Bi 101a can be selected by a database query in a step 312. Additional data can be selected for device 106 and/or initiator 101 in a query at step 312 as well, including a set of parameters 199a to use with bootstrap PKI keys and ephemeral keys and/or a responder mode 306a

for DPP server 103 and responder 102' to use when conducting a DPP 191 with initiator 101. Note that a responder mode 306a could be selected for responder 102' in either a step 306 or a query in step 312. The query in step 312 can return values for a message 312a from device database 104 to DPP server 103.

5 The values in a message 312a can comprise data for a device identity of MAC.I 106n (or hash value 101aa) and the values can include responder bootstrap public key Br 102a, initiator bootstrap public key Bi 101a, responder bootstrap private key br 102b, a device identity ID.device 101i, a selected set of cryptographic parameters 199a for the PKI keys, and optionally PKEX key 198. MAC.I 106n or hash value 101aa can be included in a message 312a so that DPP server 103 can track which
10 of a possible plurality of initiators 101 and devices 106 that message 312a is associated with. Although not depicted for a message 312a, a message 312a could also include a responder mode 306a for device 106 using MAC.I 106n.

DPP server 103 can receive the message 312a and record the data received in a server database 103a. DPP server 103 can (i) use a device identity such as MAC.I 106n or hash value 101aa in order
15 to select or update a responder mode 306a previously selected in a step 306, or (ii) conduct a step 306 at this time in order to select a responder mode 306a recorded for responder 102' in a server database 103a, where a server database 103a is depicted and described in connection with Figure 3c below. As depicted in Figure 3a, DPP server 103 can then send message 312b to responder 102', where message 312b can include (i) the selected responder mode 306a of operation for responder 102' and DPP server
20 103 to use when conducting a DPP 191 with initiator 101, (ii) a selected set of cryptographic parameters 199a for use with PKI keys and a device provisioning protocol, and (iii) optionally a PKEX key 198 for conducting a public key exchange with initiator 101. In exemplary embodiments, the PKEX key 198 can be included in message 312a for embodiments where device database 104 does not record an initiator bootstrap public key Bi 101a for device 106 with MAC.I 106n. For the
25 responder mode 306a depicted in Figure 3a and Figure 3b, message 312b can comprise a signal that DPP server 103 derives the responder ephemeral private key 102r. Message 312b can also optionally include a device identity such as MAC.I 106n or hash value 101aa in order to associate the data in message 312b with a particular initiator 101, since responder 102' could potentially communicate with a plurality of different devices 106. For some exemplary embodiments, DPP server 103 can send
30 message 312b after establishment of secure session 302b and before DPP server 103 receives a message 309.

For some embodiments, DPP server 103 can (i) store the equivalent of a responder network table 102f depicted and described in connection with Figure 1f, (ii) may not store or operate with the responder bootstrap private key br 102b for responder 102', and (iii) select a different, second DPP
35 server 103 after the receipt of a message 309 as described six paragraphs above. For these embodiments, a message 312b can (i) include an identity or domain name for the different, second

DPP server 103 and (ii) omit depicted data of responder mode 306a and parameters 199 and PKEX key 198. For these embodiments, responder 102' (i) could receive the message 312b and use the received identity or domain name of the different, second DPP server 103 in the message 312b to (ii) resend the message 309 to the different, second DPP server 103 received in the message 312b. Responder 102' and the different, second DPP server 103 could establish a second secure session 302b. For these embodiments, after resending the message 309 to the different, second DPP server 103, the subsequent steps and messages depicted in Figures 3a and 3b can apply for the different, second DPP server 103 with the responder 102' (where the different, second DPP server 103 operates as the depicted DPP server 103 in Figure 3a and Figure 3b). For example, the different, second DPP server 103 could send a message 312b to responder 102', conduct the depicted steps of 318 through 315b, send the message 313 to responder 102', etc.

Responder 102' can receive message 312b from DPP server 103 and conduct a step 312c. In step 312c, responder 102' can use and record the responder mode 306a in order to conduct subsequent steps for responder 102' and messages with DPP server 103 for device 106 with initiator 101. Note that a responder mode 306a could specify the operation of responder 102' with device 106, such that responder 102' could operate with a first responder mode 306a for a first device 106 and a second responder mode 306a with a second device 106. The depicted subsequent steps and messages for responder 102' in Figure 3a with the responder mode 306a depicted as received in a message 312b can comprise a first responder mode 306a depicted as "Fig. 3" in server database 103a. For embodiments where a responder network table 102t includes cryptographic parameters 199a associated with the responder bootstrap public key hash value 102aa, then a message 312b could optionally omit parameters 199a within message 312b, and responder 102' can determine the cryptographic parameters 199a based on the hash value 102aa received from initiator 101 in a message 122.

For embodiments where PKEX key 198 is received in a message 312b (e.g. embodiments where device database 104 may not previously record initiator bootstrap public key Bi 101a before step 312), then responder 102' can conduct a public key exchange protocol exchange 312d with initiator 101 using the PKEX key 198 depicted in Figure 3a. A PKEX 312d can comprise the PKEX protocol as specified in section 5.6 of DPP specification version 1.0. In this manner, responder 102' can receive the key Bi 101a if not previously recorded by a device database 104. Although not depicted in Figure 3a, upon conclusion of a PKEX 312d (or otherwise obtaining an initiator bootstrap public key Bi 101a such as an "out of band" message 126 in Figure 1c), responder 102' can forward the key Bi 101a to DPP server 103 after receipt of the initiator bootstrap public key Bi 101a from a step 312d. The use of a PKEX protocol and key 198 can also be optionally omitted in some embodiments, if initiator bootstrap public key Bi 101a can be recorded by a network 107 prior to a step 312c.

As depicted in Figure 3a, the collection of steps and messages in Figure 3a, from step 200 for device 106 through the optional message 312d can be grouped together into a step 301. Step 301 with the collective group of sub-steps and messages depicted for a step 301 can also be performed in other embodiments depicted in Figure 5a and Figure 5b below. Subsequent messages and steps depicted in Figure 3a starting after step 301 can represent a first responder mode 306a for DPP server 103 and responder 102', where the allocation of recording PKI keys and symmetric keys from key exchanges for the first mode 306a are depicted below in Figure 3c. The first responder mode 306a can be depicted in Figure 3c with the label "Fig. 3" is depicted in Figure 3a and Figure 3b.

For the first responder mode 306a selected in a step 306, DPP server 103 can then conduct the series of depicted steps 318 through 315b in Figure 3a in order to generate data for responder 102' to send a DPP authentication response message 123. At step 318, DPP server 103 can also generate two random numbers using a random number generator. The first random number can be used as a responder ephemeral private key pr 102d. The responder ephemeral private key pr 102p can be input into an ECC key pair generation algorithm in order to output a responder ephemeral public key Pr 102p. The set of cryptographic parameters 199a used for the responder ephemeral PKI key pair can match the recorded parameters 199a used with the bootstrap public keys such as responder bootstrap public key Br 102a. The second random number generated by DPP server 103 can be used as a responder nonce 318b. DPP server 103 can also determine responder capabilities 319b, which could comprise the capabilities for a configurator. The responder ephemeral private key pr 102d can also be referred to as responder protocol private key pr 102d, and responder ephemeral public key Pr 102c can be referred to as responder protocol public key Pr 102c. The second random number in a step 318 can comprise a responder nonce value 318b, which can be used in subsequent messages and calculation of authentication values.

DPP server 103 can conduct a first responder key exchange step 319, where the details for a key exchange step 319 are depicted in Figure 4a below. In summary, the received initiator ephemeral public key Pi 101c and the responder bootstrap private key br 102b can be input into an ECDH key exchange in order to mutually derive a shared secret key M 410. The shared secret key M 410 can be input into a key derivation function 402a in order for DPP server 103 to derive mutually shared key k1 314a. A DPP server 103 can then conduct a decryption step 320 as depicted in Figure 4a below. DPP server 103 can input the received ciphertext 315a from a message 122 into a symmetric ciphering algorithm 405b and key k1 314a using parameter 199f in order to read the plaintext 403a, which could comprise the initiator nonce 317a and the initiator capabilities. In exemplary embodiments, the responder 102' can have capabilities of a configurator, and DPP server 103 can select initiator capabilities or an enrollee in a step 320. Steps 319 and steps 320 can also comprise the first four paragraphs of section 6.2.3 of DPP specification version 1.0.

DPP server 103 can then conduct a second responder key exchange step 314b, where the details for a key exchange step 314b are depicted and described in connection with Figure 4b below. In summary, the initiator ephemeral public key P_i 101c received in message 122 can be input into an ECDH key exchange algorithm 401a along with the responder ephemeral private key pr 102r derived in a step 318 above in order to output a secret shared key N 411. The value for shared secret key N 411 can be recorded and used in the derivation of a key ke 327a in a third key exchange step 321 below. For a step 314b, the value for the shared secret key N 411 can be input into a key derivation function 402b in order to derive a mutually shared symmetric key k_2 314c.

DPP server 103 can conduct a third responder key exchange step 321, where the details for a key exchange step 321 are depicted in Figure 4c below. In summary, both (i) the derived responder ephemeral private key pr 102r from a step 318 with the recorded responder bootstrap private key br 102b, and (ii) the initiator bootstrap public key B_i 101a can be input into an ECDH key exchange algorithm 401c in order to output a secret shared key L 412. For embodiments where a DPP responder authentication 141 is conducted by DPP server 103 without the use of an initiator bootstrap public key B_i 101a, then calculation of a secret shared key L 412 can be optionally omitted in a step 321. Step 321 for DPP server 103 can then comprise a key derivation function 402c using the mutually derived secret shared keys M 410, N 411, and optionally L 412 in order to mutually derive secret shared key ke 327a. An example for a key exchange step 321 is described in point 2 of section 6.2.3 of DPP specification version 1.0 on page 48.

DPP server 103 can then conduct an encryption step 322, where the details for an encryption step 322 are depicted and described in connection with Figure 4c below. Step 322 can also comprise DPP server 103 creating a value $R\text{-auth}$ 404, where $R\text{-auth}$ 404 can comprise a hash of the initiator nonce 317a from plaintext 403a in step 320, the second random number $R\text{-nonce}$ 318b from a step 318 above, and public PKI keys. The value for $R\text{-auth}$ 404 in a step 322 can comprise the value in point 3 of section 6.2.3 of DPP specification version 1.0. An encryption step 322 can comprise DPP server 103 (a) encrypting the value for $R\text{-auth}$ 404 using the mutually derived key ke 327a from the third key exchange step 321 above, where (b) the output of encryption step 322 can comprise ciphertext 324a.

DPP server 103 can then conduct an encryption step 315b, where the details for an encryption step 315b are depicted and described in connection with Figure 4b below. Ciphertext 324a from an encryption step 322 above can also be encrypted again for an encryption step 315b. The symmetric ciphering key for encryption step 315b can comprise symmetric key k_2 314c. Encryption step 315b, encryption step 322, and decryption step 320 can use a common symmetric ciphering algorithm 405a, which could comprise the Advanced Encryption Standard with Synthetic Initialization Vectors (AES-SIV) (and deciphering algorithm) also with a common set of symmetric ciphering parameters 199f. Symmetric ciphering parameters 199f can also specify the use of a block chaining mode such as cipher block chaining (CBC), counter mode (CTR), or Galois/Counter mode (GCM) and other possibilities

exist as well. In addition, symmetric ciphering parameters 199f could specify a mode for message authentication, which could comprise a CMAC mode as specified in NIST publication SP-800-38B. In exemplary embodiments, a symmetric ciphering algorithm 405a can comprise the AES-SIV algorithm as specified in IETF RFC 5297. The output from an encryption step 315b can be ciphertext 315c, as depicted in Figure 4b below.

After the encryption step 315b, DPP server 103 can send responder 102' a message 313, where message 313 can include an identity for device 106 such as MAC.I 106n, hash value 102aa for key Br 102a, hash value 101aa for key Bi 101a, the responder ephemeral (or protocol) public key Pr 102p from a step 318 and a ciphertext 315c. The ciphertext 315c can comprise the encrypted data depicted and described in connection with Figure 4b below from the encryption step 315b. As depicted in Figure 3a, ciphertext 315c can be encrypted with key k2 314c and include the responder nonce 318c, the initiator nonce 317a, the responder capabilities (as an exemplary configurator in Figure 3a), and a ciphertext 324a. In other words, ciphertext 324a can comprise data that is encrypted twice, where the first encryption with a key ke 327a can be in a step 322 as depicted in Figure 4c below, and the second encryption step can be with key k2 314c. Ciphertext 324a can include a responder authentication value 404, where the responder authentication value 404 is encrypted by DPP server 103 with key ke 327a.

Responder 102' can receive message 313 and conduct a step 324 to process the message. Note that message 313 and other messages between responder 102' and DPP server 103 can be sent and received via secure session 302b. Responder 102' can conduct a step 324. For a step 324, responder 102' can record the received responder ephemeral public key Pr 102p in a responder network table 102t along with the identity of device 106 comprising potentially MAC.I 106n. In other words, a responder network table 102t can be updated to include information received from network 107 (and DPP server 103) regarding devices 106 communication through network 107 via responder 102'. Step 324 can also include responder 102' conducting a key validation step 308b on the received key Pr 102p from message 313 using parameters 199a from a message 312b above. In exemplary embodiments, if message 313 has been received after more than an exemplary 10 seconds after the start of the timer in step 308c (e.g. maximum exemplary time between message 122 and message 123), then step 324 comprises responder 102' caching the data from message 313 in memory 102f until responder 102' receives a retry message 122 again. Upon responder 102' receiving the message 122 again (this time after message 313 has been received with data from message 313 cached in memory 102f), then responder 102' can send the cached data from message 313 recorded in the step 324.

Using the data received from DPP server 103, responder 102' can then send initiator 101 a DPP authentication response message 123. In exemplary embodiments, message 123 is sent as a unicast message from (i) the responder MAC address MAC.R 102n (which received message 122) to (ii) the initiator MAC address MAC.I 106n (which sent message 122), over (iii) WiFi link 192. As

depicted in Figure 3a, message 123 can include hash values of the bootstrap public keys $H(\text{Br } 102a)$, $H(\text{Bi } 101a)$, the responder ephemeral public key $\text{Pr } 102p$, and the ciphertext 315c. The ciphertext 315c can have encrypted values for (i) R-nonce 318b, (ii) I-nonce 317a, (iii) R-capabilities 318c, and (iv) ciphertext 324a. The ciphertext 315c can be encrypted with mutually derived symmetric encryption key $k2$ 314c. Ciphertext 324a can be encrypted with the mutually derived symmetric key $k2$ 314c. Ciphertext 324a can include a plaintext value for R-auth 404. As described above, R-auth 404 can comprise a hash value over an initiator nonce 317a received and the responder nonce 318b.

As depicted in Figure 3a, initiator 101 can receive DPP Authentication Response message 123. Although not depicted in Figure 3a, in exemplary embodiments, initiator 101 can conduct a step 308b to process and validate the received responder ephemeral public key $\text{Pr } 102p$, such as evaluating that the received key in message 123 is securely compatible with key pi 101d and key bi 101b, using the same steps as a key validation step 308c. In other words, a key validation step 308c by responder 102' may not be as complete as a key validation step 308b by initiator 101, since initiator 101 can compare the received key $\text{Pr } 102p$ from a message 123 with data from private keys pi 101d and key bi 101b using parameters 199a for the keys.

Initiator 101 can receive DPP authentication response message 123 from responder 102' and conduct a series of steps in order to process the message and reply to the responder 102'. At step 325, initiator 101 can record the validated, received responder ephemeral public key $\text{Pr } 102p$ in a responder public key table 101ta, as depicted in Figure 1d above. In other words, although Figure 1d does not depict a column for key $\text{Pr } 102p$, the column and data could be added for a responder as the data is received. Initiator ephemeral private key pi 101d and initiator ephemeral public key Pi 101c could be recorded in a responder public key table 101ta as well, when the keys can be utilized. Any insertion of key $\text{Pr } 102p$, pi 101d and Pi 101c (for responder 102' identified by key $\text{Br } 102a$ into table 101ta preferably has the data deleted upon conclusion of a DPP 191 and receipt and installation of credentials 109 as a configuration object for device 106 with initiator 101.

At step 319a, initiator 101 can conduct a second initiator key exchange step 319a, where initiator 101 can input the received, validated responder ephemeral public key $\text{Pr } 102p$ and the derived initiator ephemeral private key pi 101d into an ECDH key exchange algorithm 401b in order to calculate a shared secret key N 411. The second server key exchange step 319a is also depicted and described in connection with Figure 4b below. Shared secret key N 411 can also be used by initiator 101 in a step 327 below. Initiator 101 can input shared secret key N 411 into a key derivation function 402b in order to mutually derive the symmetric encryption key $k2$ 314c.

At step 320b, initiator 101 can use the derived key $k2$ 314c from a step 319a in order to conduct a decryption step 320b on received ciphertext 315c. A decryption step 320b on received ciphertext 315c using key $k2$ 314c is depicted and described in connection with Figure 4b below. Initiator 101 can read the plaintext 403b values from decryption step 320b. The plaintext values 403b can include

the initiator nonce 317a from a message 122 and message 309 above. The plaintext values 403b can also include a second ciphertext 324a (e.g. ciphertext 324a can be inside ciphertext 315c). Initiator 101 verifies that the initiator nonce 317a received in message 123 and decrypted in step 320b is equal to the initiator nonce 317a value sent in message 122 above. If the two initiator nonce 317a values are not equal, then initiator 101 can abort the DPP procedure and subsequent steps.

At step 327, initiator 101 can conduct a third initiator key exchange step 327, where initiator 101 can determine the secret shared key L 412 by inputting (i) the an ECC point addition of the responder bootstrap public key Br 102a and the responder ephemeral public key Pr 102p, and (ii) the initiator bootstrap private key bi 101b into an ECDH key exchange algorithm 401c. This third server key exchange step 327 is depicted and described in connection with Figure 4b below. The shared secret key L 412 along with share secret keys M 410 and N 411 can be used with a key derivation function 402c in order to output the mutually derived symmetric ciphering key ke 327a.

At step 328, initiator 101 can conduct a decryption step 328 on the ciphertext 324a using the key ke 327a. A decryption step 328 is depicted and described in connection with Figure 4c below. Initiator 101 can use the decryption step 328 in order to read the plaintext value of R-auth 404. R-auth 404 can comprise a hash 199e value over at least a combination of initiator nonce 317a and the responder nonce 318b. At step 328, initiator 101 can then also internally calculate the same value for R-auth 404, and if the internally calculated value and the received, decrypted value are equal, then the responder 102' can be consider authenticated with initiator 101. The calculation of R-auth 404 by initiator 101 can be according to section 6.2.3 of DPP specification version 1.0.

Initiator 101 can then conduct a step 322b, where step 322b can comprise calculating an initiator authentication value I-auth 404a. Initiator authentication value 404a can be similar to responder authentication value R-auth 404, except the values of initiator nonce 317a can be sequenced before a responder nonce 318b inside a hash 199e calculation over the two nonces. An initiator 101 can then encrypt the initiator authentication value 404a using an encryption step 322 using key ke 327a, as depicted in Figure 4c below. In other words, an encryption step 322 in Figure 4c depicts the encryption of R-auth 404, and encryption step 322b can comprise a initiator 101 encrypting the initiator authentication value I-auth 404a using the key ke 327a derived in a step 327. The resulting ciphertext with the encrypted initiator nonce valued 404a from an encryption step 322b can comprise ciphertext 324b. As depicted in Figure 3a, the collection of steps for an initiator 101 of the depicted steps 325, 319a, through 322b can comprise a step 331. A step 331 in summary allows initiator 101 to process and verify the received DPP authentication response message 123. As depicted in Figure 3a, additional steps for a responder 102' in initiator 101, and DPP server 103 to conduct in order to complete a device provisioning protocol session are depicted in Figure 3b below.

Figure 3b

Figure 3b is a simplified message flow diagram illustrating an exemplary system with exemplary data sent and received by a DPP server, a responder proxy, and an initiator, in accordance with exemplary embodiments. Before starting steps and message flows depicted in Figure 3b, the device 106 with initiator 101, responder 102', and DPP server 103 can complete the series of steps and message flows depicted above in Figure 3a. System 300b can include a DPP server 103, responder proxy 102', and initiator 101 operating in device 106. Responder proxy 102' can communicate with DPP server 103 via IP network 113 as depicted in Figure 1a. Responder proxy 102' can communicate with initiator 101 via WiFi link 192.

After completing the series of steps for a step 331 in Figure 3a, initiator 101 can send responder 102' a DPP authentication confirm message 123a. The data and steps to determine and calculate a DPP authentication confirm message 123a can be compatible with section 6.2.4 of DPP specification version 1.0. Note that although the DPP authentication confirm message 123a can be equivalent to the DPP authentication confirm message from section 6.2.4 of DPP specification version 1.0, an important difference can be the operation of responder 102' in Figure 3b versus a responder as contemplated by DPP specification version 1.0. In the present disclosure as depicted in Figure 3b, responder 102' does not record the responder bootstrap private key br 102b. However, using the technology depicted in Figure 3a and Figure 3b, a responder 102' can (i) send and receive messages that are fully compatible with a DPP protocol 191 (including DPP specification version 1.0), but (ii) DPP server 103 can record and operate on key br 102b. In other words, a initiator 101 may not "know" that a responder 102' operates as either (i) an responder 102* from Figure 1b or Figure 1c, or (ii) an responder 102' from Figure 3a and Figure 3b since the messages transmitted and received by the initiator 101 from responder 102' can be fully compatible with a DPP 191.

Other differences besides the location of recording and operating on key br 102b can apply as well for the responder mode 306a depicted in Figure 3a and Figure 3b, such as key pr 102r also remains in DPP server 103 and not recorded or operated on by responder 102'. The recording and operation of responder private keys br 102b and key pr 102r by DPP server 103 can comprise a first responder mode 306a, where responder mode 306a was selected by DPP server 103 or network 107 in a step 306 and communicated to responder 102' in a message 312b.

Responder 102' can listen for either (i) a broadcast message with the hash 199e values for the bootstrap public keys 102aa and 101aa, or (ii) a unicast message to MAC.R 102n from MAC.I 106n for initiator 101 operating in device 106. Responder 102' can receive the DPP authentication confirm message 123a and process the message. DPP authentication confirm message 123a can be processed by initiator 101 using a step 331 as depicted in Figure 3a. DPP authentication confirm message 123a

can include an encrypted value for initiator authentication value 404a as a ciphertext 324b, where ciphertext 324b is encrypted with derived key ke 327a in a step 322b.

Upon receipt of DPP authentication confirm message 123a from device 106, responder 102' can forward data in the message to DPP server 103 via secure session 302b. Responder 102' can send
5 DPP server 103 a message 329, where message 329 includes an identity for device 106 (which could be the depicted value of MAC.I 106n) as well as ciphertext 324b. Although not depicted for a message 329 in Figure 3b, ciphertext 324b can also include the hash values for the bootstrap public keys of hash value 102aa and/or hash value 101aa.

DPP server 103 can receive message 329 and record data from the message 329 and process
10 the data. DPP server 103 can use the identity for device 106 comprising MAC.I 106n (or another device identity such as ID.device 106i or hash value 101aa sent in a message 329). DPP server 103 can first conduct a decryption step 328b, where decryption step 328b can use the same steps as a decryption step 328 as depicted in Figure 4c below, with a difference being decryption step 328b operates on the ciphertext 324b containing the encrypted initiator authentication value I-auth 404a instead of the R-auth 404 value. After a step 328b, DPP server 103 can read the plaintext initiator authentication value I-auth 404a.
15

In step 330, DPP server 103 can compare the decrypted value for I-auth 404a with a calculated value for I-auth 404a and if the two values match then the initiator 101 can be considered by DPP server 103 to be authenticated. The calculation of I-auth 404a for DPP server 103 in a step 330 can
20 comprise a calculation of a secure hash value over at least the responder nonce 318b and bootstrap and ephemeral public keys, with an exemplary calculation for I-auth 404a shown in the sixth paragraph of page 50 of DPP specification version 1.0. Other calculations of an initiator authentication value over the responder nonce 318b are possible as well, without departing from the scope of the present disclosure.

In a step 333, responder 102' can select a set of network credentials 109 for device 106 using
25 initiator 101. The set of network credentials 109 could comprise the exemplary set of credentials 109 depicted and described in connection with Figure 1a. The set of network credentials 109 can comprise an SSID.network-AP 109a, a pairwise master key (PMK) PMK.network-AP 109b, and a set of configuration data comprising config.network-AP 109c. The value PMK.network-AP 109b can also
30 include an identifier for a PMK such as a PMKID. Or, for legacy WiFi connections that utilize WPA2 and similar or earlier standards, PMK.network-AP 109b can comprise a value for a shared secret key such as a PSK. The set of network credentials 109 can comprise values for device 106 to authenticate and encrypt data with network access point 105 in order for device 106 to connect with network access point 105. The network credentials 109 could support use of WiFi link 192 with WPA3 standards
35 from the WiFi Alliance TM in exemplary embodiments. Note that in some exemplary embodiments, a step 333 for selecting credentials 109 for device 106 can be omitted by responder 102' and the step

333 can be performed by DPP server 103 or another server within network 107, such as a server running a RADIUS server, or performing an extensible authentication protocol (EAP) based authentication with device 106.

In a step 334, for embodiments where responder 102' conducts the step 333 to select credentials 109, responder 102' can optionally encrypt the credentials 109 with an asymmetric ciphering algorithm using a public key for initiator 101 or device 106. The use of an asymmetric ciphering algorithms and associated public keys is depicted below in a step 334 in Figure 3e. Responder 102' can send DPP server 103 a message 335, where the message 335 includes an identity for device 106 such as MAC.I 106n, and a ciphertext 334a, where the ciphertext includes the asymmetrically encrypted network credentials 109'. The encryption key depicted in Figure 3b for ciphertext 334a is initiator ephemeral public key P_i 101p, although a different public key for device 106 or initiator 101 could be used as well, such as the initiator bootstrap public key or a public key in certificate cert.device 106m. The device certificate cert.device 106m could be (i) stored in a device database 103d and (ii) sent to DPP server 103 in a message 312a in Figure 3a, and then (ii) sent from DPP server 103 to responder proxy 102' in a message 312b in Figure 3a.

The "optional" notation for a message 335 depends on the source of credentials 109 in a system 300b. If a device operating responder 102' (such as an exemplary access point 105) is responsible for generating or operating credentials 109 (such as a single access point in a user's home), then responder 102' can be the source of the credentials 109 and send the message 335. If a network 107 is responsible for generating or operating credentials 109 (such as in a corporate or campus network of LAN access points 105), including operating EAP-based authentication via RADIUS or similar standards, then the message 335 can be optionally omitted since DPP server 103 could obtain the network credentials 109 from another source in a network 107 besides the device operating responder 102'.

After transmitting the DPP authentication confirm message 123a, in step 332, initiator 101 can take on a roll of enrollee, such that initiator 101 can prepare to send a configuration attribute 330a. Configuration attribute 330a can be used to specify that initiator 101 can take the roll of enrollee and is expecting a set of network credentials 109 from responder 102'. A configuration attribute 330a can be formatted according to section 6.3.4 of DPP specification version 1.0. For the exemplary embodiment depicted in Figure 1a, device 106 with initiator 101 would prefer a role of WiFi client (or "sta") for a configuration attribute 330a, but other possibilities exist as well. For the exemplary embodiment depicted in Figure 7 below a responder 102' operating within an access point 105 would prefer a role of WiFi access point (or "ap") for a configuration attribute 330a.

In step 332, initiator 101 can also generate a third random number, for an enrollee nonce E-nonce 332a. E-nonce 332a and other nonce values in Figure 3a such as the initiator nonce I-nonce 317a and responder nonce R-nonce 318b can be processed using nonce parameters 199g. Initiator 101 can then conduct an encryption step 322c, where encryption step 322c can comprise (i) an encryption

step 322 using key ke 327a, and (ii) the plaintext to be encrypted can be at least the E-nonce 332a and the configuration attribute 330a. The output of an encryption step 322c can be a ciphertext 324c. Initiator 101 can then send responder 102' a DPP Configuration Request message 123b, where the configuration request 123b can include the ciphertext 324c, where the plaintext in ciphertext 324c can include E-nonce 332a and the configuration attribute 330a. DPP Configuration Request message 123b can be processed by initiator 101 according to section 6.3.3 of DPP specification version 1.0.

Responder 102' can receive DPP Configuration Request message 123b via connection 192 and using DPP application 102g. As contemplated herein, a DPP application 102g can perform the steps for responder 102' depicted in Figure 3a, such as transmitting/receiving DPP 191 messages using a first MAC.R 102n for connection 192 and transmitting/receiving messages with DPP server 103 using MAC.N 102q via access network 112. DPP application 102g could also perform the depicted steps 308b above to validate the received initiator ephemeral public key Pi 101c. For embodiments where responder 102' uses a WAN radio 102h to connect with IP network 113 and DPP server 103, then DPP application 102g could also use a different identity than MAC.N 102q to identify the physical interface of WAN radio 102h, such as an IMEI, or other possibilities exist as well. For other embodiments MAC.R 102n could also potentially be the same as MAC.N 102q, and other possibilities exist as well for the identification of physical network interfaces used by responder 102' or a device operating responder 102' without departing from the scope of the present disclosure.

Responder 102' can send DPP server 103 a message 336 via secure connection 302b, where message 336 can include the ciphertext 324c. Message 336 can also include an identity for device 106, such as, but not limited to, MAC.I 106n or ID.device 106i, since both DPP server 103 and responder 102' can communicate with a plurality of different devices 106 over time, and DPP server 103 may prefer to know which ciphertext 324c is associated with which device 106 in order to process the ciphertext 324c. As depicted in Figure 3b, message 336 can include the encrypted values for E-nonce 332a and configuration attribute 330a.

DPP server 103 can receive message 336 and process the message 336. DPP server 103 can conduct a decryption step 328c using decryption step 328 depicted in Figure 4c below. A difference between decryption step 328 and decryption step 328c is that decryption step 328c uses the ciphertext 324c as input into the symmetric decryption algorithm 405b. DPP server 103 can use the previously calculated key ke 327a in order to read the plaintext within ciphertext 324c. In a step 328c, DPP server 103 can record the plaintext E-nonce 332a and configuration attribute 330a.

In step 337, DPP server 103 can process the received configuration attribute 330a. For the exemplary embodiment depicted in Figure 1a, configuration attribute 330a can specify that device 106 with initiator 101 may have a role of WiFi client. For a step 337, DPP server 103 can select a set of network credentials 109 for device 106 using initiator 101 in order to create a configuration object. The configuration object in a step 337 can be compatible with a configuration object in section 6.3.6

of DPP specification version 1.0. For embodiments where DPP server 103 received a message 335 with an asymmetrically encrypted set of credentials 109' from responder 102', then in a step 337 DPP server 103 can select and use the credentials 109' as the data for a configuration object. If encrypted credentials 109' are selected by DPP server 103 in a step 337, then the credentials 109' after decryption by device 106 using a decryption step 339 can comprise a configuration object per section 6.3.6 of DPP specification version 1.0. As contemplated herein, a reference to DPP specification version 1.0 can also include subsequent and related versions of the standard that would be substantially compatible with version 1.0.

Continuing at step 337, for embodiments where DPP server 103 did not receive a message 335 from responder 102', then DPP server 103 can be responsible for obtaining credentials 109, such as if network 107 is responsible for generating or creating the network credentials 109 for device 106 (such as if access point 105 uses EAP-based authentication and other possibilities exist as well). Note that DPP server 103 could also receive an asymmetrically encrypted set of network credentials 109' from another server or external network 108 in a step 337, where DPP server 103 or network 107 sends an identity for device 106 such as Bi 101a or H(BI) 101aa to the external network 108 and the external network 108 can use the identity 106 to conduct a step 334 for credential 109'. Or, in a step 337, DPP server 103 could receive plaintext network credentials 109 from another server in network 107 or generate the network credentials 109 using data recorded for access point 105 in a server database 103a. Other possibilities exist as well for the source of a set of network credentials 109 for DPP server 103 in a step 337 without departing from the scope of the present disclosure.

Continuing at step 337, DPP server 103 can process a configuration object for network credentials 109, and the configuration object can include an SSID value of SSID.network-AP 109a, a pairwise master key value of PMK.network-AP 109b, and configuration values of config.network-AP 109c. The value of PMK.network-AP 109b can also include a value for PMKID (e.g. PMK identifier), such that PMK.network-AP 109b is used by each of a client and an access point with the PMKID value. The configuration values of config.network-AP 109c in a configuration object could also specify required supporting data for operation of device 106 with an access point 105 after a configuration step 106a, such as an operating class 131a, channel list 131b, an authentication mode for device 106 to operate, which could be PSK, PMK, SAE, EAP, also with any required supporting data and/or identities in order to use the authentication mode. The configuration values of config.network-AP 109c in a configuration object could also include a passphrase or pre-shared key values of PSK for legacy WPA2 networks. In other words, credentials 109 for other authentication schemes between a device and an access point could be supported as well, including EAP-TLS, EAP-PSK, EAP-AKA, EAP-AKA', etc.

As contemplated herein, a set of network credentials 109 can include either (i) network credentials 109 for a selected primary access network, and/or (ii) network credentials 109 for both a

selected primary access network and a selected backup access network. Or, in another exemplary embodiment, although a single configuration object of network credentials 109 is depicted in Figure 3b, DPP server 103 could process (i) a first set of network credentials 109 for selected primary access network into a first configuration object and then (ii) a second set of network access credentials 109 for selected backup access network into a second configuration object. At the conclusion of a step 337, DPP server 103 can record and process a configuration object comprising a set of plaintext network credentials 109 or asymmetrically encrypted set of network credentials 109' for device 106 operating initiator 101, where device 106 can be identified by an identity such as ID.device 106i or MAC.I 106n or H(Bi) 101aa.

For a step 338 in Figure 3b, DPP server 103 can conduct a signature creation step 338 as depicted and described in connection with Figure 3d below. The configuration object with network credentials 109 (or encrypted credentials 109') can be signed using the recorded responder ephemeral private key pr 102r, corresponding to public key Pr 102p sent in message 325. The digital signature by DPP server 103 in Figure 3b can comprise a digital signature 338c. Or, in another embodiment the configuration object with network credentials 109 can be signed using the recorded responder bootstrap private key 102b, where initiator 101 could verify the signature 338c with the bootstrap public key Br 102a. Note that some exemplary embodiments contemplate that DPP server 103 receives (i) credentials 109 or asymmetrically encrypted credentials 109' from (ii) another server in network 107 or an external network 108, and a step 338 can comprise DPP server 103 creating a digital signature 338c for the (i) credentials 109 or asymmetrically encrypted credentials 109' from (ii) another server in network 107 or an external network 108. The steps to create and verify a signature with ECC PKI keys can be according to FIPS publication 186-4, and other possibilities exist as well. The use of a digital signature 338c for credentials 109 or 109' is optional in exemplary embodiments, and for some embodiments a digital signature 338c could be omitted (as well as signature creation step 338 and signature verification step 340).

At step 332d, DPP server 103 can conduct an encryption step 332d using an encryption step 322 depicted and described in connection with Figure 4c below. Encryption step 332d can be performed using key ke 327a for device 106. For encryption step 332d, the plaintext values of (i) the E-nonce 332a and (ii) the configuration object comprising network credentials 109 (or asymmetrically encrypted credentials 109') can be input into a symmetric encryption algorithm 405a in order to generate a ciphertext 324d. Note that digital signature 338c can be included in an encryption step 332d as well, although digital signature 338c could also optionally not be included in a ciphertext 324d and sent in a subsequent message 341 and 124 external to ciphertext 324d. As depicted in Figure 3b, the collection of steps 328c through 322d can comprise a step 349.

DPP server 103 can then send responder 102' a message 341, where message 341 includes the ciphertext 324d. Message 338 can also include (i) an identity for device 106 such as MAC.I 106n or

ID.device 106i, and (ii) a signature 338c from a signature creation step 338. Message 338 can be sent via secure connection 302b, as depicted in Figure 3b. Although signature 338c is depicted as within ciphertext 324d for a message 341, signature 338c could be external to ciphertext 324d in a message 341.

5 Responder 102' using DPP application 102g can receive message 341 via secure connection 302b. Responder 102' can use an optional identity for device 106 in a message 341 in order to determine which of a potential plurality of devices 106 is associated with ciphertext 324d in order to forward the data to device 106. Responder 102' can then send initiator 101 a DPP configuration response message 124 via connection 192, where message 124 can include the ciphertext 324d, and
10 the ciphertext 324d can include the E-nonce 332a, the signature 338c, and either network credentials 109 or asymmetrically encrypted network credentials 109'. If network credentials 109' are sent in ciphertext 324d, then the network credentials 109' can comprise a set of "double encrypted" network credentials of (i) asymmetrically encrypted network credentials 109' and (ii) then encrypted again in a ciphertext 324d using the key ke 327a.

15 Initiator 101 can receive message 124 from responder 102' with ciphertext 324d. Initiator 101 can conduct a decryption step 328d using decryption step 328 depicted in Figure 4c below. A difference between decryption step 328 and decryption step 328d is that decryption step 328d uses the ciphertext 324d as input into the symmetric decryption algorithm 405b. Initiator 101 can use the previously calculated key ke 327a in order to read the plaintext within ciphertext 324d. Initiator 101
20 can confirm that plaintext in ciphertext 324d includes the previously sent E-nonce 332a. Initiator 101 can extract a plaintext value for (i) network credentials 109 or (ii) asymmetrically encrypted network credentials 109' using symmetric decryption algorithm 405b and key ke 327a. In other words, with credentials 109' decryption step 328d removes the first layer of "double encryption" where the first layer was encrypted with key ke 327a, and the "plaintext" after decryption step 328d for credentials
25 109' can comprise the asymmetrically encrypted network credentials 109' (e.g. with the first layer of encryption with key ke 327a has been removed).

Initiator 101 can conduct a signature verification step 340 as depicted and described in connection with Figure 3d above in order to verify signature 338c received in message 124. Initiator 101 can use either (i) responder ephemeral bootstrap public key Pr 102p or (ii) responder bootstrap
30 public key Br 102a in order to conduct a signature verification step 340 on digital signature 338c. If plaintext network credentials 109 are included in ciphertext 324d, then initiator 101 can read the plaintext values for credentials 109. If asymmetrically encrypted network credentials 109' are included in ciphertext 324d, then initiator 101 or device 106 operating initiator 101 can conduct an asymmetric decryption step 339 depicted below in Figure 3e in order to convert credentials 109' into
35 plaintext credentials 109.

After reading the plaintext network credentials 109 and optionally verifying a signature using recorded public keys 102p or 102a, device 106 can complete the configuration step 106a by loading the network credentials 109 and begin operating a WiFi radio 106r with the network credentials 109. In other words, the WiFi data link 192 depicted in Figure 1a can be converted into a fully authenticated and encrypted WiFi connection using network credentials 109 and standards such as WPA2, WPA3, and also subsequent or related standards where WiFi nodes such as clients or access points use a compatible set of network credentials 109. For example, device 106 could begin scanning for SSID.network-AP 109a on a channel in Config.network-AP 109c, and then send access point 105 a probe request. In this manner, device 106 can obtain connectivity to network access point 105.

Figure 3c

Figure 3c is an illustration of an exemplary server database, with tables for a server database recording exemplary data, in accordance with exemplary embodiments. A server database 103a depicted and described above in connection with Figure 1a and Figure 3a can record data for a DPP server 103 to work with a responder 102' in order for the responder 102' to conduct a device provisioning protocol 191 with a device 106 operating an initiator 101. A server database 103a could record a plurality of tables for a plurality of devices 106 and a plurality of responders 102', including the depicted device credentials table 345. Other possibilities exist as well for the organization, tables, and recorded data within a server database 103a without departing from the scope of the present disclosure. Data within server database 103a could be encrypted using a symmetric key. As depicted in Figure 1a, a DPP server 103 operating in a network 107 could operate and record a server database 103a.

Server database 103a can record a device identity for device 106 of MAC.I 106n, a responder identity which could be MAC address MAC.N 102q, a responder mode 306a comprising a mode of operation for the DPP server 103 and responder 102', a selected set of Cryptographic Parameters 199a, a Responder Ephemeral Public Key Pr 102p, a Responder Ephemeral Private Key pr 102r, an Initiator Ephemeral Public Key Pi 101c, Bootstrap keys 399, an optional PKEX Key 198, a Shared Secret k1 314a, a Shared Secret k2 314c, and a Shared Secret ke 327a. Different identities for device 106 and responder 102' could be recorded in a server database 103a as well, such as ID.device 106i or an identity for a responder 102' that is different than a MAC address for responder 102', such as a serial number or unique identifier for responder 102'. In exemplary embodiments, hash value 101aa over the initiator bootstrap public key Bi 101a can be used for a device identity in a server database 103a.

For exemplary embodiments, not all depicted values may be recorded in a server database 103a, as shown by the depicted value of "--", which can mean that the value is not recorded or stored in a server database 103a. A responder 102' could record and operate on the value depicted as "--" for

a server database 103a in Figure 3c for some responder modes 306a. For example, with the row in server database 103a depicted with a mode “Fig. 5a”, the responder 102’ could derive, record, and operate on the responder ephemeral public key Pr 102p, and DPP server 103 may not need or record the key. Similarly, with the row in server database 103a depicted in Figure 3c as a mode of “Fig. 5a”, the responder 102’ could derive or operate on shared secret keys k1 314a, k2 314c and ke 327e, while server database 103a would not need to record or operate on the shared secret keys in order for responder 102’ with a mode 306a of “Fig. 5a” to conduct a DPP 191 with initiator 101.

Note that in alternative exemplary embodiments, a server database 103a could record the values M 410, N 411, and L 412 respectively instead of key k1 314a, k2 314c, and/or ke 327e, and either server 103 or responder 102’ could use the key derivation function 402a, 402b, and/or 402c, respectively in order to convert the values M, N, and L into keys k1, k2, and ke as depicted in Figures 4a through 4c. In exemplary embodiments, the time that a device shared secret key or a derived ephemeral PKI key, or a bootstrap key is recorded in a server database 103a can be minimized to the minimum time necessary to conduct a DPP 191. For example, shared secret keys and ephemeral PKI keys recorded in a server database 103a can be deleted once a configuration step 106a has been completed for a device 101, such as at the time DPP server 103 receives a confirmation that device 106 has begun using the credentials 109 with a network access point 105.

Responder mode 306a can specify an operating mode for responder 102’ and DPP server 103, such that a sequence of steps and transmitting/receiving PKI and/or shared secret keys and/or related supporting messages can be specified by the responder mode 306a. A responder mode 306a could also be referred to as a “server mode” 306a, since the mode specifies the operation of a DPP server 103. Or, responder mode 306a could be referred to as a “mode 306a”, since both steps for responder 102’ and DPP server 102 are affected by the mode. Other terms could be used to refer to the selected mode as well, such as a “method”, “procedure”, or a “process”. Figure 2a above also described a first, second, and third operating responder modes 306a. A first mode is depicted as “Fig. 3” in Figure 3c and can comprise the mode 306a depicted in Figure 3a and Figure 3b. Note that all depicted ephemeral PKI keys and bootstrap PKI keys, as well as all depicted shared secret keys can be recorded in a server database 103a for a responder 102’ for the first mode. In other words, when conducting a DPP 191, responder 102’ and DPP server 103 could have all the depicted keys recorded in a server database 103a for the first mode. This first mode for an initiator mode 306a can be preferred for embodiments where (i) responder 102’ may have a lower level of trust with network 107, or (ii) responder 102’ could have a lower level of processing resources, or (iii) an owner for device 106 or a network 107 may prefer that responder bootstrap private key br 102b is not shared externally to network 107, (iv) regulatory requirements where device 106 could comprise a high value or sensitive piece of equipment, or (v) responder 102’ conducts a secure session 302b without a responder certificate cert.responder 102m. Other possibilities exist as well for the reasons different modes 306a may be

used in a server database 103a to specify the steps for a responder 102' and DPP server 103 to conduct a DPP 191 with device 101.

A second responder mode 306a is depicted as "Fig. 5a" in Figure 3c, where (a) the steps and message flows in Figure 5 for responder 102' and DPP server 103 can comprise (b) the second mode.

Note that all relevant bootstrap PKI keys 399 are recorded in a server database 103a for the second responder mode 306a depicted in Figure 5a. The second mode 306a can also omit (i) the initiator ephemeral public key P_i 101c, (ii) and the responder ephemeral private key pr 102r from being recorded in a server database 103a. Note that secret shared keys can optionally not be recorded in a server database 103a for the second mode depicted in Figure 5a. This exemplary embodiment of a second responder mode 306a may be preferred for cases where (i) responder 102' can handle more of the cryptographic operations (e.g. reducing DPP server 103 processing load), but (ii) a network 107 could prefer that the responder bootstrap private key br 102b is not transmitted to or recorded in responder 102'. A third responder mode 306a is depicted as "Fig. 5b" in Figure 3c, where (a) the steps and message flows in Figure 5b for responder 102' and DPP server 103 can comprise (b) the third mode.

A fourth responder mode 306a in Figure 3c can comprise responder 102' operating as a responder 102* depicted in Figure 1b and Figure 1c, and in this case responder 102* can record the keys shown for the third mode 306a in Figure 3b (depicted as "102*"). A DPP server 103 can still provide the bootstrap keys 399 shown to responder 102* in this third mode of operation, or the responder 102* could obtain the bootstrap keys 399 from other means, such as "out of band" messages 121 and 126 in Figure 1c. In addition, a responder 102* could obtain the bootstrap keys "in band" (e.g. via WiFi link 192) using PKEX key 198. In exemplary embodiments, a fourth responder mode 306a where responder 102' operates as a responder 102* as depicted in Figure 1b and Figure 1c (e.g. responder 102' would record responder bootstrap private key br 102b), responder 102 can conduct a step 604 as depicted in Figure 6 in order to obtain credentials 109 from a DPP server 103. In other words, although the conventional technology depicted for a responder 102* in Figure 1b and Figure 1c contemplates sending credentials 109 from responder 102* to initiator 101, the conventional technology does not suggest responder 102* obtains the credentials from a DPP server 103. A step 604 as depicted and described below in connection with Figure 6 could be used by a responder 102' with the fourth responder mode 306a in order for responder 102' (operating as a responder 102*) in order to obtain credentials 109 or credentials 109' from a DPP server 103.

Bootstrap keys 399 in a server database 103 can record the bootstrap keys depicted in Figure 3c. For embodiments where a device 106 can support multiple different sets of cryptographic parameters 199a, a server database 103a can record the selected set of cryptographic parameters 199a for communication between a responder 102' and device 106 with initiator 101. The keys for "bootstrap keys 399" can be according to the selected set of parameters 199a. In other words, a device

database 104 in Figure 2a above can record all versions of the bootstrap PKI keys for all supported cryptographic parameters 199a for a device 106. However, since a set of cryptographic parameters 199a can be selected for a DPP 191 (e.g. based on data received from initiator 101 in message 122 such as key Pi 101c or hash value 102aa), then the bootstrap keys 399 for that selected set of cryptographic parameters 199a can be recorded in a server database 103a.

A server database 103a can also record a pre-shared PKEX key 198, if PKEX is optionally supported by initiator 101. PKEX key 198 can be used by responder 102' to securely transfer bootstrap public keys in band (i.e. via connection 192 in Figure 1a and Figure 3a). The use of a PKEX protocol to exchange the public keys is described in section 5.6 of DPP specification version 1.0. For embodiments where responder 102' uses PKEX key 198 with initiator 101, the responder bootstrap private key br 102b preferably continues to reside within a network 107 and can be in a server database 103a. Embodiments can omit the use of PKEX key 198, as depicted with the value of "NA" for the key in Figure 3c. Derived, shared secret keys with initiator 101 can also be recorded in a server database 103a. The derived shared secret keys could comprise key k1 314a, key k2 314c, and key ke 327e. The steps to derive and use these keys are depicted and described below in Figure 4a, Figure 4b, and Figure 4c, respectively. As discussed above, some keys may reside in server database 103a, and other keys may reside within a responder 102' (depicted with a "-"), depending on the responder mode 306a.

The present disclosure contemplates other implementations of a device provisioning protocol besides the DPP specification version 1.0. These other implementations could have different mutually derived shared secret keys than those depicted in Figure 3c, and the different mutually derived shared secret keys could be recorded in a server database 103a as well. For example, different key derivation functions 402 could be used to derive a secret shared key than that specified in DPP specification version 1.0. In addition, other or different PKI keys for a responder 102' and/or initiator 101 could be recorded in a server database 103a. For example, the DPP specification version 1.0 could be extended to explicitly include a DPP server 103 public and private key, in order to extend support with a DPP server 103, and in these embodiments the DPP server 103 public and private keys could be included in a server database 103. Other possibilities exist as well for a server database 103a to record PKI keys and shared secret keys without departing from the scope of the present disclosure.

A server database 103a could also record a device credentials table 345. Device credentials table 345 can record a device identity ID.device 106i, MAC.I 106n, a set of primary network credentials 350a, and a set of backup network credentials 350b. The set of primary network credentials 350a can comprise the set of credentials (plus configuration) for a device 106 to use with a selected primary network, such as an exemplary network access point 105 in Figure 1a. The set of backup network credentials 350b can comprise the set of credentials (plus configuration) for a device 106 to use with a selected backup network. The credentials in a device credentials table 345 could be obtained

from during a step 333 or a step 337 as depicted and described above in connection with Figure 3b. Note that a DPP server 103 could conduct a step 337 in order to select the network credentials 109, including receiving the credentials 109 from an external network 108. DPP server 103 could receive and record the asymmetrically encrypted network credentials 109' in a device credentials table 345.

5 For example, with the first device depicted in a device credentials table 345, the credentials in device table for the first device depicted could comprise a set of plaintext credentials 109 depicted in Figure 1a. The depicted designation of "109-1" can identify the specific set of plaintext credentials 109 for a primary network 350a device 106 with ID.device 106i. The device in the second row can use the same plaintext network credentials 109. The credentials stored for the primary network for
10 the third device can comprise depicted credentials "109' -1", which can mean asymmetrically encrypted network credentials 109' are recorded. The asymmetrically encrypted network credentials 109' could be received from an external network 108 (or possibly just securely recorded by a network 107), and only the third device could feasibly decrypt the credentials 109' using a step 339 and a recorded private key for the device 106.

15 In exemplary embodiments, the asymmetrically encrypted credentials 109' are encrypted using Elgamal encryption and either (i) the initiator bootstrap public key Bi 101a or (ii) the initiator ephemeral public key Pi 101c. The DPP server 103 could send either (i) the initiator bootstrap public key Bi 101a or (ii) the initiator ephemeral public key Pi 101c (along with an identity for device 106 such as ID.device 106i or MAC.I 106n) to a server that conducts an asymmetric encryption step 334
20 on plaintext credentials 109. The DPP server 103 could receive the encrypted credentials 109' and record them for device 106 with ID.device 106i in a device credentials table 345. Additional details for receiving credentials from an external network 108, and recording the encrypted credentials 109' in a server database 103 are depicted and described below in connection with Figure 3b, depicted as credentials 109'.

25 Note the present disclosure contemplates that (a) credentials 109 or 109' could utilize wireless networking technology other than WiFi or 802.11 based technology. For example, surrounding wireless networks for device 101 could use exemplary potential 5G wireless technology or LPWAN networking technology potential wireless networks for device 106. Thus, credentials 109 or 109' in a server database 103a and used by device 106 can support networking technology other than WiFi. As
30 one example, an initiator 101 could be operated by an Ethernet interface on a computer or server, and an Ethernet switch could operate a responder 102'. The flow of messages for a DPP 191 protocol could be conducted via a physical wired Ethernet link instead of a WiFi link 192. In this manner, the device operating an initiator 101 could be authenticated and configured by an Ethernet switch operating the responder 102'. As one example, the values and configuration in a dynamic host configuration protocol (DHCP) could be sent as credentials 109 or credentials 109' in a message 124
35 from an Ethernet switch operating a responder 102'.

In another exemplary embodiment, a wireless WAN network, such as a 5G or 6G wireless network according to future 3GPP or ETSI standards, could implement support for the device provisioning protocol contemplated herein. A base station or base transceiver station for the wireless network could operate as a responder 102' and a user device desiring to connect with the network could operate as an initiator 101. The user device and base station could conduct a DPP 191 and the base station could send credentials 109 or credentials 109' for the 5G or 6G wireless network to the user device in a DPP message 124 after successful authentication using DPP. The credentials 109 or 109' for either a wired Ethernet network or a wireless WAN network could be recorded for a device 106 or a plurality of devices 106 in a server database 103a. The use of a primary and backup network for device 101 can provide increased reliability for the operation of device 106, such that if the primary access network for credentials 350a is not available then device 101 could use the selected backup network with credentials 350b in order for device 106 to maintain connectivity to an IP network 113.

Figure 3d

Figure 3d is a flow chart illustrating exemplary steps for creating and verifying a digital signature using PKI keys, parameters, and data input, in accordance with exemplary embodiments. Nodes in a system 100 and other systems herein can create and verify digital signatures using the steps depicted for a signature creation step 338 and a signature verification step 340. In exemplary embodiments, a DPP server 103 or a responder proxy 102 could conduct a signature creation step 338 and an initiator 101 operating in a device 106 can conduct a signature verification step 340. In Figure 3d, signature creation 338 can use the sub-steps of obtaining a message to sign, calculating a message digest 338a, using a private key such as responder ephemeral private key pr 102r, using a signature algorithm 338b, inputting parameters 199i, and calculating a resulting signature 338c. The steps and sub-steps depicted for DPP server creating signature 338 in Figure 3d can also be used by other nodes, including signature creation 338d for a responder 102' to create a signature 338c, where responder 102' creates a signature 338c for network credentials 109 received from a DPP server 103 in Figure 6 below.

Signature algorithm 338a and a corresponding signature verification algorithm for a signature verification step 340 below could comprise an RSA-based digital signature algorithm (DSA) or an ECC based elliptic curve digital signature algorithm (ECDSA), and other possibilities exist as well for signature algorithm 338a and the corresponding signature verification algorithm without departing from the scope of the present disclosure. When using a DSA or ECDSA algorithm in non-deterministic mode for a signature creation 338, a value of "k" or "r", which could comprise a random number can be associated with the digital signature 338c. When using a DSA or ECDSA in deterministic mode for preferred exemplary embodiments, such as specified in IETF RFC 6979 and

titled “Deterministic Usage of the Digital Signature Algorithm (DSA) and Elliptic Curve Digital Signature Algorithm (ECDSA)”, which are hereby incorporated by reference, then the requirement for a separately transmitted random number with digital signature 338c (such as value “k” or “r”) can be optionally omitted, such that “r” can be deterministically calculated based on the message to sign. In
5 exemplary embodiments, device 106 and servers such as DPP sever 103 can utilize deterministic ECDSA without also sending a random number along with a digital signature, although the value “r” from the deterministic mode could be sent with the digital signature 338c. In other words, a value can be sent with the digital signature 338c that is deterministically derived and associated with the message to sign. In other exemplary embodiments, a random number can be generated a derived value for the
10 random number such as “r” sent with digital signature 338c (e.g. not in deterministic mode).

For a signature creation 338 step, the exemplary message to sign comprises either plaintext network credentials 109 or asymmetrically encrypted network credentials 109'. The message to sign values can be transmitted to the initiator 101 in an encrypted format, such as shown for message 124 in Figure 3b. The message to sign values can be input into a message digest algorithm 338a, which
15 could comprise a standard algorithm such as SHA-256, SHA-3, or similar algorithms. In exemplary embodiments, the message digest algorithm 338a can be specified by a hash value 199e in a selected set of parameters 199a from parameters 199. The output of message digest algorithm 338a can be input along with parameters 199i and a private key such as pr 102d into signature algorithm 338b. Parameters 199i can specify encoding rules, padding, key lengths, selected algorithms, curve names,
20 and other values or fields necessary to utilize a signature algorithm 338b. Both a signature creation step 338 and a signature verification step 340 use the same or equivalent values for parameters 199i. A private key used in a signature creation step 338 can comprise a private key held and recorded by a network 107 where initiator 101 can record the corresponding public key.

Figure 3d depicts the private key as responder ephemeral private key pr 102r, although
25 alternatively the private key could be responder bootstrap private key br 102b or a different compatible private key recorded by network 107. The selection of (i) key pr 102r or key br 102b for a signature creation step 338 and (ii) key Pr 102p or key Br 102a for a signature verification step 340 can be specified in a set of cryptographic parameters 199a, or simply included in software for DPP server 103 and initiator 101. For embodiments where signature 338 uses a private key of responder bootstrap private key br 102b, then initiator 101 could record and use responder bootstrap public key Br 102a.
30 For embodiments where signature 338 uses a private key for network 107, then initiator 101 could record a certificate for the network 107 and use the public key from the certificate for a signature verification step 340.

Signature verification 340 can comprise a step using the sub-steps of (i) obtaining a message
35 to verify, (ii) calculating a message digest 338a for the message to verify, (iii) using a public key corresponding to the private key, such as responder ephemeral public key Pr 102p for responder

ephemeral private key pr_{102r} , (iv) using a signature verification algorithm corresponding to signature creation algorithm 338b, (v) inputting parameters 199i and received signature 338c into the signature verification algorithm, and (vi) determining a pass/fail. If the signature 338c received matches a calculated signature using the public key and message to verify, then the received signature 338c passes or is validated or verified. If the signature 338c does not match the calculated signature in a step 340, then the signature 338c is considered to fail or not be verified. In exemplary embodiments, a device 106 operating an initiator 101 can conduct a signature verification step 340 for a set of credentials 109 (or encrypted credentials 109') before accepting and applying the credentials in a configuration step 106a.

Figure 3e

Figure 3e is a flow chart illustrating exemplary steps for using asymmetric ciphering in order to encrypt credentials using a public key and decrypting credentials using the corresponding private key, in accordance with exemplary embodiments. An encryption 334 step could be performed by (i) a responder 102' as depicted and described in connection with Figure 3b above in order to create a ciphertext 334a, (ii) an external network 108 (different than network 107 operating DPP server 103), where the external network 108 could be the source of a plaintext set of credentials 109 for device 106. The purpose of an encryption 334 step can be to convert a plaintext set of network credentials 109 into the ciphertext 334a, which can comprise and be depicted as an asymmetrically encrypted set of network credentials 109' in Figure 3b, Figure 3c, Figure 6, etc. An exemplary reason that an asymmetrically encrypted set of network credentials 109' may be preferred for receipt by and operation inside DPP server 103 is that DPP server 103 is that a network access point 105 or an external network 108 may prefer that DPP server 103 or responder 102' not be able to read asymmetrically encrypted set of network credentials 109', while device 106 could decrypt and read the plaintext set of network credentials 109'.

An encryption step 334 can use an asymmetric encryption algorithm 351a, where asymmetric encryption algorithm 351a can be based on either RSA or ECC algorithms. The use of an asymmetric encryption algorithm 351a and an asymmetric decryption algorithm 351b with RSA based keys is described in Internet Engineering Task Force Request for Comments (RFC) 8017, which is titled "PKCS #1: RSA Cryptography Specifications Version 2.2", which is hereby incorporated by reference. The use of asymmetric encryption algorithm 351a and an asymmetric decryption algorithm 351b with elliptic curve cryptography can be accomplished with ElGamal asymmetric encryption, as summarized in the Wikipedia article for ElGamal encryption from April 3, 2018, which is herein incorporated by reference. Other possibilities exist as well for the use of ECC algorithms for

asymmetric encryption, including IEEE 1363a standards and ISO/IEC standard 18033-2. Other possibilities exist as well without departing from the scope of the present disclosure.

For an encryption step 334, the initiator ephemeral public key P_i 101c can be input into an asymmetric encryption algorithm 351a. As depicted, the plaintext values can be input as well, along with a set of parameters 199h. Parameters 199h can be a subset of parameters 199 depicted and described in connection with Figure 2a above, and can include a set of asymmetric encryption parameters 199h. Note that where initiator 101 or responder 102' is depicted as recording or using a set of parameters 199a, the set of parameters 199a can specify a selected row in a set of parameters 199, such that a single value for parameters 199a can identify all values for 199a through 199h in a set of cryptographic parameters 199. Parameters 199h can specify key lengths, encoding rules, byte orders, use of point compression, and other values necessary to operate an asymmetric ciphering algorithms 351a and 351b. The output of an asymmetric encryption algorithm 351a can be ciphertext 334a.

In a different, but related, exemplary embodiment, the public key used in an encryption step 334 can comprise a public key from certificate cert.device 106m as depicted for device 106 in Figure 1a. The public key from certificate cert.device 106m may be more accessible to an external network 107 or a node than key P_i 101c. The parameters 199h used for a step 334 in this different embodiment could be recorded in the certificate cert.device 106m. Cert.device 106m could be publicly shared by a device manufacturer or device distributor along with a device identity in the certificate such as ID.device 106i or MAC.I 106n. For this different, but related embodiment of encryption step 334, the secret key for device 106 or initiator 101 to utilize in a decryption step 339 below can comprise the secret key SK.device 106s.

As depicted in Figure 3e, and encryption 334 step could also comprise an embodiment of encrypting a plaintext symmetric key using asymmetric encryption, such that the symmetric key could be used with a symmetric ciphering algorithms 405a and 405b such as AES, AES-SIV, or Triple Data Encryption Standard (3DES) or Blowfish, or related algorithms. In this embodiment, (i) the symmetric key for encryption network credentials 109 could comprise a random number derived by responder 102' (or a node for the owner of credentials 109), and (ii) ciphertext 334a could comprise two parts, where the first part is the encrypted symmetric key and the second part is a symmetric encryption of network credentials 109. In other words, for this embodiment where (a) a symmetric key for encrypting/decrypting credentials 109 is asymmetrically encrypted with algorithm 351a, then (b) network credentials 109 would be symmetrically encrypted/decrypted using the asymmetrically encrypted/decrypted symmetric key.

For a decryption step 339, the initiator ephemeral private key p_i 101d for initiator 101 in device 106 can be input into an asymmetric decryption algorithm 351b. The initiator ephemeral private key p_i 101d can be the secret key corresponding to the public key P_i 101c used with the asymmetric

encryption algorithms 351a. Asymmetric decryption algorithm 351b can correspond to asymmetric encryption algorithm 351a, such that if an ECC curve with a given set of parameters 199h is used with asymmetric encryption algorithm 351a, then the same ECC curve with the same or equivalent set of parameters 199h can be used with asymmetric decryption algorithm 351b. The output of an asymmetric decryption algorithm 351b can be the plaintext, as depicted in Figure 3e.

In addition, although credentials 109 are depicted as decrypted in a decryption step 339, an asymmetric decryption algorithm 351b could be used to read a plaintext symmetric key for a symmetric ciphering algorithm, where the plaintext credentials 109 could be read by inputting the plaintext symmetric key and encrypted credentials 109 into a symmetric ciphering algorithm 405b. Standards such as IEEE 1363a standards and ISO/IEC standard 18033-2 describe using asymmetric algorithms and public keys to encrypt symmetric keys, and asymmetric decryption algorithms and private keys to decrypt the asymmetrically encrypted symmetric keys, where (i) the plaintext and ciphertext can be converted using the symmetric key, and then (ii) a second plaintext comprising credentials 109 could be read using the symmetric key and a symmetric ciphering algorithm. In other words, as described in two paragraphs above, the ciphertext 334a could comprise two parts, where (i) the first part comprises a symmetric key asymmetrically decrypted by 351b, and then (ii) the second part comprises the symmetrically encrypted credentials 109, where the symmetrically encrypted credentials 109 are decrypted with the asymmetrically decrypted (e.g. plaintext) symmetric key.

Note that encryption step 334 and decryption step 339 can be used with other plaintext values and ciphertext values as contemplated herein, such as (i) a device 106 asymmetrically encrypting in a step 334 a configuration attribute 330a for an external network 108 using a public key for the external network, and then (ii) the external network 107 asymmetrically decrypting in a step 339 the configuration attribute 330a using a private key. Other possibilities exist as well without departing from the scope of the present disclosure.

Figure 4a

Figure 4a is a flow chart illustrating exemplary steps for conducting a key exchange using PKI keys in order to derive a shared secret key, and for using the derived shared secret key to encrypt and decrypt data, in accordance with exemplary embodiments. Exemplary steps for a responder 102' and an initiator 101 to mutually derive keys can comprise (i) a key exchange 319 step for DPP server 103 or responder 102' (depending on the responder mode 306a) to derive a symmetric encryption key k1 314a and (ii) a key exchange 314 step for a initiator 101 to derive the same symmetric encryption key k1 314a. Exemplary steps for a initiator 101 to encrypt data and responder 102' or DPP server 103 (depending on a mode 306a) to decrypt data can comprise (a) an encryption step 315 for initiator 101 to utilize the symmetric encryption key k1 314a to convert plaintext to ciphertext, and (b) a decryption

320 step for responder 102' or DPP server 103 (depending on mode 306a) to utilize the symmetric encryption key k1 314a to convert the ciphertext received from initiator 101 into plaintext.

The use of the steps for a key exchange 314, a key exchange 319, encryption 315, and decryption 320 were also depicted and described in connection with Figure 3a and Figure 3b above.

Additional detail regarding the use of these steps will be provided herein. Further, although the steps are depicted specifically for the use of particular keys and plaintext/ciphertext combinations in Figure 4a, the steps illustrated can be used with different keys and plaintext/ciphertext combinations. For example, Figure 4b below depicts the same functionality used by the nodes as in Figure 4a, but with (i) different PKI keys input to mutually derive a different symmetric encryption key, (ii) a different key derivation function and (iii) different plaintext encrypted and different ciphertext decrypted.

A key exchange 314 step for initiator 101 to derive a symmetric encryption key 314a can utilize a selected set of cryptographic parameters 199a as depicted and described in connection with Figure 1d and Figure 1e above. As depicted in Figure 4a, a key exchange algorithm 401a can receive input both of a responder bootstrap public key Br 102a and an initiator ephemeral private key pi 101d. The key exchange algorithm 401a could comprise a Diffie Hellman key exchange (DH), an Elliptic Curve Diffie Hellman key exchange (ECDH), and other possibilities exist as well without departing from the scope of the present disclosure. A key exchange algorithm 401a can support either PKI keys based on elliptic curves or RSA algorithms, although support of elliptic curves may be preferred in some exemplary embodiments due to their shorter key lengths and lower processing requirements.

A summary of ECDH as a key exchange algorithm 401a is included in the Wikipedia article titled "Elliptic Curve Diffie-Hellman" from March 9, 2018, which is herein incorporated by reference. An exemplary embodiment of key exchange algorithm 401a could comprise a "One-Pass Diffie-Hellman, C(1, 1, ECC CDH)" algorithm as described in section 6.2.2.2 on page 81 of the National Institute of Standards and Technology (NIST) document "NIST SP 800-56A, Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography" from March, 2007 which is hereby incorporated by reference its entirety. Other key exchange algorithms in NIST SP 800-56A could be utilized as well for a key exchange algorithm 401a in Figure 4a, Figure 4b, and Figure 4c without departing from the scope of the present disclosure. Example calculations for an ECDH key exchange for a key exchange algorithm 401a are shown on pages 88 and 89 of DPP specification version 1.0, where a selected set of parameters 199a can use the ECC curve P-256 with PKI key lengths of 256 bits.

Other algorithms to derive a shared symmetric encryption key using public keys and private keys may also be utilized in a key exchange algorithm 401a (and 401b in Figure 4b and 401c in Figure 4c), such as, but not limited to, the American National Standards Institute (ANSI) standard X-9.63. Cryptographic parameters 199a can also include information for using a key exchange algorithm 401a and a key derivation function 402a in order to derive a commonly shared symmetric encryption key

k1 314a. As contemplated herein, the terms “selected set of cryptographic parameters 199a” and “cryptographic parameters 199a”, and “parameters 199a” can be equivalent. Parameters 199a input into a key exchange algorithm 401a can include a time-to-live for a key k1 314a that is derived, a supported point formats extension, where the supported point formats extension could comprise uncompressed, compressed prime, or “compressed char2” formats, as specified in ANSI X-9.62. In other words, an ECC keys input into a key exchange algorithm 401a may have several different formats and a set of parameters 199a can be useful to specify the format.

As depicted in Figure 4a, the output of a key exchange algorithm 401a, such as an ECDH key exchange, can comprise a secret shared key M 410. The secret shared key M 410 can be input into both (i) a first key derivation function 402a in order to calculate the key k1 314a and (ii) a second key derivation function 402c in order to calculate a symmetric encryption key ke 327a as depicted in Figure 4c below. Key derivation function 402a could use a hash algorithm 199e from parameters 199a, where an input into the hash algorithm 199e can include the value M 410, in addition to other data such as public keys. As contemplated herein, a secret shared key can also be referred to as a value, since the secret shared key can be a numeric value or a number in the form of a sequence of digits. In DPP specification version 1.0 a key derivation function to determine key k1 314a is depicted on the fourth line of page 48.

Note that in some exemplary embodiments, such a second responder mode 306a depicted and described in connection with Figure 5a below, (i) a key exchange algorithm 401a can be processed by a DPP server 103 and a key derivation function 402a can be processed by an responder 102'. In other words, for some embodiments the key exchange 314b step can be shared or distributed between a DPP server 103 and an responder 102'. Other possibilities exist as well for a key derivation function 402a without departing from the scope of the present disclosure. For example, with other embodiments besides those within DPP specification version 1.0, the output of a key derivation function 402a can be both a symmetric encryption key k1 314a and a message authentication code (MAC) key, where a MAC key can function to verify message integrity of ciphertexts generated with the symmetric encryption key 314a.

As depicted in Figure 4a, an encryption 315 step could be utilized with the derived symmetric encryption key k1 314a in order to convert plaintext 403a into ciphertext 315a. An encryption 315 step can utilize a symmetric encryption algorithm 405a, where symmetric encryption algorithm 405a could comprise an algorithm according to the Advanced Encryption Standard (AES), Blowfish, and Triple Data Encryption Standard (3DES), and other possibilities exist as well. As depicted in Figure 4a, a symmetric encryption algorithm 405a can comprise the AES-SIV algorithm as specified in IETF RFC 5297. Parameters 199f could specify settings for a symmetric ciphering algorithm 405a as well, such as key length and a block cipher modes of ECB, CBC, OFB, and CFB. Parameters 199f may

also specify an authenticated encryption mode such as CCM (NIST SP800-38C), GCM (NIST SP800-38D), and other examples exist as well.

Symmetric encryption algorithm 405a can accept input of (i) plaintext 403a, (ii) symmetric encryption key k1 314a, (iii) parameters 199f, and (iv) optionally an initialization vector (not shown in Fig. 4a and not required for AES-SIV), and output ciphertext 315a. Although not depicted in Figure 4a, and encryption 405a step can output a message authentication code (MAC), where the MAC can be calculated with the MAC key from a key derivation function 402a. In exemplary embodiments the use of a MAC and MAC key, as well as an initialization vector can be optionally omitted, such as with algorithm 405a. The exemplary listed values for plaintext and ciphertext in step 315 in Figure 4a were described in connection with depicted step 315 in Figure 3a.

Key exchange 319 step for a responder 102' or DPP server 103 (depending on a mode 306a) depicted in Figure 4a can correspond to key exchange 319 in Figure 3a. Key exchange 319 step can comprise inputting or using the initiator ephemeral Public key Pi 101c (from message 122) and the responder bootstrap private key br 102b into key exchange algorithm 401a, which can comprise the same or equivalent key exchange algorithm 401a depicted and described in connection with key exchange step 314 described above. Other elements or algorithms within a key exchange step 319 can be equivalent to a key exchange step 314 above, including the use of shared parameters 199a. In this manner, responder 102' or DPP server 103 (depending on a mode 306a) and initiator 101 can securely derive the same symmetric encryption key k1 314a as depicted in Figure 4a.

For embodiments with a first mode 306a depicted in Figure 3a, DPP server 103 can conduct key exchange step 319 and for embodiments with a second mode 306a depicted in Figure 5a, a responder 102' can conduct key derivation function 402a and DPP server 103 can conduct key exchange algorithm 401a. For embodiments where DPP server conducts a key exchange step, such as step 401a in Figure 4a, or other key exchange steps such as 401a in Figure 4b below, etc., the appropriate keys for input into the key exchange algorithm can be selected from a database (e.g. 103a for DPP server) or table (e.g. table 102u) using an identity for device 106 received in message 122. In exemplary embodiments, the identity of device 106 can comprise either MAC.I 106n or the hash value 101aa over the initiator bootstrap public key Bi 101a.

A decryption 320 step allows responder 102' or DPP server 103 (depending on mode 306a) to convert the ciphertext 315a received in a message 122 from Figure 3a into plaintext 403a. Decryption 320 step can utilize a symmetric decryption algorithm 405b, which could comprise the same algorithm used in symmetric encryption algorithm 405a except the algorithm being used for decryption instead of encryption. For example, if symmetric encryption algorithm 405a in Figure 4a comprises an AES-SIV encryption, then symmetric decryption algorithm 405b can comprise an AES-SIV decryption. Note that the same values are input into symmetric decryption algorithm 405b as symmetric encryption algorithm 405a, such as symmetric encryption key k1 314a and parameters 199f in order to convert

ciphertext 315a back into plaintext 403a. Responder 102' or DPP server 103 (depending on the mode 306a) can read and process plaintext 403a after a decryption 320 step. For example, with the first mode 306a depicted in Figure 3a and Figure 3b, DPP server 103 can perform decryption step 320 to read plaintext 403a. With the second mode 306a depicted in Figure 5, responder 102' can perform decryption step 320 to read plaintext 403a.

For embodiments where a MAC code is received with ciphertext 315a, initiator 101 can also verify the MAC code using a MAC key from a key derivation function 402a in order to verify the integrity of ciphertext 315a. Note that AES-SIV can have a message integrity check built into the algorithm, and thus a separate MAC code and MAC key can be optionally omitted in some exemplary embodiments. In exemplary embodiments, the successful decryption of a ciphertext into a plaintext using decryption algorithms 405b supports mutual authentication between the two nodes, since successful decryption can only take place when the two nodes record the depicted private keys used for the depicted key exchange.

Figure 4b

Figure 4b is a flow chart illustrating exemplary steps for conducting a key exchange using PKI keys in order to derive a shared secret key, and for using the derived shared secret key to encrypt and decrypt data, in accordance with exemplary embodiments. Exemplary steps for (i) an responder 102' or a DPP server 103 (depending on the mode 306a) and (ii) an initiator 101 to mutually derive keys can comprise (a) a key exchange 314b step for DPP server 103 or responder 102' (depending on the mode 306a) to derive a symmetric encryption key k2 314c and (b) a key exchange 319a step for an initiator 101 to derive the same symmetric encryption key k2 314c. Exemplary steps for an initiator 101 to decrypt data and responder 102' (or DPP server 103) to encrypt data can comprise (i) an encryption 315b step for responder 102' or DPP server 103 to utilize the symmetric encryption key k2 314c to convert plaintext to ciphertext, and (ii) a decryption 320b step for initiator 101 to utilize the symmetric encryption key k2 314c to convert the ciphertext received from responder 102' into plaintext. The use of the steps for a key exchange 314a, a key exchange 319a, encryption 315b, and decryption 320b were also depicted and described in connection with Figure 3a above.

Note that steps in Figure 4b and the steps in Figure 4a can share some algorithms and values, and the descriptions for the algorithms and values in Figure 4a can be applicable for Figure 4b. For example, a same key exchange algorithm 401a depicted and described in Figure 4a can also be used in Figure 4b. The key exchange algorithm 401a can comprise an ECDH key exchange. The set of parameters 199a depicted and described in Figure 4a can also be used in Figure 4b. Further, a symmetric encryption algorithm 405a and a symmetric decryption algorithm 405b can be the same in

Figure 4a and Figure 4b. A set of parameters 199f for a symmetric encryption algorithm 405a and a symmetric decryption algorithm 405b can be the same in Figure 4a and Figure 4b.

A responder 102' or DPP server 103 (depending on a mode 306a) can conduct a key exchange step 314b. In a key exchange step 314b, the initiator ephemeral public key P_i 101c received in message 122 from Figure 3a can be input into an ECDH key exchange algorithm 401a along with the responder ephemeral private key pr 102r derived in a step 318 in Figure 3a in order to output a secret shared key N 411. The value for shared secret key N 411 can be recorded and also used in the derivation of a key ke 327a in a key exchange step 321 below in Figure 4c. For a step 314b, the value for the shared secret key N 411 can be input into a key derivation function 402b in order to derive a mutually shared symmetric key k_2 314c. In exemplary embodiments which support DPP specification version 1.0, a key derivation function 402b can comprise the key derivation function for key k_2 in point 2 of section 6.2.3 of DPP specification version 1.0. For a first responder mode 306a depicted in Figure 3a, a DPP server 103 can conduct key exchange step 314b and for a second responder mode 306a depicted in Figure 5a, a responder 102' can conduct key exchange step 314b.

An initiator 101 can conduct a key exchange step 319a. At step 319a, a received, validated responder ephemeral public key Pr 102p and a derived initiator ephemeral private key pi 101d can be input into an ECDH key exchange algorithm 401a in order to calculate a shared secret key N 411. Shared secret key N 411 can also be recorded and used in a step 327 below. Initiator 101 can input shared secret key N 411 into a key derivation function 402b in order to mutually derived the symmetric encryption key k_2 314c. In exemplary embodiments which support DPP specification version 1.0, a key derivation function 402b can comprise the key derivation function for key k_2 in section 6.2.4 of DPP specification version 1.0.

Responder 102' or DPP server 103 (depending on a mode 306a) can conduct an encryption step 315b. The use for an encryption step 315b for a DPP server 103 in a first responder mode 306a is depicted and described in connection with Figure 3a above. The use for an encryption step 315b for responder 102' for a second responder mode 306a is depicted and described in connection with Figure 5a below. Ciphertext 324a can comprise a ciphertext from an encryption step 322 below in Figure 4c. In other words, a responder 102' or DPP server 103 can conduct the encryption step 322 below and also key exchange step 321 below in Figure 4c before conducting the encryption step 315b. As depicted in Figure 4b, the ciphertext 324a can also be encrypted again for an encryption step 315b. The symmetric ciphering key for encryption step 315b can comprise symmetric key k_2 314c from a key exchange step 314b in two paragraphs above. The output from an encryption step 315b can be ciphertext 315c, as depicted in Figure 4b. The plaintext 403b for an encryption step 315b can comprise the depicted values of ciphertext 324a, I-nonce 317a, R-Capabilities 318c, and R-nonce 318b.

A decryption 320b step can be performed by an initiator 101. A decryption 320b step converts the ciphertext 315c received in a message 123 from Figure 3a into plaintext 403b. Decryption 320b

step can utilize a symmetric decryption algorithm 405b, which could comprise the same algorithm used in symmetric encryption algorithm 405a except the algorithm being used for decryption instead of encryption. Note that the same values (other than ciphertext) are input into symmetric decryption algorithm 405b as symmetric encryption algorithm 405a, such as symmetric encryption key k2 314c and parameters 199f in order to convert ciphertext 315c back into plaintext 403b. Initiator 101 can read and process plaintext 403b after a decryption 320b step. In exemplary embodiments, the successful decryption of a ciphertext into a plaintext using decryption algorithms 405b supports mutual authentication between the two nodes, since successful decryption can only take place when the two nodes record the depicted private keys used for the depicted key exchange.

Figure 4c

Figure 4c is a flow chart illustrating exemplary steps for conducting a key exchange using PKI keys in order to derive a shared secret key, and for using the derived shared secret key to encrypt and decrypt data, in accordance with exemplary embodiments. Exemplary steps for (i) an responder 102' or a DPP server 103 (depending on the mode 306a) and (ii) an initiator 101 to mutually derive keys can comprise (a) a key exchange 321 step for DPP server 103 or responder 102' (depending on the mode 306a) to derive a symmetric encryption key ke 327a and (b) a key exchange 327 step for an initiator 101 to derive the same symmetric encryption key ke 327a. Note that in some exemplary embodiments, such a second responder mode 306a depicted and described in connection with Figure 5a below, (i) a key exchange algorithm 401c can be processed by a DPP server 103 a key derivation function 402c can be processed by an responder 102'. In other words, for some embodiments the key exchange 321 step can be shared or distributed between a DPP server 103 and an responder 102'.

Exemplary steps in Figure 4c for (i) a responder 102' or DPP server 103 to encrypt data and initiator 101 to decrypt data can comprise (i) an encryption 322 step for responder 102' or DPP server 103 to utilize the symmetric encryption key ke 327a to convert plaintext to ciphertext, and (ii) a decryption 328 step for initiator 101' to utilize the symmetric encryption key ke 327a to convert the ciphertext received from responder 102' into plaintext. The use of the steps for a key exchange 321, a key exchange 327, encryption 322, and decryption 328 were also depicted and described in connection with Figure 3a and Figure 3b above.

Note that steps in Figure 4c and the steps in Figure 4a can share some algorithms and values, and the descriptions for the algorithms and values in Figure 4a can be applicable for Figure 4c. For example, the key exchange algorithm 401c can comprise an ECDH key exchange. The set of parameters 199a depicted and described in Figure 4a can also be used in Figure 4c. Further, a symmetric encryption algorithm 405a and a symmetric decryption algorithm 405b can be the same in

Figure 4a and Figure 4c. A set of parameters 199f for a symmetric encryption algorithm 405a and a symmetric decryption algorithm 405b can be the same in Figure 4a and Figure 4c.

For a first responder mode 306a depicted in Figure 3a and Figure 3b, a DPP server 103 can conduct a key exchange step 321. For a second responder mode 306a depicted in Figure 5a, a key exchange step 321 can be shared between DPP server 103 and responder 102', such that DPP server 103 conducts a key exchange algorithm 401c and responder 102' conducts a key derivation function 402c. In a key exchange step 321, (i) the initiator ephemeral public key P_i 101c received in message 122 from Figure 3a can be input into an ECDH key exchange algorithm 401c along with (ii) the responder ephemeral private key pr 102r and the responder bootstrap private key br 102b.

An ECDH key exchange algorithm 401c can use $(pr\ 102r + br\ 102b)$ modulo q , as specified in DPP specification version 1.0 in point 2 of section 6.2.3, so algorithm 401c for initiator 101 can be slightly different than algorithm 401a for initiator 101 above in Figure 4a and Figure 4b. The output of key exchange algorithm 401c can be a shared secret key L 412. The value L 412 can be input into a key derivation function 402c in order to calculate a shared secret key ke 327a. Note that for a key derivation function 402c, the input can include all of key M 410 from a step 319, key N from a step 314b, and the derived key L 412. As contemplated herein, a key derivation function 402a, 402b, and 402c can also use other data mutually shared between (i) initiator 101 and (ii) responder 102' or DPP server 103 (depending on a mode 306a). In exemplary embodiments which support DPP specification version 1.0, a key derivation function 402c can comprise the key derivation function for key ke 327a in point 2 of section 6.2.3 of DPP specification version 1.0.

An initiator 101 can conduct a key exchange step 327. At step 327, (i) a combination of a recorded responder bootstrap public key Br 102a and received responder ephemeral public key Pr 102p, and (ii) the recorded initiator bootstrap private key bi 101b can be input into an ECDH key exchange algorithm 401c in order to calculate a shared secret key L 412. The recorded responder bootstrap public key Br 102a and received responder ephemeral public key Pr 102p can be combined via elliptic curve point addition. Shared secret key L 412 can be input into a key derivation function 402c, along with shared secret key M 410 from a step 314 and shared secret key N 411 from a step 319a in order to mutually derive the symmetric encryption key ke 327a. In exemplary embodiments which support DPP specification version 1.0, a key derivation function 402c can comprise the key derivation function for key ke 327e in section 6.2.4 of DPP specification version 1.0.

Responder 102' or DPP server 103 (depending on a mode 306a) can conduct an encryption step 322, where the use for an encryption step 322 are depicted and described in connection with Figure 3a above and also Figure 4b above. Plaintext 403c in a step 322 can comprise the responder authentication value 404, where value R -auth 404 can comprise a hash 199e that includes at least the initiator nonce 317a, the second random number R -nonce 318b from a step 318 above in Figure 3a, and public PKI keys such as responder bootstrap public key 102a and responder ephemeral public key

102c. The value for R-auth 404 in a step 322 can comprise the value in point 3 of section 6.2.3 of DPP specification version 1.0. The symmetric ciphering key for encryption step 322 can comprise symmetric key ke 327a from a key exchange step 321.

5 The output from an encryption step 322 can be ciphertext 324a, as depicted in Figure 4c. Note that ciphertext 324a in a step 322 can also be encrypted again in a step 315b above in Figure 4b. In other words, a responder 102' or DPP server 103 (depending on the mode 306a) can conduct the encryption step 322 herein for Figure 4c (along with the key exchange 321 herein in Figure 4b) before conducting the encryption step 315b. For a first mode 306a representing an embodiment depicted in Figure 3a, DPP server 103 can conduct encryption step 322. For a second mode 306a representing an
10 embodiment depicted in Figure 5a, a responder 102' can conduct encryption step 322.

A decryption step 328 can be performed by initiator 101. A decryption 328 step converts the ciphertext 324a received in a message 123 from Figure 3a into plaintext 403c. Decryption 328 step can utilize a symmetric decryption algorithm 405b, which could comprise the same algorithm used in symmetric encryption algorithm 405a except the algorithm being used for decryption instead of
15 encryption. Note that the same values are input into symmetric decryption algorithm 405b as symmetric encryption algorithm 405a in Figure 4c, such as symmetric encryption key ke 327a and parameters 199f in order to convert ciphertext 324a back into plaintext 403c. Initiator 101 or device 106 can then read and process plaintext 403c after a decryption 328 step. In exemplary embodiments, the successful decryption of a ciphertext into a plaintext using decryption algorithms 405b supports
20 mutual authentication between the two nodes, since successful decryption can only take place when the two nodes record the depicted private keys used for the depicted key exchange. In other words, only the nodes could derive key ke 327a in Figure 4c, and thus data that is successfully encrypted by one node and decrypted by the other node using key ke 327a would confirm the nodes are authenticated, since only the nodes recording the private keys depicted in Figure 4c would be able to
25 derive keys ke 327a.

As depicted and described in connection with Figure 3a, a responder 102' or DPP server 103 can also conduct both an encryption step 322 and a decryption step 328. The steps for DPP server 103 to conduct a decryption step 328 for a first responder mode 306a can comprise step 328b and step 328c as depicted and described in Figure 3a. When responder 102' or DPP server 103 conducts
30 decryption step 328 using symmetric encryption key ke 327a, the ciphertext and plaintext will comprise different values than those depicted in Figure 4c. For example, in Figure 3a a DPP server 103 with a first responder mode 306a can use a decryption step 328b (e.g. step 328 but with different ciphertext than that depicted in Figure 4c) in order to decrypt the encrypted initiator authentication value 404a into a plaintext authentication value 404a. Also, in Figure 5a, a responder 102' for a second
35 responder mode 306a can use a decryption step 328b (e.g. step 328 but with different ciphertext 324e than that depicted in Figure 4c) in order to decrypt the encrypted initiator authentication value 404a.

Further, an initiator 101 can use a decryption step 328d to decrypt network credentials 109 from a responder 102'.

Note that in some exemplary embodiments, the "plaintext" network credentials 109' (for the purposes of a step 328d in Figure 5a) can comprise network credentials 109' that can also be encrypted with an asymmetric ciphering algorithm such as Elgamal using a step 351a in Figure 3e above. In other words, network credentials 109' can comprise credentials that are "double encrypted" with both (i) a symmetric key 327a (which can be decrypted by initiator 101 using a decryption step 328d, where decryption step 328d uses the depicted decryption 328 in Figure 4c), and then (ii) an asymmetric ciphering algorithm 351b such as Elgamal from a step 339. In other words, the output of a decryption step 328d could be input into an asymmetric decryption algorithm 351b, so that the input into asymmetric decryption algorithm 351b could be decrypted by initiator 101 using either (i) initiator ephemeral private key pi 101d or (ii) initiator bootstrap private key bi 102b.

Figure 4d

Figure 4d is a flow chart illustrating exemplary steps for conducting key exchanges using PKI keys in order to derive a first and a second shared secret keys, where ECC point addition is performed on output from a key exchange algorithm in order to derive a third shared secret key. The exemplary key exchange step 421 can be used by a DPP server 103 in order to derive a key L(a) 421a. The exemplary key exchange step 422 can be used by a responder 102' in order to derive a key L(b) 422a. A key derivation step 423 can combine key L(a) 421a and key L(b) 422b in order to calculate or derive the key L 412, using ECC point addition on key L(a) 421a and key L(b) 422b. In this manner, key L 412 can be calculated through the use of two different key exchange steps 421 and 422 in some exemplary embodiments such as those depicted and described in connection with Figure 5a below, instead of the single key exchange step 321 depicted in Figure 4c.

Key exchange 421 can use an ECDH key exchange algorithm 401a, where the input comprises the initiator bootstrap public key Bi 101a and the responder bootstrap private key br 102b. The set of mutually agreed parameters 199a between with an initiator 101 can be used with a key exchange step 421. The output of ECDH key exchange algorithm 401a can comprise key L(a) 421a. In exemplary embodiments, where DPP server 103 conducts the key exchange step 421, the responder bootstrap private key br 102b can remain in DPP server 103 or within a network 107 and does not need to be shared with responder 102'. In addition, although a DPP server 103 is described herein for a Figure 4d as performing the key exchange step 421, another server on an external network 108 could (i) record the responder bootstrap private key br 102b and (ii) perform a key exchange step 421, and the output of key exchange step 421 in the form of key L(a) 421a could be sent to a DPP server 103 or

responder 102', such as in a message 503 depicted in Figure 5a below. Or, DPP server 103 could be operated by an external network 108 and communicate with a responder 102' via a secure session 302b. Key L(a) 421a could comprise a shared secret key, although in exemplary embodiments an initiator 101 may not need to calculate the corresponding key L(a) 421a directly, but rather initiator 101 can calculate a shared secret key L 412 using step 327 as depicted in Figure 4c.

Key exchange 422 can use an ECDH key exchange algorithm 401a, where the input comprises the initiator bootstrap public key Bi 101a and the responder ephemeral private key pr 102r. The set of mutually agreed parameters 199a between with an initiator 101 can be used with a key exchange step 422. The output of ECDH key exchange algorithm 401a can comprise key L(b) 422a. In exemplary embodiments, where responder 102' conducts the key exchange step 422, the responder does need access to the responder bootstrap private key br 102b, which can remain in DPP server 103 or within a network 107 and does not need to be shared with responder 102'. Key L(b) 422a could comprise a shared secret key, although in exemplary embodiments an initiator 101 may not need to calculate the corresponding key L(b) 421a directly, but rather initiator 101 can calculate a shared secret key L 412 using a step 327 as depicted in Figure 4c.

A key derivation step 423 can (i) combine the output of key exchange steps 421 and 422 in order to calculate or derived the shared secret key L 412 and then (ii) perform a key derivation function step 402c on the derived or calculated shared secret key L 412 in order to determine or calculate shared secret key ke 327a, which can be also mutually derived by initiator 101, where initiator 101 uses the key exchange step 327 depicted and described in connection with Figure 4c. In exemplary embodiments, a responder 102' can conduct the key derivation step 423 using (i) the value L(a) 421a received from a DPP server 103 (where receipt of L(a) 421a by responder 102' can be in a message 503 as shown in Figure 5a below), and (ii) the value or key L(b) 422b output from a key exchange step 422 in the paragraph above.

Key derivation step 423 can comprise two primary steps. A first step in key derivation 423 can comprise an ECC point addition 424 on the value L(a) 421a and the value L(b) 422a. The result of the ECC point addition will be equal to the value L 412, where the value L 412 could also be calculated without ECC point addition 424 using a step 321. Exemplary calculations for an ECC point addition 424 can comprise the calculations shown for point addition in the Wikipedia article for "Elliptic Curve Point Multiplication" dated May 15, 2018, which is herein incorporated by reference in its entirety. As depicted in Figure 4c, the calculation of L 412 using an ECC point addition 424 over L(a) 421a and L(b) 422a will equal the value for L 412 calculated by a key exchange algorithm 401c from a step 321 in Figure 4c. A second step in key derivation step 423 as depicted in Figure 4d can comprise a key derivation function step 402c using input from ECC point addition step 424 (e.g. value L 412 output from step 424), where the output of key derivation function step 402c can comprise key ke 327a, which is also depicted in Figure 4c above.

By conducting a key derivation step 423 as depicted in Figure 4c (where DPP server 103 conducts the calculations using the responder bootstrap private key br 102b), (i) responder 102' can calculate key ke 327a without recording or operating on the responder bootstrap private key br 102b, and (ii) the responder bootstrap public key Br 102a could comprise a shared key 102z as depicted in Figure 2a across multiple different devices 106. Although the responder 102' can receive the key or value L(a) 421a, the responder bootstrap private key br 102b remains secure, because responder 102' cannot feasibly calculate or determine the responder bootstrap private key br 102b using a received key L(a) 421a. In other words, the use of a key derivation step 423 can allow a responder bootstrap public key Br 102a to be securely shared between multiple devices 106 in order to conduct a DPP 191, since a responder 102' can conduct a DPP 191 as contemplated herein without recording, receiving, or operating on a responder bootstrap private key 102b.

Exemplary data and numbers can be provided to demonstrate the calculations for key exchange steps 421, key exchange step 422, and key derivation step 423 using an ECC point addition 424. The exemplary data can comprise decimal numbers for the example data and calculations listed in "Test vectors for DPP Authentication using P-256 for mutual authentication" on pages 88 and 89 of the DPP specification version 1.0. Parameters 199a can comprise the elliptic curve 199c of "secp256r1" with key lengths 199b of 256 bit long keys.

The responder bootstrap private key 102b can comprise the exemplary following number, and may be recorded in a DPP server 103:

38358416135251014160802731750427376395128366423455574545250035236739593908128

The responder ephemeral private key 102r can comprise the exemplary following number, and may be recorded by a responder 102':

111991471310604289774359152687306247761778388605764559848869154712980108827301

The initiator bootstrap public key Bi 101a can comprise the following exemplary values with X and Y numbers (or "coordinates") of:

X:

61831688504923817367484272103056848457721601106987911548515219119661140991966

Y:

436821274116052626307636850969789027573720854595612820926922498255090826944

Key exchange step 421 for an ECDH algorithm key exchange 401a can input the initiator bootstrap public key Bi 101a and the responder bootstrap private key br 102b (both with numbers above) in order to calculate a key L(a) 421a. An exemplary number or value for key L(a) 421a from the values above using parameters 199a can be:

X:

11490047198680522515311590962599671482029417064351337303313906642805743573119

Y:

27933966560238204731245097943399084523809481833434754409723604970366082021855

Key exchange step 422 for an ECDH algorithm key exchange 401a can input the initiator bootstrap public key Bi 101a and the responder ephemeral private key pr 102r (both with numbers above) in order to calculate a key L(b) 422a. An exemplary number or value for key L(b) 422a from the values above using parameters 199a can be:

X:

78944719651774206698250588701582570633503182903415394243006529481189158194650

Y:

11227712702924684581834935828837489140201820424536062912051086382324589445237

An ECC point addition 424 for the above two derived points L(a) 421a (from keys Bi 101a and br 102b) and L(b) 422b (from keys Bi 101a and pr 102r) will result in the following point that also equals L 412.

X:

113734500629065545557893524064610113740858966831672649615565042035695230713090

Y:

68961429691307429166796760881095689348088875771334970644593306388375741965262

Note that key L 412 can also be derived by initiator 101 from a key exchange step 327 using ECDH key exchange algorithm 401c. ECDH key exchange algorithm 401c can comprise initiator 101 calculating L from keys Br 102b, Pr 102p, and bi 101b, represented by $(Br\ 102b + Pr\ 102p) * bi\ 101b$. Using hexadecimal representations of the above keys Br 102b, Pr 102p, and bi 101b from the same keys values in DPP specification version 1.0 and exemplary numbers above, the X,Y numbers or point or "coordinates" derived by initiator 101 with algorithm 401c for shared secret L on page 89 equals:

X: fb737234 c973cc3a 36e64e51 70a32f12 089d198c 73c2fd85 a53d0b28 2530fd02

Y: 9876c937 b642cce8 f604bef0 c24ce0da d2c27867 08a7575a f940a58c fe8ca3ce

The hexadecimal representation of shared secret L using keys Br 102b, Pr 102p, and bi 101b by initiator 101 equals the decimal representation of shared secret key L 412 from the ECC point addition 424 by a responder 102', where ECC point addition 424 used as input key L(a) 421a and key L(b) 422a.

After an ECC point addition 424, a key derivation step 423 in Figure 4c can input the shared secret key L 412 output from the ECC point addition 424 into a key derivation function 402c. The key derivation function 402c can comprise the same key derivation function 402c as depicted and described in connection with Figure 4c above. The key derivation function 402c can also use input of shared secret key M 410 from Figure 4a in a step 319 and shared secret key N 411 from a step 314b. The output of the key derivation function 402c can be shared secret key ke 327a. An exemplary key derivation function 402c could comprise the key derivation function in point 2 of page 48 of DPP specification version 1.0. Note that initiator 101 can mutually derive the same value or number or "coordinate" for shared secret key ke 327e using a key exchange step 321. As contemplated herein, the use of the words or description "shared secret key" can also mean "shared secret". Thus, "shared secret key M 410" can be the same as "shared secret M 410", "shared secret key N 411" can be the same as "shared secret N 411", etc.

Figure 5a

Figure 5a is a simplified message flow diagram illustrating an exemplary system with exemplary data sent and received by a device database, a DPP server, a responder proxy, and an initiator, in accordance with exemplary embodiments. System 500a can include a device database 104, DPP server 103, responder proxy 102', and initiator 101 operating in device 106. Before starting steps and message flows depicted in Figure 5a, the initiator, DPP server, and device database depicted for system 500a in Figure 5a may previously complete exemplary message flows and steps depicted for a step 301 in Figure 3a above. Responder 102' can use data received from initiator 101 in a message 122 from step 301 (such as initiator ephemeral public key Pi 101c) in order to conduct the steps and process the messages depicted for a responder 102' in Figure 5a.

Responder proxy 102' can communicate with DPP server 103 via IP network 113, also as depicted in Figure 3a above. As contemplated herein, a "responder proxy 102'" depicted in Figure 5a may also be referred to as responder 102', and a responder 102' and responder proxy 102' may be equivalent. A difference between system 500a in Figure 5a and system 300 in Figure 3a can be that (i) responder ephemeral private key pr 102r, with corresponding (ii) responder ephemeral public key

Pr 102p are derived and recorded by a responder 102' instead of a DPP server 103. In exemplary embodiments depicted in Figure 5a the responder bootstrap private key br 102b can remain recorded in a DPP server 103 and not exposed to responder 102'. In exemplary embodiments depicted in Figure 5a the responder ephemeral private key pr 102r can remain recorded in responder 102' and not exposed to DPP server 103. The exemplary embodiment depicted in Figure 5a can comprise a second mode 306a of operation for responder 102' and DPP server 103, as depicted and described in connection with Figure 3c above.

At step 502, DPP server 103 can select a mode 306a for both responder 102' and DPP server 103. Note that the mode 306a selected is not normally observable to an initiator 101, since the full and complete set of DPP 191 messages between initiator 101 and responder 102' can be transmitted and received between the two nodes. In other words, (a) the different distribution of PKI and shared secret keys in a server database 103a and responder 102' for a first mode 306a in Figure 3a and a second mode 306a in Figure 5a (b) would not normally be observable to an initiator 101. The recording of PKI and shared secret keys in a server database is depicted in a server database 103a in Figure 3c, where the value "--" can mean that the key is recorded and/or operated by responder 102'. Step 502 can also comprise DPP server 103 selecting a responder bootstrap public key Br 102a and a responder bootstrap private key br 102b and initiator bootstrap public key Bi 101a for device 106 using MAC.I 106n, although a different value in server database 103a could be used as well, such as device identity ID.device 106i or hash value 101aa. Selecting keys Br 102a and br 102b in a step 502 can use the parameters 199a received in a message 312a in Figure 3a, in order to select keys that are compatible with received initiator ephemeral public key Pi 101c from a message 122 in Figure 3a. Also, although a step 502 is depicted in Figure 5a, a step 502 could be used in other Figures and embodiments herein, include embodiments depicted in Figure 3a and Figure 5b.

DPP server 103 can then conduct an ECDH key exchange step 401a using as input (i) the responder bootstrap private key br 102b and (ii) the initiator ephemeral public key Pi 101c received in a message 122 from Figure 3a during a step 301. This ECDH key exchange step 401a is depicted and described in connection with a key exchange step 319 depicted and described in connection with Figure 4a. The output of step 401a can be shared secret M 410. DPP server 103 can send (i) shared secret M 410 along with a device identity such as MAC.I 106n (or ID.device 106i or hash value 101aa) to (ii) responder 102' in a message 503. The use of a device identity in message 503 can be helpful for embodiments where responder 102' communicates with a plurality of devices 106 either simultaneously or in sequence.

Responder 102' can receive message 503 with shared secret M 410 and conduct a series of steps in order to prepare a DPP authentication response message 123. A step 318 in Figure 5a can comprise the same steps 318 as performed by DPP server 103 in Figure 3a, except responder 102' performs the actions in Figure 5a instead of DPP server 103 performing the actions. In summary for

a responder 102' in Figure 5a, responder 102' can derive two random numbers using a random number generator. The first random number can be used as a responder ephemeral private key pr_{102r} and the second random number can be used as a responder nonce 318b. Parameters 199a can specify the format and length of key pr_{102p} and nonce 318b in a step 318 for responder 102. Responder 102' can also derive a responder ephemeral public key Pr_{102p} using the derived key pr_{102r} . DPP server 103 can also determine responder capabilities 319b, which could comprise the capabilities for a configurator.

Responder 102' can process a key derivation function 402a using the received shared secret M 410 from a message 503. The key derivation function 402a can be equivalent to a function 402a depicted and described in connection with Figure 4a. Shared secret M 410 can be input into key derivation function 402a, and the output can be shared secret key $k1_{314a}$. Responder 102' can then perform a decryption step 320 using the derived key $k1_{314a}$, where a decryption step 320 is depicted and described in connection with Figure 4a. Decryption step 320 can be performed by responder 102' on the ciphertext 315a received in a DPP authentication request message 122. After a decryption step 320, responder 102' can read plaintext 403a with the values of the initiator nonce 317a and the initiator capabilities. For the embodiment depicted in Figure 5a, the initiator 101 can have the capabilities of an enrollee, although other possibilities exist as well without departing from the scope of the present disclosure.

Responder 102' can then conduct a second responder key exchange step 314b, where the details for a key exchange step 314b are depicted and described in connection with Figure 4b above. In summary, the initiator ephemeral public key Pi_{101c} received in message 122 can be input into an ECDH key exchange algorithm 401a along with the responder ephemeral private key pr_{102r} derived in a step 318 above in order to output a secret shared key N 411. The value for shared secret key N 411 can be recorded and used in the derivation of a key ke_{327a} in a key derivation function 402c below by responder 102'. For a step 314b, the value for the shared secret key N 411 can be input into a key derivation function 402b in order to derive a mutually shared symmetric key $k2_{314c}$. As depicted in Figure 5a, the collection of steps and messages beginning with a step 502 by DPP server 103 through key exchange step 314b can comprise a step 501, where a step 501 can also be used by a DPP server 103 and responder 102' in Figure 5b below.

In Figure 5a, a DPP server 103 can then conduct or calculate an ECDH key exchange step 421, where key exchange step 421 was depicted and described in connection with Figure 4d above. In summary, the initiator bootstrap public key Bi_{101a} and the responder bootstrap private key br_{102b} can be input into an ECDH key exchange algorithm 401d using a set of parameters 199a. The output of key exchange step 421 can comprise a value $L(a)_{421a}$. Note that step 421 can be performed before a step 501 or concurrently with a step 501. DPP server 103 can then send responder 102' a message 504, where message 504 include an identity for device 106 such as $MAC.I_{106n}$, the initiator bootstrap

public key Bi 101a, and the calculated value L(a) 421a from a key exchange step 421. Message 504 can be sent via secure session 302b established during a step 301 from Figure 3a.

Responder 102' can receive message 504 and begin performing a series of steps in order to process the data and prepare a DPP authentication response message 123. Responder 102' can conduct a key exchange step 422 in order to calculate a value L(b) 422a. In summary, the responder ephemeral private key pr 102r can be input into an ECDH key exchange algorithm 401a along with the initiator bootstrap public key Bi 101a, using a set of parameters 199a in order to calculate the value or key L(b) 422a using a key exchange step 422. Key Bi 101a can be received by responder 102' in message 504. A key exchange step 422 for responder 102' was depicted and described in connection with Figure 4d above. Responder 102' can then conduct a key derivation step 423 from Figure 4d above in order to determine or derived a secret shared key ke 327a.

A key derivation step 423 for responder 102' was depicted and described in connection with Figure 4d above. In summary, responder 102' can perform an elliptic curve point addition step 424 for the two values or points of L(a) 421a and L(b) 422b. As contemplated herein, the output of an ECDH key exchange function such as 401a or 401c can comprise a point value comprising an X coordinate or value and a Y coordinate or value, and when the output is described herein as a key or a value, the key or value can comprise either (i) the X value only for use in a key derivation function such as 402a, 402b, or 402c, or (ii) the X and Y values together for other calculations such as an ECC point addition step 424. The combination of the points L(a) 421a and L(b) for an ECC point addition step 424 can comprise the point or value L 412, as depicted and described in connection with a key derivation step 423 in Figure 4d. Key derivation step 423 can then input the value L 412, M 410 from message 503, and N 411 from step 314b into key derivation function 402c in order output symmetric ciphering key ke 327a.

Responder 102' can then perform an encryption step 322, where encryption step 322 was depicted and described in connection with Figure 4c above. Step 322 can also include responder 102' creating a value R-auth 404, where R-auth 404 can comprise a hash of the initiator nonce 317a from plaintext 403a in step 320, the second random number R-nonce 318b from a step 318 above, and public PKI keys. The value for R-auth 404 in a step 322 can comprise the value in point 3 of section 6.2.3 of DPP specification version 1.0. An encryption step 322 can comprise responder 102' (a) encrypting the value for R-auth 404 using the mutually derived key ke 327a from the key derivation step 423 above, where (b) the output of encryption step 322 can comprise ciphertext 324a.

Responder 102' can then conduct an encryption step 315b, where the details for an encryption step 315b are depicted and described in connection with Figure 4b above. Ciphertext 324a from an encryption step 322 in the paragraph above can also be encrypted again for an encryption step 315b. The symmetric ciphering key for encryption step 315b can comprise symmetric key k2 314c from a

key exchange step 314b. The output from an encryption step 315b can be ciphertext 315c, as depicted in Figure 4b above.

Responder 102' can conduct a step 324 before sending a message 123. For a step 324, responder 102' can record the derived responder ephemeral public key Pr 102p and responder ephemeral private key pr 102r and initiator bootstrap public key Bi 101a in an initiator table 102u from Figure 1f along with the identity of device 106 comprising potentially MAC.I 106n. In other words, an initiator table 102u can be updated to include information or data (i) received from network 107 (and DPP server 103) regarding devices 106 communication through network 107 via responder 102', and (ii) derived by responder 102' when performing key exchange steps 401a and 401c and key derivation steps 402a, 402b, and 402c. In other words, an initiator table 102u can include columns for the values of M 410, N 411, L 412, L(a) 421a, L(b) 422a, key k1 314a, key k2 314c, and key ke 327a for each device 106 with an identity such as MAC.I 106n. The recording of values for a device 106 or a plurality of devices 106 in an initiator table 102 can be equivalent to the values recorded for a server database 103a depicted and described in connection with Figure 3c above. In this manner, responder 102' can track keys and values for devices 106 since responder 102' may communicate with a plurality of different devices 106 and initiators 101 over time. In exemplary embodiments, responder 102' derives a new responder ephemeral PKI key pair for each unique DPP authentication request message 122 received. Data or values stored by responder 102' pertaining to derived encryption keys (e.g. k1, k2, and ke) and ephemeral PKI keys (e.g. Pr 102p and pr 102r) are preferably erased from an initiator table 102u upon successful completion of a configuration step 106a by device 106.

Responder 102' can then send initiator 101 a DPP authentication response message 123. In exemplary embodiments, message 123 is sent as a unicast message from (i) the responder MAC address MAC.R 102n (which received message 122) to (ii) the initiator MAC address MAC.I 106n (which sent message 122), over (iii) WiFi link 192. As depicted in Figure 5a, message 123 can include hash values of the bootstrap public keys H(Br 102a), H(Bi 101a), the responder ephemeral public key Pr 102p, and the ciphertext 315c. The ciphertext 315c can have at least encrypted values for (i) R-nonce 318b, (ii) I-nonce 317a, (iii) R-capabilities 318c, and (iv) ciphertext 324a. The ciphertext 315c can be encrypted with mutually derived symmetric encryption key k2 314c. Ciphertext 324a can be encrypted with the mutually derived symmetric key ke 327a and can include a plaintext value for R-auth 404. As described above, R-auth 404 can comprise a hash value over at least an initiator nonce 317a received and the responder nonce 318b.

Initiator 101 can receive DPP authentication response message 123 from responder 102' and conduct a series of steps in order to process the message and reply to the responder 102'. Initiator 101 can conduct a step 331, and a step 331 can comprise the series of steps 325, 319a, 320b, 327, 328, and 322b. The series of steps 331 for an initiator 101 to process a DPP authentication response 123 were depicted and described in connection with Figure 3a above. After completing the series of steps for a

step 331 in Figure 5a, initiator 101 can send responder 102' a DPP authentication confirm message 123a. The data and steps to determine and calculate an DPP authentication confirm message 123a for a step 331 can be compatible with section 6.2.4 of DPP specification version 1.0. Note that although the DPP authentication confirm message 123a can be equivalent to the DPP authentication confirm message from section 6.2.4 of DPP specification version 1.0, an important difference can be the operation of responder 102' in Figure 5a versus a responder as contemplated by DPP specification version 1.0. In the present disclosure as depicted in Figure 5a, responder 102' does not record the responder bootstrap private key br 102b.

Responder 102' can listen for either (i) a broadcast message with the hash 199e values for the bootstrap public keys 102aa and 101aa, or (ii) a unicast message to MAC.R 102n from MAC.I 106n for initiator 101 operating in device 106. Responder 102' can receive the DPP authentication confirm message 123a and process the message. DPP authentication confirm message 123a can include an encrypted value for initiator authentication value 404a as a ciphertext 324b, where ciphertext 324b is encrypted with derived key ke 327a in a step 322b within step 331. Upon receipt of DPP authentication confirm message 123a from device 106, responder 102' can first conduct a decryption step 328b, where decryption step 328b can use the same steps as a decryption step 328 as depicted in Figure 4c above, with a difference being decryption step 328b operates on the ciphertext 324b containing the encrypted initiator authentication value I-auth 404a instead of the R-auth 404 value. After a step 328b, responder 102' can read the plaintext initiator authentication value I-auth 404a.

In step 330, responder 102' can compare the decrypted value for I-auth 404a with a calculated value for I-auth 404a and if the two values match then the initiator 101 can be considered by responder 102' to be authenticated. The calculation of I-auth 404a for responder 102' in a step 330 can comprise a calculation of a secure hash value over the responder nonce 318b and bootstrap and ephemeral public keys, with an exemplary calculation for I-auth 404a shown in the sixth paragraph of page 50 of DPP specification version 1.0. Other calculations of an initiator authentication value over the responder nonce 318b are possible as well, without departing from the scope of the present disclosure.

After authentication step 330 is completed, in a step 505, responder 102' can determine the source of network credentials 109 or encrypted network credentials 109' for initiator 101. In a step 505, responder 102' can use the selected responder mode 306a as depicted and described in connection with Figure 3c. For embodiments where a network 107, an external network 108, or DPP server 103 can be the source of network credentials 109 or encrypted network credentials 109' (as specified in a responder mode 306a), then responder 102' can use the series of steps and messages in Figure 6 in order to (i) receive network credentials 109 or encrypted network credentials 109' from a DPP server 103 and (ii) send the credentials to initiator 101. For other embodiments where a responder 102' already records the network credentials 109 for initiator 101 and device 106 (such as responder 102' uses network credentials 109 to connect with network access point 105), then responder 102' can read

the locally stored credentials 109 and send the credentials 109 to initiator 101 in a DPP configuration response message 124 in Figure 6 below. Responder 102' can use a step 349 as depicted in Figure 3b in order to send locally stored credentials 109 to initiator 101, where responder 102' performs the actions instead of a DPP server 103. As depicted in Figure 5a, the collection of steps and messages beginning with responder 102' sending message 123 through step 505 can comprise a step 506. Step 506 can subsequently be used by a responder 102' for a different responder mode 306a depicted in Figure 5b below.

Figure 5b

Figure 5b is a simplified message flow diagram illustrating an exemplary system with exemplary data sent and received by a device database, a DPP server, a responder proxy, and an initiator, in accordance with exemplary embodiments. System 500b can include a device database 104, DPP server 103, responder proxy 102', and initiator 101 operating in device 106. Before starting steps and message flows depicted in Figure 5b, the initiator, DPP server, and device database depicted for system 500b in Figure 5b may previously complete exemplary message flows and steps depicted for a step 301 in Figure 3a above. Responder proxy 102' can communicate with DPP server 103 via IP network 113, also as depicted in Figure 3a above.

A difference between system 500b in Figure 5b and system 500a in Figure 5a can be that (i) responder ephemeral PKI keys pr 102r and Pr 102p are derived and recorded by a responder 102' (instead of a DPP server 103 in Figure 3a), (ii) responder 102' sends responder ephemeral private key pr 102r to DPP server 103, and (iii) DPP server 103 derives key L 412 instead of responder 102' in Figure 5a. In exemplary embodiments depicted in Figure 5b the responder bootstrap private key br 102b can remain recorded in a DPP server 103 and not exposed to responder 102'. The exemplary embodiment depicted in Figure 5b can comprise a third mode 306a of operation for responder 102' and DPP server 103, as depicted and described in connection with Figure 3c above.

As depicted in Figure 5b, responder 102' and DPP server 103 can conduct a step 501, where a step 501 is depicted and described in connection with Figure 5a above. Responder 102' can use data received from initiator 101 in a message 122 from step 301 in order to conduct step 501. In summary, a step 501 in Figure 5b can comprise, (i) a step 502 for DPP server 103 to select a mode 306a for both responder 102' and DPP server 103, (ii) DPP server 103 conducting an ECDH key exchange to calculate the value or key M 410, (iii) responder 102' receiving the key M 410 from DPP server 103, (iv) responder 102' deriving the responder ephemeral PKI keys of pr 102r and Pr 102p, (v) responder 102' performing a key derivation step 402a with M 410 in order to calculate key k1 314a, (vi)

responder 102' decrypting ciphertext in message 122 using key k1 314a, and (vii) responder 102' conducting a key exchange step 320 to calculate key k2 314c.

Note that the selected responder mode 306a in a step 501 is not normally observable to an initiator 101, since the full and complete set of DPP 191 messages between initiator 101 and responder 102' can be transmitted and received between the two nodes in Figure 5b. In other words, the different distribution of PKI and shared secret keys in a server database 103a and responder 102' for the responder mode 306a used for Figure 5b would not normally be observable to an initiator 101. The recording of PKI and shared secret keys in a server database is depicted in a server database 103a in Figure 3c, where the value "--" can mean that the key is recorded and/or operated by responder 102'.

Upon completion of a step 501 in Figure 5b, responder 102' can send DPP server 103 a message 507 via secure session 302b established during step 301 from Figure 3a. Message 507 can include (i) an identity for device 106 and initiator 101, such as MAC.I 106n or hash value 101aa, and (ii) the derived responder ephemeral private key pr 102r. DPP server 103 can receive message 507 and use the received device identity to select the initiator bootstrap public key Bi 101a and responder bootstrap private key br 102b from a server database 103a. DPP server can use the received responder ephemeral private key pr 102r from message 507 and the selected data in order to conduct an ECDH key exchange step 401c as depicted and described in connection with Figure 4c above. The output of a step 401c can be key L 412.

After deriving key L 412, DPP server 103 can then send responder 102' a message 508, where message 508 can include an identity for device 106 such as MAC.I 106n (or hash value 101aa), the initiator bootstrap public key Bi 101a (from a server database 103a), and the derived key L 412 from a step 402c. Responder 102' can receive message 508 and record and process the data. Responder 102' can conduct a key derivation function 402c using (i) the received key L 412, (ii) the received key M 310 from a message 503 in a step 501, and (ii) the key N 311 derived by responder 102 in a step 314b in step 501. The output of a key derivation function 402c can be key ke 327a, as depicted and described for a key exchange step 402c in Figure 4c.

Responder 102' can then calculate a responder authentication value 404 and conduct an encryption step 322 using the plaintext 403c as depicted and described for a step 322 in Figure 4c. The output of an encryption step 322 can comprise ciphertext 324a. Responder 102' can then assemble plaintext 403b (which includes ciphertext 324a in the plaintext values for plaintext 403b) and conduct an encryption step 315b, where encryption step 315b is depicted and described in connection with Figure 4b above. Responder 102' can conduct a step 324 before sending a message 123. For a step 324, responder 102' can record the derived responder ephemeral public key Pr 102p and responder ephemeral private key pr 102r and initiator bootstrap public key Bi 101a in an initiator table 102u from Figure 1f along with the identity of device 106 comprising potentially MAC.I 106n.

Responder 102' and initiator 101 can then conduct a step 506, where the series of steps and messages for a step 506 were depicted and described for a responder 102' and initiator 101 in Figure 5a above. In summary for a step 506 depicted above in a Figure 5a, a responder 102' can send initiator 101 a DPP authentication response message 123 via WiFi link 192. Initiator 101 can process the DPP authentication response message 123 using a step 331, which was depicted and described in connection with Figure 3a above. Initiator 101 can then send a DPP authentication confirm message 123a and responder 102' can receive the message 123a via WiFi link 192. Responder 102' can decrypt ciphertext 324b from message 123a using a decryption step 328b in order to read plaintext initiator authentication value 404a. In step 330 of a step 506, responder 102' can compare the decrypted value for I-auth 404a with a calculated value for I-auth 404a and if the two values match then the initiator 101 can be considered by responder 102' to be authenticated.

After authentication step 330 in step 506 is completed, in a step 505 of a step 506, responder 102' can determine the source of network credentials 109 or encrypted network credentials 109' for initiator 101. For embodiments where a network 107, an external network 108, or DPP server 103 can be the source of network credentials 109 or encrypted network credentials 109' (as specified in a responder mode 306a), then responder 102' can use the series of steps and messages in Figure 6 in order to (i) receive network credentials 109 or encrypted network credentials 109' from a DPP server 103 and (ii) send the credentials to initiator 101. For other embodiments where a responder 102' already records the network credentials 109 for initiator 101 and device 106 (such as responder 102' uses network credentials 109 to connect with network access point 105), then responder 102' can read the locally stored credentials 109 and send the credentials 109 to initiator 101 in a DPP configuration response message 124 in Figure 6 below. Responder 102' can use a step 349 as depicted in Figure 3b in order to send locally stored credentials 109 to initiator 101, where responder 102' performs the actions instead of a DPP server 103.

Figure 6

Figure 6 is a simplified message flow diagram illustrating an exemplary system with exemplary data sent and received by a DPP server, a responder proxy, and an initiator, in accordance with exemplary embodiments. System 600 can include a DPP server 103, responder proxy 102', and initiator 101 operating in device 106. Before starting steps and message flows depicted in Figure 6, DPP server 103, responder 102', and initiator 101 can previously complete exemplary message flows and steps depicted for the nodes in either Figure 5a or Figure 5b. System 600 depicts the transfer of credentials 109 or 109' from responder 102' when responder 102' calculates and operates with symmetric ciphering key ke 327a. Although the source of credentials 109 or 109' is depicted as network 107 with DPP server 103 in Figure 6, the source of credentials 109 or 109' could be either

DPP server 103 or responder 102'. A difference between system 600 and system 300b in Figure 3b is that for system 300b, DPP server 103 calculates, records, and operates with key ke 327a for use with initiator 101 in device 106.

After transmitting or sending DPP authentication confirm message 123a in a step 506 as depicted and described in connection with Figure 5a and Figure 5b, initiator 101 can conduct a step 332. Initiator 101 can take on a roll of enrollee, such that initiator 101 can prepare to send a configuration attribute 330a. Configuration attribute 330a can be used to specify that initiator 101 can take the roll of enrollee and is expecting a set of network credentials 109 from responder 102', with associated data for responder 102' to prepare or format a set of network credentials 109.

In step 332, initiator 101 can also generate a random number, for an enrollee nonce E-nonce 332a. E-nonce 332a and other nonce values can be processed using nonce parameters 199g. Initiator 101 can then conduct an encryption step 322c, where encryption step 322c can comprise (i) an encryption step 322 using key ke 327a, and (ii) the plaintext to be encrypted can be the E-nonce 332a and the configuration attribute 330a. The output of an encryption step 322c can be a ciphertext 324c, which is also described for a step 322c in Figure 3b above. Initiator 101 can then send via WiFi link 192 to responder 102' a DPP Configuration Request message 123b, where the configuration request 123b can include the ciphertext 324c. The plaintext in ciphertext 324c can include E-nonce 332a and the configuration object 330. DPP Configuration Request message 123b can be processed by initiator 101 according to section 6.3.3 of DPP specification version 1.0.

Responder 102' can receive DPP Configuration Request message 123b via connection 192 and using DPP application 102g. Responder 102' can conduct a decryption step 328c using decryption step 328 depicted in Figure 4c above. A difference between decryption step 328 and decryption step 328c is that decryption step 328c uses the ciphertext 324c as input into the symmetric decryption algorithm 405b. Responder 102' can use the previously calculated key ke 327a in order to read the plaintext within ciphertext 324c. Responder 102' can record the plaintext E-nonce 332a and configuration attribute 330a.

Responder 102' can then conduct a step 601 in order to determine the source of credentials 109 for initiator 101 and device 106. For embodiments where responder 102' previously records a valid set of network credentials 109 for use with network access point 105 depicted in Figure 1a, then responder 102' could select the recorded network access credentials 109 for use as a configuration object for initiator 101 and device 106. In other words, for embodiments where responder 102' previously connects with network access point 105 (or responder 102' could operate within network access point 105), then responder 102' could record a valid set of network credentials 109 in order to successfully connect with network access point 105. For these embodiments, responder 102' could select the recorded network access credentials 109 for transfer to initiator 101 and used by initiator 101 and device 106 in order to also connect with access point 105 using the same set of network

credentials 109 recorded by responder 102'. Responder 102' can read the locally stored credentials 109 and send the credentials 109 to initiator 101 in a DPP configuration response message 124 in Figure 6. Responder 102' can use a step 349 as depicted in Figure 3b in order to send locally stored credentials 109 to initiator 101, where responder 102' performs the actions instead of a DPP server 103.

For other embodiments, such as responder 102' does not previously record a set of network credentials 109 for use by initiator 101 and device 106, then in a step 601 responder 102' could determine a query to DPP server 103 or network 107 is required for the network credentials 109 or encrypted network credentials 109' for initiator 101 and device 106. The determination of a source for network credentials (e.g. either responder 102' or DPP server 103 could also be specified in a responder mode 306a, such that a first responder mode 306a selects responder 102' as the source of credentials 109 and a second responder mode 306a selects DPP server 103 as the source of credentials 109 or encrypted credentials 109'). Other possibilities exist as well for a step 601 by responder 102' in Figure 6 to determine a source of credentials 109 for device 106 without departing from the scope of the present disclosure.

For embodiments where step 601 by responder 102' determines that network 107 or DPP server 103 is the source of network credentials 109 for device 106, responder 102' can then send DPP server 103 a message 602 through secure session 302b (depicted and described in Figure 3a), where message 602 include an identity for device 106 such as MAC.I 106n (or device identity 106i or hash value 101aa) and the plaintext configuration attribute 330a. DPP server 103 can receive message 602 and can conduct a step 337 in order to obtain network credentials 109 or asymmetrically encrypted credentials 109'. A step 337 for DPP server 103 was depicted and described in connection with Figure 3b above, and the same step can be used by DPP server 103 for system 600 in Figure 6 (for embodiments where step 601 by responder 102' determines that network 107 or DPP server 103 is the source of network credentials 109 for device 106).

Continuing with embodiments where step 601 by responder 102' determines that network 107 or DPP server 103 is the source of network credentials 109 for device 106, DPP server 103 can then send responder 102' a message 603, where message 603 includes (i) an identity for device 106 such as MAC.I 106n or ID.device 106i or hash value 101aa, and (ii) a set of network credentials 109 or asymmetrically encrypted network credentials 109'. Asymmetrically encrypted credentials 109' could be in message 603 if DPP server 103 obtained the credentials outside of or external to DPP server 103, such as an external network 108 (where external network 108 may prefer that only device 106 be able to read the plaintext network credentials 109 inside asymmetrically encrypted network credentials 109'). A difference between message 603 in Figure 6 and message 341 in Figure 3b can be (i) the message 603 is not encrypted with key ke 327a, and (ii) the message 603 may not include a digital signature 338c. Or, for other embodiments, then a system 600 could include the use of a digital

signature creation step 338 as depicted and described in connection with Figure 3b and Figure 3d and the message 603 and subsequent message 124 can include the digital signature 338c. Responder 102' can receive message 603 and process the data. Note that message 603, step 337, and message 603 can be omitted for embodiments where step 601 determines that responder 102' is the source of credentials 109 for device 106.

For embodiments where message 603 does not include a digital signature 338c as depicted in Figure 6, responder 102' can then record the network credentials 109 or asymmetrically encrypted network credentials 109' and conduct a signature creation step 338. Note that the source of credentials 109 can be either responder 102' or DPP server 103, depending on the source selected in step 601. In exemplary embodiments, asymmetrically encrypted network credentials 109' can be from DPP server 103 or potentially an external network 108. For a step 338 in Figure 6, responder 102' can conduct a signature creation step 338 as depicted and described in connection with Figure 3d above. The configuration object with network credentials 109 (or encrypted credentials 109') can be signed using the recorded responder ephemeral private key pr 102r, corresponding to public key Pr 102p sent in message 325 in Figure 3a above. The digital signature by responder 102' in Figure 3b can comprise a digital signature 338c. Or, in another embodiment the configuration object with network credentials 109 can be signed using the recorded responder bootstrap private key br 102b recorded by DPP server 103 and digital signature 338c included in a message 603, where initiator 101 could verify the signature 338c with the bootstrap public key Br 102a.

At step 332d, responder 102' can conduct an encryption step 332d using an encryption step 322 depicted and described in connection with Figure 4c above. Encryption step 332d can be performed using key ke 327a for device 106 and initiator 101 from an initiator table 102u. For encryption step 332d, the plaintext values of (i) the E-nonce 332a and (ii) the configuration object comprising network credentials 109 (or asymmetrically encrypted credentials 109') can be input into a symmetric encryption algorithm 405a in order to generate a ciphertext 324d. As depicted, the collection of steps and messages from 602 through 322d can comprise a step 604. In some exemplary embodiments using a fourth responder mode as described in Figure 3c where responder 102 operates as a responder 102* using conventional technology in Figure 1b and Figure 1c, then a responder 102* could use the step 604 in order to obtain the credentials 109 or 109' from a DPP server 103.

Responder 102' can then send initiator 101 a DPP configuration response message 124 via connection 192, where message 124 can include the ciphertext 324d, and the ciphertext 324d can include the E-nonce 332a, the signature 338c, and either network credentials 109 or asymmetrically encrypted network credentials 109'. If network credentials 109' are sent in ciphertext 324d, then the network credentials 109' can comprise a set of "double encrypted" network credentials of (i) asymmetrically encrypted network credentials 109' and (ii) then encrypted again in a ciphertext 324d using the key ke 327a.

Initiator 101 can receive message 124 from responder 102' with ciphertext 324d. Initiator 101 can conduct a decryption step 328d using decryption step 328 depicted in Figure 4c above. A difference between decryption step 328 and decryption step 328d is that decryption step 328d uses the ciphertext 324d as input into the symmetric decryption algorithm 405b. Initiator 101 can use the previously calculated key ke 327a from a step 331 in order to read the plaintext within ciphertext 324d. Initiator 101 can confirm that plaintext in ciphertext 324d includes the previously sent E-nonce 332a. Initiator 101 can extract a plaintext value for (i) network credentials 109 or (ii) asymmetrically encrypted network credentials 109' using symmetric decryption algorithm 405b and key ke 327a. In other words, with credentials 109' decryption step 328d removes the first layer of "double encryption" where the first layer was encrypted with key ke 327a, and the "plaintext" after decryption step 328d for credentials 109' can comprise the asymmetrically encrypted network credentials 109' (e.g. with the first layer of encryption with key ke 327a has been removed).

Initiator 101 can conduct a signature verification step 340 as depicted and described in connection with Figure 3d above in order to verify signature 338c received in message 124. Initiator 101 can use either (i) responder ephemeral bootstrap public key Pr 102p or (ii) responder bootstrap public key Br 102a in order to conduct a signature verification step 340 on digital signature 338c. The use of either (i) responder ephemeral bootstrap public key Pr 102p or (ii) responder bootstrap public key Br 102a can be specified in a set of parameters 199 or as part of asymmetric encryption parameters 199h or digital signature parameters 199i. Note that if responder bootstrap public key Br 102a is used to verify digital signature 338c in a step 340, then DPP server 103 would create digital signature 338c in a step 338 with responder bootstrap private key br 102b. If plaintext network credentials 109 are included in ciphertext 324f, then initiator 101 can read the plaintext values for credentials 109. If asymmetrically encrypted network credentials 109' are included in ciphertext 324d, then initiator 101 or device 106 operating initiator 101 can conduct an asymmetric decryption step 339 depicted above in Figure 3e in order to convert credentials 109' into plaintext credentials 109.

After reading the plaintext network credentials 109 and optionally verifying a signature using a recorded public key 102p or 102a, device 106 can complete the configuration step 106a by loading the network credentials 109 and begin operating a WiFi radio 106r with the network credentials 109. In other words, the WiFi data link 192 depicted in Figure 1a can be converted into a fully authenticated and encrypted WiFi connection using network credentials 109 and standards such as WPA2, WPA3, and also subsequent or related standards where WiFi nodes such as clients or access points use a compatible set of network credentials 109. For example, device 106 could begin scanning for SSID.network-AP 109a on a channel in Config.network-AP 109c (which can be included in credentials 109 or 109'), and then send access point 105 a probe request. In this manner, device 106 can obtain connectivity to network access point 105 through a configuration step 106a.

Figure 7

Figure 7 is a graphical illustration of an exemplary system, where an access point with an initiator communicates with a responder proxy using a hosted device provisioning protocol, in accordance with exemplary embodiments. The present disclosure contemplates that the systems and methods for using a responder 102' with a DPP server 103 can support a device provisioning protocol 191 in order to configure a network access point 105'. System 700 in Figure 7 can include an unconfigured network access point 105', a mobile phone 701, and a network 107. Unconfigured network access point 105' can include an initiator 101, where initiator 101 can have the same or equivalent functionality as an initiator 101 depicted and described above in Figure 1a, Figure 1b, Figure 1c, Figure 3a, Figure 3b, Figure 5a, Figure 5b, and Figure 6.

Initiator 101 can include a responder bootstrap public key Br 102a, an initiator bootstrap private key bi 101b, and an initiator bootstrap public key Bi 101a. PKI keys could be recorded in access point 105' before access point 105' is installed at an operating location, such as during manufacturing or distribution of unconfigured access point 105'. Initiator 101 operating in unconfigured access point 105' can have responder capabilities of an enrollee. Before installation of access point 105' at an operating location, initiator 101 could record a responder public key table 101ta (depicted and described in connection with Figure 1d and also Figure 1e) with a plurality of responder bootstrap public keys Br 102a for a plurality of different networks 107.

In exemplary embodiments, the responder public key table 101ta could be loaded into access point 105' during manufacturing or distribution of access point 105' or a configuration step for access point 105' before conducting DPP 191. As contemplated herein, an access point 105 or access point 105' can comprise a "WiFi router", where one interface is a WiFi radio similar to 106r or 102r and the other interface is a wired connection such as Ethernet, DSL, or a coaxial cable. For some exemplary embodiments where a radio 102x uses wireless technology other than WiFi, then access point 105' could comprise a "gnodeb" for a 5G network, and "enodeb" for a 4G LTE network, and other possibilities exist as well for the wireless technology used by a radio 102x in access point 105'.

Unconfigured access point 105' can power up and conduct a step 200 from Figure 2a and Figure 3a in order to periodically transmit DPP authentication request messages 122 using different recorded responder bootstrap public keys (combined with possibly different initiator bootstrap PKI keys) from an initiator key database 101t in order to "search" or "scan" for a compatible responder to communicate with. When mobile phone 701 is taken to the vicinity or within WiFi range of AP 105' and activates DPP application 102g and responder 102', then responder 102' can receive the DPP authentication request message 122 and begin processing the message if mobile phone 701 records a matching hash value 102aa over at least the hash over responder bootstrap public key Br 102a recorded in memory, such as in a responder network table 102t depicted in Figure 1f.

Mobile phone 701 can comprise a smart phone or a mobile handset, such as a mobile phone manufactured by Apple® or running an Android based operating system. Mobile phone 701 could also comprise another mobile and networked computing device, such as a laptop or a tablet. Mobile phone 701 could include a responder 102', where responder 102' could have the same or equivalent functionality as responder 102' depicted and described in connection with Figure 1a, Figure 1d, Figure 1f, Figure 3a, etc. In exemplary embodiments, responder 102' can operate as an application for mobile phone 701 downloaded from a repository of mobile phone applications, such as an "app store" or "play store", or possibly a private application. In other words, embodiments herein contemplate that a mobile phone 701 could comprise the physical entity operating or processing a responder 102' and the software or firmware instructions for operation of responder 102' could comprise a DPP application 102g. Other physical arrangements are possible as well, without departing from the scope of the present disclosure. Mobile phone 701 can have both a WiFi radio 102x and a WAN radio 102h. Mobile phone 701 can communicate with unconfigured access point 105' via the WiFi radio 102x and a network 107 and a DPP server 103 via the WAN radio 102h and an IP network 113. Responder 102' for mobile phone 701 can have capabilities of a configurator.

Network 107 can operate a DPP server 103, where a DPP server 103 in Figure 7 can have the functionality of a DPP server 103 depicted and described in connection with Figures 1a, Figure 3a, Figure 3b, Figure 3c, Figure 5a, Figure 5b, and Figure 6. Although not depicted in Figure 7, a network 107 can also operate other supporting servers and/or running processes depicted in Figure 1a, Figure 2a, etc., such as a device database 104. Network 107 could be connected to mobile phone 107 via the IP network 113 depicted in Figure 1a and using a secure session 302b from Figure 3a.

As depicted in Figure 7, mobile phone 107 using responder 102' could conduct the series of messages for a device provisioning protocol 191 with initiator 101 operating in unconfigured access point 105'. The series of messages for a device provisioning protocol 191 could comprise the messages 122, 123, 123a, 123b, and 124 as depicted and described in connection with Figures 1a, 3a, 3b, 5a, 5b, and 6 above. In order for responder 102' to send and receive the device provisioning protocol messages 191 depicted in Figure 7, responder 102' transmits and receives a series of messages with DPP server 103 in network 107, where the series of messages could comprise a hosted device provisioning protocol 702. For a first mode 306a, representing embodiments depicted in Figure 3a (where responder bootstrap and ephemeral secret keys remain in DPP server 103), the series of messages for a hosted device provisioning protocol 702 could comprise the messages between responder 102' and DPP server 103 that are depicted in Figure 3a and Figure 3b (e.g. messages 302b, 305, 309, 312b, etc.).

For a second mode 306a, representing embodiments depicted in Figure 5a where the responder bootstrap private key br 102b can remain in DPP server 103 (but DPP server 103 does not operate on responder ephemeral private key pr 102r), the series of messages for a hosted device provisioning

protocol 702 could comprise the messages between responder 102' and DPP server 103 that are depicted in Figure 5a (e.g. messages in step 301 and then 503, 504, etc.). For a third mode 306a, representing embodiments depicted in Figure 5b where (i) the responder bootstrap private key br 102b remains in DPP server 103 and (ii) responder 102' derives key pr 102r and sends the key to DPP server 103, the series of messages for a hosted device provisioning protocol 702 could comprise the messages between responder 102' and DPP server 103 that are depicted in Figure 5b (e.g. messages in step 301 and then message 507, 508, etc.).

Upon completion of a device provisioning protocol 191 and a hosted device provisioning protocol 702, unconfigured network access point 105' can conduct a configuration step 106a to load a set of network credentials 109 and begin operating with the credentials 109, as depicted in Figure 7. The configuration attribute in a message 123b can specify that access point 105' is to be configured as an access point (or with a value of "ap"). The configuration step 106a can convert the unconfigured network access point 105' into an operating network access point 105, such that WiFi clients (or "sta") can connect with network access point 105 using a compatible set of network credentials 109. As depicted in Figure 7, (i) a mobile phone 701 could also operate as a responder 102' with a roll of a configurator and (ii) the access point 105' operating with a roll of enrollee.

As depicted in Figure 7, mobile phone 701 could also conduct a device provisioning protocol 191 using a hosted device provisioning protocol 702 with DPP server 103 in order to transfer a set of compatible network credentials 109 to device 106, such that device 106 could connect with network access point 105. The compatible network credentials 109 for device 106 can specify that device 106 operates in a WiFi client mode with network access point 105. Other possibilities exist as well for the operation of a responder 102' to support the transfer of credentials to devices operating as initiators 106 without departing from the scope of the present disclosure.

Figure 8

Figure 8 is a graphical illustration of an exemplary system, where an access point operates as a responder proxy for a device with an initiator, in order transfer a set of network credentials to the device, in accordance with exemplary embodiments. The present disclosure contemplates embodiments where a network access point 105 can operate as a responder 102' for a device 106 operating as an initiator 101, in order for the network access point 105 to transfer network credentials 109 to the device 106. System 800 in Figure 8 can include an unconfigured device 106', a network access point 105, and a network 107.

Network access point 105 in a system 800 can operate with a set of network access credentials 109 and also operate as a responder 102'. Note that network access point 105 as a WiFi router can simultaneously support other devices routing WiFi data through network access point 105

concurrently with network access point 105 operating as a WiFi router. In other words, the present disclosure contemplates that when a radio such as WiFi radio 102x operates for a responder 102' listening for incoming DPP authentication request messages 122, the radio 102x can concurrently support regular WiFi communication according to standard protocols such as WPA2, WPA3, etc.

5 Network access point 105 can connect with a network 107 via an IP network 113 and a secure session 302b as depicted and described in connection with Figure 3a. Network 107 can operate a DPP server 103, where the function and operation of a DPP server 103 is depicted and described above in connection with Figure 1a, Figure 2a, Figure 3a, etc. Although not depicted in Figure 8, network 107 could also operate a device database 104 and DPP server 103 could include a server database 103b.

10 Network access point 105 could comprise a WiFi router and operate a WiFi radio and communicate with a plurality of devices 106 that have already performed a configuration step 106a or a manual configuration to connect the configured devices 106. For some exemplary embodiments where a radio 102x uses wireless technology other than WiFi, then access point 105' could comprise a "gnodeb" for a 5G network, and "enodeb" for a 4G LTE network, and other possibilities exist as well for the wireless

15 technology used by a radio 102x in access point 105'.

Network access point 105 can have the internal electrical components equivalent to a responder 102' depicted and described above in connection with Figure 1f, including a CPU, RAM, nonvolatile memory, and a system bus. For the embodiment depicted in Figure 8, the responder 102' in network access point 105 could comprise software, firmware, or a running process within network

20 access point 105. Responder 102' in network access point 105 could have the same or equivalent functionality as responder 102' depicted and described in connection with Figure 1a, Figure 1f, Figure 3a, Figure 3b, Figure 5a, Figure 5b, and Figure 6. Network access point 105 could operate with capabilities as a configurator 801, as depicted in Figure 8.

An unconfigured device 106' could be a device with a WiFi radio 106r and operate an initiator

25 101. The electrical components inside device 106' could be similar to the electrical components depicted for a responder 102' in Figure 1f, such as including a CPU, RAM, nonvolatile memory, an operating system, and a system bus. Initiator 101 in unconfigured device 106' could have the same or equivalent functionality as a initiator 101 depicted and described above in Figure 1a, Figure 1b, Figure 1c, Figure 1d, Figure 1e, Figure 3a, etc.. Unconfigured device 106' can have capabilities as an enrollee

30 802.

Unconfigured device 106' can power up and conduct a step 200 from Figure 2b and Figure 3a in order to periodically transmit DPP authentication request messages 122 using different recorded responder bootstrap public keys (combined with possibly different initiator bootstrap PKI keys) from an initiator key database 101t in Figure 1d in order to "search" or "scan" for a compatible responder

35 to communicate with. When unconfigured device 106' is powered on or taken to the vicinity or within WiFi range of AP 105, then responder 102' in WiFi access point 105 can receive the DPP

authentication request message 122 from unconfigured device 106' and begin processing the message 122 if WiFi access point 105 records a matching hash value 102aa over the hash over responder bootstrap public key Br 102a recorded in memory, such as in a responder network table 102t depicted and described in connection with Figure 1f.

5 As depicted in Figure 8, network access point 105 with responder 102' could conduct the series of messages for a device provisioning protocol 191 with initiator 101 operating in unconfigured device 106'. The series of messages for a device provisioning protocol 191 could comprise the messages 122, 123, 123a, 123b, and 124 as depicted and described in connection with Figure 3a, Figure 3b, Figure 5a, Figure 5b, and Figure 6. In order for responder 102' in WiFi access point 105 to send and
10 receive the device provisioning protocol messages 191 depicted in Figure 8, responder 102' (or AP 105 operating the responder 102') transmits and receives a series of messages with DPP server 103 in network 107, where the series of messages could comprise a hosted device provisioning protocol 702. The series of messages for a hosted device provisioning protocol 702 were depicted and described in connection with Figure 7 above. Note that the steps and messages for conducting a hosted device
15 provisioning protocol 702 can depend on a responder mode 306a, where the responder mode 306a can be selected by a DPP server 103 in a step 306. The present disclosure also contemplates that a responder mode 306a for a hosted device provisioning protocol 702 can be selected by a responder 102'. For these embodiments, where mode 306a is selected by a responder 102', a selected responder mode 306a by the responder 102' could be transmitted from responder 102' to DPP server 103 in a
20 message 305 depicted and described in connection with Figure 3a above.

The combined (i) device provision protocol 191 between network access point 105 and unconfigured device 106' and (ii) hosted device provisioning protocol 702 between network access point 105 and DPP server 103 can securely transfer a set of compatible network access credentials 109 to device 106'. Upon completion of a device provisioning protocol 191 and a hosted device
25 provisioning protocol 702, unconfigured device 106' can conduct a configuration step 106a to (i) load a compatible set of network credentials 109 for network access point 105 and (ii) begin operating with the credentials 109, as depicted in Figure 8. The configuration step 106a can convert the unconfigured device 106' into a configured and operating device 106, such that device 106 can conduct a WiFi connection setup step 803 using the received set of compatible network credentials 109. The WiFi
30 connection setup step 803 could use the series of steps and messages as specified by standards such as WPA2 and WPA3 with credentials 109, and other possibilities exist as well for a WiFi connection setup step 803 without departing from the scope of the present disclosure.

Conclusion

Various exemplary embodiments have been described above. Those skilled in the art will
35 understand, however, that changes and modifications may be made to those examples without departing from the scope of the claims.

CLAIMS

1. A method for conducting a device provisioning protocol (DPP), the method performed by a responder, the method comprising:

a) establishing a secure session with a DPP server using at least a certificate for the DPP
5 server;

b) receiving, from an initiator, a DPP authentication request, wherein the DPP authentication request includes at least an initiator ephemeral public key, a hash value for a responder bootstrap public key, and a first ciphertext;

c) sending, to the DPP server, at least the initiator ephemeral public key, the hash value,
10 and the first ciphertext;

d) receiving, from the DPP server, a responder ephemeral public key and a second ciphertext, wherein the second ciphertext is encrypted using at least a responder ephemeral private key corresponding to the responder ephemeral public key, and wherein the second ciphertext includes at least a responder nonce and a responder authentication value;

e) sending, to the initiator, a DPP authentication response, wherein the DPP authentication response includes at least the received responder ephemeral public key and the second ciphertext;
15 and

f) receiving, from the DPP server, a third ciphertext, wherein the third ciphertext includes network credentials for the initiator, and wherein the third ciphertext is encrypted using at least (i)
20 the responder ephemeral private key, and (ii) the responder bootstrap private key.

2. The method of claim 1, wherein the DPP server derives the responder ephemeral private key and the responder ephemeral public key using a set of cryptographic parameters, and wherein the set of cryptographic parameters are selected using at least the hash value.

3. The method of claim 2, wherein the hash value, the set of cryptographic parameters, and the responder bootstrap private key are stored in a device database, and wherein the DPP server uses the hash value to query the device database for the set of cryptographic parameters and the responder bootstrap private key.

5 4. The method of any one of claims 1 to 3, wherein the responder sends the hash value for the responder bootstrap public key to the DPP server before step b), and wherein the responder stores the hash value before a step a).

5. The method of any one of claims 1 to 4, further comprising, after step a) and before step d), receiving a responder mode from the DPP server, wherein the responder mode signals the
10 DPP server to derive the responder ephemeral private key.

6. The method of any one of claims 1 to 5, wherein the second ciphertext includes the responder authentication value as a fourth ciphertext, and wherein the fourth ciphertext is encrypted by the DPP server using at least (i) the responder ephemeral private key, and (ii) the responder bootstrap private key.

15 7. The method of claim 6, wherein the fourth ciphertext is encrypted using at least an initiator bootstrap public key, and wherein a network operating the DPP server stores the initiator bootstrap public key.

8. The method of any one of claims 1 to 7, wherein the first ciphertext is encrypted with a symmetric ciphering key, and wherein the symmetric ciphering key is derived by the
20 initiator using the responder bootstrap public key and an initiator ephemeral private key corresponding to the received initiator ephemeral public key.

9. The method of any one of claims 1 to 8, wherein the DPP server conducts an elliptic curve Diffie-Hellman key exchange with the responder ephemeral private key and the initiator ephemeral public key in order to derive a symmetric ciphering key for the second ciphertext.

10. The method of any one of claims 1 to 9, wherein the DPP server conducts an elliptic curve Diffie-Hellman key exchange with the responder ephemeral private key, the responder bootstrap private key, and an initiator bootstrap public key in order to derive a symmetric ciphering key for the third ciphertext.

5 11. The method of claim 10, wherein the DPP authentication request includes a second hash value for the initiator bootstrap public key, wherein the responder sends to the DPP server the second hash value in a step c), and wherein the DPP server selects the initiator bootstrap public key for the initiator using the second hash value.

10 12. The method of any one of claims 1 to 11, wherein the responder comprises a responder proxy, and wherein the responder uses a WiFi radio with a responder configuration in order to receive the DPP authentication request message from the initiator.

13. The method of any one of claims 1 to 12, further comprising after step e) and before step f), receiving, from the initiator, a DPP authentication confirm message with an encrypted initiator authentication value, wherein the responder sends to the DPP server at least the encrypted
15 initiator authentication value.

14. The method of claim 13, wherein the DPP server decrypts the encrypted initiator authentication value using at least (i) the responder ephemeral private key, and (ii) the responder bootstrap private key, and wherein the DPP server compares the received initiator authentication value with a calculated initiator authentication value.

20 15. The method of any one of claims 1 to 14, further comprising:
recording a responder network table, wherein the responder network table includes at least the hash value; and
using the hash value and the responder network table to select the DPP server.

16. A method for conducting a device provisioning protocol (DPP), the method performed by a responder, the method comprising:

a) recording a set of cryptographic parameters and a responder network table;

b) receiving a DPP authentication request from an initiator, wherein the DPP authentication request includes at least an initiator ephemeral public key and a hash value for a responder bootstrap public key;

c) using the hash value and the responder network table to select a DPP server;

d) sending at least the initiator ephemeral public key and the hash value to the selected DPP server;

e) deriving (i) a responder ephemeral private key, (ii) a corresponding responder ephemeral public key, and (iii) a responder nonce using the set of cryptographic parameters;

f) receiving an initiator bootstrap public key and a secret key from the DPP server;

g) conducting an elliptic curve Diffie-Hellman (ECDH) key exchange using the derived responder ephemeral private key and the received initiator bootstrap public key in order to derive a first shared secret key;

h) conducting an elliptic curve point addition with the secret key and the derived first shared secret key in order to derive a second shared secret key; and

i) encrypting at least the responder nonce using the derived second shared secret key.

17. The method of claim 16, wherein the responder comprises a responder proxy, and wherein the responder establishes a secure session with the DPP server using at least a certificate for the responder.

18. The method of claim 16 or 17, further comprising (i) transmitting the encrypted responder nonce to the initiator in a DPP authentication response message, and (ii) receiving from the initiator a DPP authentication confirm message.

19. The method of claim 18, further comprising (i) receiving from the initiator a DPP configuration request after receiving the DPP authentication confirm message, and (ii) transmitting a DPP configuration response message, wherein the DPP configuration response message includes network credentials for the initiator.

5 20. The method of claim 19, wherein the responder stores the network credentials.

21. The method of claim 19, further comprising (i) sending at least a configuration attribute from the DPP configuration request to the DPP server, and (ii) receiving the network credentials from the DPP server before sending the DPP configuration response message.

22. The method of claim 21, wherein the network credentials received from the DPP
10 server comprise asymmetrically encrypted network credentials.

23. The method of any one of claims 16 to 22, wherein the responder conducts the ECDH key exchange and the elliptic curve point addition using the set of cryptographic parameters.

24. The method of any one of claims 16 to 23, wherein the responder network table
15 includes at least (i) a plurality of hash values for a plurality of responder bootstrap public keys and (ii) a list of DPP servers.

25. The method of any one of claims 16 to 24, wherein the derived second shared secret key comprises a key "L", wherein the responder derives a symmetric ciphering key using a key derivation function and at least the key "L", and wherein the responder encrypts at least the
20 responder nonce using the symmetric ciphering key.

26. The method of any one of claims 16 to 25, wherein the DPP authentication request includes a first ciphertext of at least an initiator nonce, and wherein the first ciphertext is encrypted with a symmetric ciphering key, and wherein the symmetric ciphering key is derived by the initiator using the responder bootstrap public key and an initiator ephemeral private key
25 corresponding to the initiator ephemeral public key.

27. A method for conducting a device provisioning protocol (DPP), the method performed by a responder, the method comprising:

a) recording a set of cryptographic parameters and a responder network table, wherein the responder network table includes at least (i) a plurality of hash values for a plurality of responder bootstrap public keys and (ii) a list of DPP servers;

b) receiving, from an initiator, a DPP authentication request, wherein the DPP authentication request includes at least an initiator ephemeral public key, a hash value for a responder bootstrap public key, and a first ciphertext;

c) using the hash value and the responder network table to select a DPP server;

d) sending the initiator ephemeral public key, the hash value for a responder bootstrap public key, and the first ciphertext to the selected DPP server;

e) receiving, from the DPP server, a responder ephemeral public key and a second ciphertext, wherein the second ciphertext is encrypted using at least a responder ephemeral private key corresponding to the responder ephemeral public key, and wherein the second ciphertext includes at least a responder nonce and a responder authentication value;

f) sending, to the initiator, a DPP authentication response, wherein the DPP authentication response includes at least the received responder ephemeral public key and the second ciphertext; and

g) receiving, from the DPP server, a third ciphertext, wherein the third ciphertext includes network credentials for the initiator, and wherein the third ciphertext is encrypted using at least (i) the responder ephemeral private key, and (ii) the responder bootstrap private key.

28. The method of claim 27, wherein the responder comprises a responder proxy, and wherein the responder establishes a secure session with the DPP server using at least a certificate for the DPP server.

29. The method of claim 27 or 28, wherein the DPP server derives the responder ephemeral private key and the responder ephemeral public key using the set of cryptographic parameters, and wherein the set of cryptographic parameters are selected by the DPP server using at least the hash value.

5 30. The method of claim 29, wherein the hash value, the set of cryptographic parameters, and the responder bootstrap private key are stored in a device database, and wherein the DPP server uses the hash value to query the device database for the set of cryptographic parameters and the responder bootstrap private key.

31. The method of any one of claims 27 to 30, wherein the responder sends the hash
10 value for the responder bootstrap public key to the DPP server before step b), and wherein the responder network table includes the hash value.

32. The method of any one of claims 27 to 31, further comprising, after step a) and before step e), receiving a responder mode from the DPP server, wherein the responder mode signals the DPP server to derive the responder ephemeral private key.

15 33. The method of any one of claims 27 to 32, wherein the first ciphertext is encrypted with a symmetric ciphering key, and wherein the symmetric ciphering key is derived by the initiator using the responder bootstrap public key and an initiator ephemeral private key corresponding to the initiator ephemeral public key.

34. The method of any one of claims 27 to 33, wherein the DPP server derives the
20 responder ephemeral private key, and wherein the DPP server conducts an elliptic curve Diffie-Hellman key exchange with the responder ephemeral private key and the initiator ephemeral public key in order to derive a symmetric ciphering key for the second ciphertext.

35. The method of any one of claims 27 to 34, wherein the second ciphertext includes the responder authentication value as a fourth ciphertext, and wherein the fourth ciphertext is
25 encrypted by the DPP server using at least (i) the responder ephemeral private key, and (ii) the responder bootstrap private key.

36. The method of claim 35, wherein the fourth ciphertext is encrypted using at least an initiator bootstrap public key, and wherein a network operating the DPP server stores the initiator bootstrap public key.

37. The method of any one of claims 27 to 36, wherein the DPP server conducts an elliptic curve Diffie-Hellman key exchange with the responder ephemeral private key, the responder bootstrap private key, and an initiator bootstrap public key in order to derive a symmetric ciphering key for the third ciphertext.

38. The method of claim 37, wherein the DPP authentication request includes a second hash value for an initiator bootstrap public key, wherein the responder sends to the DPP server the second hash value in a step d), and wherein the DPP server selects the initiator bootstrap public key for the initiator using the second hash value.

39. The method of any one of claims 27 to 38, wherein the responder comprises a responder proxy, and wherein the responder uses a WiFi radio with a responder configuration in order to receive the DPP authentication request message from the initiator.

40. The method of any one of claims 27 to 39, further comprising after step f) and before step g), receiving, from the initiator, a DPP authentication confirm message with an encrypted initiator authentication value, wherein the responder sends to the DPP server at least the encrypted initiator authentication value.

41. The method of claim 40, wherein the DPP server decrypts the encrypted initiator authentication value using at least (i) the responder ephemeral private key, and (ii) the responder bootstrap private key, and wherein the DPP server compares the received initiator authentication value with a calculated initiator authentication value.

42. A system for supporting a device provisioning protocol (DPP), the system comprising:

a wireless device for operating a DPP application, wherein the DPP application comprises at least, in part, a responder, for establishing a secure session with a DPP server using at least a certificate for the DPP server, for storing at least one hash value for a responder bootstrap public key, for using a responder configuration to communicate over a WiFi link with an initiator, for using the WiFi link to receive a DPP authentication request message with at least (i) an initiator ephemeral public key and (ii) the hash value, for receiving an encrypted set of network credentials from the DPP server for the initiator, and for sending over the WiFi link the encrypted set of network credentials to the initiator in a DPP configuration response message;

a database for recording (i) a responder bootstrap private key and (ii) a set of cryptographic parameters for the responder bootstrap private key;

a DPP server for receiving the initiator ephemeral public key and the hash value from the wireless device, for querying the database using the hash value in order to receive the responder bootstrap private key, and for conducting an elliptic curve Diffie-Hellman (ECDH) key exchange using at least (i) the received initiator ephemeral public key, (ii) the received responder bootstrap private key, and (iii) the set of cryptographic parameters in order to derive a shared secret;

a processor in the DPP server for deriving a symmetric ciphering key using at least the derived shared secret, and for encrypting the set of network credentials using the symmetric ciphering key and the set of cryptographic parameters; and

a LAN interface in the DPP server for using the secure connection (i) to send the hash value to the wireless device before the wireless device receives the DPP authentication request and (ii) to send the encrypted set of network credentials to the initiator.

43. The system of claim 42, further comprising the DPP server for selecting a responder mode for the wireless device, and for sending the responder mode to the wireless device.

44. The system of claim 42 or 43, wherein the wireless device operates as a responder proxy, and wherein the wireless device does not receive and does not operate with the responder bootstrap private key.

45. The system of any one of claims 42 to 44, further comprising a network access
5 point for authenticating with the set of network access credentials, and for providing connectivity with an IP network to the initiator after authenticating with the set of network access credentials.

46. The system of any one of claims 42 to 45, further comprising the DPP server for
deriving (i) a responder ephemeral public key for a DPP authentication response message after
receiving the hash value and (ii) a ciphertext of a responder authentication value, and wherein the
10 ciphertext is encrypted using at least the derived shared secret, wherein the DPP server sends the
responder ephemeral public key and the ciphertext to the wireless device before sending the
encrypted set of network credentials.

47. The system of any one of claims 42 to 46, wherein the DPP server uses the
responder bootstrap private key with a plurality of different initiator bootstrap public keys,
15 wherein the plurality of different initiator bootstrap public keys are associated with a plurality of
different initiators.

48. A wireless device for supporting a device provisioning protocol (DPP), the wireless
device comprising:

a WAN radio for establishing a secure session with a server, and for receiving from the
20 server (i) a responder configuration, (ii) a first hash value of a responder bootstrap public key, and
for sending to the server a first ciphertext for a DPP authentication request message from an
initiator;

a WiFi radio for operating with the responder configuration, for receiving from the initiator
the DPP authentication request message with (i) the first ciphertext, (ii) the first hash value, and
25 (iii) an initiator ephemeral public key, and for transmitting to the initiator a DPP authentication
response message with a second ciphertext and a responder ephemeral public key;

a nonvolatile memory for storing (i) a responder network table with at least the first hash value and an identity for the server, and (ii) a DPP application for conducting the DPP over the WiFi radio using the responder configuration;

a processor for selecting the identity from the responder network table using the first hash value, for validating the initiator ephemeral public key, for encrypting the initiator ephemeral public key and the first ciphertext for the secure session, and for storing in volatile memory a third ciphertext from the server, wherein the third ciphertext includes data for a DPP authentication confirm message; and

a data bus for electrically connecting the WAN radio, WiFi radio, nonvolatile memory, and processor, and for transferring from the WAN radio to the WiFi radio a fourth ciphertext of network credentials for the initiator.

49. The wireless device of claim 48, wherein the wireless device (i) does not store and (ii) does not receive a responder bootstrap private key corresponding to the responder bootstrap public key.

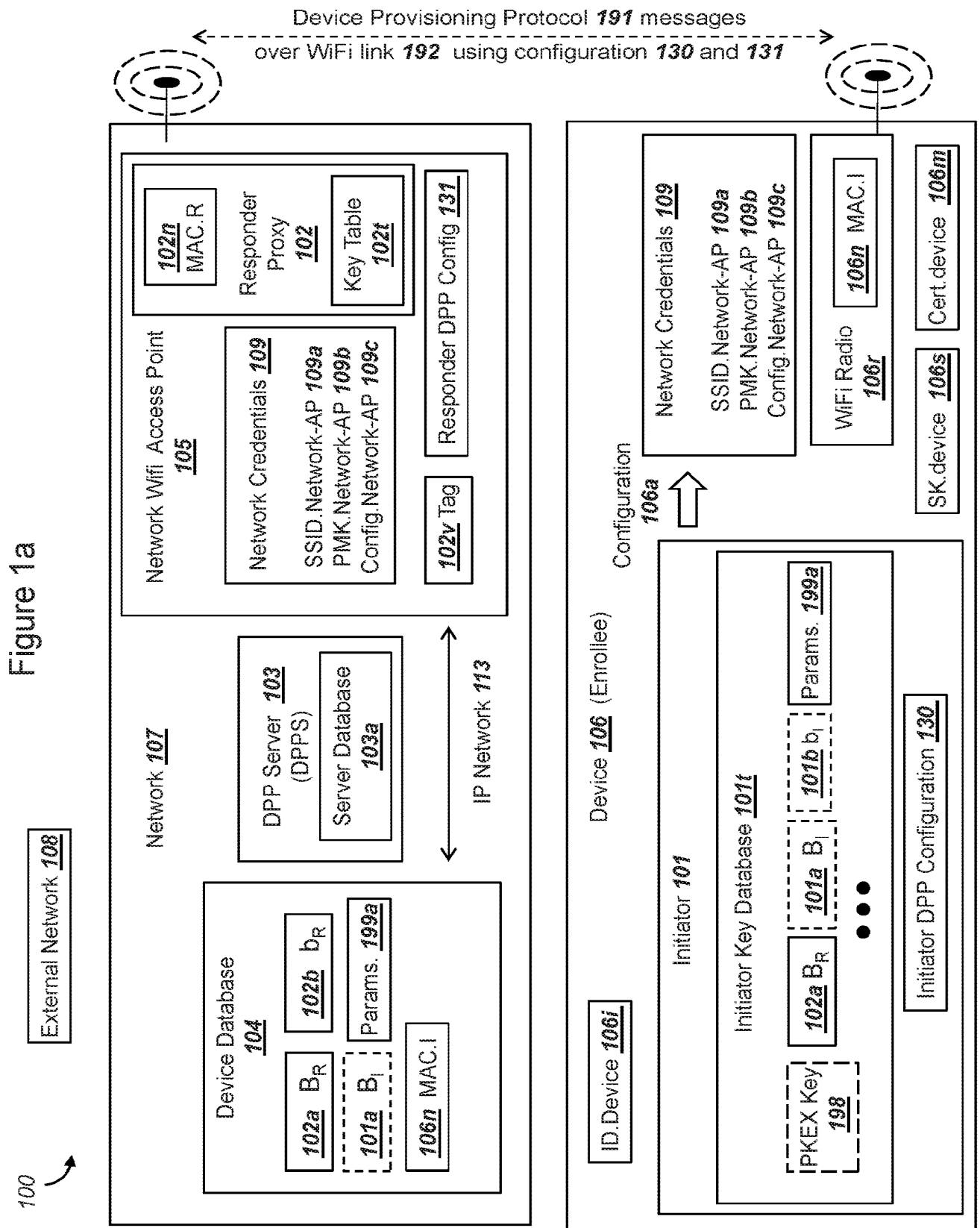
50. The wireless device of claim 48 or 49, wherein the wireless device comprises a responder for the DPP, and wherein the DPP authentication request message is received by the wireless device as a broadcast message over a WiFi link.

51. The wireless device of any one of claims 48 to 50, wherein the server records a responder bootstrap private key corresponding to the responder bootstrap public key, wherein the WiFi radio receives a second hash value of an initiator bootstrap public key with the DPP authentication request, and wherein the WAN radio sends the second hash value to the server.

52. The wireless device of any one of claims 48 to 51, wherein the third ciphertext is encrypted with a symmetric ciphering key, and wherein the symmetric ciphering key is derived by the server using an elliptic curve Diffie Hellman (ECDH) key exchange and at least (i) the responder bootstrap private key, and (ii) the initiator ephemeral public key.

53. The wireless device of claim 52, wherein the symmetric ciphering key is derived using an initiator bootstrap public key.

54. The wireless device of any one of claims 48 to 53, wherein the WiFi radio transmits the DPP authentication confirm message that includes the third ciphertext, wherein the WiFi radio
5 receives a DPP configuration request message after the WiFi radio transmits the DPP authentication confirm message, wherein the WiFi radio transmits a DPP configuration response message with the fourth ciphertext to the initiator after the WiFi radio receives the DPP configuration request message.



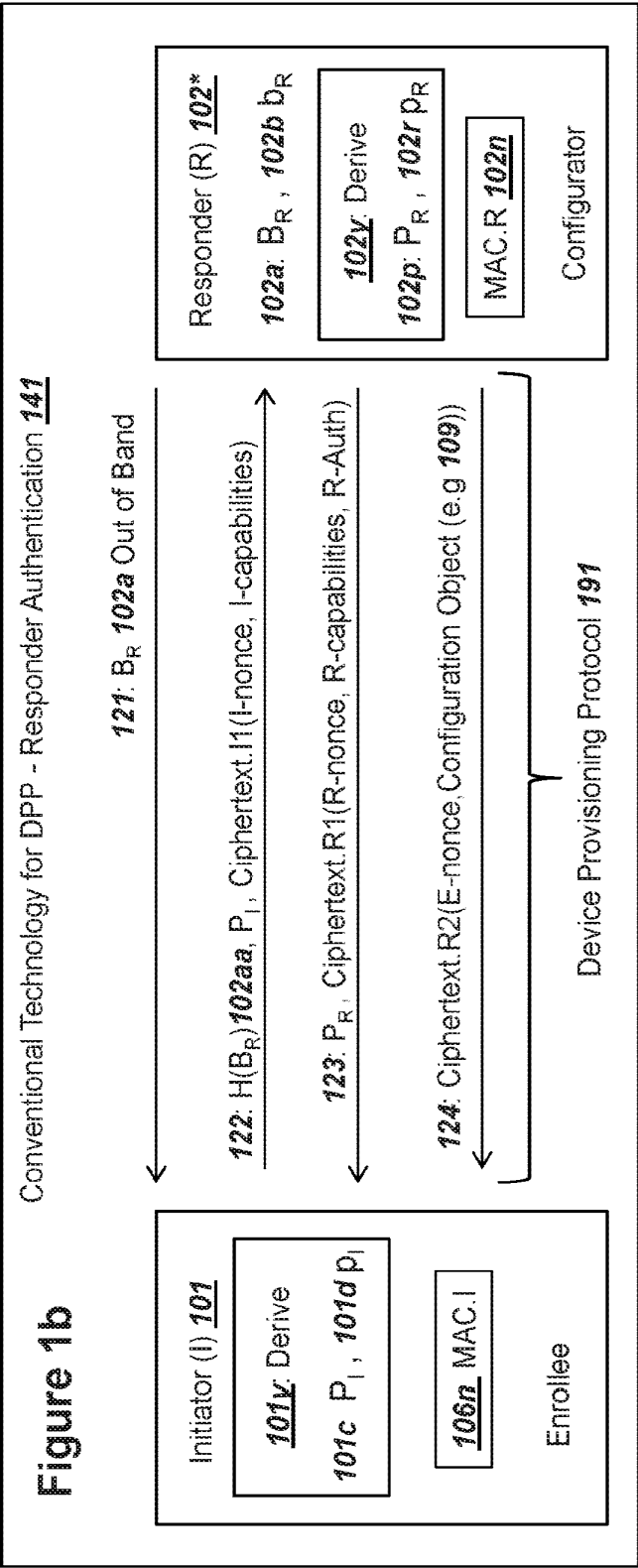
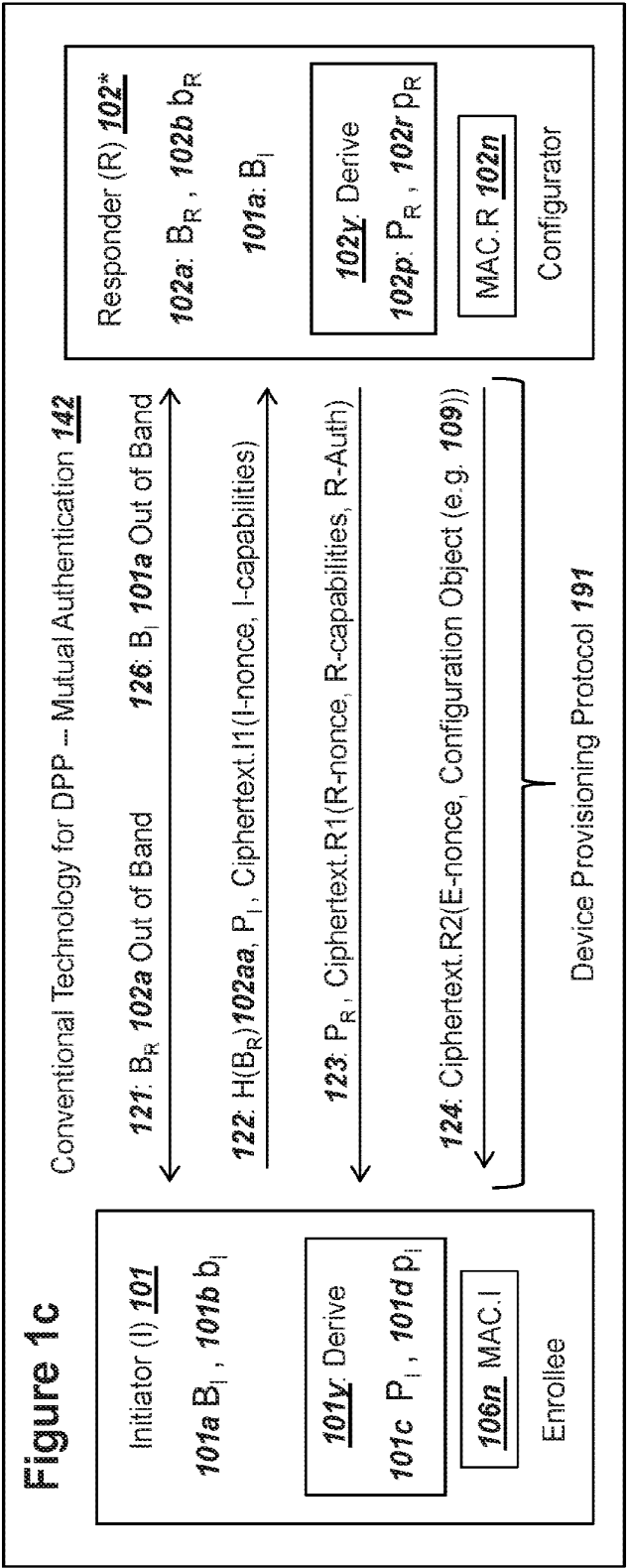
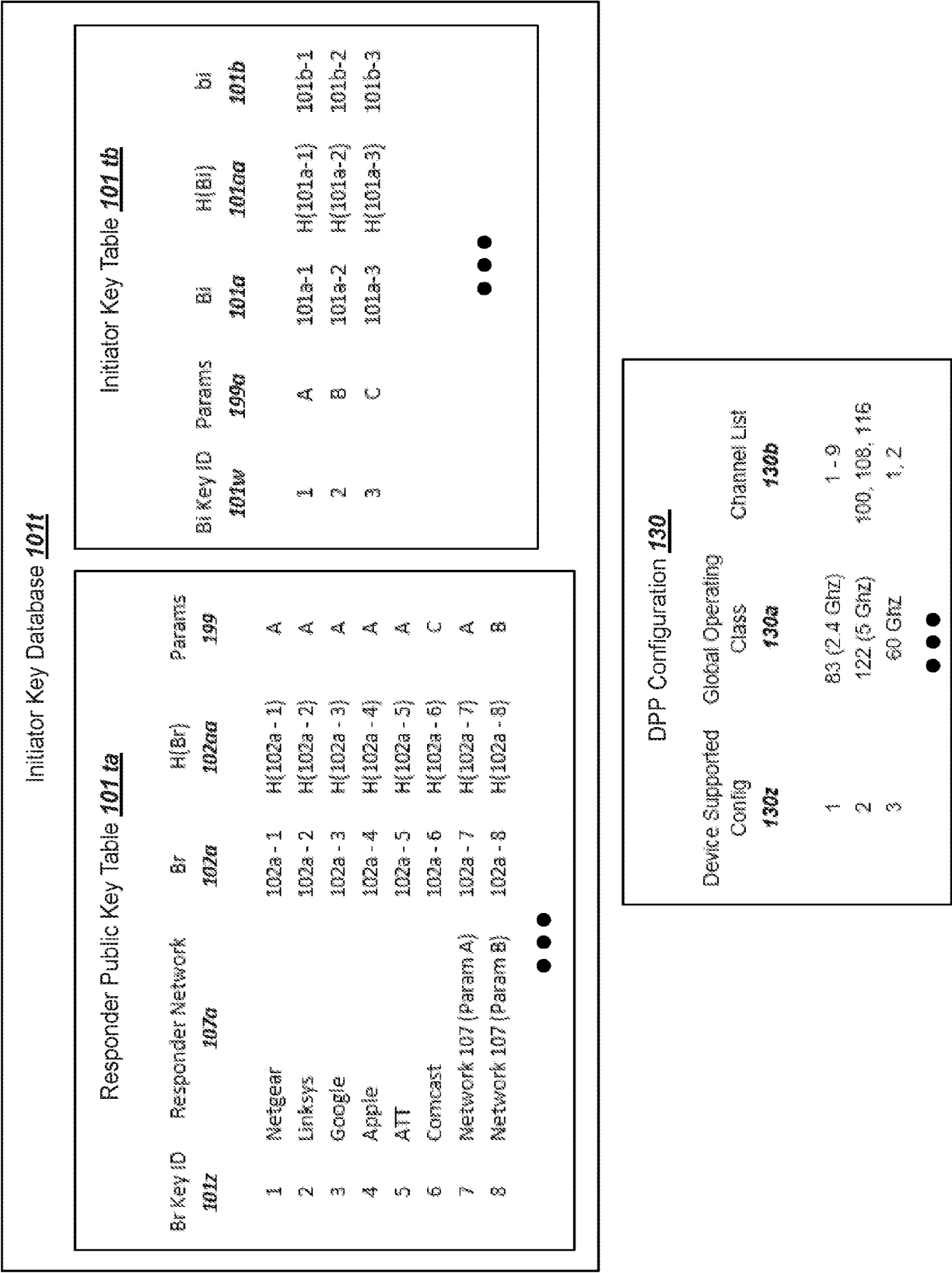


Figure 1d



DPP Configuration **130**

Device Supported Config	Global Operating Class	Channel List
130z	130a	130b
1	83 (2.4 Ghz)	1-9
2	122 (5 Ghz)	100, 108, 116
3	60 Ghz	1, 2
...		

Figure 1e

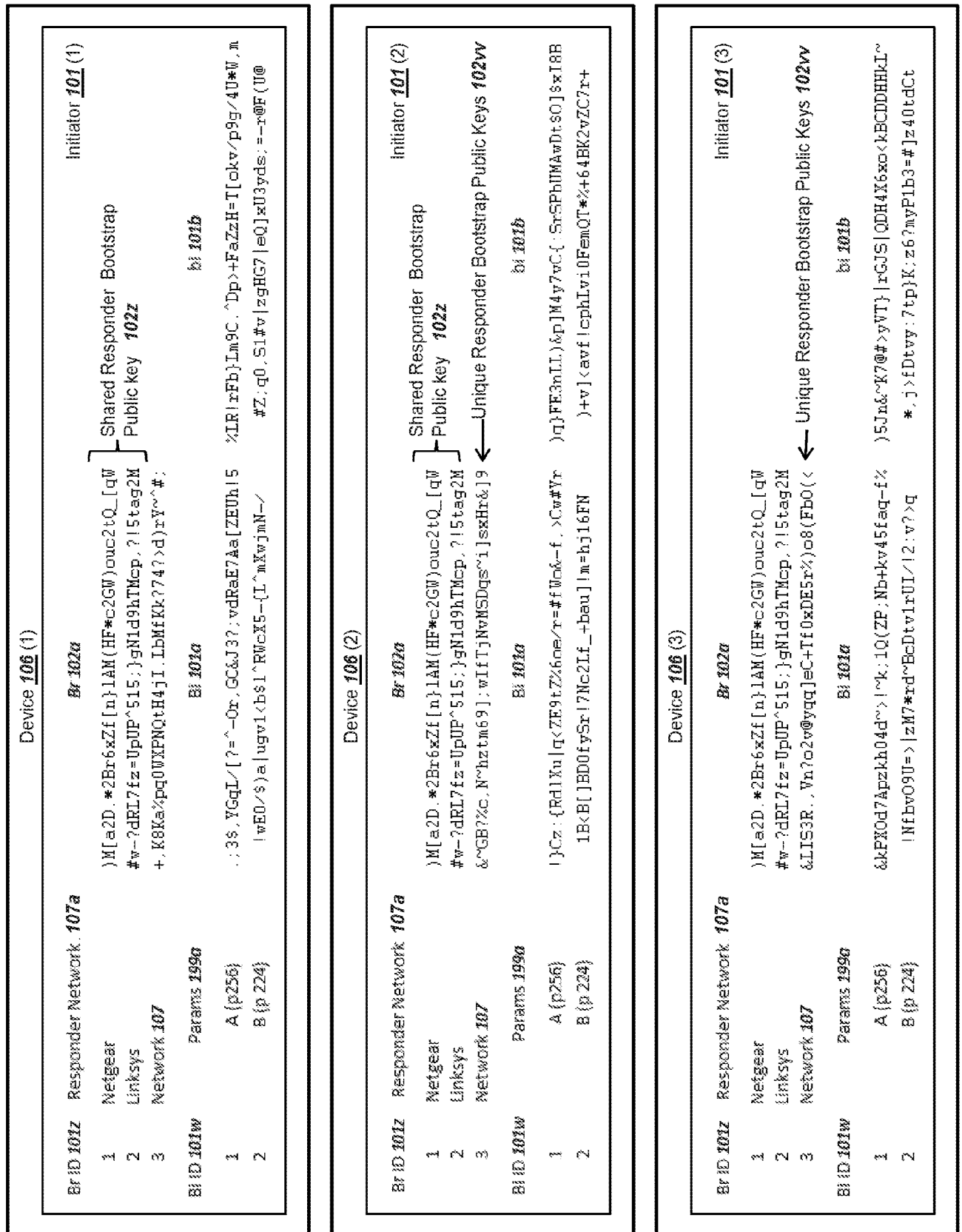
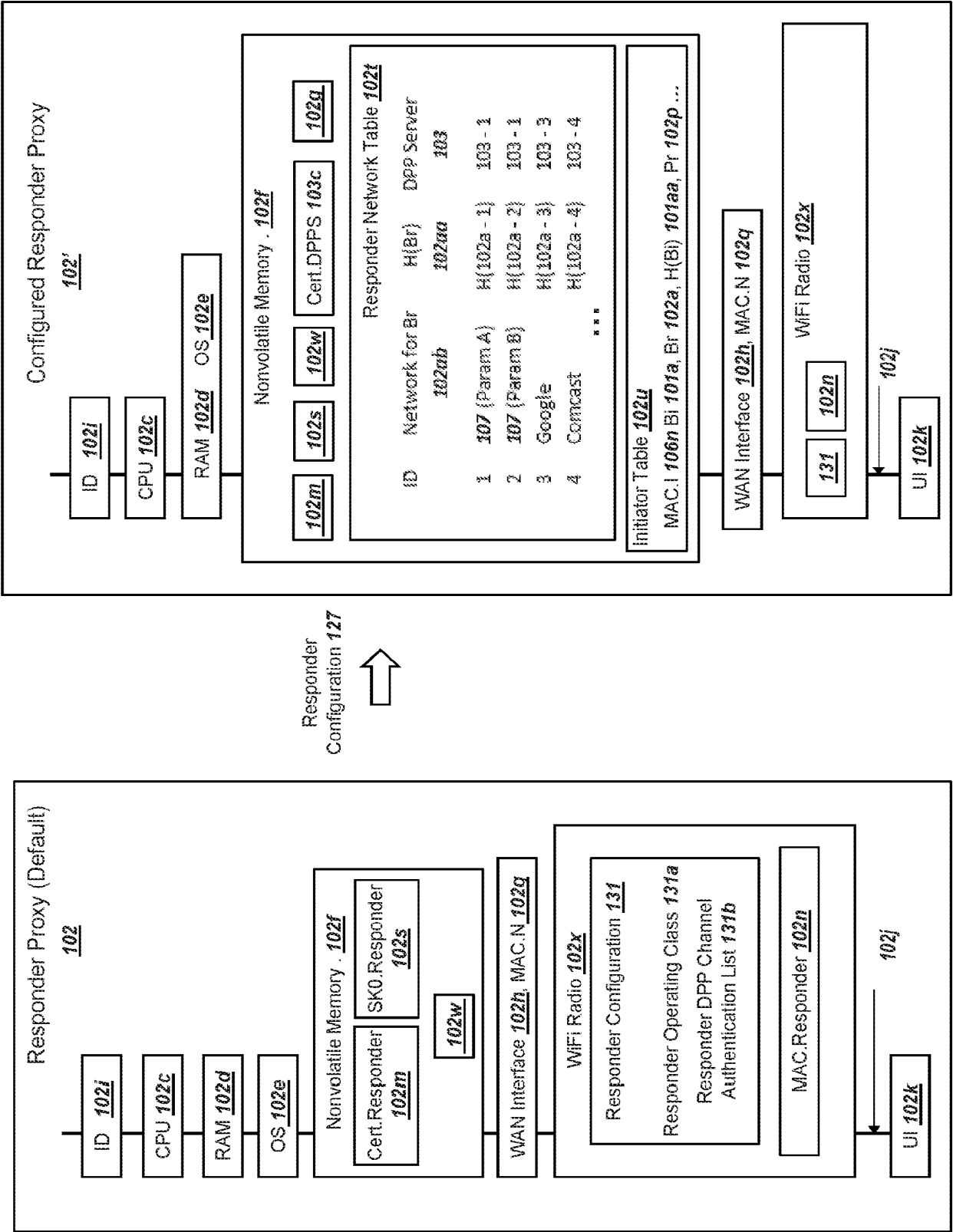


Figure 1f



7/20

Figure 2b

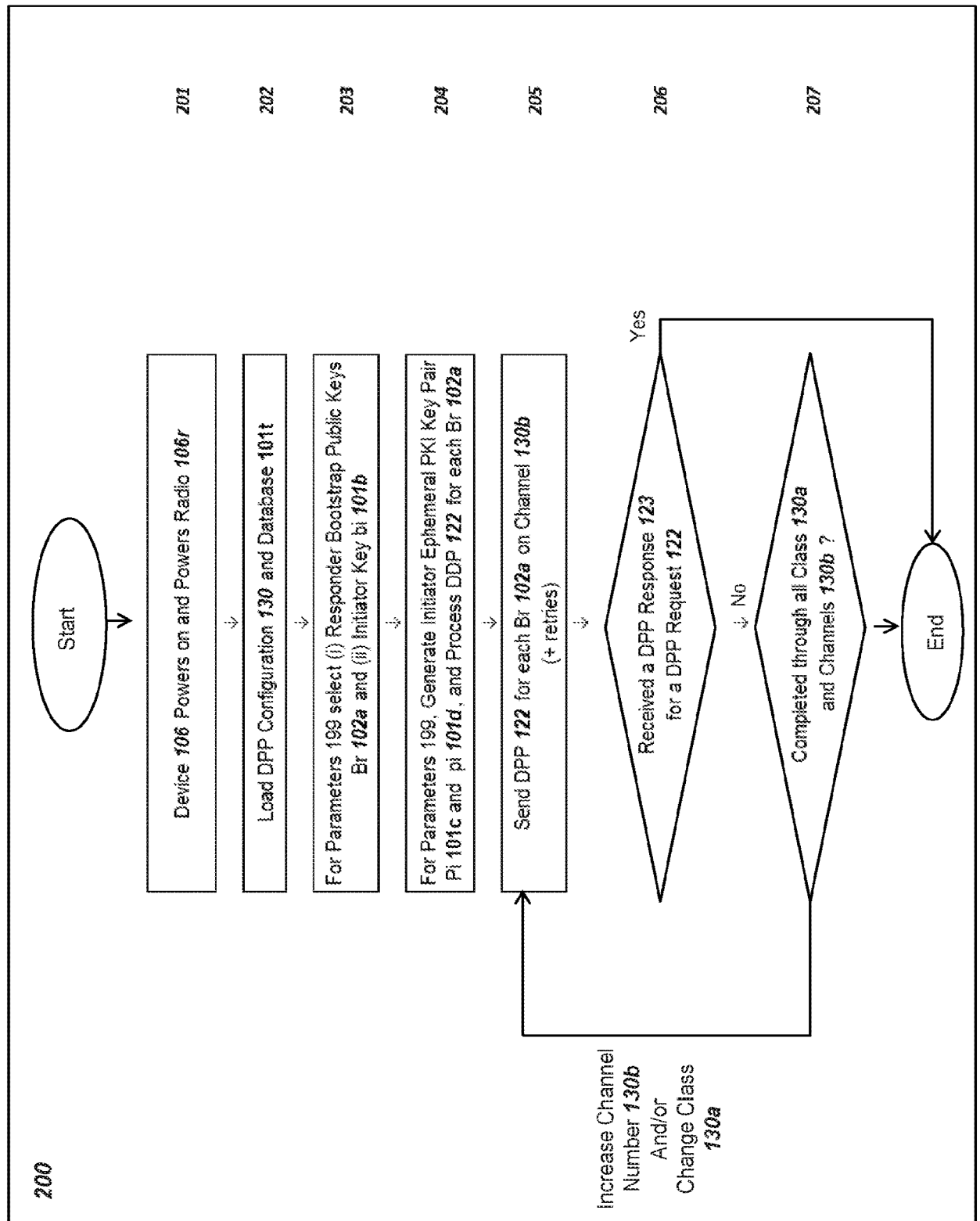


Figure 3b

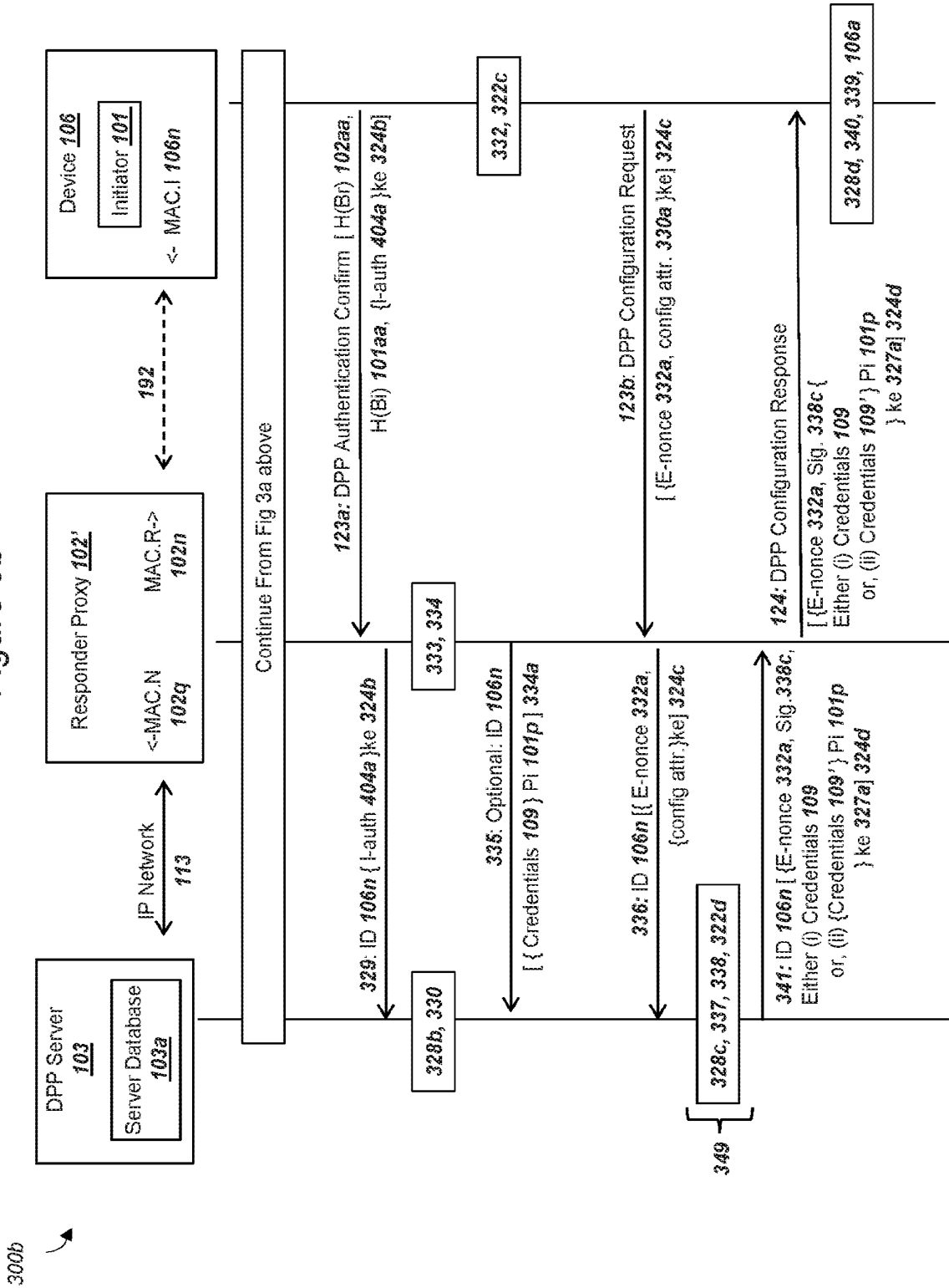


Figure 3c

Server Database <u>103a</u>												
MAC Initiator <u>106n</u>	MAC <u>102n</u>	Responder Mode <u>306a</u>	Responder Cryptog. Params <u>199a</u>	Responder		Initiator		PKEX		Shared		Shared Secret ke <u>327a</u>
				Ephem. Public Key <u>102p</u>	Ephem. Private Key <u>102r</u>	Ephem. Public Key <u>101c</u>	Ephem. Private Key <u>101r</u>	Key <u>198</u>	Secret k1 <u>314a</u>	Secret k2 <u>314c</u>	Secret ke <u>327a</u>	
38607720A	6cf049080	Fig. 3	A	Pr-1	pr-1	Pl-1	Pl-1	X1	k1-1	k2-1	ke-1	
386077208	4a508b99	Fig. 5a	A	-	-	Pl-2	Pl-2	NA	-	-	-	
386077209	1c6f654d2	Fig. 5b	A	-	pr-2	Pl-2	Pl-2	NA	-	-	-	
386077400	7071bce29	102*	B	-	-	-	-	NA	-	-	-	
3860773cc	7071bce29	Fig. 3	C	Pr-4	pr-4	Pl-4	Pl-4	X4	k1-4	k2-4	ke-4	

• • •

Device Credentials Table <u>345</u>				
Device Identity <u>106i</u>	MAC. Initiator <u>106n</u>	Primary Network Credentials <u>350a</u>	Backup Network Credentials <u>350b</u>	
20001	38607720A	109 - 1	109 - 3	
20002	386077209	109 - 1	109 - 4	
20003	386077400	109' - 1	109' - 3	
20004	3860773cc	109' - 2	109' - 4	
20005	386077a09	109 - 2	109 - 5	

Figure 3e

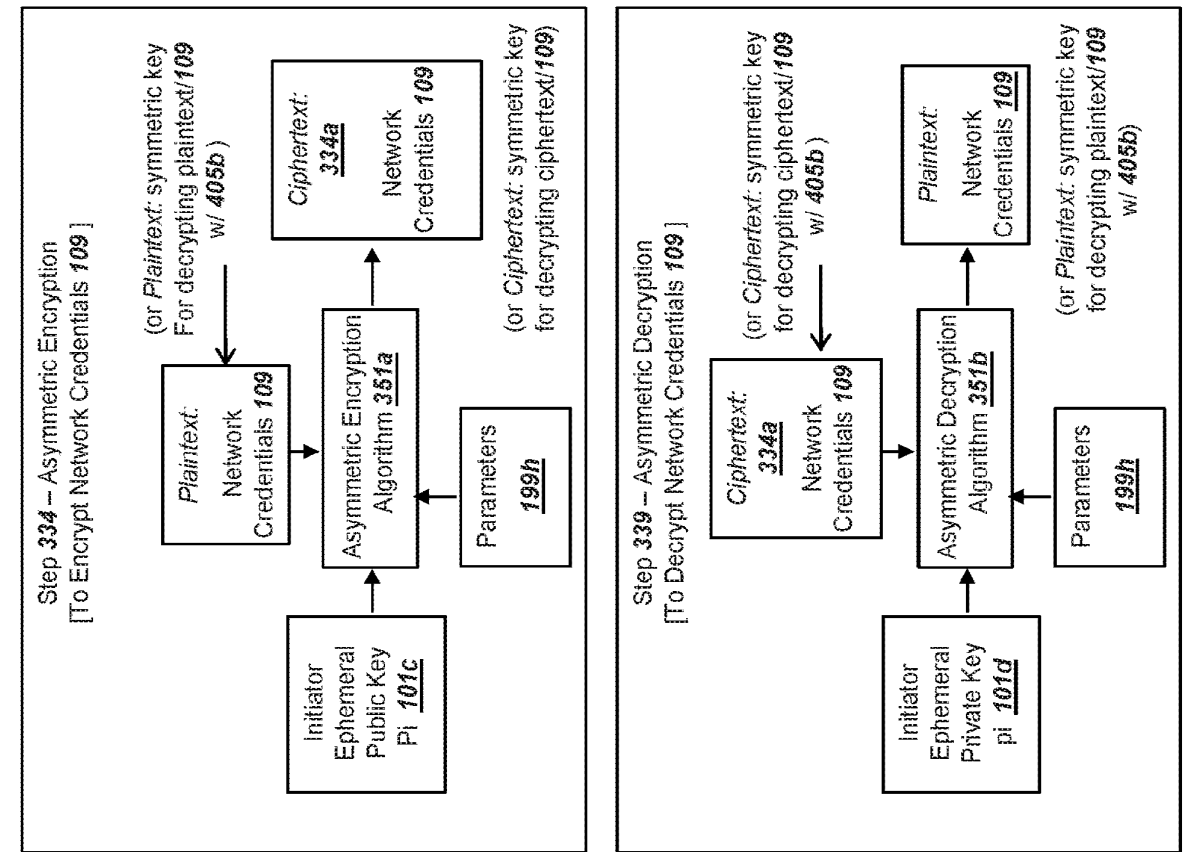
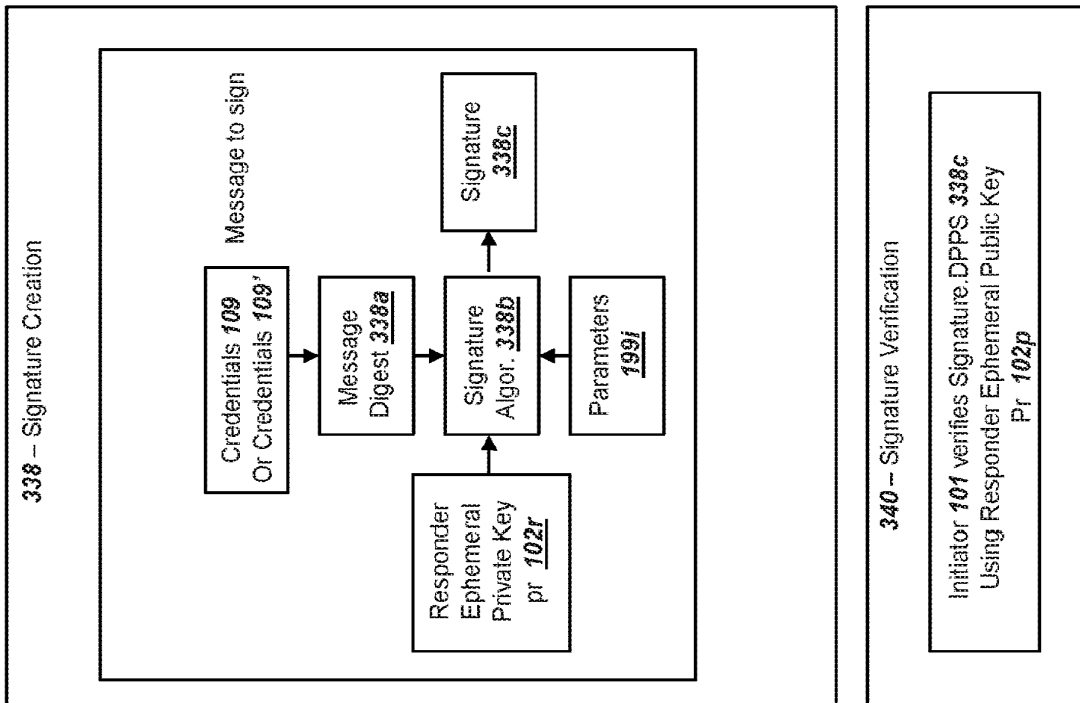


Figure 3d



12/20

Figure 4a

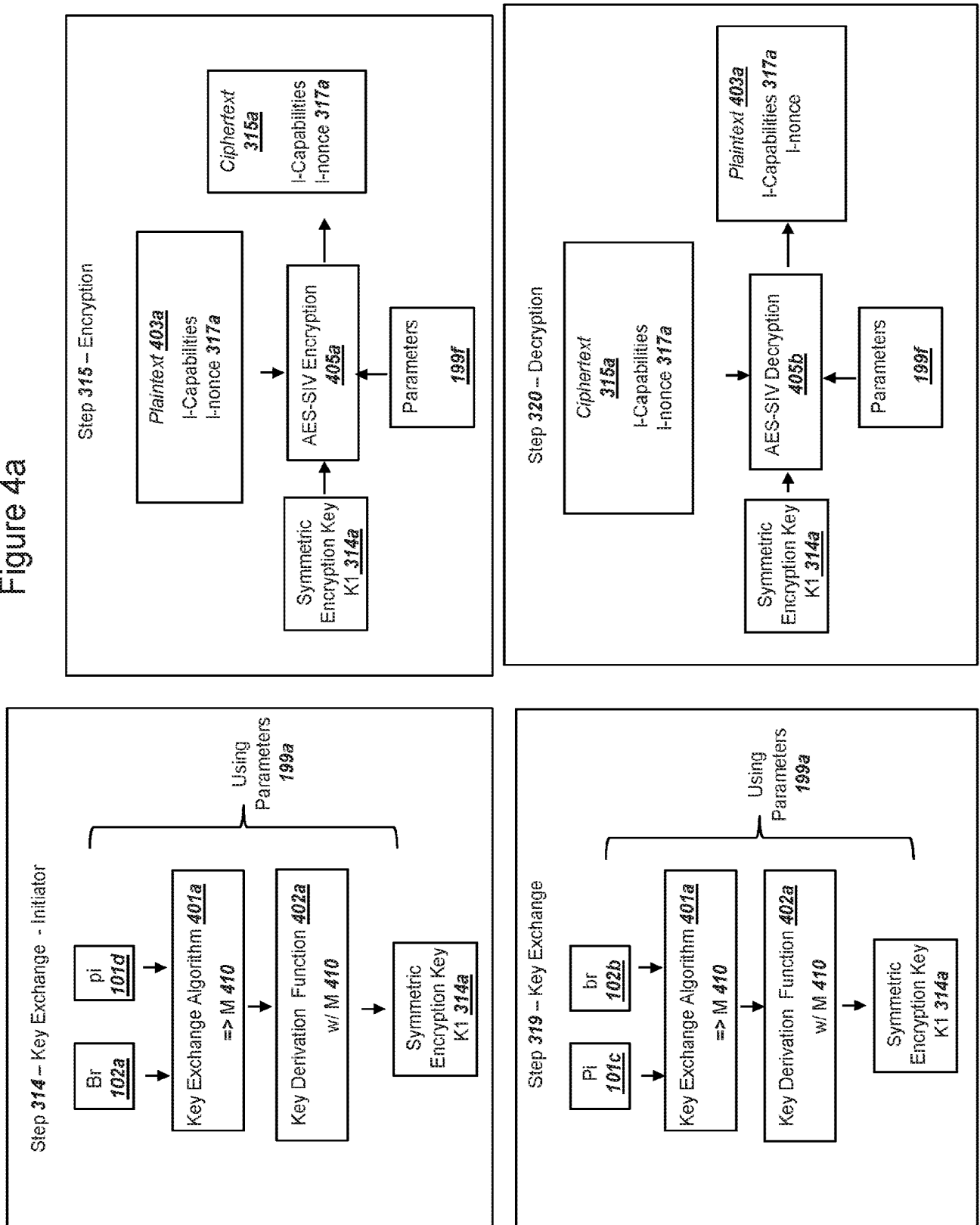


Figure 4b

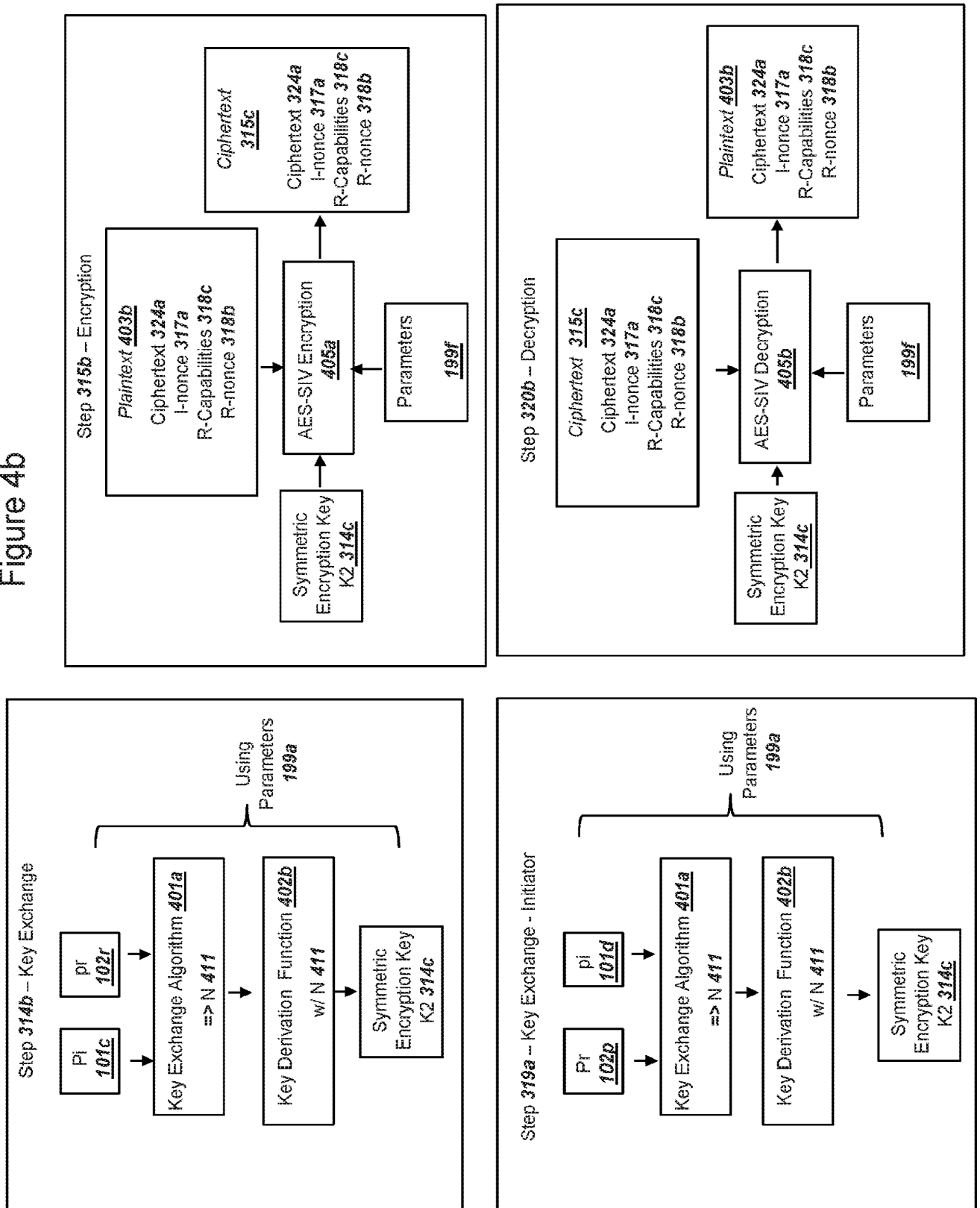


Figure 4c

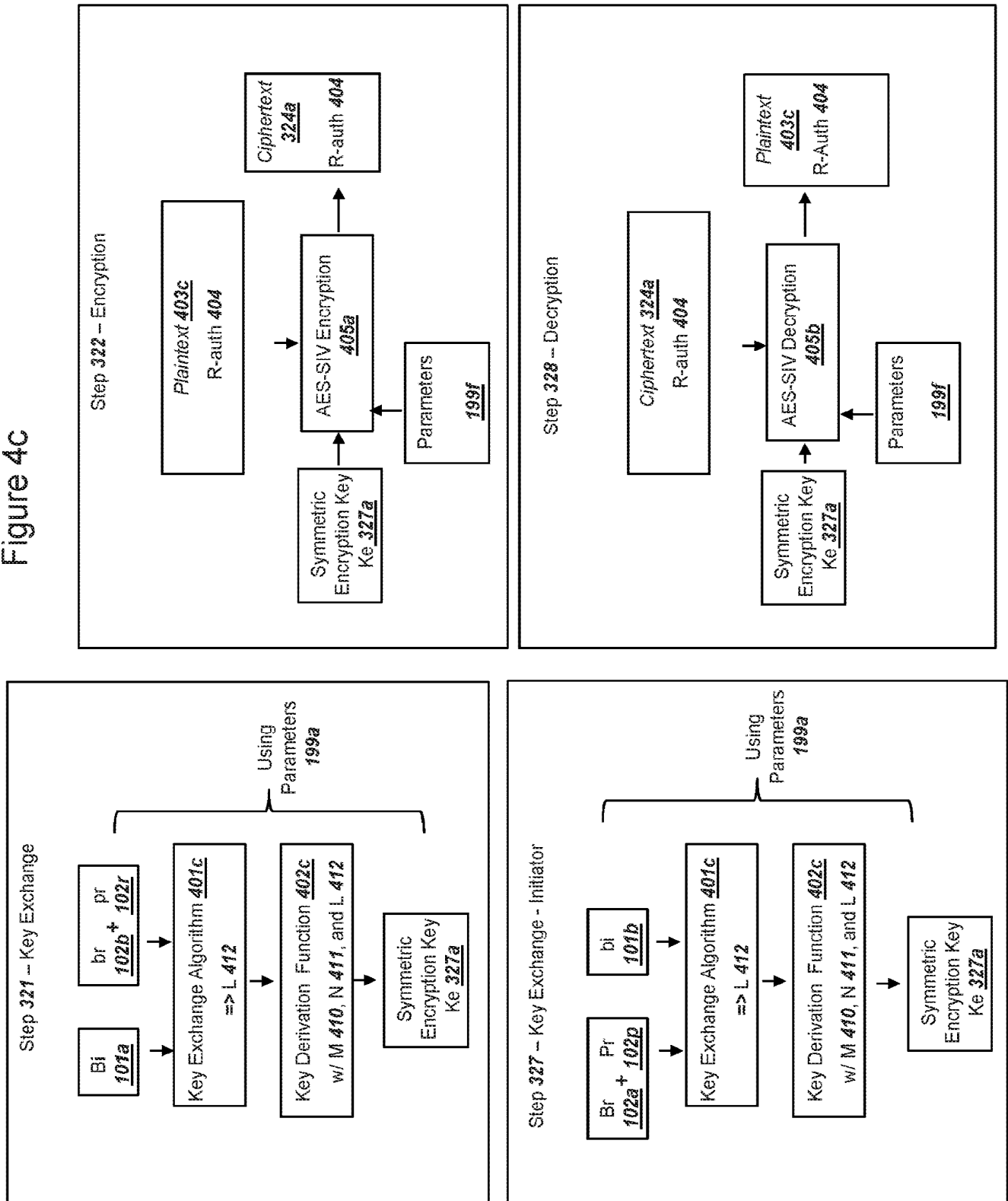


Figure 4d

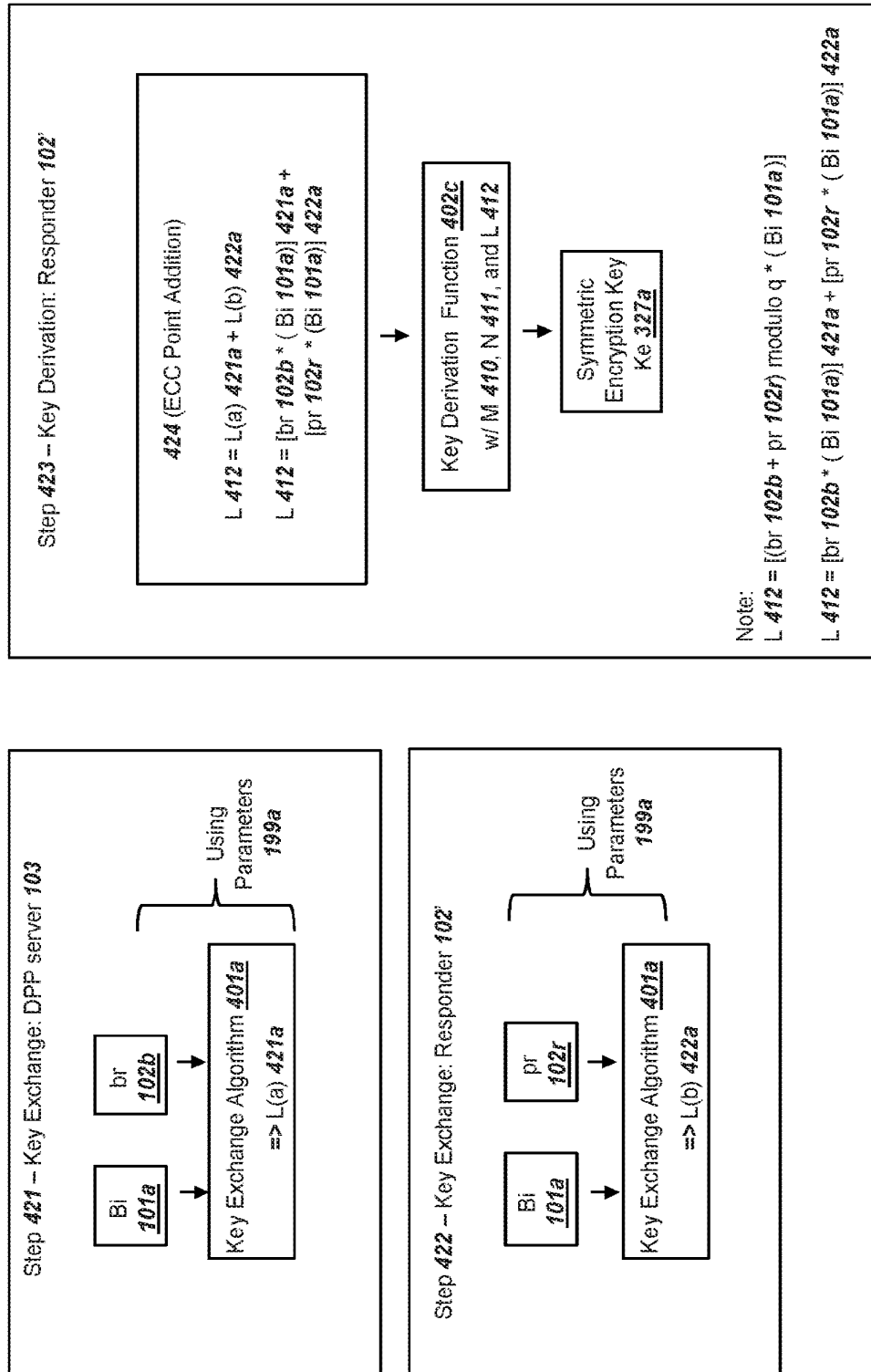


Figure 5a

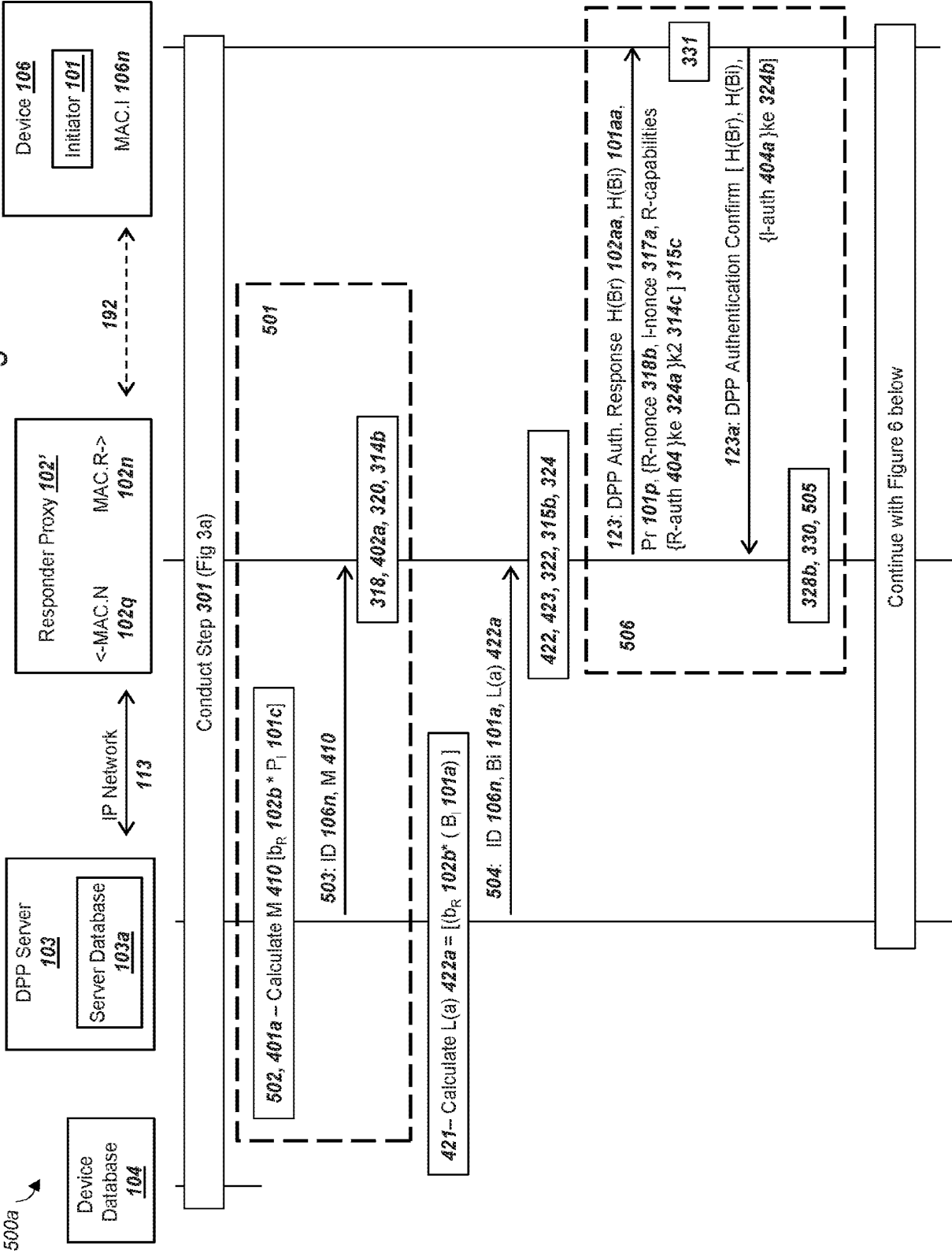


Figure 5b

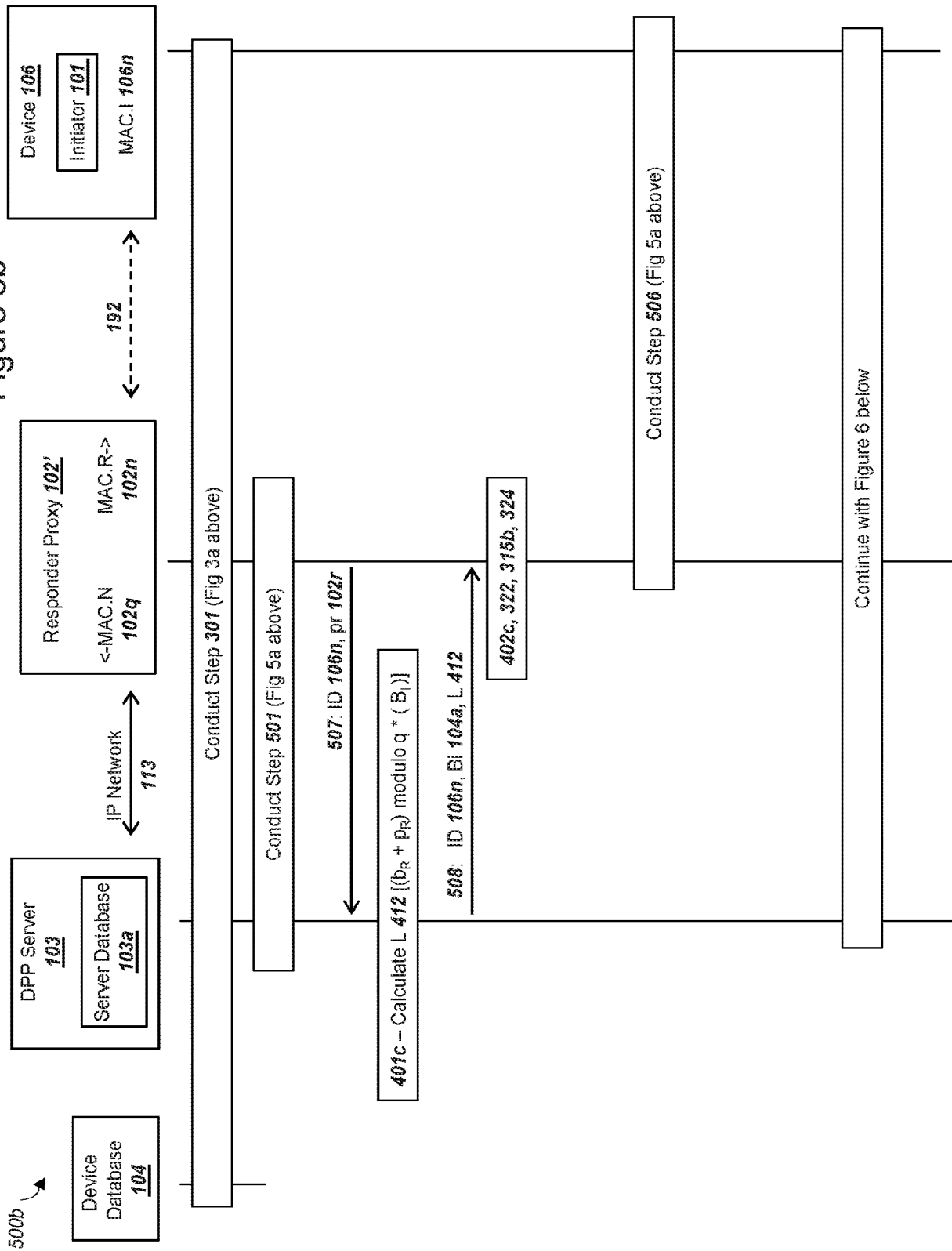


Figure 6

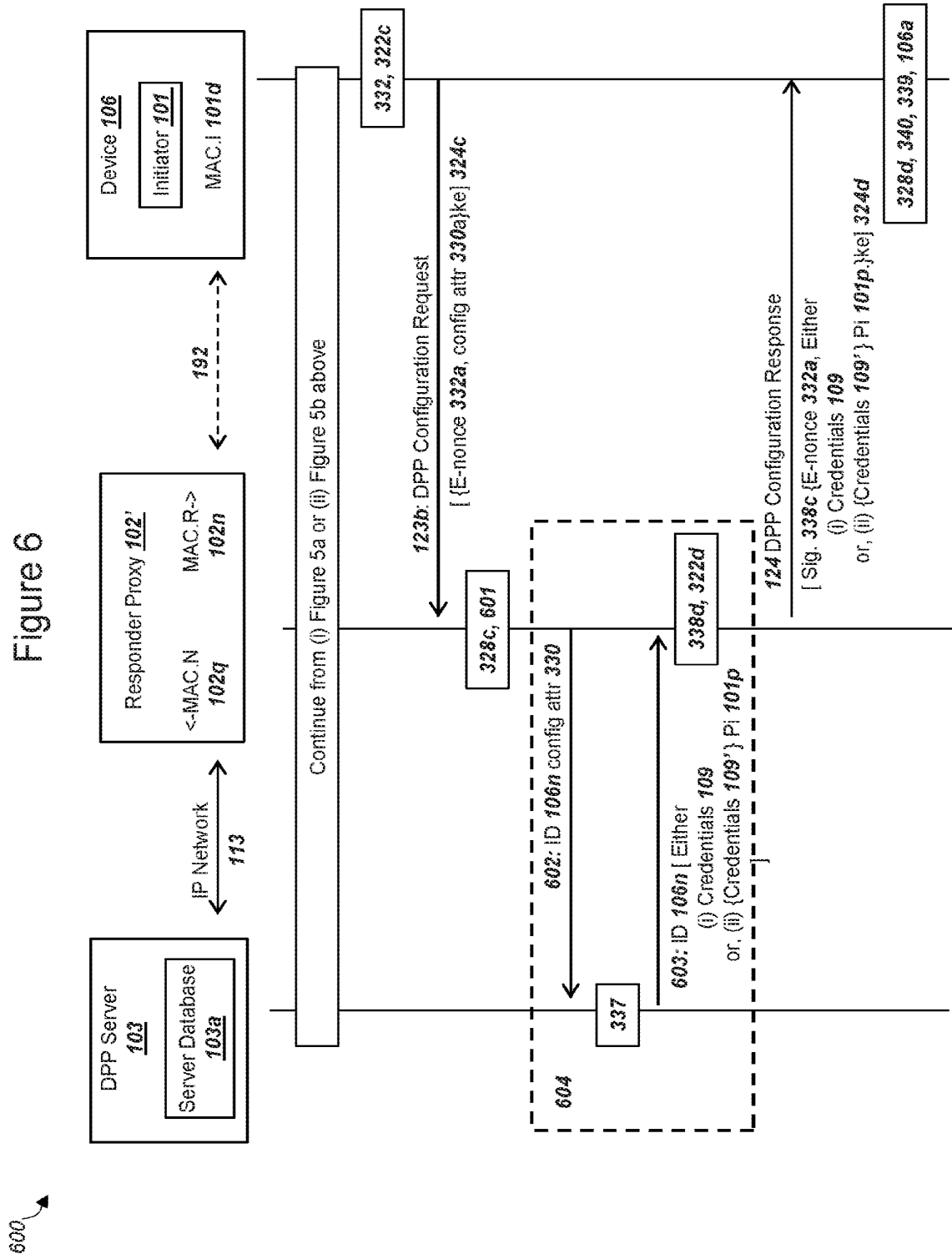


Figure 7

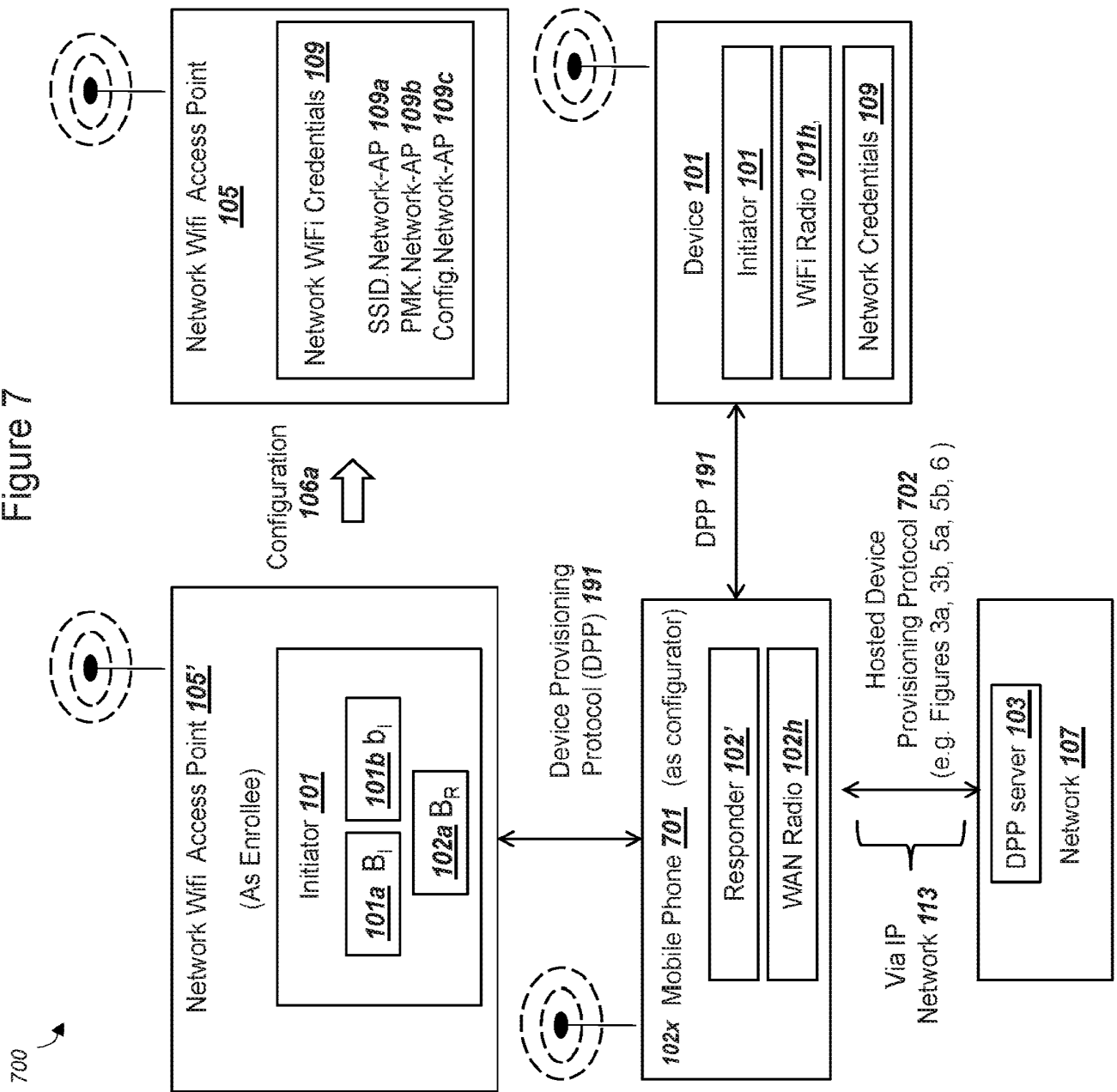
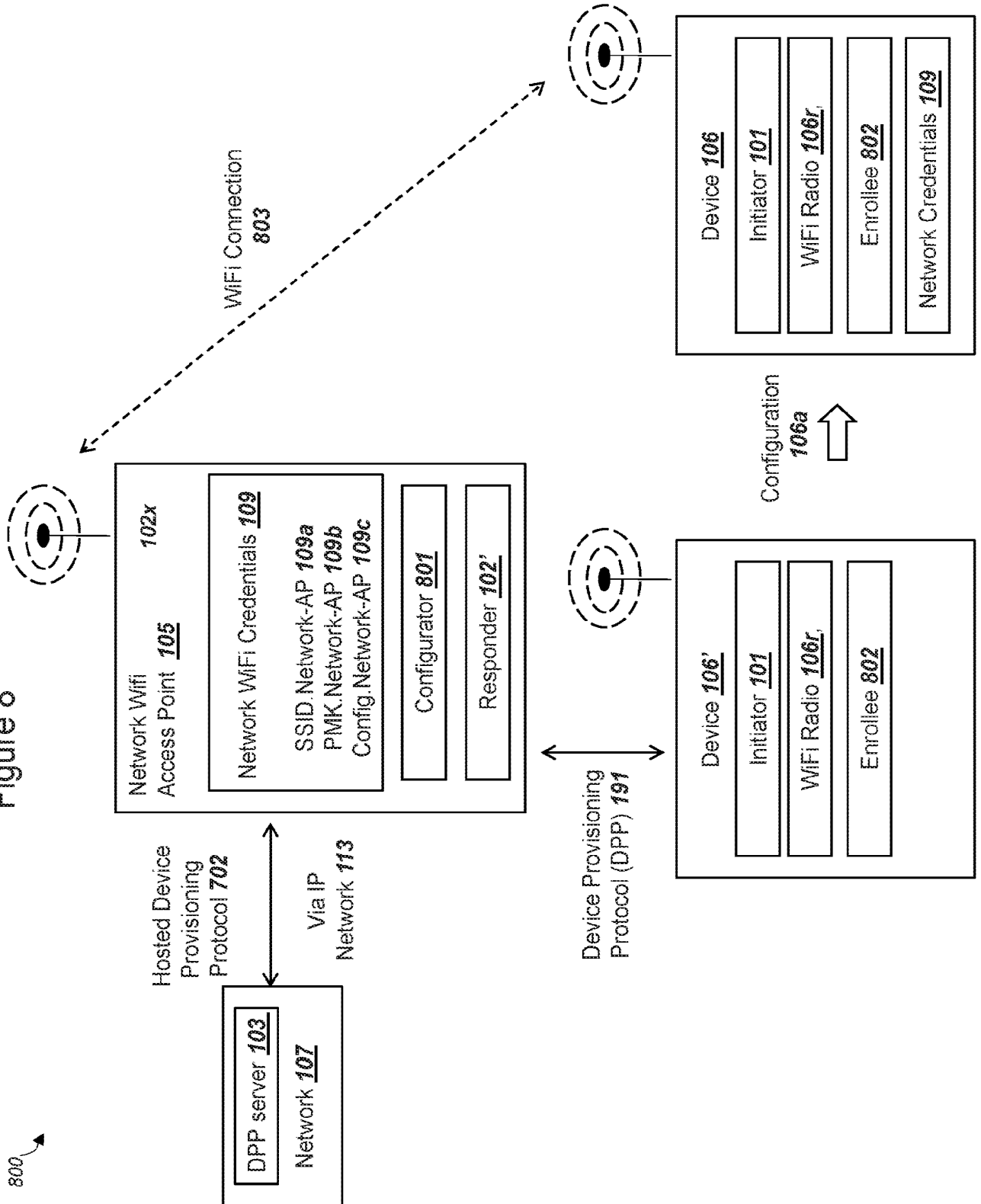


Figure 8



INTERNATIONAL SEARCH REPORT

International application No.

PCT/US 2019/032371

A. CLASSIFICATION OF SUBJECT MATTER		
<p style="text-align: center;"><i>G06F 21/30 (2013.01)</i> <i>G06F 21/62 (2013.01)</i></p> <p>According to International Patent Classification (IPC) or to both national classification and IPC</p>		
B. FIELDS SEARCHED		
Minimum documentation searched (classification system followed by classification symbols)		
G06F H04L		
Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched		
Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)		
PatSearch (RUPTO internal), USPTO, PAJ, K-PION, Esp@cenet, Information Retrieval System of FIPS		
C. DOCUMENTS CONSIDERED TO BE RELEVANT		
Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	Device Provisioning Protocol Specification Version 1.0. Wi-Fi Alliance [online] 2018-04-09 [retrieved on 2019-08-05]. Retrieved from < http://www.sandelman.ca/tmp/ieee/dev-prov.pdf >	1-4, 16-22, 27-30, 42-44, 48-50
A	US 2015/0229475 A1 (QUALCOMM INCORPORATED) 13.08.2015	1-4, 16-22, 27-30, 42-44, 48-50
A	US 2014/0223016 A1 (LEVEL 3 COMMUNICATIONS LLC) 07.08.2014	1-4, 16-22, 27-30, 42-44, 48-50
☐ Further documents are listed in the continuation of Box C. ☐ See patent family annex.		
<p>* Special categories of cited documents:</p> <p>“T” later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention</p> <p>“A” document defining the general state of the art which is not considered to be of particular relevance</p> <p>“X” document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone</p> <p>“E” earlier document but published on or after the international filing date</p> <p>“Y” document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art</p> <p>“L” document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)</p> <p>“O” document referring to an oral disclosure, use, exhibition or other means</p> <p>“P” document published prior to the international filing date but later than the priority date claimed</p> <p>“&” document member of the same patent family</p>		
Date of the actual completion of the international search		Date of mailing of the international search report
26 August 2019 (26.08.2019)		29 August 2019 (29.08.2019)
Name and mailing address of the ISA/RU: Federal Institute of Industrial Property, Berezhkovskaya nab., 30-1, Moscow, G-59, GSP-3, Russia, 125993 Facsimile No: (8-495) 531-63-18, (8-499) 243-33-37		Authorized officer A. Reshetnikova Telephone No. (495) 531-65-15

Form PCT/ISA/210 (second sheet) (January 2015)

INTERNATIONAL SEARCH REPORT

International application No.

PCT/US 2019/032371

Box No. II Observations where certain claims were found unsearchable (Continuation of item 2 of first sheet)

This international search report has not been established in respect of certain claims under Article 17(2)(a) for the following reasons:

1. ☐ Claims Nos.:
because they relate to subject matter not required to be searched by this Authority, namely:
2. ☐ Claims Nos.:
because they relate to parts of the international application that do not comply with the prescribed requirements to such an extent that no meaningful international search can be carried out, specifically:
3. ☒ Claims Nos.: 5-15, 23-26, 31-41, 45-47, 51-54
because they are dependent claims and are not drafted in accordance with the second and third sentences of Rule 6.4(a).

Box No. III Observations where unity of invention is lacking (Continuation of item 3 of first sheet)

This International Searching Authority found multiple inventions in this international application, as follows:

1. ☐ As all required additional search fees were timely paid by the applicant, this international search report covers all searchable claims.
2. ☐ As all searchable claims could be searched without effort justifying additional fees, this Authority did not invite payment of additional fees.
3. ☐ As only some of the required additional search fees were timely paid by the applicant, this international search report covers only those claims for which fees were paid, specifically claims Nos.:
4. ☐ No required additional search fees were timely paid by the applicant. Consequently, this international search report is restricted to the invention first mentioned in the claims; it is covered by claims Nos.:

Remark on Protest

- ☐ The additional search fees were accompanied by the applicant's protest and, where applicable, the payment of a protest fee.
- ☐ The additional search fees were accompanied by the applicant's protest but the applicable protest fee was not paid within the time limit specified in the invitation.
- ☐ No protest accompanied the payment of additional search fees.

Form PCT/ISA/210 (continuation of first sheet (2)) (January 2015)



(51) International Patent Classification:

H04L 9/08 (2006.01) H04W 12/00 (2009.01)
H04L 9/32 (2006.01) G06F 21/62 (2013.01)
H04L 29/06 (2006.01) H04L 9/14 (2006.01)
H04W 12/06 (2009.01) H04W 12/04 (2009.01)
H04W 12/08 (2009.01) H04W 84/12 (2009.01)
H04L 29/08 (2006.01) H04W 88/02 (2009.01)

(21) International Application Number:

PCT/US2019/027968

(22) International Filing Date:

17 April 2019 (17.04.2019)

(25) Filing Language:

English

(26) Publication Language:

English

(30) Priority Data:

62/664,057 27 April 2018 (27.04.2018) US

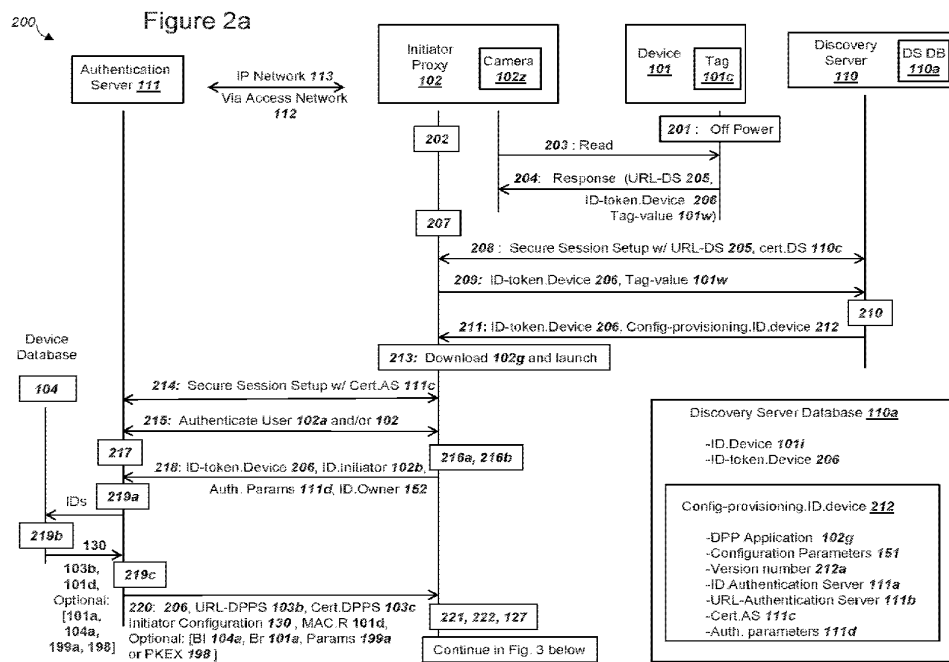
(71) Applicant: **IOT AND M2M TECHNOLOGIES, LLC**
[US/US]; 807 Davis Street, Unit 2207, Evanston, IL
60201-7104 (US).

(72) Inventor: **NIX, John, A.**; c/o Iot and M2M Technolo-
gies, LLC, 807 Davis Street, Unit 2207, Evanston, IL
60201-7104 (US).

(74) Agent: **THOMSON, Kirsten, L.**; McDonnell Boehnen
Hulbert & Berghoff LLP, 300 South Wacker Drive, Chica-
go, IL 60606 (US).

(81) Designated States (unless otherwise indicated, for every
kind of national protection available): AE, AG, AL, AM,
AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ,
CA, CH, CL, CN, CO, CR, CU, CZ, DE, DJ, DK, DM, DO,
DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN,
HR, HU, ID, IL, IN, IR, IS, JO, JP, KE, KG, KH, KN, KP,
KR, KW, KZ, LA, LC, LK, LR, LS, LU, LY, MA, MD, ME,

(54) Title: A HOSTED DEVICE PROVISIONING PROTOCOL WITH SERVERS AND A NETWORKED INITIATOR



(57) **Abstract:** A network can operate a WiFi access point with credentials. An unconfigured device can (i) support a Device Provisioning Protocol (DPP), (ii) record responder bootstrap public and private keys, and (iii) be marked with a tag. The network can record initiator bootstrap public and private keys, as well as derived initiator ephemeral public and private keys. An initiator can (i) operate a DPP application, (ii) read the tag, (iii) establish a secure and mutually authenticated connection with the network, and (iv) send the network data within the tag. The network can record the responder bootstrap public key and derive an encryption key with the (i) recorded responder bootstrap public key and (ii) derived initiator ephemeral private key. The network can encrypt credentials using the derived encryption key and send the encrypted credentials to the initiator, which can forward the encrypted credentials to the device, thereby supporting a device configuration.

[Continued on next page]



MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ,
OM, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SA,
SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN,
TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(84) Designated States (*unless otherwise indicated, for every kind of regional protection available*): ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, ST, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, RU, TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, KM, ML, MR, NE, SN, TD, TG).

Published:

— with international search report (Art. 21(3))

A HOSTED DEVICE PROVISIONING PROTOCOL WITH SERVERS AND A NETWORKED INITIATOR

CROSS-REFERENCE TO RELATED APPLICATIONS

5 This international PCT application claims the benefit of the filing date of U.S. Provisional Patent Application Serial No. 62/664,057, filed on April 27, 2018, which is hereby incorporated by reference in its entirety.

BACKGROUND

Technical Field

10 The present systems and methods relate to authentication and configuration of WiFi radios, and more particularly, to using a Device Provisioning Protocol with a networked initiator and a set of servers that record PKI keys, in order securely transfer a set of credentials to a device.

Description of Related Art

15 The ability to connect transducers such as sensors and actuators with a network is a growing field with many economical applications. As the costs for both electronic hardware and bandwidth continue to decrease, the use of networked transducers is expected to continue increasing over the coming decades. Connecting transducers to a network can be referred to as “machine-to-machine (M2M) communications” or “the Internet of Things (IoT).” Among many potential benefits, IoT technologies allow automated monitoring and/or control of assets, equipment, personnel, or a physical location where manual monitoring may not be economical. Many applications for the “Internet of Things” significantly reduce costs by using automated monitoring instead of manual techniques. Prominent examples of IoT applications today include monitoring and control for building heating/air-conditioning, automobiles, alarm systems, and tracking devices. Fast growing markets for IoT applications today include health applications such as the remote monitoring of a person’s fitness activity, heartbeat, or glucose levels, monitoring of industrial equipment deployed remotely in the field, and also security systems.

20 Many IoT applications can leverage wireless networking technologies, in addition to wired technologies such as Ethernet. Wireless technologies such as wireless local area networks and wireless wide area networks have proliferated around the world over the past 30 years, and usage of these wireless networks is also expected to continue to grow. Wireless local area network (LAN) technologies include WiFi based on the series of 802.11 standards from the Institute of Electrical and Electronics Engineers (IEEE). The many options to connect a transducer device to a wireless network creates opportunities for new products and services, but also creates several classes of

problems that need to be solved. Many of these problems or needs in the art pertain to securely and efficiently configuring the transducer devices. A need exists in the art to allow a user or a network to securely and easily upload to the device a set of network access credentials for a wireless network.

Manually configuring devices to (i) use different security keys, (ii) apply new network access credentials to connect with the owner's selected wireless network, and/or (iii) update running firmware can be time consuming and difficult for users. Entering network access credentials is also prone to errors and especially difficult for devices with limited user interfaces, where a limited user interface is common for many devices supporting applications for the "Internet of Things". Loading credentials and network configuration into a device in order to obtain connectivity to a network can also be known as device provisioning or device configuration. In an attempt to address these needs to simplify or automate device provisioning, the WiFi Alliance™ released a Device Provisioning Protocol (DPP) specification on April 9, 2018 as version 1.0 (DPPv1.0). Although the DPPv1.0 defines a series of messages between an initiator and a responder in order to support a device configuration, several unmet needs remain in the art in order to securely and easily deploy a Device Provisioning Protocol at wide scale with potentially billions of devices over many years.

To highlight one example unmet need by a DPPv1.0, WiFi capable devices currently being designed and deployed may have an expected life of more than a decade. During that timeframe, the device may need to undergo a configuration or device provisioning step more than one time. Reasons could include either (i) the device undergoes a "factory reset", or (ii) the device needs to be provisioned a second time, such as if the device owner changes or the device is moved to a different network, and other reasons exist as well. But, as stated in DPPv1.0, "threats against this bootstrapping method are possible if public keys and/or codes are reused." A need exists in the art to support a Device Provisioning Protocol in a manner where bootstrapping public keys can be securely reused. A need exists in the art for bootstrapping PKI keys to be securely updated after a first instance of conducting a DPP, such that the updated bootstrapping PKI keys can be subsequently used at a later time for a second instance of conducting a DPP.

In order to securely provision a device, a device may prefer to conduct mutual authentication with an initiator that records and operates with initiator PKI keys. The device could be a high value piece of equipment, such as an automobile, health monitoring equipment in a hospital, or industrial equipment, etc. that could be targets for potential hackers seeking to "provision" the devices to a network under their control or a network providing the hackers access to the device. In these cases, an owner or user of a device may prefer that a device provisioning protocol use mutual authentication. The DPPv1.0 supports mutual authentication in order to securely authenticate a device and an initiator before transferring network access credentials to the device, but creates a new and shifted problem of both (i) securely providing an initiator bootstrap private key to an initiator and (ii) the corresponding initiator bootstrap public key to a device. In other words, recording an

initiator bootstrap public key in a device can be difficult if there is no pre-existing relationship or communication between a device and an initiator before the start of a device provisioning protocol. In addition, an insecure initiator may have no pre-existing relationship with a target network to provide connectivity to the device, yet the network may need to securely transfer credentials to the device through the insecure initiator. A need exists in the art to securely and readily enable an initiator to have access to an initiator bootstrap private key while also enabling a device to have access to the corresponding initiator bootstrap public key. A need exists in the art for a network to securely transfer credentials to a device through an insecure initiator.

With the widespread deployment of WiFi access points, a device may have access to potentially a plurality of different access points associated with potentially different networks. Although DPPv1.0 supports configuration of a device, DPPv1.0 does not suggest (i) which of a possible plurality of potential different access points should be selected, nor (ii) how credentials for a selected access point could be obtained by an initiator. Need exists in the art for an initiator to automatically determine a list of available networks and to automatically and securely receive a set of credentials for a preferred network for the device. A need exists in the art for the credentials to be encrypted in a manner such that the initiator cannot feasibly read the credentials, but the device can receive the encrypted credentials and convert the encrypted credentials into plaintext form in order for the device to load and use the plaintext credentials.

A network can use DPPv1.0 in order to connect devices to the network using an initiator. The security for conducting a device provisioning protocol according to DPPv1.0 depends on the security of an initiator, since as envisioned by DPPv1.0 the initiator (i) records an initiator bootstrap private key and (ii) conducts key exchanges with the initiator bootstrap private key. However, an initiator for conducting a DPP may be outside the control of a network and may also be an insecure device. An initiator could be a “rooted” mobile phone operated by a hacker, as one example. Or, the initiator could simply be a mobile device that is insecure or operates with security policies that do not meet the standards of the network. A need exists in the art for an insecure initiator to conduct a device provisioning protocol that is compatible with DPPv1.0, while simultaneously keeping (i) the initiator bootstrap private key and (ii) cryptographic operations with the initiator bootstrap private key under the control or within the security policies of a network.

The secure use of a device provisioning protocol can depend on an initiator having access to a responder bootstrap public key. Access to the responder bootstrap public key for the initiator is described in DPPv1.0 as supported in an “out of band” operation. The security of a device provisioning protocol can depend on the security of the steps for an initiator to gain access to the responder bootstrap public key. The initiator could also comprise an insecure device, as described in the paragraph above. If the responder bootstrap public key is externally readable by an initiator, such as contained in a QR code, then potentially many different initiators could read the responder

bootstrap public key and attempt a DPP with a device. Although this ease may be preferred for some devices and applications, many devices may require a higher level of security such that a responder bootstrap public key can only be accessed by an authorized initiator. A need exists in the art for an initiator to have access to a responder bootstrap public key, while simultaneously keeping
5 the responder bootstrap public key secured.

Many other examples exist as well for needs in the art to support efficient yet secure device configuration or provisioning, and the above are examples are just a few and intended to be illustrative instead of limiting.

SUMMARY

Methods and systems are provided for a networked initiator to conduct a device provisioning protocol (DPP) with a responder operating in a device. The device provisioning protocol can support the Device Provisioning Protocol specification version 1.0 and subsequent versions from the WiFi Alliance™. Conducting a DPP with a device can transfer a set of network credentials to the device, in order for the device to connect with a network using the network credentials. The set of network credentials can also include a network configuration, where the network configuration includes settings to use with a WiFi radio in the device, in order for the WiFi radio to communicate with other nodes with the credentials, including an access point which can provide connectivity to an IP network for the device.

A device can operate a responder, which listens for an authentication request from an initiator. The responder can record and operate an initiator configuration, specifying a frequency band to utilize, a WiFi networking technology to implement (e.g. 802.11 ac, 802.11ah, etc.) and a channel list. When the device is awake or powered on, the responder can listen for incoming authentication requests using a WiFi radio for the device. The responder can record a responder bootstrap public key, a responder bootstrap private key, and, when supporting mutual authentication with an initiator, an initiator bootstrap public key. The PKI keys can be associated with a selected set of cryptographic parameters, where the parameters specify settings to use for cryptographic operations by a responder. The responder can also include a random number generator, a PKI key pair generation algorithm, a secure hash algorithm, an elliptic curve Diffie Hellman (ECDH) key exchange algorithm, and a symmetric ciphering algorithm. The device with the responder can be marked with a tag, such that the tag records an identity for the device and a uniform resource locator (URL) for a server.

A network can operate an access point, where the access point operates with a set of credentials. The credentials can include an identifier for the network such a service set identifier (SSID), a pairwise master key (PMK) or a preshared key (PSK), a pairwise master key identifier (PMKID), and also a configuration. The access point can support wireless networking standards in the series of 802.11 standards. A network can also operate a plurality of access points. The network can operate or be associated with a Device Provisioning Protocol (DPP) server, a device database, an authentication server, and a discovery server. The network can be connected to an IP network, which can comprise the globally routable public Internet. Different servers within a network can be connected using an Intranet, a virtual private network (VPN), Internet Protocol Security (IPSEC), or similar techniques to secure communication between the servers.

The network can operate or communicate with the device database. The DPP server in the network can also communicate with the device database. The device database can record information for a plurality of devices before responders in the devices conduct the device provisioning protocol with the initiator. The device database can record the device identity, a responder identity, which could be in the form of a responder medium access control (MAC) address, an initiator configuration, and a key table. The key table in a device database can record for each device a responder bootstrap public key, an initiator bootstrap public key, and an initiator bootstrap private key. Recorded data within a device database can be obtained by a network (i) upon device manufacturing from a device manufacturer, or (ii) before distribution of a device to the location of the network's access point.

The DPP server for a network can operate a server database, where the server database records information for a plurality of devices connecting with the network. The server database can record the device identity, medium access control (MAC) addresses for devices and initiators, an initiator mode, a selected set of cryptographic parameters for each device, an initiator bootstrap private key, the initiator bootstrap public key, the responder bootstrap public key, an initiator ephemeral public key, and initiator ephemeral private key, an initiator bootstrap key sequence number to identify and track the initiator bootstrap PKI keys, a public key exchange (PKEX) key, a first, second, and third derived shared secret key for a responder. Depending on the initiator mode, not all PKI keys and secret shared keys may be recorded in the server database. The DPP server can also include a set cryptographic algorithms, where the cryptographic algorithms can include a random number generator, a PKI key pair generation algorithm, a secure hash algorithm, an elliptic curve Diffie Hellman key exchange algorithm, and a symmetric ciphering algorithm. The cryptographic algorithms within a DPP server can be compatible with the equivalent set of cryptographic algorithms in the responder and also the initiator.

An initiator can be a computing device that includes a WiFi radio and can be associated with an initiator user. The initiator can operate the WiFi radio with an initiator configuration in order for the initiator to send and receive messages with the responder. The initiator is located in sufficient proximity with the responder so that the WiFi radio for the initiator and the WiFi radio for the responder can communicate using 802.11 standards. The initiator can operate a DPP application, where the DPP application can also include the set of cryptographic algorithms, which are compatible with cryptographic algorithms in both the responder and the DPP server. The initiator can use an access network in order to communicate with the network via an IP network. The initiator can record a user configuration for the WiFi radio, where the user configuration can be temporarily replaced with the initiator configuration in order to conduct the device provisioning protocol (DPP) session with the responder. After the DPP session, the initiator configuration can be

replaced with the user configuration, such that the initiator is restored to its previous state before the DPP session.

The discovery server for a network can operate a discovery server database. The discovery server can receive requests from initiators, where the requests include a device identity from the tag of the device. The discover server database can record configuration provisioning data for the device with the device identity, and a URL for an authentication server for use by the initiator in order for the initiator to (i) authenticate with the network before conducting the device provisioning protocol and (ii) receive information from the DPP server. The initiator can send a request to the discovery server with the device identity. The discovery server can query the discovery server database with the device identity in order to respond to the initiator with the configuration provisioning data for the initiator, where the configuration provisioning data includes the URL for the authentication server.

The initiator can call the URL for the authentication server and conduct an authentication step with the authentication server. After authentication of the initiator, the authentication server can query the device database in order to obtain information for the initiator to use in order to conduct a device provisioning protocol. The information for the initiator can include a URL for the DPP server, an initiator configuration, and optionally the initiator bootstrap public key and the responder bootstrap public key, where the optional information can depend on the initiator mode for a DPP selected by the network.

The initiator can call the URL for the DPP server and setup a secure session via TLS or similar standards. The initiator can then use the WiFi radio and wireless WAN radio to conduct a radio frequency scan in order to collect a networks available list. The networks available list can include identities broadcast by wireless networks and available at the physical location of the initiator near the device. The initiator can send the networks available list to the DPP server. The DPP server can use the networks available list to select a primary access network for the device and obtain credentials for the primary access network for the device to use. The DPP server can also select an initiator mode, which can specify the subsequent series of steps for the initiator to take in order to conduct a DPP session with the responder. The DPP server can send the initiator the initiator mode, where a DPP application operating in the initiator implements the mode. The DPP server can obtain the initiator bootstrap PKI keys and the responder public key from the device database, as well as a selected set of cryptographic parameters in order to use the PKI keys.

In a first initiator mode, the DPP server can then conduct a series of steps in order to generate data for a Device Provisioning Protocol (DPP) authorization request. The DPP server can derive an initiator ephemeral PKI key pair. The DPP server can conduct a first initiator ECDH key exchange with the responder bootstrap public key and the derived initiator private key to derive a key k1. The initiator can derive an initiator nonce and encrypt the initiator nonce using the derived

key k_1 , where the encrypted initiator nonce can comprise a first ciphertext. The initiator can send (i) secure hash values for the initiator bootstrap public key and the responder bootstrap public key, the (ii) derived initiator ephemeral public key, and (iii) the first ciphertext to the initiator. The initiator can send a DPP authentication request to the responder.

5 The responder can receive and process with DPP authentication request. The responder can record a plurality of initiator bootstrap public keys, where the plurality of initiator bootstrap public keys could be recorded during manufacturing or distribution of the device with the responder before the device was placed in the physical proximity of the network access point. At least one of the recorded initiator bootstrap public keys can be selected based on the hash value for the key received
10 in the DPP authentication request. The responder can generate a responder ephemeral PKI key pair. The responder can conduct a first responder key exchange using the recorded responder bootstrap private key and the received initiator ephemeral public key in order to derive the key k_1 . The responder can conduct a second responder key exchange using the derived responder ephemeral private key and the received initiator ephemeral public key in order to derive a key k_2 . The
15 responder can conduct a third key exchange using the selected initiator bootstrap public key and a combination of (i) the responder ephemeral private key and the responder bootstrap private key in order to derive a key k_e . The responder can decrypt the first ciphertext using the key k_1 and read the plaintext initiator nonce. The responder can calculate a responder authentication value using the initiator nonce and encrypt the value using the key k_e , creating a second ciphertext. The responder
20 can generate a responder nonce and encrypt both the responder nonce and the second ciphertext using the key k_2 , creating a third ciphertext. The responder can send a DPP authentication response to the initiator, where the response includes the third ciphertext and the responder ephemeral public key.

The initiator can receive the DPP authentication response from the responder. The initiator
25 can send the third ciphertext and the responder ephemeral public key to the DPP server, along with an identity for the device. The DPP server can conduct a second initiator key exchange using the responder ephemeral public key and the initiator ephemeral private key in order to derive the key k_2 . The DPP server can conduct a third initiator key exchange using (i) a combination of the responder bootstrap public key and the responder ephemeral public key and (ii) the initiator bootstrap private
30 key in order to derive the key k_e . The DPP server can decrypt the third ciphertext using the derived key k_2 . The DPP server can read the plaintext responder nonce and the second ciphertext. The DPP server can decrypt the second ciphertext using the derived key k_e . The DPP server can read the plaintext responder authentication value from the second ciphertext. The DPP server can calculate the responder authentication value, and if the received responder authentication value matches the
35 calculated responder authentication value, then the responder is authenticated.

The DPP server can calculate an initiator authentication value using the received responder nonce. The DPP server can encrypt the initiator authentication value using the key ke , thereby creating a fourth ciphertext. The DPP server can send the fourth ciphertext to the initiator. The initiator can send a DPP authentication confirm message to the responder with the fourth ciphertext.

5 The responder can decrypt the fourth ciphertext using the derived key ke and read the plaintext initiator authentication value. The responder can calculate the same initiator authentication value, and if the received authentication value matches the calculated authentication value then the initiator is authenticated with the responder.

After the mutual authentication, the responder can take the roll of an enrollee and the

10 initiator can take the roll of a configurator. The responder can generate an enrollee nonce and also a configuration attribute, where the configuration attribute describes WiFi capabilities for the device. The responder can create a fifth ciphertext by encrypting the enrollee nonce and the configuration attribute with the derived key ke . The responder can send a DPP configuration request to the initiator, where the DPP configuration request can include the fifth ciphertext. The initiator can send

15 the fifth ciphertext to the DPP server.

The DPP server can receive the fifth ciphertext from the initiator and decrypt the fifth ciphertext using the derived key ke , in order to read the plaintext enrollee nonce and the configuration attribute. Using the selected primary network from above and the configuration attribute, the DPP server can process the selected network credentials. The selected network

20 credentials were described above and can be obtained by a DPP server using the networks available list from the initiator. The DPP server can encrypt the received enrollee nonce and the network credentials using the derived key ke , creating a sixth ciphertext. The DPP server can send the sixth ciphertext to the initiator. The initiator can forward the sixth ciphertext to the responder in a DPP configuration response message.

25 The responder can receive the DPP configuration response message and decrypt the sixth ciphertext using the derived key ke in order to read the plaintext network credentials. The responder can pass the network credentials to the device, which can apply the credentials with the WiFi radio for the device. The device can subsequently connect with the network access point using the received network credentials. The network access point can provide connectivity to an IP network

30 for the device. The device can establish a secure session with the DPP server using the network access point. The DPP server can derive a new initiator bootstrap PKI key pair and send the new initiator bootstrap public key to the device via the secure session. The device can record the new initiator bootstrap public key for use in a subsequent DPP session, if necessary, for a later time. The DPP server can send the device database the new initiator bootstrap PKI keys. The DPP server and

35 the device can record that the DPP session is successfully completed.

These as well as other aspects and advantages will become apparent to those of ordinary skill in the art by reading the following detailed description, with reference where appropriate to the accompanying drawings.

BRIEF DESCRIPTION OF THE DRAWINGS

Various exemplary embodiments are described herein with reference to the following drawings, wherein like numerals denote like entities.

5 Figure 1a is a graphical illustration of an exemplary system, where an initiator communicates with a network and a responder in a device, in order to conduct a device provisioning protocol, in accordance with exemplary embodiments;

Figure 1b is a graphical illustration of a device provisioning protocol for authentication and configuration of a responder, in accordance with conventional technology;

10 Figure 1c is a graphical illustration of a device provisioning protocol for (i) authentication and configuration of a responder and (ii) authentication of an initiator, in accordance with conventional technology;

Figure 1d is a graphical illustration of hardware, firmware, and software components for an initiator, including an initiator configuration step, in accordance with exemplary embodiments;

15 Figure 2a is a simplified message flow diagram illustrating an exemplary system with exemplary data sent and received by an initiator, in accordance with exemplary embodiments;

Figure 2b is an illustration of an exemplary device database, with tables for a device database recording exemplary data, in accordance with exemplary embodiments;

20 Figure 3a is a simplified message flow diagram illustrating an exemplary system with exemplary data sent and received by a device database, a DPP server, an initiator proxy, and a responder, in accordance with exemplary embodiments;

Figure 3b is an illustration of an exemplary server database, with tables for a server database recording exemplary data, in accordance with exemplary embodiments;

25 Figure 4a is a flow chart illustrating exemplary steps for conducting a key exchange using PKI keys in order to derive a shared secret key, and for using the derived shared secret key to encrypt and decrypt data, in accordance with exemplary embodiments;

Figure 4b is a flow chart illustrating exemplary steps for conducting a key exchange using PKI keys in order to derive a shared secret key, and for using the derived shared secret key to encrypt and decrypt data, in accordance with exemplary embodiments;

30 Figure 4c is a flow chart illustrating exemplary steps for conducting a key exchange using PKI keys in order to derive a shared secret key, and for using the derived shared secret key to encrypt and decrypt data, in accordance with exemplary embodiments;

35 Figure 5a is a simplified message flow diagram illustrating an exemplary system with exemplary data sent and received by a device database, a DPP server, an initiator proxy, and a responder, in accordance with exemplary embodiments;

Figure 5b is a simplified message flow diagram illustrating an exemplary system with exemplary data sent and received by a device database, a DPP server, an initiator proxy, and a responder, in accordance with exemplary embodiments;

Figure 6a is a simplified message flow diagram illustrating an exemplary system with exemplary data sent and received by a device database, a DPP server, a WiFi access point, and a device, in accordance with exemplary embodiments;

Figure 6b is a graphical illustration of an exemplary system, where a responder records a plurality of initiator bootstrap public keys and the responder selects and uses an initiator bootstrap public key, in accordance with exemplary embodiments;

Figure 7 is a graphical illustration of an exemplary system, where an access point conducts a configuration step for a device using a hosted device provisioning protocol, in accordance with exemplary embodiments; and,

Figure 8 is a graphical illustration of an exemplary system, where an access point operates as an initiator for a device with a responder, in order transfer a set of network credentials to the device, in accordance with exemplary embodiments.

DETAILED DESCRIPTION OF EXEMPLARY EMBODIMENTS OF THE INVENTION**Figure 1a**

Figure 1a is a graphical illustration of an exemplary system, where an initiator communicates with a network and a responder in a device, in order to conduct a device provisioning protocol, in accordance with exemplary embodiments. The system 100 can include an initiator 102, a device 101, a network 107, and a discovery server 110. In exemplary embodiments, initiator 102 can communicate with network 107 and discovery server 110 through an access network 112, which provides communication to an Internet Protocol (IP) network 113. Initiator 102 can also be associated with an initiator user 102u, which could be an end user, group of people, or business entity controlling or operating initiator 102. Initiator 102 could comprise a computing device that operates a radio and processor in order to transmit and receive wireless data. Additional details for components within an initiator 102 are provided below in Figure 1d. In some exemplary embodiments, initiator 102 could comprise a mobile phone or “smart phone” with an operating system such as Android ® from Google ® or IOS® from Apple ®. In other exemplary embodiments, an initiator 102 could comprise a portable device, such as laptop, tablet, wearable device such as a smart watch, a USB device including a “USB memory stick”, etc. In exemplary embodiments, the function of initiator 102 could be combined with network WiFi access point 105 (such as depicted and described in connection with Figure 8 below), and a separate physical unit for initiator 102 could be omitted for some embodiments of a system 100.

Initiator 102 could operate as an initiator proxy 102 and send and receive messages with device 101 according to a device provisioning protocol (DPP) 191 as specified in the “Device Provisioning Protocol Specification version 1.0” from the WiFi Alliance™ (DPPv1.0), which is hereby incorporated by reference in its entirety. Subsequent and related versions of a DPP 191 could be supported as well by device 101 and an initiator 102. A summary of the message flows between initiator 102 and device 101 according to the DPP 191 are depicted and described in connection with Figure 1b and Figure 1c below, as well as subsequent figures herein. The message flows for a DPP 191 between initiator 102 and device 101 could be via a WiFi data link 192, where the WiFi data link 192 can use a configuration 130. Initiator proxy 102 could operate a camera 102z and a WiFi radio as depicted, where the WiFi radio could record a medium access control (MAC) address MAC.I 102n. MAC.I 102n for initiator 102 could comprise a string of octets to identify initiator 102 with any of a WiFi access point 105, device 101, network 107, and discovery server 110. In order to acquire the data to function as an initiator according to a DPP 191 and/or DPPv1.0, initiator 102 in Figure 1a can communicate and transmit/receive data from network 107 via access network 112.

Additional details regarding the electrical components within an initiator 102 are depicted and described in connection with Figure 1d below.

Embodiments of the present invention contemplate both (i) some of the functionality of an initiator in the DPP specification DPPv1.0 operates within initiator proxy 102 depicted in Figure 1a, and (ii) other functionality of an initiator according to the DPP protocol in DPPv1.0 resides within network 107. Depictions and descriptions below in the present invention may utilize the term “initiator 102”, which can also comprise an “initiator proxy 102” depicted in Figure 1a. In other words, (x) from the perspective of device 101, initiator proxy 102 can operate as an initiator according to the DPP Specification DPPv1.0, but (y) in the present invention some of the functionality and data for an initiator as contemplated by the DPP Specification remains in the network 107. This configuration of operating an initiator proxy 102 in a system 100 (along with proper operation of a network 107) can address and solve the needs in the art described above. Initiator proxy 102 can communicate with network 107 via access network 112.

Access network 112 could be either a Local Area Network (LAN) or a Wide Area Network (WAN), or potentially a combination of both. Access network 112 could comprise a network supporting either IEEE 802.11 (WiFi) standards. Initiator 102 and access network 112 could also utilize a variety of WAN wireless technologies to communicate, including Low Power Wide Area (LPWA) technology, 3rd Generation Partnership Project (3GPP) technology such as, but not limited to, 3G, 4G Long-Term Evolution (LTE), or 4G LTE Advanced, NarrowBand – Internet of Things (NB-IoT), LTE Cat M, proposed 5G networks, and other examples exist as well. A wired initiator 102 can connect to the access network 112 via a wired connection such as, but not limited to, an Ethernet, a fiber optic, or a Universal Serial Bus (USB) connection (not shown). In some exemplary embodiments, access network 112 for initiator proxy 102 can be provided by access point 105, where initiator proxy 102 may also record a compatible set of network credentials 109.

IP network 113 could be a public or private network supporting Internet Engineering Task Force (IETF) standards such as, but not limited to, such as, RFC 786 (User Datagram Protocol), RFC 793 (Transmission Control Protocol), and related protocols including IPv6 or IPv4. A public IP network 113 could utilize globally routable IP addresses, and a private IP network 113 could utilize private IP addresses which could also be referred to as an Intranet. Other possibilities for IP Network 113 exist as well without departing from the scope of the invention. Although not depicted in Figure 1a, elements within a system 100 could each operate or be associated with a firewall, such that packets from IP network 113 are filtered or restricted in order to secure the communication between the nodes.

Device 101 can comprise a computing device with a WiFi radio for transmitting and receiving data such as via WiFi data link 192. Device 101 can include a tag 101c and a WiFi radio 101h. WiFi radio 101h can operate with a medium access control (MAC) address MAC.R 101d.

The tag 101c can record an identity for the device 101, which can be ID-token.device 206 as described in Figure 2a below. For the system depicted in Figure 1a, device 101 operating a responder 101x can have a roll of an enrollee, and initiator proxy 102 can operate with a roll of a configurator. Device 101 can operate a responder 101x and, after a configuration step 106, also
5 operate a set of network WiFi credentials 109 that are compatible with an access point 105. Device 101 could be powered via any of (i) traditional “wall power” potentially with an AD/DC adapter, (ii) a battery which may be periodically recharged or replaced, (iii) power over a wired LAN connection such as “power over Ethernet”, and other possibilities exist as well. A device 101 can record a certificate cert.device 101m, which can record a device public key, which can be associated with a
10 device private key also recorded in device 101, but not shown in Figure 1a.

The responder 101x in device 101 can record a responder bootstrap public key Br 101a, a responder bootstrap private key br 101b, an initiator bootstrap public key Bi 104a, a set of cryptographic parameters 199a for the PKI keys, and optionally (i) an initiator bootstrap public key sequence number 197, and (ii) a public key exchange protocol (PKEX) shared secret key 198. The
15 responder 101x in device 101 can comprise a process operated by a processor and recorded in memory for device 101. Although not depicted in Figure 1a, a responder 101x can also include a random number generator, a PKI key pair generation algorithm, a secure hash algorithm, an elliptic curve Diffie Hellman key exchange algorithm, and a symmetric ciphering algorithm.

A network 107 can operate an access point 105, where the access point operates with a set of
20 credentials 109. The credentials 109 can include an identifier for the network such an SSID.network-AP 109a, a pairwise master key (PMK) PMK.network-AP 109b or a preshared key (PSK), a pairwise master key identifier (PMKID) (not shown), and also a configuration Config.network-AP 109c. The access point can support wireless networking standards in the series of 802.11 standards. A network 107 can also operate a plurality of access points 105. In addition,
25 access point 105 can comprise a plurality of different WiFi radios to collectively function as one access point 105, such as with “mesh” WiFi networks. Or, access point 105 can operate as a single WiFi router. The network 107 can include or be associated with a Device Provisioning Protocol (DPP) server 103, a device database 104, an authentication server 111, and a discovery server 110. Different servers within the network 107 can be connected using an Intranet, a virtual private
30 network (VPN), Internet Protocol Security (IPSEC), or similar techniques to secure communication between the servers.

The network 107 can operate or communicate with the device database 104. The DPP server 103 in the network 107 can also communicate with the device database 104. The device database 104 can record information for a plurality of devices 101 before responders 101x for the device 101
35 conducts the device provisioning protocol 191 with the initiator 102. The device database 104 can record the device identity 206, a responder identity 101d, which could be in the form of a responder

medium access control (MAC) address MAC.R 101d, an initiator configuration 130, and a key table. The key table in a device database 104 can record for each device 101 a responder bootstrap public key Br 101a, an initiator bootstrap public key Bi 104a, and an initiator bootstrap private key bi 104b. Recorded data within a device database 104 can be obtained by a network from either (i) upon
5 manufacturing from a device manufacturer, or (ii) before distribution of a device to the location of the network's access point. Some data in a device database 104 could also be acquired by an initiator 102 prior to the start of a DPP 191.

The DPP server 103 for a network 107 can operate a server database 103a, where the server database 103a records information for a plurality of devices 101 connecting with the network 107.
10 The server database 103a can record a medium access control (MAC) address MAC.R 101d in the device 101 a set of cryptographic parameters 199a for each device, an initiator bootstrap private key bi 104b, the initiator bootstrap public key Bi 104a, and the responder bootstrap public key Br 101a. A server database can also record additional data for responders 101x in devices 101, as depicted and described in connection with Figure 3b below. Depending on an initiator mode 306a in Figure 3
15 below, not all PKI keys and secret shared keys may be recorded in the server database 103a. The DPP server 103 can also include a set cryptographic algorithms, where the cryptographic algorithms can include a random number generator, a PKI key pair generation algorithm, a secure hash algorithm, an elliptic curve Diffie Hellman key exchange algorithm, and a symmetric ciphering algorithm. The cryptographic algorithms within a DPP server 103 can be compatible with the
20 equivalent set of cryptographic algorithms in the responder 101x and also the initiator proxy 102.

Note that the security of communications between a device 101 and an access point 105 may generally depend on the security of a configuration step 106, which in turn may rely on a device provisioning protocol 191. Further, a manufactured device 101 may not have the credentials 109 or configuration in order to communicate with an access point 105 without successfully completing a
25 DPP 191 and a configuration step 106 in order for device 101 to secure receive and use network credentials 109. After a DPP 191 and a configuration step 106 described in the present invention and in multiple figures below, device 101 can communicate with access point 105 using credentials 109.

30 **Figure 1b**

Figure 1b is a graphical illustration of a device provisioning protocol for authentication and configuration of a responder, in accordance with conventional technology. Figure 1b depicts a summary of the WiFi Device Provisioning Protocol (DPP) specification, version 1.0 which was
35 published on April 9, 2018. The summary depicted in Figure 1b highlights recorded bootstrap PKI

keys, derived ephemeral PKI keys, and messages transmitted and received between an initiator 102* and a responder 101x. The use of keys and messages for a DPP 191 can accomplish both (i) authenticating the responder 101x and (ii) transfer a configuration object to responder 101x, where the configuration object could comprise a set of network access credentials 109 from Figure 1a.

5 Initiator 102* is depicted with a "*" and has several differences than an initiator proxy 102 depicted in Figure 1a and subsequent figures below, although some functionality is shared. Both initiator 102* and initiator proxy 102 (from Figure 1a and other Figures below) can send and receive the full and complete set of a DPP messages for a Device Provisioning Protocol 191.

In other words, initiator 102* and an initiator proxy 102 can send equivalent messages such

10 that a responder 101x would process the messages in the same manner and would not normally be able to detect to discern whether a initiator proxy 102* or an initiator 102 was transmitting and receiving messages for the Device Provisioning Protocol 191. In the present invention, initiator proxy 102* can operate as an initiator according to the DPP Specification 1.0. Initiator proxy 102* or an initiator 102 can be (x) a process, program, firmware, or software application operating in a

15 computing device with a WiFi compatible radio, in order to (y) communicate with a responder 101x via a WiFi data link 192 (in Figure 1a). Exemplary additional configurations for an initiator 102 and a responder 101x within computing devices is depicted and described in connection with Figures 7 and 8 below, and other possibilities exist as well without departing from the scope of the present invention.

20 Other differences between an initiator 102* and an initiator proxy 102 can exist in the present invention. For a system 100 with an initiator proxy 102 in Figure 1a and Figures below, the recordation of PKI keys, PKI key derivation, and cryptographic processes such as symmetric encryption and decryption can be distributed between an initiator proxy 102 and a DPP server 103, while an initiator 102* can internally record the keys necessary for symmetric key exchange, key

25 derivation, and symmetric encryption and decryption. The distributed storage and operation with PKI keys for a system that includes an initiator proxy 102 is depicted and described in connection with Figure 3, Figure 5a, Figure 5b, and Figure 6a below. Additional details for internal components, memory structures, values, and operation of an initiator proxy 102 in order to provide equivalent functionality as initiator 102* for a responder 101x are described in Figures below.

30 Initiator 102* can record bootstrap keys, ephemeral keys, and derive symmetric encryption keys in order to authenticate the responder 101x and transfer the network access credentials 109 according to a Device Provisioning Protocol 191. For a system with responder authentication 141, an initiator proxy 102* can record a medium access control (MAC) address MAC.I 102n. MAC.I 102n can be associated with a WiFi radio within Initiator proxy 102* and comprise a string of octets

35 in order to uniquely identify initiator 102* within a wireless network. With current, convention technology, MAC.I 102n can comprise a string of 6 octets, although other possibilities exist as well

for a MAC.I 102n. Initiator 102* can also operate a key pair generation function 102y in order to derive ephemeral PKI keys such as an initiator ephemeral public key Pi 102a and an initiator ephemeral private key pi 102b. As contemplated herein, an ephemeral PKI key can also be referred to as a “protocol” key. In other words, key Pi 102a can be referred to as initiator protocol public key
5 Pi 102a or initiator ephemeral public key Pi 102a, which can both represent the same numeric value, number, or sequence of digits.

Although not depicted in Figure 1b, a key pair generation function 102y could use (i) a random number generator in order to derive the PKI private key and (ii) a selected set of cryptographic parameters 199a (depicted in Figure 2b below) in order to specify an elliptic curve
10 cryptography (ECC) curve name, key length, key formatting, etc. As contemplated herein, the algorithms used with ECC PKI keys in the present invention can be compatible and substantially conform with ECC algorithms and keys as specified in (i) the IETF Request for Comments (RFC) 6090 titled “Fundamental Elliptic Curve Cryptography Algorithms”, and (ii) IETF RFC 5915 titled “Elliptic Curve Private Key Structure”, and also subsequent and related versions of these standards.
15 With a random number generated by initiator 102* or an initiator proxy 102, a key pair generation function 102y, a derived ephemeral initiator private key pi 102b can be used to derive a corresponding ephemeral initiator public key Pi 102a. The two derived PKI keys can be used in conjunction for subsequent operations such as an Elliptic Curve Diffie Hellman key exchange. As contemplated in the present invention, the use of a capital letter as the first character for a PKI key
20 can represent a public key, the use of a lower case letter as the first character for a PKI key can represent a private key, and the second letter for a PKI key can represent the entity the key is associated with or belongs to. Further, the letter “B” or “b” as the first character for a PKI key can represent a recorded bootstrap key, which could be static or relatively static over the lifetime of a responder or initiator, and the letter “P” or “p” the first character for a PKI key can represent an
25 ephemeral key.

Responder 101x in Figure 1b, Figure 1c, and other Figures below can include functionality to operate as a responder according to the DPP Specification version 1.0. As depicted in Figure 1a, responder 101x could operate within a device 101, although other possibilities exist as well for the location or equipment operating a responder 101x without departing from the scope of the present
30 invention. For a DPP 191, a responder 101x can record a responder bootstrap public key Br 101a, a responder bootstrap private key br 101b, a MAC address MAC.R 101d, and also operate a key pair generation algorithm 101y. Responder bootstrap public key Br 101a and responder bootstrap private key br 101b could comprise an ECC key pair selected and formatted according to a selected set of parameters 199a (in Figure 1a above).

35 Ephemeral bootstrap PKI keys for responder 101x could also be formatted and selected with a compatible set of cryptographic parameters 199a as Pr 101e and pr 101f. Bootstrap PKI keys for

responder 101x could be recorded in responder 101x or nonvolatile memory for a device 101 operating a responder 101x. The bootstrap PKI keys could be recorded in responder 101x during manufacturing or a configuration of responder 101x before delivery of a device 101 to the location of an access point 105 from Figure 1a. MAC address MAC.R 101d can comprise an identity of responder 101x or a device 101 on a WiFi network, and can be similar to MAC.I 102d for an initiator 102* as described above. Key pair generation algorithm 101y can be equivalent to key pair generation algorithm 102y described above for initiator proxy 102*. Key pair generation algorithm 101y can derive ephemeral PKI keys for responder 101x comprising public key Pr 101e and private key pr 101f, which could also be derived using a compatible set of parameters 199a for Br 101a, br 101b, Pi 102a and pi 102b.

An initiator 102* and a responder 101x can conduct a Device Provisioning Protocol (DPP) 191. Again, as contemplated herein, an exemplary Device Provisioning Protocol 191 according to conventional technology could comprise the “Device Provisioning Protocol Specification” version 1.0 as published by the WiFi Alliance TM. In message 121, an initiator 102* can receive the bootstrap public key Br 101a for responder 101x. The “out of band” transfer of Br 101a in message 121 could comprise several different possible methods for initiator 102* to receive Br 101a, which are also described in the DPP specification 1.0, such as via reading a QR code, reading a near-field communications tag, using a Bluetooth wireless connection to read Br 101a from responder 101x, and other possibilities exist as well. A DPP 191 also contemplates that associated information for responder 101x could be transferred in message 121, such as a device identity for device 101, MAC.R 101d, in addition to Br 101a and possible parameters 199a.

Initiator 102* can receive Br 101a and derive PKI keys Pi 102a and pi 102b. Initiator 102* can conduct an Elliptic Curve Diffie Hellman key exchange algorithm 401a as depicted and described below in key exchange 314 step for Figure 4a in order to derive a shared secret key with responder 101x. Initiator 102* can derive a random number comprising an initiator nonce and also determine a set of capabilities for the initiator, such as a configurator or enrollee as specified in DPP specification version 1.0. Initiator 102* can encrypt the initiator nonce and the initiator capabilities using the derived shared secret key as depicted and described in connection with an encryption step 315 in Figure 4a below. Initiator 102* can send the derived public key Pi 102a and a ciphertext with the encrypted data in a message 122. Although not depicted in Figure 1b, initiator 102* can also send a hashed value of Br 101a in message 122.

As specified in DPP specification version 1.0, message 122 could also include a hashed value for initiator bootstrap public key Bi 104a, but for a responder authentication 141 the value of Bi 104a may not be used (since responder 101x may not record the corresponding initiator bootstrap public key Bi 104a for a responder authentication 141 as depicted in Figure 1b). The hash algorithm on a bootstrap public key could comprise the SHA-256 hash algorithm according to IETF RFC 5754

titled "Using SHA2 Algorithms with Cryptographic Message Syntax". Other hash algorithms could be used as well and specified in a selected set of cryptographic parameters 199 as depicted and described in connection with Figure 2b below, such as SHA-3, SHA-384, etc. Message 122 can comprise a Device Provisioning Protocol Authentication Request according to DPP specification version 1.0. Additional details for a message 122 are provided in section 6.2.2 of DPP specification version 1.0.

Responder 101x can receive message 122 and follow the steps for a responder according to the section 6.2.3 of DPP specification version 1.0, in order to generate a Device Provisioning Protocol Authentication Response, which could comprise message 123. Responder 101x can conduct an Elliptic Curve Diffie Hellman key exchange algorithm 401a as depicted and described below in key exchange 319 step for Figure 4a in order to mutually derive a first shared secret key with initiator 102*. Responder 101x could decrypt the initiator nonce and the initiator capabilities using the derived shared secret key as depicted and described in connection with an decryption step 320 in Figure 4a below. Responder 101x can use the key pair generation algorithm 101y in order to derive an ephemeral responder public key Pr 101e and a corresponding ephemeral private key pr 101f. Responder 101x can use the derived responder ephemeral private key pr 101f and the received initiator ephemeral public key Pi 102a in order to conduct a key exchange algorithm 314b as depicted below in Figure 4b and derive a second secret shared key. The first mutually derived symmetric key can comprise k1 and the second mutually derived symmetric key can comprise k2 according to the DPP specification version 1.0. Responder 101x can derive a random number comprising a responder nonce and also determine a set of capabilities for the responder, such as a configurator or enrollee as specified in DPP specification version 1.0.

Responder 101x could also generate a responder authentication value, which could comprise a hash value over the initiator nonce received in message 122 and the generated responder nonce. Responder 101x could use the first shared secret key and the second shared secret key in order to derive a third shared secret key. Responder 101x could encrypt the responder authentication value with the third shared secret key and also then include the encrypted responder authentication value with the responder nonce and responder capabilities in a ciphertext. The ciphertext could be encrypted with the second symmetric key, as depicted and described in connection with an encryption step 315b in Figure 4b below. Responder 101x could send the derived ephemeral responder public key Pr 101e and the ciphertext in a message 123.

Initiator 102* can receive message 123 and record the received ephemeral responder public key Pr 101e. Initiator 102* can follow the steps in section 6.2.4 of DPP specification version 1.0 in order to process message 123 and confirm the authentication response from responder 101x in message 123. Initiator 102* can mutually derive the second symmetric key using the received ephemeral responder public key Pr 101e and the derived initiator ephemeral private key pi 102b,