'117 Claims	Exemplary Disclosures
	this is the first time program 51 is executed, in step S4. If this is not the first time (NO), the process will move on to step 31 (FIG. 4), for receiving data. If this is the first time execution of the program, (YES) in step S4, the next step S5 determines whether or not a shared encryption key exists on the removable storage 5. This operation is directed to a shared encryption key being originally stored on the removable storage 5. Alternatively, the user may enter and store the shared encryption key after obtaining it separately by post mail commerce, or the like. Consequently, if the shared encryption key already exists on the removable storage 5 (YES), the process moves on to the step 31 describing the operating procedure for receiving data shown in FIG. 4 below. In step S5, if the shared encryption key does not exist in the removable storage 5 (NO), then a request for the shared encryption key will be transmitted to the data input site 1 in step S6. The data input site 1 may allow the client site 3 to choose the algorithms to apply and the validity time or lifespan of the key in step S7. The client site 3 will provide a selection of one of these options in step S8 via a touch screen, keyboard voice, tactile or like interface. Then, the data input site 1 produces and transmits a number that will be used to generate the key in step S9, and the client site 3 will produce and transmit a number that will be used to generate the key in step S10. This leads to a key exchange in accordance with Diffie-Hellman protocol. The key generated in step S10 will be transmitted to the data input site 1, so the data input site 1 will transmit the necessary ID and passwords in step S11. Similarly, the client site 3 will transmit the necessary ID and passwords in step S12, after which the data input site 1 will save the encryption key on the storage device in step S13. Furthermore, the client site 3 will send the encryption key to the removable storage 5 in step S14. When the removable storage 5 receives the

'117 Claims	Exemplary Disclosures
	encryption key from the client site 3 in step S15, it stores the encryption key into the memory device in the removable storage in step S16. By using the key exchange algorithm above, the distribution of keys is carried out between the data input site 1 and the client site 3. The flow chart in FIG. 3 shows processing operations to transmit data from the data input site 1 to the database server 2. FIG. 3 shows the data input site 1 on the left and the database server 2 on the right. When the program is executed in step S21, user authentication is performed at the data input site 1. If authentication is not confirmed (NO), step S22 will be repeated. If authentication is confirmed (OK), the instruction for starting the data transmission will be issued in step S23. Then the data is compressed in step S24 and the compressed data is encrypted with the shared encryption key in step S25. Data encrypted in step S25 is transmitted to the database server 2 in step S26. The database server 2 receives the encrypted data which was encrypted using the shared encryption key from the data input site 1 in step S27. The shared key encrypted data is stored on the storage device in step S28. By this procedure, the shared key encrypted data is transmitted from the data input site 1 to the database server 2." Qumei at 3:25-27 In an embodiment according to the present invention, an enciphering algorithm and an enciphering key may be stored in the electronic devices. Qumei at 7:66-9:9 Ciphering/enciphering may comprise scrambling information to hide the actual content from un-intended viewers. Several methodologies of ciphering/enciphering exist. One method of ciphering/enciphering firmware images is by applying symmetric ciphering/enciphering. Symmetric ciphering/enciphering may use one or

'117 Claims	Exemplary Disclosures
	multiple keys for both encryption and decryption. Symmetric ciphering/enciphering may be performed in at least the following two embodiments: stream symmetric ciphering/enciphering and block symmetric ciphering/enciphering. Stream symmetric ciphering/enciphering may operate on information in a byte by byte manner. A key stream may be combined with a message to produce a ciphered/enciphered message. Stream symmetric ciphering/enciphering may be mathematically represented, for example, as follows: C[i]=KS[i] ⊗ M[i]; Encryption M[i]=KS[i] ⊗ C[i]; Decryption For example, to encrypt a message according to the stream symmetric ciphering/enciphering method of an embodiment of the present invention, the ith byte of key stream KS may operate on a byte of message M to produce the ith cipher C, for example. To decrypt a message, the ith byte of key stream KS may operate on the ith cipher C to reproduce a byte of the message M, for example. Stream symmetric ciphering techniques are known to have been employed in general cryptography, however, stream symmetric ciphering techniques are not known to have been previously used in techniques of updating firmware/software in mobile electronic devices. An example of stream symmetric ciphering used in general cryptography, for example, is RC4 (Ron's Code or Rivest's Cipher level 4). RC4 is a stream ciphering

Headwater Research LLC
Ex. 2011, IPR2024-00003
Page 297 of 442

'117 Claims	Exemplary Disclosures
	technology designed by Ronald Rivest for RSA Security™. RC4 is a variable key-size stream ciphering technique having byte-oriented operations. The algorithm for RC4 is based on the use of a random permutation. Analysis shows that the period of the cipher using RC4 is likely to be greater than 10100. Eight to sixteen machine operations are required per output byte using RC4. The RC4 cipher and corresponding algorithm are considered secure. Block symmetric ciphering/enciphering, in contrast to stream symmetric ciphering/enciphering, may break the information into blocks of data, instead of bytes. The blocks may comprise 8-16 bytes, for example. A key block may be applied to a message to produce ciphered/enciphered text. In an embodiment according to the present invention, block symmetric ciphering/enciphering may be performed by cipher block chaining, for example. Block symmetric ciphering/enciphering may be mathematically represented, for example, as follows: C[i]=E(K[i],M[i] ⊗ C[i−1]); Encryption M[i]=D(KS[i],C[i]) ⊗ C[i−1]; Decryption For example, using block symmetric ciphering/enciphering, the message data may be expanded to accommodate variable block size requirements. For example, large messages may be impacted (e.g., expanded) toward the end of the message. Block symmetric ciphering techniques are known to have been employed in general cryptography, however, block symmetric ciphering techniques are not known to

'117 Claims	Exemplary Disclosures
	have been previously used in techniques of updating firmware/software in mobile electronic devices. Examples of block symmetric ciphering techniques used in general cryptography are DES (data encryption standard), 3DES (triple-DES) and RC2 (Ron's Code or Rivest's Cipher level 2). RC2 is a variable key-size block cipher designed by Ronald Rivest for RSA Security™. (RC stands for "Ron's Code" or "Rivest's Cipher"). RC2 is faster than DES and is designed as a "drop-in" replacement for DES. RC2 can be made more secure or less secure than DES against exhaustive key search by using appropriate key sizes. RC2 has a block size of 64 bits and is about two to three times faster than DES. An additional string (40 to 88 bits long) called a salt can be used to thwart attackers who try to precompute a large look-up table of possible encryptions. The salt may be appended to the encryption key. The lengthened key is used to encrypt the message. The salt is then sent, unencrypted, with the message. Tyhurst at [0028]: The mobile data server 120 also provides a secure exchange of data (e.g., email messages, personal information manager (PIM) data, and IM data) with the mobile communication devices 201. In at least some embodiments, communications between the mobile data server 120 and the mobile communication devices 201 are encrypted. In some embodiments, communications are encrypted using a symmetric encryption key implemented using Advanced Encryption Standard (AES) or Triple Data Encryption Standard (Triple DES) encryption. Private encryption keys are generated in a secure environment and are used for both encryption and decryption of data. In some embodiments, the private encryption key is stored on the messaging server 132 (for example, in the user's mailbox) and on the mobile communication device 201. In other embodiments, the private encryption key maintained by the mobile data server 120 is stored elsewhere. The

'117 Claims	Exemplary Disclosures
	location of the private encryption key maintained by the mobile data server 120 typically depends on the messaging server 132 which is used. The private encryption key may, in some embodiments, be regenerated by the user on mobile communication devices 201. Data sent to the mobile communication devices 201 is encrypted by the mobile data server 120 using the private encryption key retrieved from the user's mailbox or other location. The encrypted data, when received on the mobile communication devices 201, is decrypted using the private encryption key stored in memory. Similarly, data sent to the mobile data server 120 from the mobile communication devices 201 is encrypted using the private encryption key stored in the memory of the mobile communication device 201. The encrypted data, when received on the mobile data server 120, is decrypted using the private encryption key retrieved from the user's mailbox or other location. Tyhurst at [0049]: The mobile communication device 201 includes a microprocessor 240 which controls the overall operation of the mobile communication device 201. The microprocessor 240 interacts with communication subsystem 211 which performs communication functions. The microprocessor 240 also interacts with additional device subsystems which may include but are not limited to a display device 242 which may be a liquid crystal display (LCD) screen, a flash memory 244, a random access memory (RAM) 246, a read only memory (ROM) 248, auxiliary input/output (I/O) subsystems 250, a data port 252 such as serial data port (for example, a Universal Serial Bus (USB) data port), a keyboard or keypad 254, a speaker 256, microphone 258, a navigation tool 260 such as a clickable scroll wheel (also referred to as a track wheel or thumbwheel) or trackball, a short-range communication subsystem 262, and other device subsystems generally designated as 264. Different numbers and combinations of the aforementioned device subsystems may be present in various embodiments, and

'117 Claims	Exemplary Disclosures
	not all need be present in some embodiments. The keypad 254 maybe either a complete alphanumeric keypad, a limited, reduced, or simplified alphanumeric keypad, or a telephone-type keypad. In some embodiments, the mobile communication device 201 may comprise a touchscreen display which, in at least some embodiments, includes a touch-sensitive input surface which overlays the display device 242. In such embodiments, the keypad 254 may be omitted, and possibly the navigation tool 260 may be realized using the touch-sensitive input surface. Tyhurst at [0050]: The microprocessor 240 operates under stored program control and executes software modules 220 stored in memory such as persistent memory, for example, in the flash memory 244. The software modules 220 comprise operating system software 254 and software applications 258. Those skilled in the art will appreciate that the software modules 220 or parts thereof may be temporarily loaded into volatile memory such as the RAM 246. The RAM 246 is used for storing runtime data variables and other types of data or information, as will be apparent to those skilled in the art. Although specific functions are described for various types of memory, this is merely an example, and those skilled in the art will appreciate that a different assignment of functions to types of memory could also be used. Tyhurst at [0026]: The mobile data server 120 may be implemented as one or more server modules, and is typically located behind a firewall 113. The mobile data server 120 manages communications to and from a set of managed mobile communication devices 201 such that the mobile communication devices 201 are each enabled to exchange electronic messages and other information with each other and computers, for example via the Internet 112. The mobile data server 120 also

'117 Claims	Exemplary Disclosures
	provides administrative control and management capabilities over users and mobile communication devices 201 which may connect to the mobile data server 120. Cohen at [0042]: The wireless device 403 securely browses to the initial browse point using, for example, the Secure Socket Layer (also called "SSL") protocol. Using this protocol, the wireless device 403 provides a Security Parameter Index (also called "SPI") that may be used to identify a notification session. The SPI is, for example, a 32-bit value that is unique to the wireless device 403, and that may serve as a handle into the appropriate security context for the notification session. The wireless device 403 may also suggest other security information such as the master secret, the cryptographic algorithm (e.g., NULL for none, DES, 3DES, RC5 or the like), the key length, the hash algorithm (e.g., NULL, MD5-HMAC, SHA_HMAC, MD5_RCA, SHA_RCA or the like), the session expiration (in time and/or in usage), and potentially an exportable keys flag that indicates whether or not this security information is exportable. The wireless device 403 may suggest these parameters in a GET method of a HyperText Transport Protocol (HTTP) request with the parameters includes as arguments in the method. The initial browse point of the notification source 401 may then response with an acceptance or a rejection of these parameters. A rejection may also include a suggestion for alternative values. Ultimately, the required security information for the session is agreed upon. Cohen at [0047]: [0047] Referring to FIG. 3, once the security information is negotiated out-of-band from the notification mechanism (act 302), the notification source uses the security information to generate a push message (act 303) thus completing the step for drafting a push message so as to ensure secure end-to-end notification (step 301) between the notification source and the notification sink. Referring to FIG. 5, the push message 500 includes an encrypted form of the notification represented as

'117 Claims	Exemplary Disclosures
	notification message (encrypted) 502, as well as supplemental security information 501 that is represented in clear text and that may be used along with the security information originally negotiated to decrypt the notification message. Cohen at [0048]: The notification source 401 then initiates transmission of the push message 500 to the notification sink 403 (act 304) via the at least one message transit point 404. Although the push message 500 traverses through the at least one transit points 404, the notification message portion of the push message is encrypted. Since the at least one message transit point 404 is not privy to the security information originally negotiated in act 302, the at least one message transit point 404 is not able to decrypt the notification message 502. Once the notification sink 403 receives the push message 500 (act 305), the notification sink 403 decrypts the encrypted notification message 502 using the security information previously negotiated (security information 421) along with potentially the supplemental security information 501 (act 306). Thus end-to-end security is provided since only the notification source 401 and the notification sink 403 are able to access the clear text content of the notification message. Cohen at [0072]: [0072] The encrypted payload data field 704 is a variable length field that actually contains the encrypted form of the notification, as well as potentially some encrypted header fields. For example, the field 704 may include a content type field as well as a Wireless Application Protocol (also called "WAP") application ID field. The content type field may identify the presence of XML in the multipart message, while the WAP application ID field identifies the application the device should call to process the push message. The padding field 705 ranges from zero to 255 bytes and is often used when cryptographic algorithms require the cipher text to be a certain length or, regardless of the cryptographic algorithm, padding is

'117 Claims	Exemplary Disclosures
	needed to terminate the cipher text on a 4-byte boundary. The padding length field 706 is an 8-bit value that specifies the length of the padding field 704 in bytes. The payload ID field 707 (also called the "next field" in the ESP standard) identifies the type of data included in the payload data field 703. The payload data field 704, the padding field 705, the padding length field 706, and the payload ID field 707 are all encrypted. The payload data field 704 represents an example of the notification message 502 of the push message 500. ESP object 700 may also include authentication data field 708 that includes a digital signature for authenticating the push initiator 601. Shen at [0054]-[0060] As shown in FIG. 5, in a fourth embodiment, an authentication and key management module 502 is added in the MMS gateway to distribute symmetric keys to users and a certificate management module 504 is added in the user device. Both the MMS gateway and the user device deploy the same symmetric cipher to encode and decode MMS messages to protect the privacy of these messages. The encryption may be realized in any possible manner. The following is an example. 1. If the MMS user device has no certificate of the MMS gateway, it first sends a CA (certificate) requirement to the gateway. After the gateway confirms that the user has the access to send and receive MMS according to the user device number, the gateway's CA is sent to the user device. There are three methods to distribute a certificate to a user device: manually synchronize a certificate into the user device via an infrared port, or send a certificate via SMS or WAP response. 2. The MMS user device sends a Master Key requirement to the gateway and the gateway creates a session ID and a master key, generally a random object, and

'117 Claims	Exemplary Disclosures
	sends them to the device via SMS or WAP response. The random object is encrypted by the gateway's private key. 3. The random object is decoded by the user device and mixed with its phone number and session ID to generate a shared secret key. The mixing operation may be any operation, the simplest is XOR operation. At the same time, the gateway also generates a shared secret key with the same algorithm. 4. Both the MMS user device and the MMS gateway can secretly communicate with each other by encrypting MMS messages with the shared secret key. In above encryption process, both the MMS user device and the MMS gateway can fresh the shared secret key by changing the session ID to prevent the threat of attacks stemming from key reuse. Novack at [0013]: "The present invention enables secure inter-process communications. The implementation of secure inter-process communications will be described in the context of a client-server network and a peer-to-peer network. Of course, the present invention is also applicable to environments other than the networks described herein, including any network environment that includes inter-process communications between different processes running on one or more computers." Novack at [0021]: "According to an aspect of the present invention, a computer-readable medium is provides that stores a program that secures inter-process communications between a first process and a second process. The computer-readable medium includes a receiving code segment that receives, from a device that supports the first process, a request to initiate inter-process communications. The computer-readable medium also includes a digital certificate requesting code segment that replies to the device

'117 Claims	Exemplary Disclosures
	with a request for information of a digital certificate that authenticates the identity of the first process." *See also* Novack at [0027]. Novack at [0064]: "The device in FIG. 4 may include multiple processes that communicate with each other using the secure inter-process communications of the present invention. Additionally, although the terms "client" and "server" are used to refer to devices, the devices may be defined and distinguished by a set of software instructions installed as a pre-packaged program. Further, while examples of the uses of the present invention include secure client/server networks and peer-to-peer networks, secure inter-process communications can be used to enhance security in any inter-process communications environment. Further, while the embodiment shown in FIG. 2 only shows a **single computer** 201 and a **single application server** 230, secure inter-process communications can be used to enhance security for **multiple application servers** 230 belonging to a single entity or multiple **users using computers** 201. Additionally, while **networks** 220 and 240 are separately designated in FIG. 2, the networks may be the same, such as an internet protocol (IP) network. Further, while a certification authority is discussed in the context of the present invention, a certification authority is not shown in the figures. In this regard, a certification authority may communicate over the **network** 220 and/or the **network** 240 to provide digital certificates to the **application server** 230 and the **management server** 250. Of course, the digital certificates may be provided by the certification authority as portable memory modules that are installed on the **application server** 230 or the **management server** 250. Furthermore, the entity that provides the **application server** 230 and/or the **management server** 250 may also act as a certification authority for its customers using the computers 102."

Headwater Research LLC
Ex. 2011, IPR2024-00003
Page 306 of 442

'117 Claims	Exemplary Disclosures
	Okonnen at [0010]: "In an embodiment according to the present invention, an electronic device may comprises random access memory and non-volatile memory. The non-volatile memory may comprise a plurality of components. The plurality of components may comprise at least one of an update application loader, firmware, an operating system, and provisioned data. The provisioned data may comprise update agent provisioning information and a number assignment module." Okonnen at [0011]: "In an embodiment according to the present invention, the network may further comprise at least one of an update server, and a plurality of generators. The generators may be adapted to generate updates able to be processed by at least one provisioned update agent in electronic device. The update server may be adapted to store updates accessible by the plurality of servers. The plurality of servers may be communicatively coupled to the update server." Okonnen at [0021]: "In an embodiment according to the present invention, the index of provisioned update agents may provide an index value used to compute an address location of a provisioned update agent. The index of provisioned update agents may be an index to a table containing an address for an update agent in non-volatile memory the electronic device." Okonnen at [0023]: "In an embodiment according to the present invention, the electronic device may further comprise an update agent table resident in non-volatile memory. The update agent table may contain references to a plurality of update agents currently

'117 Claims	Exemplary Disclosures
	available and provisioned in the electronic device. The update agent table may map update agent names, update agent address locations, and types of updates that the update agents are adapted to process, and provisioning status of the update agents for all available update agents in the electronic device." Okonnen at [0049]: "The electronic device, for example, mobile handset 107, may comprise random access memory (RAM) 125 and non-volatile memory 111. The non-volatile memory 111 may comprise a plurality of components, for example, an update application (UA) loader 127, an update agent A 113, an update agent B 115, a firmware 117, an operating system (OS) 119, and a provisioned data 129. The provisioned data 129 may comprise update agent provisioning information 131 and a number assignment module (NAM) 133." Okonnen at [0052]: "The provisioned data 129 may also be used to store update agent related provisioning information, such as a universal resource locator (URL) of a device server or a management server that may be used to retrieve updates, a security key(s) that may be used to authenticate messages to/from the device server, and selective encryption/decryption of the messages, etc. Each update agent, such as update agent A 113 and update agent B 115, may have corresponding entries in provisioned data 129." Okonnen at [0060]: "In an embodiment according to the present invention, update agent A 113 may be employed to update firmware 117 and operating system 119 during reboot of the mobile handset 107. After a reboot update, update agent B 115 may be used to update other components in the mobile handset 107, for example, configuration parameters, gaming engines, third-party software, etc., that may not be updated

'117 Claims	Exemplary Disclosures
	during reboot of the mobile handset 107. Both of update agent A 113 and update agent B 115 may be provisioned with security keys for secure communications with device servers 135 and 137 during update agent provisioning activity, such as provisioning activity accompanying NAM provisioning activity and/or incorporated into NAM programming activity." Okonnen at [0063]-[0064]: "FIG. 2A is a block diagram 205 illustrating update agent provisioning information 207 associated with an electronic device, for example, mobile handset 107, according to an embodiment of the present invention. The provisioning information may comprising a device server URL 211, an index of provisioned update agents 209, a security key(s) 213, and other related information 215. The device server URL 211 may provide references to remote device servers providing access to updates that may be downloaded. The updates may be compatible with update agents currently available and provisioned in the mobile handset 107. "In an embodiment according to the present invention, the index of provisioned update agent 209 may provide an index value that may be used to compute an address location of a provisioned update agent. For example, if the update agents are located in consecutive blocks of 64 Kbytes each, then an index value of one multiplied by a block size of 64 K and added to an offset value (or preset starting address) provides the address of the provisioned update agent." Okonnen at [0065]: "In another embodiment according to the present invention, the index of provisioned update agent 209 may be an index to a table containing an address for an update agent in storage, such as for example, non-volatile memory 111 in mobile handset 107. The security key(s) 213 may be used to authenticate updates, for example, to authenticate during download of updates or during update activity."

'117 Claims	Exemplary Disclosures
	Okonnen at [0066]: "In an embodiment according to the present invention, a separate key may be employed to authenticate updates by a download agent/download client, (e.g., a browser), and by the update agent. In an embodiment according to the present invention, the security key(s) 213 may also incorporate keys employed for secure communications with external systems, such as keys used for encryption and decryption of data and messages during communications with external systems, such as a device server." Hashimoto at [0030]: "In order to realize this, the microprocessor has a task state table for assigning an identifier to the task that can take a plurality of states and maintaining a state of the task in correspondence to the task identifier, a task register for maintaining the currently executed task identifier, an interface for reading in units of cache lines the program stored in the encrypted state by using a prescribed program key in the external memory when there is a request for the task, an encryption processing unit for generating a decryption key for each cache line according to the program key and decrypting the content read by the interface, a first cache memory formed by a plurality of cache lines having respective tags for maintaining a task identifier corresponding to the decryption key used in decrypting each cache line in the tag, and an access verification unit for comparing the task identifier maintained in the tag of the cache line and the value of the task register and discarding the content of the cache line when these values do not coincide." Hashimoto at [0031]: "In order to solve the second problem, it is made possible to carry out the decryption by using keys that are different for different blocks according to a small amount of secrets."

'117 Claims	Exemplary Disclosures
	Hashimoto at [0032]: "In order to realize this, the microprocessor is provided with the following. (a) a processor core that issues a reading request by specifying an address of the external memory that stores a plurality of programs that are respectively encrypted according to the unique program keys: (b) an interface for reading a data block of the specified address of the external memory in response to the reading request of the processor core; (c) a task key table for maintaining the program key in correspondence to the task identifier assigned to each task which is a unit of the program execution control; (d) an offset table for maintaining a top address at which the program is arranged as an offset value, for each task identifier; (e) a key generation unit for calculating a relative address value from the address specified by the reading request and the offset value specified by the task identifier of the offset table, and generating a block corresponding key in which the relative address value is encrypted by using the program key; (f) an encryption processing unit for decrypting the read out data block by using the block corresponding key; and (g) a cache memory into which the decrypted data block is read in units of cache lines." Hashimoto at [0033]-[0034]: "Also, at a time of the task execution interruption, the physical addresses at which the program and the data are arranged are often changed by the system resource management (re-location), and in order to decrypt and normally execute the

'117 Claims	Exemplary Disclosures
	program encrypted in units of blocks, there is a need for the information to be arranged in a certain address space. For this reason, a virtual address is used in the case of calculating the relative address that is expressed by a difference between the target address and the program start address (offset address). "In this way, the block interchange can be prevented effectively, and the limited physical memory can be utilized efficiently." Hashimoto at [0040]: "More specifically, the program formed by execution codes and data such as initialization data is encrypted by using a program key selected by the software vendor 121, and the program key used for the encryption is encrypted by using a public key-A of the processor-A of the target system 102 to generate a distribution key." Hashimoto at [0041]: "At this point, the software vendor 121 divides the program and the initialization data into a plurality of blocks, and encrypts them by using a different key for each block. Note however that there is no need for the software vendor 121 to select as many keys as the number of blocks, and it suffices to select two things including a constant Cx as a basis for generating a plurality of keys, and a key Kx for encrypting the plurality of keys generated according to Cx. As will be described in detail below, a block encrypted program 123 is generated by the encryption using a different key for each block, according to these two constants. Then, Cx and Kx are encrypted by using the public key-A of the processor-A to produce a distribution key Kx[A], Cx[A]." Hashimoto at [0043]:

'117 Claims	Exemplary Disclosures
	"The distributed block encrypted program is stored into a secondary memory 107 such as a hard disk once along with the distribution key, and read into a region 109-1 secured in a memory 108 at a time of the execution. This reading into the memory is carried out by an OS 110 for managing resources of the system 102. The program itself and the distribution key may be stored into a single file or separate files. The memory 108 also has regions 109-n for other programs of the same vendor 121 or other programs of the other vendors, but their description will be omitted here for the sake of explanation." Hashimoto at [0044]: "The processor-A reads the distribution key 125, and decrypts the distribution key by using the secret key-A at a protection logic 106. The encrypted program arranged in the memory 108 is read into the cache memory 104 in units of blocks and decrypted in units of blocks by using the decrypted distribution key. At this point, a feature of the processor-A is that it has a capability for generating a block corresponding key in correspondence to each one of a plurality of encrypted blocks from a limited number of secrets, such as two constants Kx and Cx, for example, and carrying out the decryption in units of blocks." Hashimoto at [0050]: "The fast (symmetric) encryption key processing module 911 has a key generation module 1312 and an encryption processing module 1313. The key generation module 1312 acquires a block key and a relative address value of a processing target block through the internal bus 905, and generates the encryption key corresponding to each block by the method to be described below. The encryption processing module 1313 executes the encryption/decryption processing of the corresponding block by using the generated encryption key for each block. The

'117 Claims	Exemplary Disclosures
	block encryption processing requires a plurality of clocks, so that a dispatcher for enabling the parallel processing of a plurality of modules is also included." Hashimoto at [0051]: "The key value table (KVT) 912 searches through the table according to the table ID information sent from the cache, and supplies the key value to the fast encryption key processing module 911." Hashimoto at [0052]: "The distribution key management module 913 sets up the decryption of the distribution key by using the public key and a decryption result table." Hashimoto at [0054]: "Fig. 3 shows a procedure for the encryption of the program in units of blocks. This processing is carried out by the block corresponding encryption processing unit 126 of the software vendor 121 of Fig. 1." Hashimoto at [0055]: "A program region 1001 of the plaintext application program developed by the software vendor is formed by a plurality of blocks (block-0 to block-n) in units of 128 bits (16 bytes). With respect to a whole of this region, two constants Kx and Cx are selected in advance." Hashimoto at [0056]: "First, a sum of the constant Cx and a relative top address of each block with respect to a top of the program is obtained. For example, the top byte of the first block-0 is "0", so that Cx+0 is calculated (S1002-0). The value Cx+0 is encrypted by using the secret key Kx, to generate a block corresponding key Kv0 (S1003-0).

'117 Claims	Exemplary Disclosures
	The plaintext block 1001-0 is encrypted by using this block corresponding key Kv0 (S1004-0), to generate an encrypted block 1005-0." Fahrny at [0008]: "According to the present invention, a controller for managing media stream decryption keys is provided. The controller comprises a media decryption engine, a table, and a content key list. The media decryption engine generally receives an encrypted media stream from a headend and presents a decrypted media stream to a receiving device in response to a decryption key. The decryption key is generally a function of a content key. The table may contain a content key and index and a plurality of corresponding content keys. Content keys that correspond to a particular encrypted media stream may be selected from the content key list using an index from the content key table, and that is referenced by an identifier received from a headend in connection with the encrypted media stream. The table may optionally (i.e., alternatively) contain initialization vector (IV) values that may be indexed and selected." Fahrny at [0023]: "Keylist: A list of decoder addresses and respective decoder keys in ordered pairs. Keylists may be used by the Uplink Control System (UCS) for generation of authorization messages that are addressed to the diagnostic circuit that is embedded in decoders that are specific to the encoder system." Fahrny at [0023]: "Keylist: A list of decoder addresses and respective decoder keys in ordered pairs. Keylists may be used by the Uplink Control System (UCS) for generation of authorization messages that are addressed to the diagnostic circuit that is embedded in decoders that are specific to the encoder system."

'117 Claims	Exemplary Disclosures
	Fahrny at [0037]-[0038]: "For VOD services, a table of keys can be generated and delivered at session setup time. The keys for VOD service may be delivered with synchronization information related to key change as well as other information for short term working key epochs. A VOD Session ID or, alternatively, a Program ID may be used as an index to reference the keys list with the appropriate record of information for the VOD transport decryption. In alternative embodiments of reduced hierarchy key management of the present invention, one-way hashing may be implemented in the protection, selection and processing of the decryption key. "The reduced hierarchy key management of the present invention generally provides a new, more secure, and elegant system and method to deliver content keys for decrypting the program streams in conditional access systems (e.g., Broadcast and Video On Demand applications). The key management of the present invention may dramatically reduce the complexity required to deliver new content keys when a first Entitlement Message has been sent (i.e., presented, transmitted, provided, broadcast, etc.) to each set top box. The reduced hierarchy key management system and method of the present invention may be implemented as a portion of a new CAS system. The new CAS system generally provides for the manufacture and distribution of devices that are compatible with infrastructure, regardless of specific content security mechanisms that are used in that infrastructure. The new CAS system may provide far more efficient manufacturing, distribution and operations, and in fact enable new business models, including the retail availability of extremely low cost customer premises equipment (CPE) when compared to conventional approaches." Fahrny at [0041]: "Referring to FIG. 1 a, a diagram illustrating a media decoder (i.e., controller, processor, apparatus, circuit, device, etc.) 100 of the present invention is shown.

'117 Claims	Exemplary Disclosures
	The decoder 100 may be implemented in connection with a digital media stream distribution system (described in more detail in connection with FIGS. 2(a-b)). The controller 100 is generally implemented as a security processor (or processing system) that provides at least one security feature (e.g., encryption, decryption, authentication, security key management, copy protection, digital rights management, etc.) to at least one digital media input/output stream. The decoder 100 generally has an input 102 that receives at least one signal (e.g., VIDIN and PID), an input/output 104 that receives/presents a signal (e.g., TFHE) as well as additional data, signals, messages, and the like, an input 106 that receives a working key modifier and application function signal (e.g., WKM), and an output 108 that presents a signal (e.g., VIDOUT)." Fahrny at [0042]: "The streams VIDIN and VIDOUT may be implemented as digital media streams that may be in an encrypted and in a clear (i.e., unencrypted or decrypted) state (or condition), respectively. The streams VIDIN and VIDOUT are each generally implemented as a digital media signal stream (e.g., an MPEG, MPEG-2, etc. stream or other transport stream). In one example, the stream VIDOUT may be implemented as a decrypted (and decompressed) version of the stream VIDIN. However, the streams VIDIN and VIDOUT may be implemented having any appropriate format and protocol to meet the design criteria of a particular application." Fahrny at [0043]-[0044]: "The signal PID may be implemented as a program identifier for the respective program that has been selected by a user (e.g., customer, client, viewer, listener, etc.). The signal TFHE may be implemented as at least one entitlement management message (EMM) that is received from and presented to the headend via an out-of-band (OOB) transmission. The working key modifier and application

'117 Claims	Exemplary Disclosures
	function (e.g., a factor, an operator, or a combination of a factor and an operator that was applied to the content key to enhance encryption) WKM is generally combined with a content key to generate a working key that is used to decrypt an encrypted media stream (e.g., the stream VIDIN) to generate a clear output media stream (e.g., the media stream VIDOUT). "The controller 100 generally comprises a media stream transport decryption engine 110, a table 112, a list 114, and a combiner 116. The decoder 100 is generally implemented via at least one processor (e.g., microprocessor, controller, etc.) and at least one memory (e.g., random access memory (RAM), read only memory (ROM), NVROM, flash, EPROM, etc.) where one or more processes, routines, engines, lists, tables, etc. may be stored. The engine 110, the table 112, the list 114, and the combiner 116 are generally implemented within the processor and memory of the decoder 100." Fahrny at [0048]: "The list 114 generally contains a list of content keys that may be referenced by respective index values (e.g., the index IND). The contents of the list 114 may be loaded via the input/output 104. The list 114 may be configured to present a content key to the combiner 116 in response to the respective index IND. The content keys (and, alternatively or optionally, IV values) in the list 114 that correspond to a particular encrypted media stream VIDIN are selected from the content key (and, alternatively or optionally, IV) table 112 using an entry in the content key (and, alternatively or optionally, IV) index that is referenced by the identifier PID that is received from the headend in connection with the encrypted media stream VIDIN. Content keys and IVs that correspond to a particular encrypted media stream are selected from the content key and IV list using the index IND from a content key and IV table that is referenced by the identifier PID that is received from the

'117 Claims	Exemplary Disclosures
	headend in connection with the encrypted media stream when Cipher Block Chaining is used as the mode of a selected algorithm." Fahrny at [0054]-[0056]: "The controller 100" generally comprises the media stream transport decryption engine 110, a table 112", and a list 114". The decoder/controller 100" is generally implemented without a combiner such as the combiner 116 of the decoder 100. The engine 110 may receive a video content key (e.g., VK) instead of the working key WK. The engine 110 may generate and present the clear media stream VIDOUT in response to the media stream VIDIN and the decryption key VK. "The table 112" generally comprises a content key (and, alternatively or optionally, IV) index table. The contents of the table 112′ are generally loaded from the headend (e.g., via the input/output 104). The table 112" may receive the identifier PID or, alternatively, the identifier VODID via the input 102. The table 112" may present an index (e.g., IND") to the content key list 114" in response to the identifier PID or, alternatively, the identifier VODID based on the respective value in the table 112" using a key record index contained therein. "The list 114" generally contains a list of VOD content keys (e.g., the keys VK) that may be referenced by respective index values (e.g., the index IND"). The contents of the list 114" may be loaded via the input/output 104. The list 114" may be configured to present a content key to the engine 110 in response to the respective index IND". The VOD content keys (and, alternatively or optionally, IVs) VK in the list 114" that correspond to a particular VOD encrypted media stream VIDIN are selected from the content key (and, alternatively or optionally, IV) table 112" using an entry in the content key record index that is referenced by the identifier PID or, alternatively, the identifier VODID that is received from the headend in connection with the encrypted media stream VIDIN. The stream

'117 Claims	Exemplary Disclosures
	decryption keys VK are generally presented to the engine 110 on respective key epochs." *See also* Fahrny at Fig. 1a. Assmann at Abstract: "An electronic mail ("e-mail") system is described that enforces authentication or encryption to/from Mail Transfer Agents and from Mail User Agents. In one embodiment, for example, an electronic mail (e-mail) system is described that comprises: a message transfer agent (MTA) available for a client to connect to; program logic for authenticating the client, through use of a certificate, in order to identify the client as approved for establishing a secured connection with the MTA; program logic for establishing a secured connection between the client and the MTA in instances where the client can be authenticated; program logic for testing encryption strength of the secured connection; and program logic for terminating the secured connection in instances where the secured connection has inadequate encryption strength." Assmann at 4:25-36: "An electronic mail ("e-mail") system is described that provides methodology to enforce authentication or encryption to/from Mail Transfer Agents and from Mail User Agents. In accordance with the present invention, support is added to enforce certain restrictions on the connections between two hosts (a server and a client), depending on whether sendmail (Message Transfer Agent) acts as a server (receiving e-mail) or as a client (sending e-mail). In this manner, it is possible for each type of connection to enforce an authentication of the other side or at least a certain key length of the symmetric cipher used for encryption." Assmann at 7:62-8:19:

'117 Claims	Exemplary Disclosures
	"Methodology to Enforce Authentication or Encryption to/from an MTA or from a MUA "A. Including TLS Support within the Sendmail MTA "STARTTLS has been implemented in the sendmail MTA for Sendmail Secure Switch 2.0. It is based on existing solutions, including OpenSSL (see e.g., OpenSSL: The open source toolkit for ssl/tls, currently available online at http://www.openssl.org/) and example programs and a patch for Postfix (see e.g., Lutz Jaenicke, Postix, 2000, currently available at http://www.aet.tu-cottbus.de/personen/jaenicke/pfixtls/). X.509 certificates tie a Distinguished Name (DN) together with a public key and those data structures are digitally signed by a Certificate Authority (CA). According to the X.509 standard, a Distinguished Name consists of several named fields, such as Country (C), State (ST), Location (L), Organization (O), and a Common Name (CN). The DN of the CA is called the cert issuer, and the DN of the cert itself is called the cert subject. The sendmail binary (i.e., "sendmail") uses a configuration file (sendmail.cf) to define several options, routing of e-mail, rewriting of addresses, and rejection or acceptance of connections and relaying attempts. The interaction of sendmail with its configuration file is defined by options, macros, classes, rulesets, and mailer definitions. For example, the decision whether an e-mail is relayed is based on a ruleset called check rcpt." Assmann at 8:34-45: "TLS only allows one to control the connection between two hosts: a server and a client. In accordance with the present invention, therefore, support is added to enforce certain restrictions on these connections, depending on whether sendmail acts as a server (receiving e-mail) or as a client (sending e-mail). In this manner, it is possible for each type of connection to enforce an authentication of the other side or at least a certain key length of the symmetric cipher used for encryption. In order

'117 Claims	Exemplary Disclosures
	to understand how that may be done in accordance with the present invention, it is first helpful to briefly review (E)SMTP, the (Extended) Simple Mail Transfer Protocol." White at [0064]: "The present invention provides a comprehensive application-to-application messaging system and infrastructure for mobile users, enabling users to develop and deploy applications that exchange messages with remote applications located on a variety of devices (e.g., server systems and mobile devices running Windows, Windows CE, or other operating systems). The store-and-forward nature of the messaging solution provided by the present invention means that messages can be constructed even when the destination application is not reachable over the network; the message is delivered when the network becomes available. The system and methodology of the present invention provides for messages (sometimes referred to herein as "QAnywhere" messages) to be exchanged via a central server, so that the sender and recipient of a message never have to be connected to the network at the same time. Applications store messages in queues until a connection between the client and the server is available for message transmission. Applications use relational databases as a temporary message store. The relational database ensures that the message store has security, transaction-based computing, and the other benefits of relational databases. The use of a relational database as a message store also facilitates the use of the present invention in conjunction with a data synchronization solution, such as that available in SQL Anywhere® Studio. The present invention provides for extending this data synchronization mechanism to enable application-to-application messages to be exchanged between and among mobile devices and enterprise systems." White at [0065]:

'117 Claims	Exemplary Disclosures
	"The present invention also provides network-independent communication. QAnywhere messages can be transported over TCP/IP, HTTP, or HTTPS protocols. They can also be delivered from a Windows CE handheld device by ActiveSync. The message itself is independent of the network protocol, and can be received by an application that communicates over a different network. The present invention also handles the challenges of wireless networks, such as slow speed, spotty geographic coverage, and dropped network connections. It protects proprietary or sensitive information by encrypting all messages sent over public networks. In its currently preferred embodiment, the system compresses and, optionally, encrypts data sent between mobile applications and enterprise servers." Holbrook at [0107]: "The network communication events between messaging node 20 and central server 12 may be intermittent and unreliable, and messaging system 10 is robust in environments where reliable network communications cannot be assured. The system imposes no strict schedule concerning the frequency or timing of data exchange between central server 12 and any individual messaging node 20. Messaging node 20 does not need to maintain an accurate clock, or be assured continuous or regular access to an operational first communications link 70. Message traffic between elements of messaging system 10 may be encrypted to provide security and privacy."
5. The network system of claim 4, wherein the secure Internet data messages are transported to the device messaging agent on each device using one or more of encryption on a transport services stack, IP (Internet	The references below disclose and/or render obvious this limitation. For example, see the following passages and/or figures, as well as related disclosures: Huskins at 4:20-33: "In accordance with one embodiment of the present invention, the server 10 may implement a unidirectional messaging system. In a unidirectional messaging system,

'117 Claims	Exemplary Disclosures
Protocol) layer encryption, and tunneling.	the server 10 may transmit messages to a plurality of clients that are unable to respond in any way. One example of such a network is a direct-to-home (DTH) broadcast network that may be compliant with the DVB protocol. The network may use a connection oriented communication protocol or a real time connectionless communication protocol as two examples. There are many applications of unidirectional messaging from server to client such as instant messaging, command and control and notification and signaling, as examples. In other cases, the network may be a bidirectional network, for example with an Internet Protocol (IP) multicast backbone." Huskins at 5:1-18: "In one embodiment of the invention, the server 10 may include a unidirectional messaging server (UMS) 22 that is coupled to the servers 16, 18, 20 and 21 to generate messages in an appropriate format. The messages transmitted by the UMS server 22 may include messages originally generated by one of the servers 16, 18, 20 or 21. The UMS server 22 may then be coupled to an Internet Protocol multicast module 24 that places the messages in an appropriate multicast protocol format. Finally, a DVB Multiprotocol Encapsulation (MPE) 26 is coupled to the Internet protocol multicast module 24. The MPE is described in DVB Specification for Data Broadcasts (EN 301 192) and Specification for Service Information (SI) in DVB Systems (EN 300 468 VI .3.1 1998-02) both available from the ETS. The output of the DVB MPE 26 and a DVB-Service Information (SI) generator 28 are coupled to the transport 14. Service Information is digital data describing the delivery system, content and scheduling/timing of broadcast data streams." Huskins at 6:28-7:6: "Turning next to Figure 2, the software 50 on the client 12 initially receives the unidirectional messaging server address and port from the server 10. The client 12 may also be assigned a client identifier as indicated in block 52. Thus, an Internet

'117 Claims	Exemplary Disclosures
	Protocol multicast system may be established wherein each client has a UMS address and port as well as a unique client identifier, assigned by the server 10. In some embodiments, the server 10 may dynamically adjust addresses and ports as well as client identifiers to enable communication of particular messages, message groups or types of messages to particular clients in a dynamic and reconfigurable fashion." Huskins at 11:3-10: "Referring to Figure 4, the software 82 on the client 12 for implementing a network management session, initially receives the unidirectional messaging server address and port from the server 10. The client 12 may also be assigned a client identifier as indicated in block 84. Thus, an Internet Protocol multicast system may be established wherein each client has a UMS address and port as well as a unique client identifier, assigned by the server 10. Having receives its address, port and client identifier, the client 12 receiver joins a multicast group and listens for messages addressed specifically to it or to any groups that the client 12 belongs to, as indicated in block 86." Chou at Abstract: "Push content is generated in a wireless secure server [WSS] or other type of server, responsive to information received in the wireless secure server from the enterprise application. The push content is deliverable from the wireless secure server to the mobile client device over a wireless network." Chou at 3:33-47: "FIG. 1A shows an example communication system 100 in which the present invention is implemented. The system 100 includes a network 102, an enterprise application 104, and a wireless secure server (WSS) 106. The enterprise application 104, which is assumed to run on a computer, server or other processing element associated with a given enterprise, is separated from the WSS 106 via an enterprise

'117 Claims	Exemplary Disclosures
	firewall 108, and receives multimodal input such as voice, data, etc. from sources not shown in the figure. The system 100 further includes a wireless application protocol (WAP) mobile client device 110, which communicates with the WSS 106 over network 102. More specifically, the mobile client device 110 communicates with the WSS 106 via WAP gateway (WG) 112, and via a push proxy gateway (PPG) 114 and a push initiator (PI) 116, as shown." Chou at 6:42-45: "5. The gateway converts the HTTP response back into an encrypted, binary format and sends it to the mobile client device. 6. The mobile client device decodes the response and displays the results." Chou at 4:14-21: "The WSS 106 further includes a secure connector 120 which illustratively includes a password-protected register connector 122 and an application connector 124. Security can be provided in these elements using conventional encryption or other cryptographic operations of a type known in the art. The application connector 124 communicates alert and/or notification information to the mobile client 110 via PI 116 and PPG 114 as shown." Chou at 10:30-52: "The illustrative embodiment of the invention as described above provides a secure information access technique in which a user can make a service request through multimodal dialogue interaction with the enterprise application but does not have direct access to all the associated information content through the request channel. The enterprise application instead controls the content delivery through an HTTP-based interface to the WSS. Further, the content is transcoded and delivered in such a way that it is sent to a unique registered mobile client device. The location of the

'117 Claims	Exemplary Disclosures
	content and the associated procedure of pulling the content are protected by encryption, so as not to be apparent to a third party or even an intended user. Therefore, the mobile client device can be safely utilized in a crowded public area, such as in a bus or an airplane. In order to access the content, a potential attacker has to pass the authentication process through multimodal interaction with the enterprise application, and therefore must obtain both a registered mobile client device and the unique access code for that device. Moreover, even if the attacker is able to access the content in this manner, there is no direct pipe provided through the enterprise firewall, so the overall security of the enterprise communication services platform remains uncompromised." *See also* Chou at Fig. 1A. Anderson at 669-670: "Another approach is to do encryption and/or authentication at the IP layer using a protocol suite known as Ipsec. IPsec defines a *security association* as the combination of keys, algorithms and parameters used to protect a particular packet stream. Protected packets are either encrypted or authenticated (or both); in the latter case, an authentication header is added that protects data integrity using HMAC-SHA1, while in the former the packet is encrypted and encapsulated in other packets. (The use of encryption without authentication is discouraged as it's insecure [151].) There's also an *Internet Key Exchange* (IKE) protocol to set up keys and negotiate parameters. IKE has been through a number of versions (some of the bugs that were fixed are discussed in [465]). "IPsec is widely used by firewall vendors who offer a *virtual private network* facility with their products; that is, by installing one of their boxes in each branch between the local LAN and the router, all the internal traffic can pass encrypted over the Internet. Individual PCs, such as workers' laptops and home PCs, can in theory join a VPN given a firewall that supports IPsec, but this is harder than it

'117 Claims	Exemplary Disclosures
	looks. Compatibility has been a major problem with different manufacturers' offerings just not working with each other; although firewall-to-firewall compatibility has improved recently, getting random PCs to work with a given VPN is still very much a hit-or-miss affair." Anderson at 670-71: "Recall that when discussing public key encryption, I remarked that a server could publish a public key *KS* and any web browser could then send a message *M* containing a credit card number to it encrypted using *KS*: $\{M\}_{KS}$. This is in essence what the TLS protocol (formerly known as SSL) does, although in practice it is more complicated. It was developed to support encryption and authentication in both directions, so that both http requests and responses can be protected against both eavesdropping and manipulation. It's the protocol that's activated when you see the padlock on your browser toolbar." "Here is a simplified description of the version as used to protect web pages that solicit credit card numbers: 1. the client sends the server a *client hello* message that contains its name *C*, a transaction serial number *C#*, and a random nonce N_C; 2. the server replies with a *server hello* message that contains its name *S*, a transaction serial number *S#*, a random nonce N_S, and a certificate *CS* containing its public key *KS*. The client now checks the certificate *CS* back to a root certificate issued by a company such as Verisign and stored in the browser; 3. the client sends a *key exchange* message containing a *pre-master-secret* key, K_0, encrypted under the server public key *KS*. It also sends a *finished* message with a message authentication code (MAC)computed on all the messages to date. The key for this MAC is the *master-secret*, K_1. This key is computed by hashing the pre-master-secret key with the nonces sent by the client and server: $K_1 = h(K_{CS},N_C,N_S)$. From this point onward, all the traffic is encrypted; we'll write this

'117 Claims	Exemplary Disclosures
	as $\{\ldots\}_{KCS}$ in the client-server direction and $\{\ldots\}_{KSC}$ from the server to the client. These keys are generated in turn by hashing the nonces with K_1. 4. The server also sends a finished message with a MAC computed on all the messages to date. It then finally starts sending the data. $C \rightarrow S : C, C\#, N_C$ $S \rightarrow C : S, S\#, N_S, CS$ $C \rightarrow S : [K0]_{KS}$ $C \rightarrow S : [finished, MAC(K_1, everythingtodate)]_{KCS}$ $S \rightarrow C : [finished, MAC(K_1, everythingtodate)\}_{KSC}, [data]_{KSC}$ "The design goals included minimizing the load on the browser, and then minimizing the load on the server. Thus the public key encryption operation is done by the client, and the decryption by the server; the standard encryption method (*ciphersuite*) uses RSA for which encryption can be arranged to be very much faster than decryption. (This was a wrong design decision as browsers generally have a lot more compute cycles to spare than servers; it has created a brisk aftermarket for crypto accelerator boards for web servers.) Also, once a client and server have established a pre-master secret, no more public key operations are needed as further master secrets can be obtained by hashing it with new nonces." *See generally* Anderson at Ch. 20 ("Telecom System Security"); § 21.4.5.6 ("TLS"). Ozaki at 3:11-39: "The data-link layer (second layer) is a layer which defines packet format, physical node addresses, a transmitting and receiving method of packets and the like. Layer 2 defines a protocol for interacting packets between two nodes through a physical communication medium. In this second layer, some kind of address or Medium Access Control (MAC) identifier is added with respect to each node and packets are

'117 Claims	Exemplary Disclosures
	transmitted on the communication medium by specifying a transmission source of the packets. As to communication mediums, there are diversified types of copper wiring, wireless, optical fiber and the like. Also, with respect to the connection mode (topology), there is not only point-to-point connection but also bus, star, and link topologies. The packet transmitted on the communication medium is provides to a node at a reception side where it is processed with respect to the remaining OSI layers. "For example, a NIC (Network Interface Card) functions in conjunction with an associated driver to operate at the physical layer and data-link layer for connecting a personal computer, a printer or the like to a premises network (LAN). In many cases, the NIC interfaces to an Ethernet. "By means of the NIC, a node (apparatus) which wants to transmit data awaits a vacant time slot to start a transmission via the Ethernet. At that time, if a plurality of nodes start transmissions simultaneously, data collide in the cable and are broken-down, so that transmitting nodes stop transmissions and will restart transmissions after waiting a random of time. In this manner, it is possible for a plurality of nodes to share a single cable so as to communicate mutually." Ozaki at 10:55-64: "FIG. 3 is a communication layer diagram of a system in a second exemplified embodiment (broadcasting application by UDPsec) of a communication system using the protocol according to the present invention; "FIG. 4A is a diagram showing a packet structure of encryption regions and application regions of completeness authentication with respect to TCPsec/IPsec; "FIG. 4B is a diagram showing a packet structure of encryption regions and application regions of completeness authentication with respect to TCPsec/IP;"

'117 Claims	Exemplary Disclosures
	See also Ozaki at Figs. 3, 4A, 4B. Novack at [0039]: "As an example, a secure sockets layer (SSL) may be used to demand authentication from the requesting computer. According to the protocols of the secure sockets layer, also known as hyper-text transfer protocol secure, a communications request involves a request for authentication. Accordingly, the computer that receives the initial communications request may initiate a secure sockets layer session to authenticate the requestor." Novack at [0043]: "Confidentiality of the transmission is accomplished by encrypting it with the message recipient's public key, as only the holder of the associated private key can decrypt the message. The authentication of source is assured by a "digital signature" mechanism, which is generated by encrypting a secure digest of the message with the private key of the sender. A digital certificate may be stored on any type of computer-readable media, including hard drives, smart cards, flash memory, floppy memory and floppy drives. The digital certificate is installed on a computer and made available to provide secure communications. The digital certificate is retrieved and forwarded to a recipient when a particular process is run and inter-process communications are initiated with the recipient. The digital certificate may be used as a generic authentication tool for all communications involving a particular device or user (i.e., including all inter-process communications), or may be used for only one or more predetermined processes involved in inter-process communications. As an example of the use of digital certificates, according to the secure sockets layer protocols, computers exchange digital certificates to secure communications and authenticate sources and destinations.." Novack at [0054]:

'117 Claims	Exemplary Disclosures
	"FIG. 4 shows an exemplary device layout for a device that employs secure inter-process communications. The **device** 400 includes a **port** 405 which is used to communicate according to a particular communications protocol. The communications sent and received by the **port** 405 are processed by a **processor** 410. The **device** 400 also includes **internal memory** 420, which may be read only memory (ROM) and/or random access memory (RAM). As an example, read only memory may store fixed operating procedures, e.g., operating system instructions, for the device including instructions for routines and subroutines unrelated to the security of inter-process communications. The random access memory may store data as directed by a programmer or user of the device." Novack at [0064]: "The device in **FIG.** 4 may include multiple processes that communicate with each other using the secure inter-process communications of the present invention. Additionally, although the terms "client" and "server" are used to refer to devices, the devices may be defined and distinguished by a set of software instructions installed as a pre-packaged program. Further, while examples of the uses of the present invention include secure client/server networks and peer-to-peer networks, secure inter-process communications can be used to enhance security in any inter-process communications environment. Further, while the embodiment shown in **FIG.** 2 only shows a **single computer** 201 and a **single application server** 230, secure inter-process communications can be used to enhance security for **multiple application servers** 230 belonging to a single entity or multiple **users using computers** 201. Additionally, while **networks** 220 and 240 are separately designated in **FIG.** 2, the networks may be the same, such as an internet protocol (IP) network. Further, while a certification authority is discussed in the context of the present invention, a certification authority is not shown in the figures. In this regard, a certification authority may communicate over the **network** 220 and/or the **network** 240 to provide digital certificates to the **application server** 230 and the **management server** 250. Of

'117 Claims	Exemplary Disclosures
	course, the digital certificates may be provided by the certification authority as portable memory modules that are installed on the **application server** 230 or the **management server** 250. Furthermore, the entity that provides the **application server** 230 and/or the **management server** 250 may also act as a certification authority for its customers using the computers 102." Novack at [0068]: "Although the present specification describes components and functions implemented in the embodiments with reference to particular standards and protocols, the invention is not limited to such standards and protocols. Each of the standards for digital certificate format (e.g., X.509), transfer protocols (e.g., HTTP, HTTPS/SSL), packet switched network transmission (e.g., IP) and interface protocols and architectures (e.g., CORBA, RSI) represent examples of the state of the art. Such standards are periodically superseded by faster or more efficient equivalents having essentially the same functions. Accordingly, replacement standards and protocols having the same functions are considered equivalent." Santamaria at [0007]: "A mobile device establishes a data context (e.g., a Packet Data Protocol (PDP) context) with one or more network elements that provide data services to the mobile device (e.g., Internet Protocol traffic from the Internet) over a network (e.g., a cellular network such as a Generic Packet Radio Services (GPRS) network, and/or a Local Area Network (LAN)). The data context is a logical association between the mobile device and the network elements, and includes information relating to routing (e.g., IP address information), Quality of Service (QoS), billing, authentication, etc. Since maintaining a data context consumes network resources of the network elements, the one or more servers may teardown a data context associated with a mobile device if the mobile device is not actively using the data context. For example, if the mobile device is turned off (and thus is not using the data context and not contacting the

'117 Claims	Exemplary Disclosures
	network elements), the network elements may teardown the data context after some time of not receiving information from the mobile device. The mobile device may periodically refresh the data context to keep the data context connection up." Santamaria at [0038]: "At block 210, the mobile device 160 establishes a data context. The data context may be established by any number of methods known in the art. From block 210, flow moves to block 220 where the mobile device 160 establishes one or more notification services connections. For example, referring to FIG. 1, the mobile device 160 establishes a connection with the push notification services 110 and 120 and the pull notification service 130. In some embodiments of the invention, the connections are each Transmission Control Protocol (TCP) connections." Okonnen at [0020]: "In an embodiment according to the present invention, the network may further comprise update agent provisioning information stored in the electronic device. The update agent provisioning information may comprise a device server URL, an index of provisioned update agents, a security key, and electronic device related information. The device server URL may provide references to servers hosting updates to be downloaded. The updates may be compatible with update agents currently available and provisioned in the electronic device." Okonnen at [0052]: "The provisioned data 129 may also be used to store update agent related provisioning information, such as a universal resource locator (URL) of a device server or a management server that may be used to retrieve updates, a security key(s) that may be used to authenticate messages to/from the device server, and selective encryption/decryption of the messages, etc. Each update agent, such as

'117 Claims	Exemplary Disclosures
	update agent A 113 and update agent B 115, may have corresponding entries in provisioned data 129." Okonnen at [0060]: "In an embodiment according to the present invention, update agent A 113 may be employed to update firmware 117 and operating system 119 during reboot of the mobile handset 107. After a reboot update, update agent B 115 may be used to update other components in the mobile handset 107, for example, configuration parameters, gaming engines, third-party software, etc., that may not be updated during reboot of the mobile handset 107. Both of update agent A 113 and update agent B 115 may be provisioned with security keys for secure communications with device servers 135 and 137 during update agent provisioning activity, such as provisioning activity accompanying NAM provisioning activity and/or incorporated into NAM programming activity." Assmann at 7:62-8:19: "Methodology to Enforce Authentication or Encryption to/from an MTA or from a MUA "A. Including TLS Support within the Sendmail MTA "STARTTLS has been implemented in the sendmail MTA for Sendmail Secure Switch 2.0. It is based on existing solutions, including OpenSSL (see e.g., OpenSSL: The open source toolkit for ssl/tls, currently available online at http://www.openssl.org/) and example programs and a patch for Postfix (see e.g., Lutz Jaenicke, Postix, 2000, currently available at http://www.aet.tu-cottbus.de/personen/jaenicke/pfixtls/). X.509 certificates tie a Distinguished Name (DN) together with a public key and those data structures are digitally signed by a Certificate Authority (CA). According to the X.509 standard, a Distinguished Name consists of several named fields, such as Country (C), State (ST), Location

'117 Claims	Exemplary Disclosures
	(L), Organization (O), and a Common Name (CN). The DN of the CA is called the cert issuer, and the DN of the cert itself is called the cert subject. The sendmail binary (i.e., "sendmail") uses a configuration file (sendmail.cf) to define several options, routing of e-mail, rewriting of addresses, and rejection or acceptance of connections and relaying attempts. The interaction of sendmail with its configuration file is defined by options, macros, classes, rulesets, and mailer definitions. For example, the decision whether an e-mail is relayed is based on a ruleset called check rcpt." Assmann at 8:34-45: "TLS only allows one to control the connection between two hosts: a server and a client. In accordance with the present invention, therefore, support is added to enforce certain restrictions on these connections, depending on whether sendmail acts as a server (receiving e-mail) or as a client (sending e-mail). In this manner, it is possible for each type of connection to enforce an authentication of the other side or at least a certain key length of the symmetric cipher used for encryption. In order to understand how that may be done in accordance with the present invention, it is first helpful to briefly review (E)SMTP, the (Extended) Simple Mail Transfer Protocol." White at [0035]: "Synchronization: Synchronization refers the process of data and/or message synchronization between a client and a server in which data and/or messages are downloaded (i.e., transmitted from the server to the client) and/or uploaded (i.e., transmitted from the client to the server). In the currently preferred embodiment of the present invention, a MobiLink data synchronization mechanism is used for synchronization of data and/or messages. The MobiLink synchronization system is available as a component of SQL Anywhere® Studio (available from iAnywhere Solutions, Inc. of Dublin, Calif.). MobiLink offers bidirectional exchange of

'117 Claims	Exemplary Disclosures
	information between remote databases (e.g., Adaptive Server® Anywhere or UltraLite databases available from iAnywhere Solutions, Inc.) and a variety of enterprise data sources including Adaptive Server® Anywhere, Sybase® Adaptive Server® Enterprise, Oracle, Microsoft SQL Server, IBM DB2, and other ODBC-compliant databases. Remote systems connect via standard Internet protocols such as TCP/IP, HTTP or HTTPS to the MobiLink synchronization server, which communicates with a back end (consolidated) database. MobiLink is a session-based synchronization system that allows two-way synchronization between a main database, called the consolidated database, and many remote databases. The consolidated database holds the master copy of all the data. Synchronization typically begins when a MobiLink remote site opens a connection to a MobiLink synchronization server. During synchronization, the MobiLink client at the remote site uploads database changes that were made to the remote database since the previous synchronization. On receiving this data, the MobiLink synchronization server updates the consolidated database, and then downloads changes on the consolidated database to the remote database." White at [0065]: "The present invention also provides network-independent communication. QAnywhere messages can be transported over TCP/IP, HTTP, or HTTPS protocols. They can also be delivered from a Windows CE handheld device by ActiveSync. The message itself is independent of the network protocol, and can be received by an application that communicates over a different network. The present invention also handles the challenges of wireless networks, such as slow speed, spotty geographic coverage, and dropped network connections. It protects proprietary or sensitive information by encrypting all messages sent over public networks. In its currently preferred embodiment, the system compresses and, optionally, encrypts data sent between mobile applications and enterprise servers."

'117 Claims	Exemplary Disclosures
6. The network system of claim 1, wherein the device messaging agent executes in a secure execution environment on at least one of the devices, and at least one of the applications executes outside of the secure execution environment on that device.	The references below disclose and/or render obvious this limitation. For example, see the following passages and/or figures, as well as related disclosures: Chou at 2:14-18: "The enterprise application is preferably separated from the wireless secure server via an enterprise firewall, with the enterprise application being inside the enterprise firewall and the wireless secure server being outside the enterprise firewall." Ellison at 4:57-5:10: "The processor 110 represents a central processing unit of any type of architecture, such as complex instruction set computers (CISC), reduced instruction set computers (RISC), very long instruction word (VLIW), or hybrid architecture. In one embodiment, the processor 110 is compatible with an Intel Architecture (IA) processor, such as the Pentium series, the IA-32™ and the IA-64™. The processor 110 includes a normal execution mode 112 and an isolated execution circuit 115. The normal execution mode 112 is the mode in which the processor 110 operates in a non-secure environment, or a normal environment without the security features provided by the isolated execution mode. The isolated execution circuit 115 provides a mechanism to allow the processor 110 to operate in an isolated execution mode. The isolated execution circuit 115 provides hardware and software support for the isolated execution mode. This support includes configuration for isolated execution, definition of an isolated area, definition (e.g., decoding and execution) of isolated instructions, generation of isolated access bus cycles, and generation of isolated mode interrupts." Ellison at 6:1-26: "The system memory 140 stores system code and data. The system memory 140 is typically implemented with dynamic random access memory (DRAM) or static random access memory (SRAM). The system memory 140 includes the accessible physical memory 60 (shown in FIG. 11B). The accessible physical memory

'117 Claims	Exemplary Disclosures
	includes a loaded operating system 142, the isolated area 70 (shown in FIG. 1B), and an isolated control and status space 148. The loaded operating system 142 is the portion of the operating system that is loaded into the system memory 140. The loaded OS 142 is typically loaded from a mass storage device via some boot code in a boot storage such as a boot read only memory (ROM). The isolated area 70, as shown in FIG. 1B, is the memory area that is defined by the processor 110 when operating in the isolated execution mode. Access to the isolated area 70 is restricted and is enforced by the processor 110 and/or the MCH 130 or other chipset that integrates the isolated area functionalities. The isolated control and status space 148 is an input/output (I/O)-like, independent address space defined by the processor 110 and/or the MCH 130. The isolated control and status space 148 contains mainly the isolated execution control and status registers. The isolated control and status space 148 does not overlap any existing address space and is accessed using the isolated bus cycles. The system memory 140 may also include other programs or data which are not shown." *See also* Ellison at Figs. 1A, 1C. Anderson at 669-670: "Another approach is to do encryption and/or authentication at the IP layer using a protocol suite known as Ipsec. IPsec defines a *security association* as the combination of keys, algorithms and parameters used to protect a particular packet stream. Protected packets are either encrypted or authenticated (or both); in the latter case, an authentication header is added that protects data integrity using HMAC-SHA1, while in the former the packet is encrypted and encapsulated in other packets. (The use of encryption without authentication is discouraged as it's insecure [151].) There's also an *Internet Key Exchange* (IKE) protocol to set up keys and negotiate parameters. IKE has been through a number of versions (some of the bugs that were fixed are discussed in [465]). "IPsec is widely used by firewall vendors who offer a *virtual private network* facility with their products; that is, by installing one of their boxes in each branch

'117 Claims	Exemplary Disclosures
	between the local LAN and the router, all the internal traffic can pass encrypted over the Internet. Individual PCs, such as workers' laptops and home PCs, can in theory join a VPN given a firewall that supports IPsec, but this is harder than it looks. Compatibility has been a major problem with different manufacturers' offerings just not working with each other; although firewall-to-firewall compatibility has improved recently, getting random PCs to work with a given VPN is still very much a hit-or-miss affair." Anderson at 670-71: "Recall that when discussing public key encryption, I remarked that a server could publish a public key *KS* and any web browser could then send a message *M* containing a credit card number to it encrypted using *KS*: $\{M\}_{KS}$. This is in essence what the TLS protocol (formerly known as SSL) does, although in practice it is more complicated. It was developed to support encryption and authentication in both directions, so that both http requests and responses can be protected against both eavesdropping and manipulation. It's the protocol that's activated when you see the padlock on your browser toolbar." "Here is a simplified description of the version as used to protect web pages that solicit credit card numbers: 1. the client sends the server a *client hello* message that contains its name *C*, a transaction serial number *C#*, and a random nonce N_C; 2. the server replies with a *server hello* message that contains its name *S*, a transaction serial number *S#*, a random nonce N_S, and a certificate *CS* containing its public key *KS*. The client now checks the certificate *CS* back to a root certificate issued by a company such as Verisign and stored in the browser; 3. the client sends a *key exchange* message containing a *pre-master-secret* key, K_0, encrypted under the server public key *KS*. It also sends a *finished* message with a message authentication code (MAC)computed on all the messages to date. The key for this MAC is the *master-secret*, K_1. This key is computed by hashing

'117 Claims	Exemplary Disclosures
	the pre-master-secret key with the nonces sent by the client and server: $K_1 = h(K_{CS},N_C,N_S)$. From this point onward, all the traffic is encrypted; we'll write this as $\{\ldots\}_{KCS}$ in the client-server direction and $\{\ldots\}_{KSC}$ from the server to the client. These keys are generated in turn by hashing the nonces with K_1. 4. The server also sends a finished message with a MAC computed on all the messages to date. It then finally starts sending the data. $C \rightarrow S : C,C\#, N_C$ $S \rightarrow C : S, S\#,N_S,CS$ $C \rightarrow S : [K0]_{KS}$ $C \rightarrow S : [finished,MAC(K_1,everythingtodate)]_{KCS}$ $S \rightarrow C : [finished,MAC(K_1,everythingtodate)\}_{KSC},[data]_{KSC}$ "The design goals included minimizing the load on the browser, and then minimizing the load on the server. Thus the public key encryption operation is done by the client, and the decryption by the server; the standard encryption method (*ciphersuite*) uses RSA for which encryption can be arranged to be very much faster than decryption. (This was a wrong design decision as browsers generally have a lot more compute cycles to spare than servers; it has created a brisk aftermarket for crypto accelerator boards for web servers.) Also, once a client and server have established a pre-master secret, no more public key operations are needed as further master secrets can be obtained by hashing it with new nonces." *See generally* Anderson at Ch. 20 ("Telecom System Security"); § 21.4.5.6 ("TLS"). Okonnen at [0060]: "In an embodiment according to the present invention, update agent A 113 may be employed to update firmware 117 and operating system 119 during reboot of the mobile handset 107. After a reboot update, update agent B 115 may be used to update other components in the mobile handset 107, for example, configuration parameters, gaming engines, third-party software, etc., that may not be updated

'117 Claims	Exemplary Disclosures
	during reboot of the mobile handset 107. Both of update agent A 113 and update agent B 115 may be provisioned with security keys for secure communications with device servers 135 and 137 during update agent provisioning activity, such as provisioning activity accompanying NAM provisioning activity and/or incorporated into NAM programming activity." Hashimoto at [0038]-[0039]: "Fig. 1 shows a multi-party application protection environment to which the present invention is to be applied. "A software vendor 121 produces a program 122, and distributes it to a microprocessor-A 103 provided inside a target system 102 of a system user 112. In order to protect the program to be distributed, the plaintext program 122 is encrypted." Hashimoto at [0040]: "More specifically, the program formed by execution codes and data such as initialization data is encrypted by using a program key selected by the software vendor 121, and the program key used for the encryption is encrypted by using a public key-A of the processor-A of the target system 102 to generate a distribution key." Hashimoto at [0041]: "At this point, the software vendor 121 divides the program and the initialization data into a plurality of blocks, and encrypts them by using a different key for each block. Note however that there is no need for the software vendor 121 to select as many keys as the number of blocks, and it suffices to select two things including a constant Cx as a basis for generating a plurality of keys, and a key Kx for encrypting the plurality of keys generated according to Cx. As will be described in detail below, a block encrypted program 123 is generated by the encryption using a different key for each block, according to these two constants. Then, Cx and Kx are

'117 Claims	Exemplary Disclosures
	encrypted by using the public key-A of the processor-A to produce a distribution key Kx[A], Cx[A]." *See also* Hashimoto at Fig. 1. James at 1: **Secure Portable Execution Environments: A Review of Available Technologies** Peter James[1] Secure Systems Ltd pjames@securesystems.com.au **Abstract** *Live operating systems and virtualisation allow a known, defined, safe and secure execution environment to be loaded in to a PC's memory and executed with either minimal or possibly no reliance on the PC's internal hard disk drive. The ability to boot a live operating system or load a virtual environment (containing an operating system) from a USB storage device allows a secure portable execution environment to be created. Portable execution environments have typically been used by technologists, for example to recover data from a failing PC internal hard disk drive or to perform forensic analysis. However, with the commercial potential of portable execution environments becoming realised the requirement for such environments to be secure is becoming increasingly important. To be considered truly secure a portable execution environment should require authentication prior to loading the executing environment (from the USB mass storage device) and provide full encryption of the whole mass storage device.* *This paper discusses the outcomes from building four portable execution environments, using commercially available and/or freeware technologies. An overview is given of the emerging commercial requirement for secure portable USB execution environments, the security threats addressed and research performed in the area. The technologies and products considered in the review are outlined together with rationale behind the selection. The findings from the implementation of the four portable execution environments are discussed including successes, failures and difficulties encountered. A set of security requirements is defined which is used to gauge the effectiveness of each of the four environments.* *See also* James at 1-17. Taylor at 1:

Extended Abstract: *A Hardware-Assisted Data Hiding Based Approach in Building High Performance Secure Execution Systems*

Malcolm Taylor
Computer Science and Electrical Engineering Department
University of Maryland, Baltimore County, MD, USA
mtaylor2@umbc.edu

Chi-En Yin, Min Wu, and Gang Qu
Electrical and Computer Engineering Department and
Institute for Advanced Computer Studies
University of Maryland, College Park, MD, USA
{chienyin, minwu, gangqu}@umd.edu

***Abstract*— Recently, a novel data hiding technique was proposed to embed information into compiled binary codes in order to enhance system performance. Using this technique as a vehicle, we propose a framework to build high-performance secure execution systems by embedding data for security and trust purposes. We illustrate this approach in a mobile computing environment by an instruction-block level data hiding technique, which can be used to defend against malicious attacks (such as Trojan injection). This improves the trustworthiness of mobile codes. It also protects the code provider's intellectual property because the code can be executed only on the designated device. When combined with the existing approach in [4], the proposed data hiding framework can provide trust and high-performance simultaneously. Finally, we conduct a proof-of-the-concept FPGA prototyping to validate the data hiding technique and evaluate the hardware cost in terms of gate count, power consumption, and gate delay.**

Keywords - data hiding, secure execution, Trojan horse detection

I. INTRODUCTION

A. Motivation

With the increasing popularity of embedded and portable devices, a wide range of mobile codes are created to meet a diverse range of demands such as gaming, video on demand, security patches, and service updates. Trustworthiness of these mobile codes is one of the many open issues that need to be addressed. Consider the scenario that one downloads a game onto his cell phone. The user wants to make sure that the code running on his cell phone is trusted. However, such trust cannot be taken for granted even when the code provider is trusted. There are so many malicious and unintentional attempts that could cause damage to the device and leak of personal information stored on the device when untrusted codes are running.

For example, attackers can forge the code or insert Trojans into the code, among many other networking attacks. Although there exist security countermeasures to defend these attacks (such as encryption and data integrity), adding them onto the device inevitably increases the cost (in terms of price, speed, power, and so on). It may also cause user inconvenience.

Meanwhile, code/application providers may also demand a secure execution environment on the user's mobile device to protect their intellectual properties. They want to ensure that a copy of the code that they send to Alice can only be executed on her device. If Bob copies the code from Alice's device or Oscar intercepts the code during its transmission, neither Bob nor Oscar can run the code on their devices. This can be achieved by current technologies (such as that in iPod), but again at the price of higher cost. The main challenge that we will address in this paper is how to balance the tradeoff between the necessary security measures needed to build trust and the cost overhead to do so in terms of silicon area, computational complexity, and power consumption

B. Contributions

In this paper, we propose a data hiding based approach that simultaneously provides high performance and security, two of the most important design requirements for modern computing systems. The enabling technology is a recently proposed method that embeds information into compiled program binaries in order to enhance system performance [4].

We first extend this technique to build secure execution by hiding data integrity related information at instruction level. (1) The proposed scheme can detect code modification at instruction-block level. (2) It also allows successful execution if the modified code is not in the same block of the execution path. (3) It helps to protect code provider's intellectual property by allowing the code to be recovered only on designated device. (4) As we will survey in the next section, existing Trojan detection schemes either cannot detect new Trojan horses at real time or have to sacrifice system performance to achieve that. Ours has the potential to improve performance if combined with the previously proposed method in [4].

We then propose to use a light add-on hardware component to extract the embedded data and perform security related check. We conduct a proof-of-concept FPGA prototype to evaluate the hardware cost in terms of gate count, power consumption, and delay. To illustrate our design concept, the block-based hash function SHA-1 and AES with ECB mode are used to guarantee data integrity and confidentiality respectively.

'117 Claims	Exemplary Disclosures
	See also Taylor at 5-8.
9. The network system of claim 1, further comprising, on at least one of the devices, the secure interprocess communication service, and wherein the secure interprocess communication service and the secure Internet data connection from that device to the network message server are separately secured.	The references below disclose and/or render obvious this limitation. For example, see the following passages and/or figures, as well as related disclosures: Ellison at 4:57-5:10: "The processor 110 represents a central processing unit of any type of architecture, such as complex instruction set computers (CISC), reduced instruction set computers (RISC), very long instruction word (VLIW), or hybrid architecture. In one embodiment, the processor 110 is compatible with an Intel Architecture (IA) processor, such as the Pentium series, the IA-32™ and the IA-64™. The processor 110 includes a normal execution mode 112 and an isolated execution circuit 115. The normal execution mode 112 is the mode in which the processor 110 operates in a non-secure environment, or a normal environment without the security features provided by the isolated execution mode. The isolated execution circuit 115 provides a mechanism to allow the processor 110 to operate in an isolated execution mode. The isolated execution circuit 115 provides hardware and software support for the isolated execution mode. This support includes configuration for isolated execution, definition of an isolated area, definition (e.g., decoding and execution) of isolated instructions, generation of isolated access bus cycles, and generation of isolated mode interrupts." Ellison at 6:1-26: "The system memory 140 stores system code and data. The system memory 140 is typically implemented with dynamic random access memory (DRAM) or static random access memory (SRAM). The system memory 140 includes the accessible physical memory 60 (shown in FIG. 11B). The accessible physical memory includes a loaded operating system 142, the isolated area 70 (shown in FIG. 1B), and an isolated control and status space 148. The loaded operating system 142 is

'117 Claims	Exemplary Disclosures
	the portion of the operating system that is loaded into the system memory 140. The loaded OS 142 is typically loaded from a mass storage device via some boot code in a boot storage such as a boot read only memory (ROM). The isolated area 70, as shown in FIG. 1B, is the memory area that is defined by the processor 110 when operating in the isolated execution mode. Access to the isolated area 70 is restricted and is enforced by the processor 110 and/or the MCH 130 or other chipset that integrates the isolated area functionalities. The isolated control and status space 148 is an input/output (I/O)-like, independent address space defined by the processor 110 and/or the MCH 130. The isolated control and status space 148 contains mainly the isolated execution control and status registers. The isolated control and status space 148 does not overlap any existing address space and is accessed using the isolated bus cycles. The system memory 140 may also include other programs or data which are not shown." *See also* Ellison at Figs. 1A, 1C. Anderson at 669-670: "Another approach is to do encryption and/or authentication at the IP layer using a protocol suite known as Ipsec. IPsec defines a *security association* as the combination of keys, algorithms and parameters used to protect a particular packet stream. Protected packets are either encrypted or authenticated (or both); in the latter case, an authentication header is added that protects data integrity using HMAC-SHA1, while in the former the packet is encrypted and encapsulated in other packets. (The use of encryption without authentication is discouraged as it's insecure [151].) There's also an *Internet Key Exchange* (IKE) protocol to set up keys and negotiate parameters. IKE has been through a number of versions (some of the bugs that were fixed are discussed in [465]). "IPsec is widely used by firewall vendors who offer a *virtual private network* facility with their products; that is, by installing one of their boxes in each branch between the local LAN and the router, all the internal traffic can pass encrypted over the Internet. Individual PCs, such as workers' laptops and home PCs, can in

'117 Claims	Exemplary Disclosures
	theory join a VPN given a firewall that supports IPsec, but this is harder than it looks. Compatibility has been a major problem with different manufacturers' offerings just not working with each other; although firewall-to-firewall compatibility has improved recently, getting random PCs to work with a given VPN is still very much a hit-or-miss affair." Anderson at 670-71: "Recall that when discussing public key encryption, I remarked that a server could publish a public key *KS* and any web browser could then send a message *M* containing a credit card number to it encrypted using *KS*: $\{M\}_{KS}$. This is in essence what the TLS protocol (formerly known as SSL) does, although in practice it is more complicated. It was developed to support encryption and authentication in both directions, so that both http requests and responses can be protected against both eavesdropping and manipulation. It's the protocol that's activated when you see the padlock on your browser toolbar." "Here is a simplified description of the version as used to protect web pages that solicit credit card numbers: 1. the client sends the server a *client hello* message that contains its name *C*, a transaction serial number *C#*, and a random nonce N_C; 2. the server replies with a *server hello* message that contains its name *S*, a transaction serial number *S#*, a random nonce N_S, and a certificate *CS* containing its public key *KS*. The client now checks the certificate *CS* back to a root certificate issued by a company such as Verisign and stored in the browser; 3. the client sends a *key exchange* message containing a *pre-master-secret* key, K_0, encrypted under the server public key *KS*. It also sends a *finished* message with a message authentication code (MAC)computed on all the messages to date. The key for this MAC is the *master-secret*, K_1. This key is computed by hashing the pre-master-secret key with the nonces sent by the client and server: $K_1 = h(K_{CS},N_C,N_S)$. From this point onward, all the traffic is encrypted; we'll write this

'117 Claims	Exemplary Disclosures
	as $\{\ldots\}_{KCS}$ in the client-server direction and $\{\ldots\}_{KSC}$ from the server to the client. These keys are generated in turn by hashing the nonces with K_1. 4. The server also sends a finished message with a MAC computed on all the messages to date. It then finally starts sending the data. $C \rightarrow S : C, C\#, N_C$ $S \rightarrow C : S, S\#, N_S, CS$ $C \rightarrow S : [K0]_{KS}$ $C \rightarrow S : [finished, MAC(K_1, everythingtodate)]_{KCS}$ $S \rightarrow C : [finished, MAC(K_1, everythingtodate)\}_{KSC}, [data]_{KSC}$ "The design goals included minimizing the load on the browser, and then minimizing the load on the server. Thus the public key encryption operation is done by the client, and the decryption by the server; the standard encryption method (*ciphersuite*) uses RSA for which encryption can be arranged to be very much faster than decryption. (This was a wrong design decision as browsers generally have a lot more compute cycles to spare than servers; it has created a brisk aftermarket for crypto accelerator boards for web servers.) Also, once a client and server have established a pre-master secret, no more public key operations are needed as further master secrets can be obtained by hashing it with new nonces." *See generally* Anderson at Ch. 20 ("Telecom System Security"); § 21.4.5.6 ("TLS"). Novack at Abstract: "Inter-process communications between a first process and a second process are secured. A first process recognizes that inter-process communications must be initiated with a second process. A first computer that supports the first process submits a request to initiate inter-process communications with a second computer that supports the second process. The second computer replies to a request to initiate

'117 Claims	Exemplary Disclosures
	inter-process communications by replying to the first computer with a request for information of a digital certificate that authenticates the first process." Novack at [0013]: "The present invention enables secure inter-process communications. The implementation of secure inter-process communications will be described in the context of a client-server network and a peer-to-peer network. Of course, the present invention is also applicable to environments other than the networks described herein, including any network environment that includes inter-process communications between different processes running on one or more computers." Novack at [0015]: "According to an aspect of the present invention, a method is provided for securing inter-process communications between a first process and a second process. The method includes receiving, from a device that supports the first process, a request to initiate inter-process communications. The method also includes replying to the device with a request for information of a digital certificate that authenticates the identity of the first process." Novack at [0021]: "According to an aspect of the present invention, a computer-readable medium is provides that stores a program that secures inter-process communications between a first process and a second process. The computer-readable medium includes a receiving code segment that receives, from a device that supports the first process, a request to initiate inter-process communications. The computer-readable medium also includes a digital certificate requesting code segment that replies to the device with a request for information of a digital certificate that authenticates the identity of the first process."

Headwater Research LLC
Ex. 2011, IPR2024-00003
Page 349 of 442

'117 Claims	Exemplary Disclosures
	See also Novack at [0027]. Novack at [0033]: "FIG. 1 shows an exemplary method for securing inter-process communications. The process starts at S100, and a first process begins to run on a first computer at S102. A determination is made that the services of a second process are needed at S105, and the first process requests initiation of an inter-process communications session at S110. The request at S110 marks the beginning of inter-process communications. The first computer may send the request to a second computer that runs a second process in the same network. The first computer receives an acknowledgement and a request for authentication at S115. In the example above, the second computer analyzes the initial request and generates an acknowledgement and request for authentication that is received by the first computer at S115." Novack at [0034]: "The analysis on the second computer may be performed by the requested process or another process. As an example, the second computer may analyze all incoming communications requests and demand authentication of all requesters. The second computer may demand authentication of only requestors requesting inter-process communications sessions. Of course, the second computer may demand authentication of requestors for only one or more particularly sensitive processes among many. The analysis at the second computer may be performed by the requested process or by a secondary security program that performs security analysis for all communications or all inter-process communications. Accordingly, when the request is initially received, the second computer determines that authentication is necessary before an inter-process communications session can proceed." Novack at [0035]:

Headwater Research LLC
Ex. 2011, IPR2024-00003
Page 350 of 442

'117 Claims	Exemplary Disclosures
	"In one example, the first process is running on an application server that provides application services over the internet to requesting clients. The application server interacts with a management server that manages multiple databases as a back-end service. The first process may be run when a user contacts an application server and downloads presentation layer elements of an application to interact with the application server. The application includes instructions to invoke additional processes for predetermined user requests. Accordingly, the application will request the services of a second process at S110." Novack at [0037]: "Alternatively, the inter-process communications may occur on a single computer. For example, dynamic data exchange (DDE) is an inter-process communications system built into the Macintosh, Windows and OS/2 operating systems that enables two running applications to share the same data." Novack at [0038]: "At S120, the computer that supports the first process forwards information of a first digital certificate to the computer that supports the second process. The computer that supports the first process forwards a first decryption key to the second process at S125. At S130, the computer that supports the first process receives digital certificate information from the computer that supports the second process, and at S135 the computer that supports the first process receives a second decryption key from the computer that supports the second process. Accordingly, the computer that supports the first process and the computer that supports the second process exchange keys at S125 and S135 respectively. As noted above, the logic for the secure inter-process communications may be included in the processes that interact in the inter-process communications. Alternatively, the logic may be provided by secondary authentication processes on the computers."

'117 Claims	Exemplary Disclosures
	Novack at [0042]: "Digital certificates provide numerous benefits, including authentication of source and security of communications. The authentication of source results from the certification authority certifying that the public key is bound to the source. The communications security results from the strong encryption provided using a private/public key pair. The security can be used to exchange symmetric encryption keys that can be used to more efficiently communicate by the sender and recipient of messages. According to an aspect of the present invention, the source may be a process, as well as the device and/or an owner. The owner may be an individual assignee of the public key and/or an entity such as a corporation or organization. In any case, one or more digital certificates are used to secure the communications channel for inter-process communications, according to an aspect of the present invention. As a result of the present invention, a strong assertion of identity is made from either side of a communications channel which can then be used to support authorization and audit needs. Accordingly, the use of digital certificates for inter-process communications provides the benefits of integrity and confidentiality to inter-process communications." Novack at [0043]: "Confidentiality of the transmission is accomplished by encrypting it with the message recipient's public key, as only the holder of the associated private key can decrypt the message. The authentication of source is assured by a "digital signature" mechanism, which is generated by encrypting a secure digest of the message with the private key of the sender. A digital certificate may be stored on any type of computer-readable media, including hard drives, smart cards, flash memory, floppy memory and floppy drives. The digital certificate is installed on a computer and made available to provide secure communications. The digital certificate is retrieved and forwarded to a recipient when a particular process is run and inter-process

Headwater Research LLC
Ex. 2011, IPR2024-00003
Page 352 of 442

'117 Claims	Exemplary Disclosures
	communications are initiated with the recipient. The digital certificate may be used as a generic authentication tool for all communications involving a particular device or user (i.e., including all inter-process communications), or may be used for only one or more predetermined processes involved in inter-process communications. As an example of the use of digital certificates, according to the secure sockets layer protocols, computers exchange digital certificates to secure communications and authenticate sources and destinations." Novack at [0044]: "FIG. 2 shows an exemplary network architecture for secure inter-process communications. As shown, a **computer** 201 communicates through a **network** 220 with an **application server** 230. As an example, the **application server** 230 may be a web server that provides internet applications for users. The user downloads a web page onto the **computer** 201 when the user contacts an internet address corresponding to the desired internet application. Typically, the web page is accessed and downloaded using hyper text transport protocol (HTTP)." Novack at [0048]: "The **management server** 250 may require authentication of the **application server** 230 for each instance of inter-process communications. In this regard, even if the management server's 250 only function is to process requests for the **application server** 230, the **management server** 250 may need to authenticate the application server for each request for inter-process communications when, for example, the **application server** 230 widely interacts with the public and is therefore subject to unknown risks. The **management server** 250 may request verification of the application server's 230 address, the particular requesting function, and any other information that will help authenticate the request. Of course, the **management**

'117 Claims	Exemplary Disclosures
	server 250 may demand a digital certificate that is bound to the application server 230 or even a particular process that runs on the application server 230." Novack at [0049]: "Digital certificates are provided to the application server 230 and/or the management server 250 by a certification authority upon the request and verification of, e.g., the operator of the application server 230 and the management server 250. However, the digital certificates may also be bound to one or more individual processes on the application server 230 or the management server 250, so that inter-process communications may be authenticated as originating from an authorized process, and not just a predetermined device or user. Alternatively, the digital certificate may be used as appropriate to secure different types of communications, including inter-process communications." *See also* Novack at [0050]-[0053] (further discussing the digital certificates referenced in [0049]). Novack at [0060]: "An exemplary use of secure inter-process communications includes securing so-called secure networks. In this regard, secure networks often focus security efforts on preventing impostors from gaining access to the network. For example, a secure local computer network may route all communications through a firewall server that examines all incoming and outgoing communications to ensure they do not present a security risk. However, a lower level of security is provided for communications between addresses within the secure network. In particular, security for communications between addresses in a network is often limited to source and destination authentication. Accordingly, once an impostor gains access to the secure network, the impostor can wreak havoc on the devices in the network. However, using secure inter-process communications according to the present invention, each

'117 Claims	Exemplary Disclosures
	inter-process communications session may be secured, even between devices in the network." Novack at [0061]: "As another example, inter-process communications between devices on an internal network for a corporation can be secured using the present invention. Digital certificates can be installed for one or more processes on particular devices on the network such that inter-process communications are only enabled for the specified processes and/or specified devices." Novack at [0062]: "Additionally, as noted above, the use of secure inter-process communications according to the present invention is particularly well-suited to peer-to-peer networks, where devices are often connected over insecure public networks. In this regard, if a device in a peer-to-peer network only accepts communications from other devices which have digital certificates installed for specified processes, inter-process communications will be authenticated to ensure that unauthorized processes are not being propagated through the network." Novack at [0063]: "Although the invention has been described with reference to several exemplary embodiments, it is understood that the words that have been used are words of description and illustration, rather than words of limitation. Changes may be made within the purview of the appended claims, as presently stated and as amended, without departing from the scope and spirit of the invention in its aspects. Although the invention has been described with reference to particular means, materials and embodiments, the invention is not intended to be limited to the particulars disclosed;

Headwater Research LLC
Ex. 2011, IPR2024-00003
Page 355 of 442

'117 Claims	Exemplary Disclosures
	rather the invention extends to all functionally equivalent structures, methods, and uses such as are within the scope of the appended claim." Novack at [0064]: "The device in FIG. 4 may include multiple processes that communicate with each other using the secure inter-process communications of the present invention. Additionally, although the terms "client" and "server" are used to refer to devices, the devices may be defined and distinguished by a set of software instructions installed as a pre-packaged program. Further, while examples of the uses of the present invention include secure client/server networks and peer-to-peer networks, secure inter-process communications can be used to enhance security in any inter-process communications environment. Further, while the embodiment shown in FIG. 2 only shows a single computer 201 and a single application server 230, secure inter-process communications can be used to enhance security for multiple application servers 230 belonging to a single entity or multiple users using computers 201. Additionally, while networks 220 and 240 are separately designated in FIG. 2, the networks may be the same, such as an internet protocol (IP) network. Further, while a certification authority is discussed in the context of the present invention, a certification authority is not shown in the figures. In this regard, a certification authority may communicate over the network 220 and/or the network 240 to provide digital certificates to the application server 230 and the management server 250. Of course, the digital certificates may be provided by the certification authority as portable memory modules that are installed on the application server 230 or the management server 250. Furthermore, the entity that provides the application server 230 and/or the management server 250 may also act as a certification authority for its customers using the computers 102." *See also* Novack at [0054]-[0059] (further describing the device in [0064]).

'117 Claims	Exemplary Disclosures
	Santamaria at [0041]: "FIG. 3A illustrates a framework used by an exemplary mobile device managing notification service connections according to one embodiment of the invention. For example, the mobile device 160 uses the framework illustrated in FIG. 3A to manage the connections to the push notification services 110 and 120 and the pull notification service 130. The framework includes one or more notification service clients 310. For example, with reference to FIG. 1, there may be a total of three notification service clients 310 (e.g., a notification service client for the push notification service 110, a notification service client for the push notification service 120, and a notification service client for the pull notification service 130). Each notification service client 310 individually is coupled with the service connection manager 320 and the network stack 330. For example, in one embodiment of the invention, each notification service client is an independent process, and does not share memory with other notification service clients and/or does not communicate with other notification service clients. Each notification service client registers 360 with the notification service connection manager 320." Okonnen at [0060]: "In an embodiment according to the present invention, update agent A 113 may be employed to update firmware 117 and operating system 119 during reboot of the mobile handset 107. After a reboot update, update agent B 115 may be used to update other components in the mobile handset 107, for example, configuration parameters, gaming engines, third-party software, etc., that may not be updated during reboot of the mobile handset 107. Both of update agent A 113 and update agent B 115 may be provisioned with security keys for secure communications with device servers 135 and 137 during update agent provisioning activity, such as provisioning activity accompanying NAM provisioning activity and/or incorporated into NAM programming activity." Hashimoto at [0038]-[0039]:

'117 Claims	Exemplary Disclosures
	"Fig. 1 shows a multi-party application protection environment to which the present invention is to be applied. "A software vendor 121 produces a program 122, and distributes it to a microprocessor-A 103 provided inside a target system 102 of a system user 112. In order to protect the program to be distributed, the plaintext program 122 is encrypted." Hashimoto at [0040]: "More specifically, the program formed by execution codes and data such as initialization data is encrypted by using a program key selected by the software vendor 121, and the program key used for the encryption is encrypted by using a public key-A of the processor-A of the target system 102 to generate a distribution key." Hashimoto at [0041]: "At this point, the software vendor 121 divides the program and the initialization data into a plurality of blocks, and encrypts them by using a different key for each block. Note however that there is no need for the software vendor 121 to select as many keys as the number of blocks, and it suffices to select two things including a constant Cx as a basis for generating a plurality of keys, and a key Kx for encrypting the plurality of keys generated according to Cx. As will be described in detail below, a block encrypted program 123 is generated by the encryption using a different key for each block, according to these two constants. Then, Cx and Kx are encrypted by using the public key-A of the processor-A to produce a distribution key Kx[A], Cx[A]." *See also* Hashimoto at Fig. 1.
10. The network system of claim 1, wherein at least one of the secure	The references below disclose and/or render obvious this limitation. For example, see the following passages and/or figures, as well as related disclosures:

'117 Claims	Exemplary Disclosures
Internet data messages comprises multiple identifier/data pairs.	Shenfield at [0056]: "Content dependencies block 462 is used by push proxy 410 to advertise services that a mobile device user can utilize. Thus, if a mobile device user does not have a screen or bandwidth or memory sufficient for the service, content dependencies block 462 could block the advertisement of that service to the user." Shenfield at [0069]: "Flow control management block 542 is used to notify push proxy 410 of Figure 4 to stop pushing content or to resume pushing content. This can be used, for example, when the push client 510 has a limited amount of memory that it can accept pushed content. In this case, before the push content is consumed push client 510 needs to stop the flow of content from push proxy 410. Once the content has been consumed, flow control management block 542 can be used to start the flow of data again." Mostafa at 732: "Different MMS standardization bodies have recently started working to enrich MMS with the feature of transporting application data. Figure 3 depicts different layers of communication in transporting application data using MMS, showing the vertical interaction within an MMS-capable terminal and an MMS R/S to achieve communication between two applications residing in two different MMS-capable terminals. The figure assumes a simplified scenario, where the terminals belong to the same MMS R/S. Only HTTP is shown there, as it is the mostly used protocol across the terminal-interface. Lower layer protocols, below MMS in the figure, are MMS-specific, and are beyond the scope of this paper. "MMS architecture has been lately extended to include applications [11]. Application is defined as a system element that may interact with an MMS UA or a VASP to transport its data to its peer application, interacting with another MMS

'117 Claims	Exemplary Disclosures
	UA or a VASP, using MMS. The extension of MMS architecture is depicted in Figure 4, where unrelated entities and interfaces are not shown. Though an application can interact with either an MMS UA or a VASP to transport its data, MMS UA is mostly mentioned in this regard in this paper to cover both for brevity. Though only one application is shown within a mobile terminal and VASP in Figures 3 and 4, there can be multiple applications, as shown in Figure 5. The application interface between an application and an MMS UA is described in Section 4." *See also* Mostafa at Figs. 1, 3; *see generally* Mostafa at 731-732. Hwang at [0108]: "Request ID, a first item of Table 18, is an identifier given to consistently identify and manage one roaming registration request procedure. The Request ID is an identifier generated for allowing a terminal to uniquely identify its own roaming registration request. The Request ID is constant from the time it is generated in FIG. 8 until the time the roaming registration request procedure of FIG. 8 ends. Roaming Authorization Status is used for allowing the Home SP 820 to determine whether the roaming is authorized by retrieving subscription of the roaming-requested Terminal 800 in the Visited SP 810 using the Terminal ID received in step 804, and to provide the results to the Visited SP 810. Terminal Subscription Type is information provided so that the Visited SP 810 can evaluate which reception right the roaming Terminal 800 will have in the Visited SP 810. The Terminal Subscription Type can be a class of the service that the roaming-requested Terminal 800 can receive from the roaming-requested Visited SP 810 based on the roaming agreement made between the Home SP 820 and the Visited SP 810. This can be defined in the form of a roaming allowed class number or code agreed between the two service providers, and its form is not defined in an exemplary embodiment of the present invention."

Headwater Research LLC
Ex. 2011, IPR2024-00003
Page 360 of 442

'117 Claims	Exemplary Disclosures
	Hwang at [0112]: "Roaming Service Allowed Scope, when roaming is available, is used for allowing a value of the Terminal Subscription Type received in step 805 to indicate which reception right the roaming Terminal 800 will actually have in the Visited SP 810. The reception possibility of the service corresponding to the Purchase Item ID requested by the Terminal 800 is also defined in the Roaming Service Allowed Scope. In addition, the information related to occurrence of additional cost and a change in the charging system is also added to the Roaming Service Allowed Scope during roaming." Hwang at [0121]: "If the BSM 813 of the Visited SP 810 can authorize the Terminal 800 to perform the roaming, it generates a response message taking into account the additional charge occurring during roaming and the change in the charging system using the subscription information of the Terminal 800, and transmits the response message to the terminal in step 908. After transmitting the response message, the BSM 813 of the Visited SP 810 waits for a response from the Terminal 800 in step 909. Upon receipt of the Final Confirm message for the roaming from the Terminal 800, the BSM 813 of the Visited SP 810 deciphers the received message in step 910, and determines in step 911 whether the terminal has roamed. If the contents of the received message indicate roaming rejection, the roaming request procedure is terminated. Otherwise, if the contents of the received message indicate roaming acceptance, a Long-Term Key message is transmitted to the terminal in step 912." Isomursu at 2:56-3:33: "The intention is that any user messages can be used, but in following mainly short messages will be referred to as an example. In addition to being fast to send, the advantages of a short message service can be utilised, such as not reserving the voice

'117 Claims	Exemplary Disclosures
	channel. Application related information can either be pre-stored in a terminal memory (permanent memory) or a user may store the application related information in a terminal memory (cache memory) by contacting a server by means of a terminal. Depending on the application, the user may enter user input or modify the information in the applications. In another application the information relating to an application may be sent by a service provider and the information may be such that it is not possible for the user to modify it, only to request the service provider to modify it. The information readily printed in the application can also be transmitted. An application type identifier or header is preferably added to the transmission, so that a receiving terminal identifies the short message as not an ordinary short message, but as a short message containing information relating to and intended for a specific application. The identifier can be a code in an address or a control field of the short message, or it can be a code in the message part of the short message. Because it has been realised that the short message service, already existing in the mobile station system, can be utilised for sending information on applications, the advantages are, e.g., that there is no need to establish an off-line server for sending the application related information, such as, for example, in the system presented in Publication WO 94/23394. Especially advantageous is that one and the same server, i.e. the SMS server (the Short Message Service Centre SM-SC) can be used for sending and forwarding information relating to any application, so there is no need to have separate servers for each application. The SMS server will forward any short message and the terminal will address the information to the correct application according to the header or identifier in the message. And since a short message can be sent simultaneously with a circuit-coupled connection, the sending of the application related information does not engage the terminal's communication line, e.g., in case of a simultaneously incoming call. A network like the GSM network is maintained by several operators and usually each operator has at least one SMS

'117 Claims	Exemplary Disclosures
	server of their own. In this case naturally any SMS server or several servers may be used for the invention." Isomursu at 6:28-59: "In the present invention, using short messages as an example, the transmission of each application related information will be identified by means of a specific code, i.e., an identifier, which enables the receiving terminal to process the received message directly into an application, as specified, containing the received data. The identifier is preferably implemented by using ASCII characters in an information field of the short message transmission frame, i.e., in a field INFO (FIG. 5), which contains the actual short message in characters. Alternatively the identifier may be in the form of some other character or string code, such as bits, since for sending a short message the data is anyway converted into bits. Because the information relating to the applications is transmitted in a short message, it can also be received by means of an ordinary mobile station, which does not support the application service, but is capable of transmitting and/or receiving short messages. By placing the application identifier in the field INFO, there is also the advantage that in an ordinary mobile station, which does not support this type of application service, but is capable of transmitting and/or receiving short messages, the application identifier is displayed to a user of the terminal, e.g. a communicator and, hence, the user notices not having received an ordinary short message, but the information relating to a specific application. In addition, a user of this type of ordinary mobile station can also transmit a message, such as mentioned above, by first writing, on the message, the identifier of the application in question in characters, and the rest of the information correctly divided. The reception of such a transmission by means of a terminal, according to the present invention, will produce a fully received application related information record." Okonnen at [0013]-[0014]:

'117 Claims	Exemplary Disclosures
	"In an embodiment according to the present invention, the number assignment module programming activity may comprise at least one of over-the-air service provisioning (OTASP) activity and over-the-air parameter administration (OTAPA) activity. "In an embodiment according to the present invention, the provisioned data unit may be adapted to store at least one of update agent related provisioning information, a universal resource locator of a server used to retrieve updates, and a security key used to authenticate server messages." Staack at [0021]: "FIG. 1 shows a first embodiment of the invention and illustrates a case where a message is to be sent from a first network element 1 (user A) to a second network element 3 (user B). The network elements 1, 3 are, in the 25 present embodiment, client or user equipments such as terminals. In the present example, the network element 1 (user A) is an equipment trying to send a message (e.g. "MESSAGE user b@sonera.com" addressed to user b@sonera.com) to the receiving element 3 (user B) which is presently out 30 of reach, e.g. switched-off, busy, or located in a nonsupported area, or the like. The connection request of network element 1 is handled by a network element 2 which may be a server (such as a proxy server) which provide e.g. CSCF (Call Server Control Function), and/or is a home location server which contains a database storing information on the present locations of network element 3 and further network elements, reachability thereof, and the like." *See also* Staack at Figs. 1, 2. Fan at 13: Third, business control. The control of the service received by the terminal is mainly performed by setting a service identifier corresponding to the service type that the terminal allows or disallows or is not currently allowed to receive. The

'117 Claims	Exemplary Disclosures
	Push sending proxy may send the Push message sent by the PI to the Push receiving proxy in the terminal by using a SIP message, and the Acceptable Contact Head field Accept-Contact header in the SIP message indicates the service type. Fan at 13: If the Push message control policy is stored on the terminal, a list of service types that the terminal does not want or does not want to receive may be defined, or a service list that the terminal wishes to receive may be defined, and the terminal completes control of the service type, and selects a pair. Push messages are discarded or sent to other terminals for reception, or temporarily stored. Cooley at ¶[0015]: "When the incoming text message arrives at the user's device it is intercepted, under control of the device processor and an application, such as application 31 (as will be discussed), running on the device. This is possible because application 31 is registered with the device operating system when it was loaded on the device. Application 31 then routes the incoming message to the proper specific application in accordance with the data contained in the text message. Thus, in the situation with the flight reservation to Las Vegas (as discussed above), information pertaining to that flight is routed to a specific application, such as to air travel application 22-1 (shown in FIG. 2B), which application is dedicated to air travel data pertaining to the device user." Cooley at ¶[0017]: "When the user touches tile 23-1 all the flights that the user has booked on, for example, AMERICAN™ airlines (AA) are displayed. Touching tile 24-1 then yields the screen shown in FIG. 2E which then displays the information that has been received from an external source. In this case this information would have

'117 Claims	Exemplary Disclosures
	come typically from the AA server in the manner discussed above. There could be any amount of information and tiles (or layers) to this information, all controlled by application 30 residing on the user's device with the updated information coming, from time to time, via messages from various external locations. Note that in FIG. 2F the gate and status of the flight have been updated. This update could have come as a result of a request from the user or could have come as a text message from the airline as part of an update service for customers. In any event, the message would have gone first to service center 12 and then via SMS broker 13 to the user's device where application 30 would have taken the data and placed it in the proper tile." Cooley at ¶[0018]: "FIG. 3 shows one embodiment of a system for controlling messages on a user device. The incoming SMS message would normally be delivered to level 32 of network stack 30 for subsequent display to device user via, for example, display 16 (FIG. 1). The display would be in text or visual form and would then go into the SMS box in text, graphics or video form as is well-known. However, application 31 intercepts the message because that application is registered with the platform to receive certain SMS messages. This protocol varies from device to device based on operating system. SMS or HTTP transports can be used via the side channel process. SMS has the advantage of being a true "push" notification. Application 31 then routes the incoming message, based on the context and content of the data contained in the message to an individual tile, such as to tile 22-1 (FIG. 2B) or even more specifically to tile 24-1 (FIG. 2D), as discussed above." Cooley at ¶[0021]: "FIG. 4 shows one embodiment of a method, such as method 40, for controlling applications resident on a mobile device from a remote

'117 Claims	Exemplary Disclosures
	location. Process 401 determines if a message is incoming to a particular device. If there is a message available, then process 402 determines if the message contains any code or data that identifies the message as one that is to be delivered to a specific application resident on the device and/or delivered customized in a particular manner. If not, then the message is stored in its normal location, for example in SMS layer 32 (FIG. 3) and delivered without modification." Cooley at ¶[0022]: "If the message does contain a "diversion" code or other data that indicates that the message is to be delivered to another location or delivered in a customized manner, then process 404 delivers the incoming message (or other data) accordingly as discussed above, such that a particular user's tile is customized for a particular application. Customizing could, for example, include one or more of the following: appearance, context, display parameters, including audio, visual, application state change. In some situations, a control command is sent to a user's tile, for example, using an SMS transport protocol such that the command controls the display parameters and/or the display message instead of actual text being sent to the tile." Wakasa at ¶[0040]: "The server device 200 generates a transmitted message including parameter information and the identification information of an application to be executed by the portable terminal device 100 on the basis of the parameter information. Then, the server device 200 transmits the transmitted message to the portable terminal device 100 via the network 300. The parameter information is typically information used for requesting the application of the portable terminal device 100 to display information desired by the user or carry out predetermined processing. For example, the parameter information is information requesting the application of the portable terminal device 100 to display "Hello" on the display screen. The portable terminal device 100 executes the application corresponding to the

'117 Claims	Exemplary Disclosures
	identification information of the application included in transmitted message by using the parameter information as input data. For example, by using the parameter information requesting the application of the portable terminal device 100 to display "Hello" on the display screen as input data, the portable terminal device 100 executes the application corresponding to the identification information, displaying "Hello" on the display screen of the portable terminal device 100. As an alternative, the server device 200 holds an application to be provided to the portable terminal device 100 and, in accordance with a request made by the portable terminal device 100, the server device 200 transmits the application to the portable terminal device 100 via the network 300." Wakasa at ¶[0042]: "The portable terminal device 100 receives a transmitted message generated from the server device 200. The portable terminal device 100 then executes the application corresponding to the identification information of the application on the basis of the parameter information included in transmitted message. The portable terminal device 100 determines as to whether or not the application corresponding to the identification information of the application included in the transmitted message exists in the portable terminal device 100. If the result of the determination indicates that the application corresponding to the identification information does not exist in the portable terminal device 100, the portable terminal device 100 generates a request for transmission of the application corresponding to the identification information and transmits the request to the server device 200. In this case, the portable terminal device 100 receives the requested application from the server device 200 and the portable terminal device 100 being able to execute the application. Examples of the portable terminal device 100 are a portable phone, a PDA (personal digital assistant), a portable game device and a watch."

Headwater Research LLC
Ex. 2011, IPR2024-00003
Page 368 of 442

'117 Claims	Exemplary Disclosures
11. The network system of claim 1, wherein the secure interprocess communication service forwards the application data to at least one of the software processes in an encrypted format.	The references below disclose and/or render obvious this limitation. For example, see the following passages and/or figures, as well as related disclosures: Ellison at 9:48-9:62: "The usage protector 250 is coupled to the key generator 240 to protect usage of the software environment 210 or the subset 230, using the OSNK 203. The usage protection includes protection against unauthorized reads, and detection of intrusion, tampering or unauthorized modification. If the two hash values are not the same, then the usage protector 250 knows that there is a change in the subset 230. If it is known that this change is authorized and an updated hash value has been provided, the usage protector 250 would merely report the result. Otherwise, the usage protector 250 may generate an error or a fault function. When a user is notified of the error, fault condition, he or she would know that the subset 230 has been tampered, modified. The user may take appropriate action. Therefore, the usage of the subset 230 is protected." *See also* Ellison at 9:63-13:3, Figs. 3A, 3B, 3C, 3D, 3E. Anderson at 669-670: "Another approach is to do encryption and/or authentication at the IP layer using a protocol suite known as Ipsec. IPsec defines a *security association* as the combination of keys, algorithms and parameters used to protect a particular packet stream. Protected packets are either encrypted or authenticated (or both); in the latter case, an authentication header is added that protects data integrity using HMAC-SHA1, while in the former the packet is encrypted and encapsulated in other packets. (The use of encryption without authentication is discouraged as it's insecure [151].) There's also an *Internet Key Exchange* (IKE) protocol to set up keys and negotiate parameters. IKE has been through a number of versions (some of the bugs that were fixed are discussed in [465]). "IPsec is widely used by firewall vendors who offer a *virtual private network* facility with their products; that is, by installing one of their boxes in each branch

'117 Claims	Exemplary Disclosures
	between the local LAN and the router, all the internal traffic can pass encrypted over the Internet. Individual PCs, such as workers' laptops and home PCs, can in theory join a VPN given a firewall that supports IPsec, but this is harder than it looks. Compatibility has been a major problem with different manufacturers' offerings just not working with each other; although firewall-to-firewall compatibility has improved recently, getting random PCs to work with a given VPN is still very much a hit-or-miss affair." Anderson at 670-71: "Recall that when discussing public key encryption, I remarked that a server could publish a public key *KS* and any web browser could then send a message *M* containing a credit card number to it encrypted using *KS*: $\{M\}_{KS}$. This is in essence what the TLS protocol (formerly known as SSL) does, although in practice it is more complicated. It was developed to support encryption and authentication in both directions, so that both http requests and responses can be protected against both eavesdropping and manipulation. It's the protocol that's activated when you see the padlock on your browser toolbar." "Here is a simplified description of the version as used to protect web pages that solicit credit card numbers: 1. the client sends the server a *client hello* message that contains its name *C*, a transaction serial number *C#*, and a random nonce N_C; 2. the server replies with a *server hello* message that contains its name *S*, a transaction serial number *S#*, a random nonce N_S, and a certificate *CS* containing its public key *KS*. The client now checks the certificate *CS* back to a root certificate issued by a company such as Verisign and stored in the browser; 3. the client sends a *key exchange* message containing a *pre-master-secret* key, K_0, encrypted under the server public key *KS*. It also sends a *finished* message with a message authentication code (MAC)computed on all the messages to date. The key for this MAC is the *master-secret*, K_1. This key is computed by hashing

'117 Claims	Exemplary Disclosures
	the pre-master-secret key with the nonces sent by the client and server: $K_1 = h(K_{CS},N_C,N_S)$. From this point onward, all the traffic is encrypted; we'll write this as $\{\ldots\}_{KCS}$ in the client-server direction and $\{\ldots\}_{KSC}$ from the server to the client. These keys are generated in turn by hashing the nonces with K_1. 4. The server also sends a finished message with a MAC computed on all the messages to date. It then finally starts sending the data. $C \rightarrow S : C, C\#, N_C$ $S \rightarrow C : S, S\#, N_S, CS$ $C \rightarrow S : [K0]_{KS}$ $C \rightarrow S : [finished, MAC(K_1, everythingtodate)]_{KCS}$ $S \rightarrow C : [finished, MAC(K_1, everythingtodate)\}_{KSC}, [data]_{KSC}$ "The design goals included minimizing the load on the browser, and then minimizing the load on the server. Thus the public key encryption operation is done by the client, and the decryption by the server; the standard encryption method (*ciphersuite*) uses RSA for which encryption can be arranged to be very much faster than decryption. (This was a wrong design decision as browsers generally have a lot more compute cycles to spare than servers; it has created a brisk aftermarket for crypto accelerator boards for web servers.) Also, once a client and server have established a pre-master secret, no more public key operations are needed as further master secrets can be obtained by hashing it with new nonces." *See generally* Anderson at Ch. 20 ("Telecom System Security"); § 21.4.5.6 ("TLS"). Novack at Abstract: "Inter-process communications between a first process and a second process are secured. A first process recognizes that inter-process communications must be initiated with a second process. A first computer that supports the first process submits a request to initiate inter-process communications with a second computer

'117 Claims	Exemplary Disclosures
	that supports the second process. The second computer replies to a request to initiate inter-process communications by replying to the first computer with a request for information of a digital certificate that authenticates the first process." Novack at [0013]: "The present invention enables secure inter-process communications. The implementation of secure inter-process communications will be described in the context of a client-server network and a peer-to-peer network. Of course, the present invention is also applicable to environments other than the networks described herein, including any network environment that includes inter-process communications between different processes running on one or more computers." Novack at [0015]: "According to an aspect of the present invention, a method is provided for securing inter-process communications between a first process and a second process. The method includes receiving, from a device that supports the first process, a request to initiate inter-process communications. The method also includes replying to the device with a request for information of a digital certificate that authenticates the identity of the first process." Novack at [0021]: "According to an aspect of the present invention, a computer-readable medium is provides that stores a program that secures inter-process communications between a first process and a second process. The computer-readable medium includes a receiving code segment that receives, from a device that supports the first process, a request to initiate inter-process communications. The computer-readable medium also includes a digital certificate requesting code segment that replies to the device

'117 Claims	Exemplary Disclosures
	with a request for information of a digital certificate that authenticates the identity of the first process." *See also* Novack at [0027]. Novack at [0033]: "FIG. 1 shows an exemplary method for securing inter-process communications. The process starts at S100, and a first process begins to run on a first computer at S102. A determination is made that the services of a second process are needed at S105, and the first process requests initiation of an inter-process communications session at S110. The request at S110 marks the beginning of inter-process communications. The first computer may send the request to a second computer that runs a second process in the same network. The first computer receives an acknowledgement and a request for authentication at S115. In the example above, the second computer analyzes the initial request and generates an acknowledgement and request for authentication that is received by the first computer at S115." Novack at [0034]: "The analysis on the second computer may be performed by the requested process or another process. As an example, the second computer may analyze all incoming communications requests and demand authentication of all requesters. The second computer may demand authentication of only requestors requesting inter-process communications sessions. Of course, the second computer may demand authentication of requestors for only one or more particularly sensitive processes among many. The analysis at the second computer may be performed by the requested process or by a secondary security program that performs security analysis for all communications or all inter-process communications. Accordingly, when the request is initially received, the second computer determines that authentication is necessary before an inter-process communications session can proceed."

'117 Claims	Exemplary Disclosures
	Novack at [0035]: "In one example, the first process is running on an application server that provides application services over the internet to requesting clients. The application server interacts with a management server that manages multiple databases as a back-end service. The first process may be run when a user contacts an application server and downloads presentation layer elements of an application to interact with the application server. The application includes instructions to invoke additional processes for predetermined user requests. Accordingly, the application will request the services of a second process at S110." Novack at [0037]: "Alternatively, the inter-process communications may occur on a single computer. For example, dynamic data exchange (DDE) is an inter-process communications system built into the Macintosh, Windows and OS/2 operating systems that enables two running applications to share the same data." Novack at [0038]: "At S120, the computer that supports the first process forwards information of a first digital certificate to the computer that supports the second process. The computer that supports the first process forwards a first decryption key to the second process at S125. At S130, the computer that supports the first process receives digital certificate information from the computer that supports the second process, and at S135 the computer that supports the first process receives a second decryption key from the computer that supports the second process. Accordingly, the computer that supports the first process and the computer that supports the second process exchange keys at S125 and S135 respectively. As noted above, the logic for the secure inter-process communications may be included in the processes that interact in the inter-

'117 Claims	Exemplary Disclosures
	process communications. Alternatively, the logic may be provided by secondary authentication processes on the computers." Novack at [0042]: "Digital certificates provide numerous benefits, including authentication of source and security of communications. The authentication of source results from the certification authority certifying that the public key is bound to the source. The communications security results from the strong encryption provided using a private/public key pair. The security can be used to exchange symmetric encryption keys that can be used to more efficiently communicate by the sender and recipient of messages. According to an aspect of the present invention, the source may be a process, as well as the device and/or an owner. The owner may be an individual assignee of the public key and/or an entity such as a corporation or organization. In any case, one or more digital certificates are used to secure the communications channel for inter-process communications, according to an aspect of the present invention. As a result of the present invention, a strong assertion of identity is made from either side of a communications channel which can then be used to support authorization and audit needs. Accordingly, the use of digital certificates for inter-process communications provides the benefits of integrity and confidentiality to inter-process communications." Novack at [0043]: "Confidentiality of the transmission is accomplished by encrypting it with the message recipient's public key, as only the holder of the associated private key can decrypt the message. The authentication of source is assured by a "digital signature" mechanism, which is generated by encrypting a secure digest of the message with the private key of the sender. A digital certificate may be stored on any type of computer-readable media, including hard drives, smart cards, flash memory, floppy memory and floppy drives. The digital certificate is installed on a computer and made

Headwater Research LLC
Ex. 2011, IPR2024-00003
Page 375 of 442

'117 Claims	Exemplary Disclosures
	available to provide secure communications. The digital certificate is retrieved and forwarded to a recipient when a particular process is run and inter-process communications are initiated with the recipient. The digital certificate may be used as a generic authentication tool for all communications involving a particular device or user (i.e., including all inter-process communications), or may be used for only one or more predetermined processes involved in inter-process communications. As an example of the use of digital certificates, according to the secure sockets layer protocols, computers exchange digital certificates to secure communications and authenticate sources and destinations." Novack at [0044]: "FIG. 2 shows an exemplary network architecture for secure inter-process communications. As shown, a **computer** 201 communicates through a **network** 220 with an **application server** 230. As an example, the **application server** 230 may be a web server that provides internet applications for users. The user downloads a web page onto the **computer** 201 when the user contacts an internet address corresponding to the desired internet application. Typically, the web page is accessed and downloaded using hyper text transport protocol (HTTP)." Novack at [0048]: "The **management server** 250 may require authentication of the **application server** 230 for each instance of inter-process communications. In this regard, even if the management server's 250 only function is to process requests for the **application server** 230, the **management server** 250 may need to authenticate the application server for each request for inter-process communications when, for example, the **application server** 230 widely interacts with the public and is therefore subject to unknown risks. The **management server** 250 may request verification of the application server's 230 address, the particular requesting function, and any other information that will help authenticate the request. Of course, the **management**

'117 Claims	Exemplary Disclosures
	server 250 may demand a digital certificate that is bound to the application server 230 or even a particular process that runs on the application server 230." Novack at [0049]: "Digital certificates are provided to the application server 230 and/or the management server 250 by a certification authority upon the request and verification of, e.g., the operator of the application server 230 and the management server 250. However, the digital certificates may also be bound to one or more individual processes on the application server 230 or the management server 250, so that inter-process communications may be authenticated as originating from an authorized process, and not just a predetermined device or user. Alternatively, the digital certificate may be used as appropriate to secure different types of communications, including inter-process communications." *See also* Novack at [0050]-[0053] (further discussing the digital certificates referenced in [0049]). Novack at [0060]: "An exemplary use of secure inter-process communications includes securing so-called secure networks. In this regard, secure networks often focus security efforts on preventing impostors from gaining access to the network. For example, a secure local computer network may route all communications through a firewall server that examines all incoming and outgoing communications to ensure they do not present a security risk. However, a lower level of security is provided for communications between addresses within the secure network. In particular, security for communications between addresses in a network is often limited to source and destination authentication. Accordingly, once an impostor gains access to the secure network, the impostor can wreak havoc on the devices in the network. However, using secure inter-process communications according to the present invention, each

'117 Claims	Exemplary Disclosures
	inter-process communications session may be secured, even between devices in the network." Novack at [0061]: "As another example, inter-process communications between devices on an internal network for a corporation can be secured using the present invention. Digital certificates can be installed for one or more processes on particular devices on the network such that inter-process communications are only enabled for the specified processes and/or specified devices." Novack at [0062]: "Additionally, as noted above, the use of secure inter-process communications according to the present invention is particularly well-suited to peer-to-peer networks, where devices are often connected over insecure public networks. In this regard, if a device in a peer-to-peer network only accepts communications from other devices which have digital certificates installed for specified processes, inter-process communications will be authenticated to ensure that unauthorized processes are not being propagated through the network." Novack at [0063]: "Although the invention has been described with reference to several exemplary embodiments, it is understood that the words that have been used are words of description and illustration, rather than words of limitation. Changes may be made within the purview of the appended claims, as presently stated and as amended, without departing from the scope and spirit of the invention in its aspects. Although the invention has been described with reference to particular means, materials and embodiments, the invention is not intended to be limited to the particulars disclosed;

Headwater Research LLC
Ex. 2011, IPR2024-00003
Page 378 of 442

'117 Claims	Exemplary Disclosures
	rather the invention extends to all functionally equivalent structures, methods, and uses such as are within the scope of the appended claim." Novack at [0064]: "The device in FIG. 4 may include multiple processes that communicate with each other using the secure inter-process communications of the present invention. Additionally, although the terms "client" and "server" are used to refer to devices, the devices may be defined and distinguished by a set of software instructions installed as a pre-packaged program. Further, while examples of the uses of the present invention include secure client/server networks and peer-to-peer networks, secure inter-process communications can be used to enhance security in any inter-process communications environment. Further, while the embodiment shown in FIG. 2 only shows a single computer 201 and a single application server 230, secure inter-process communications can be used to enhance security for multiple application servers 230 belonging to a single entity or multiple users using computers 201. Additionally, while networks 220 and 240 are separately designated in FIG. 2, the networks may be the same, such as an internet protocol (IP) network. Further, while a certification authority is discussed in the context of the present invention, a certification authority is not shown in the figures. In this regard, a certification authority may communicate over the network 220 and/or the network 240 to provide digital certificates to the application server 230 and the management server 250. Of course, the digital certificates may be provided by the certification authority as portable memory modules that are installed on the application server 230 or the management server 250. Furthermore, the entity that provides the application server 230 and/or the management server 250 may also act as a certification authority for its customers using the computers 102." *See also* Novack at [0054]-[0059] (further describing the device in [0064]).

'117 Claims	Exemplary Disclosures
	Santamaria at [0041]: "FIG. 3A illustrates a framework used by an exemplary mobile device managing notification service connections according to one embodiment of the invention. For example, the mobile device 160 uses the framework illustrated in FIG. 3A to manage the connections to the push notification services 110 and 120 and the pull notification service 130. The framework includes one or more notification service clients 310. For example, with reference to FIG. 1, there may be a total of three notification service clients 310 (e.g., a notification service client for the push notification service 110, a notification service client for the push notification service 120, and a notification service client for the pull notification service 130). Each notification service client 310 individually is coupled with the service connection manager 320 and the network stack 330. For example, in one embodiment of the invention, each notification service client is an independent process, and does not share memory with other notification service clients and/or does not communicate with other notification service clients. Each notification service client registers 360 with the notification service connection manager 320." Hashimoto at [0038]-[0039]: "Fig. 1 shows a multi-party application protection environment to which the present invention is to be applied. "A software vendor 121 produces a program 122, and distributes it to a microprocessor-A 103 provided inside a target system 102 of a system user 112. In order to protect the program to be distributed, the plaintext program 122 is encrypted." Hashimoto at [0040]: "More specifically, the program formed by execution codes and data such as initialization data is encrypted by using a program key selected by the software vendor 121, and the program key used for the encryption is encrypted by using a

'117 Claims	Exemplary Disclosures
	public key-A of the processor-A of the target system 102 to generate a distribution key." Hashimoto at [0041]: "At this point, the software vendor 121 divides the program and the initialization data into a plurality of blocks, and encrypts them by using a different key for each block. Note however that there is no need for the software vendor 121 to select as many keys as the number of blocks, and it suffices to select two things including a constant Cx as a basis for generating a plurality of keys, and a key Kx for encrypting the plurality of keys generated according to Cx. As will be described in detail below, a block encrypted program 123 is generated by the encryption using a different key for each block, according to these two constants. Then, Cx and Kx are encrypted by using the public key-A of the processor-A to produce a distribution key Kx[A], Cx[A]." *See also* Hashimoto at Fig. 1. Cooley at ¶[0015]: "When the incoming text message arrives at the user's device it is intercepted, under control of the device processor and an application, such as application 31 (as will be discussed), running on the device. This is possible because application 31 is registered with the device operating system when it was loaded on the device. Application 31 then routes the incoming message to the proper specific application in accordance with the data contained in the text message. Thus, in the situation with the flight reservation to Las Vegas (as discussed above), information pertaining to that flight is routed to a specific application, such as to air travel application 22-1 (shown in FIG. 2B), which application is dedicated to air travel data pertaining to the device user." Cooley at ¶[0017]:

'117 Claims	Exemplary Disclosures
	"When the user touches tile 23-1 all the flights that the user has booked on, for example, AMERICAN™ airlines (AA) are displayed. Touching tile 24-1 then yields the screen shown in FIG. 2E which then displays the information that has been received from an external source. In this case this information would have come typically from the AA server in the manner discussed above. There could be any amount of information and tiles (or layers) to this information, all controlled by application 30 residing on the user's device with the updated information coming, from time to time, via messages from various external locations. Note that in FIG. 2F the gate and status of the flight have been updated. This update could have come as a result of a request from the user or could have come as a text message from the airline as part of an update service for customers. In any event, the message would have gone first to service center 12 and then via SMS broker 13 to the user's device where application 30 would have taken the data and placed it in the proper tile." Cooley at ¶[0018]: "FIG. 3 shows one embodiment of a system for controlling messages on a user device. The incoming SMS message would normally be delivered to level 32 of network stack 30 for subsequent display to device user via, for example, display 16 (FIG. 1). The display would be in text or visual form and would then go into the SMS box in text, graphics or video form as is well-known. However, application 31 intercepts the message because that application is registered with the platform to receive certain SMS messages. This protocol varies from device to device based on operating system. SMS or HTTP transports can be used via the side channel process. SMS has the advantage of being a true "push" notification. Application 31 then routes the incoming message, based on the context and content of the data contained in the message to an individual tile, such as to

'117 Claims	Exemplary Disclosures
	tile 22-1 (FIG. 2B) or even more specifically to tile 24-1 (FIG. 2D), as discussed above." Cooley at ¶[0021]: "FIG. 4 shows one embodiment of a method, such as method 40, for controlling applications resident on a mobile device from a remote location. Process 401 determines if a message is incoming to a particular device. If there is a message available, then process 402 determines if the message contains any code or data that identifies the message as one that is to be delivered to a specific application resident on the device and/or delivered customized in a particular manner. If not, then the message is stored in its normal location, for example in SMS layer 32 (FIG. 3) and delivered without modification." Cooley at ¶[0022]: "If the message does contain a "diversion" code or other data that indicates that the message is to be delivered to another location or delivered in a customized manner, then process 404 delivers the incoming message (or other data) accordingly as discussed above, such that a particular user's tile is customized for a particular application. Customizing could, for example, include one or more of the following: appearance, context, display parameters, including audio, visual, application state change. In some situations, a control command is sent to a user's tile, for example, using an SMS transport protocol such that the command controls the display parameters and/or the display message instead of actual text being sent to the tile."
12. The network system of claim 1, the device messaging agent on at least one of the devices further to initiate	The references below disclose and/or render obvious this limitation. For example, see the following passages and/or figures, as well as related disclosures: Huskins at 3:20-29:

Headwater Research LLC
Ex. 2011, IPR2024-00003
Page 383 of 442

'117 Claims	Exemplary Disclosures
the secure connection to the network message server.	"In addition, a client disk management server 20 may be provided as well. The client disk management server 20 may transmit messages that include one or more distinct service identifiers (e.g., service_id=0x01) . In some cases, a variety of messages may be issued by the client disk management server 20 in order to initiate desired functions on a client's storage device or disk drive 43. For example, separate service identifiers may be utilized for the commands to create partitions, delete partitions, or modify partitions, as examples." Chou at 2:54-59: "Advantageously, the request for additional information may be initiatable in the mobile client device utilizing a single-key operation. Such a request for additional information may initiate a WAP pull operation that pulls the information from a content generator associated with the wireless secure server and displays it on the mobile client device." Chou at 8:46-67: "Event triggering in the system 100 may be initiated by the enterprise application 104 and relayed to mobile client 110 via the WSS 106. For example, upon receiving a given event trigger from the enterprise application, the WSS first prepares the corresponding push content and then invokes the PI to initiate a WAP push operation. When the push content reaches the mobile client device via the PPG, the user in response can request more detailed information, possibly using only a single key stroke, a process referred to herein as single-key interactive switching. Such a client-originated request for additional information initiates a WAP pull operation that pulls the multimedia content, WAP application or other information from the content generator 136 of the WSS and displays it on the mobile client device. The illustrative embodiment thus seamlessly couples WAP push and WAP pull operations. This is advantageously accomplished in a manner that not only makes the transport model transparent to users, but also facilitates a server-centric architecture by eliminating any need for a corresponding client application running on the mobile client device. The latter advantage makes the

'117 Claims	Exemplary Disclosures
	application platform device-independent and suitable for use with any type of WAP-enabled terminal." Novack at [0035]: "In one example, the first process is running on an application server that provides application services over the internet to requesting clients. The application server interacts with a management server that manages multiple databases as a back-end service. The first process may be run when a user contacts an application server and downloads presentation layer elements of an application to interact with the application server. The application includes instructions to invoke additional processes for predetermined user requests. Accordingly, the application will request the services of a second process at S110." Novack at [0044]: "FIG. 2 shows an exemplary network architecture for secure inter-process communications. As shown, a **computer** 201 communicates through a **network** 220 with an **application server** 230. As an example, the **application server** 230 may be a web server that provides internet applications for users. The user downloads a web page onto the **computer** 201 when the user contacts an internet address corresponding to the desired internet application. Typically, the web page is accessed and downloaded using hyper text transport protocol (HTTP)." Santamaria at [0038]: "At block 210, the mobile device 160 establishes a data context. The data context may be established by any number of methods known in the art. From block 210, flow moves to block 220 where the mobile device 160 establishes one or more notification services connections. For example, referring to FIG. 1, the mobile device 160 establishes a connection with the push notification services 110 and 120

Headwater Research LLC
Ex. 2011, IPR2024-00003
Page 385 of 442

'117 Claims	Exemplary Disclosures
	and the pull notification service 130. In some embodiments of the invention, the connections are each Transmission Control Protocol (TCP) connections." Santamaria at [0061]: "At time 47, the mobile device 160 is awakened from the sleep mode 516 from an independent event (represented by independent wake 566). There may be many causes of an independent wake, including user interaction (e.g., the user disables sleep mode), receiving an item of interest (e.g., receiving an email message from the push notification service 110), receiving a phone call and/or text message, etc. Once awake, the mobile device determines whether the time is within any notification service connection message transmission window. For example, time 47 is within the window 552 for the push notification service 110 (which starts at time 45 and ends at time 55) and within the window 550 for the pull notification service 130 (which starts at time 42 and ends at time 50). Leveraging the fact that the mobile device is awake and within notification service connection message transmission windows, the mobile device transmits a notification service connection message to the push notification service 110 (represented by the beat 582) to refresh the push notification service connection 110, and transmits a notification service connection message to the pull notification service 130 (represented by the check 580) to check for items of interest. In addition, while not illustrated in FIG. 5 for simplicity purposes, the mobile device 160 resets the notification service connection message transmission timer and reschedules the notification service connection message transmission time and wake for the push notification service 110 and the pull notification service 130. At time 49, the mobile device enters sleep mode 518." Isomursu at 21:21-39: "In the implementation illustrated in FIG. 8, a first memory 13 is coupled to the main control circuit 6. The first memory can be a volatile memory, e.g., RAM, where the main control circuit stores in-use data. In addition, the terminal has a second memory 14, which is preferably a permanent memory 14, wherein the

'117 Claims	Exemplary Disclosures
	terminal's application programs performing different kinds of services and running the different types of applications, other data essential for the functioning of the terminal, and any other data which a user wants to store permanently, are stored. Alternatively, the application related information can be stored off-line in a memory of an external smart card, coupled to the terminal, wherefrom there is a connection to the **main control circuit** 6. This type of smart card is known, e.g., from the GSM mobile communications system, as a SIM card (Subscriber Identity Module), which usually has storage, e.g., for storing telephone numbers. In this case, new applications can be updated in the terminal by simply updating the memory of the smart card." Okonnen at [0013]-[0014]: "In an embodiment according to the present invention, the number assignment module programming activity may comprise at least one of over-the-air service provisioning (OTASP) activity and over-the-air parameter administration (OTAPA) activity. "In an embodiment according to the present invention, the provisioned data unit may be adapted to store at least one of update agent related provisioning information, a universal resource locator of a server used to retrieve updates, and a security key used to authenticate server messages." Okonnen at [0020]: "In an embodiment according to the present invention, the network may further comprise update agent provisioning information stored in the electronic device. The update agent provisioning information may comprise a device server URL, an index of provisioned update agents, a security key, and electronic device related information. The device server URL may provide references to servers hosting updates to be downloaded. The updates may be compatible with update agents currently available and provisioned in the electronic device." Okonnen at [0047]-[0048]:

'117 Claims	Exemplary Disclosures
	"FIG. 1 is a block diagram an electronic device network 105 according to an embodiment of the present invention. In the electronic device network illustrated in FIG. 1, an electronic device, for example mobile handset 107, may be communicatively coupled to at least one of a plurality of device servers, such as device server A 135 and device server B 137, for example. "The electronic device, for example, mobile handset 107, may be capable of employing one of several available update agents, for example, update agent A 113 and update agent B 115, resident in the electronic device to update software and/or firmware in the electronic device. The electronic device, for example, mobile handset 107, may also capable of provisioning the available update agents, such as for example, update agent A 113 and update agent B 115, with the necessary parameters and data needed to facilitate the respective update operations." Okonnen at [0058]: "On power up, the mobile handset 107 may execute the update application (UA) loader 127. The UA loader 127 may invoke a boot initialization code before determining to update the mobile handset 107 and determining the list of available update agents already provisioned (such as, during an initial provisioning operation initiated by the electronic device network 105) in the mobile handset 107. A list of provisioned update agents containing references to update agent A 113 and the update agent B 115 may be displayed to the end-user. When the end-user selects one of the provisioned update agents displayed, the UA loader 127 may invoke the selected provisioned update agent and transfer control to the selected update agent." *See also* Okonnen at Fig. 1. Fahrny at [0008]: "According to the present invention, a controller for managing media stream decryption keys is provided. The controller comprises a media decryption engine, a table, and a content key list. The media decryption engine generally receives an encrypted media stream from a headend and presents a decrypted media stream to a

'117 Claims	Exemplary Disclosures
	receiving device in response to a decryption key. The decryption key is generally a function of a content key. The table may contain a content key and index and a plurality of corresponding content keys. Content keys that correspond to a particular encrypted media stream may be selected from the content key list using an index from the content key table, and that is referenced by an identifier received from a headend in connection with the encrypted media stream. The table may optionally (i.e., alternatively) contain initialization vector (IV) values that may be indexed and selected." Assmann at Abstract: "An electronic mail ("e-mail") system is described that enforces authentication or encryption to/from Mail Transfer Agents and from Mail User Agents. In one embodiment, for example, an electronic mail (e-mail) system is described that comprises: a message transfer agent (MTA) available for a client to connect to; program logic for authenticating the client, through use of a certificate, in order to identify the client as approved for establishing a secured connection with the MTA; program logic for establishing a secured connection between the client and the MTA in instances where the client can be authenticated; program logic for testing encryption strength of the secured connection; and program logic for terminating the secured connection in instances where the secured connection has inadequate encryption strength." Assmann at 2:33-45: "The MTA, which is responsible for queuing up messages and arranging for their distribution, is the workhorse component of electronic mail systems. The MTA "listens" for incoming e-mail messages on the SMTP port, which is generally port 25. When an e-mail message is detected, it handles the message according to configuration settings, that is, the settings chosen by the system administrator, in accordance with relevant standards such as Request For Comment documents (RFCs). Typically, the mail server or MTA must temporarily store incoming and

'117 Claims	Exemplary Disclosures
	outgoing messages in a queue, the "mail queue." Actual queue size is highly dependent on one's system resources and daily volumes." Assmann at 4:25-36: "An electronic mail ("e-mail") system is described that provides methodology to enforce authentication or encryption to/from Mail Transfer Agents and from Mail User Agents. In accordance with the present invention, support is added to enforce certain restrictions on the connections between two hosts (a server and a client), depending on whether sendmail (Message Transfer Agent) acts as a server (receiving e-mail) or as a client (sending e-mail). In this manner, it is possible for each type of connection to enforce an authentication of the other side or at least a certain key length of the symmetric cipher used for encryption." Assmann at 4:37-58: "A method of the present invention for sending an e-mail message using a secured connection that employs encryption includes the following method steps. A client's request (e.g., Mail User Agent's request) is received at a Message Transfer Agent (MTA) for establishing a secured connection with the MTA for sending an e-mail message. The method attempts to authenticate the client, through use of a certificate. If the client cannot be authenticated, the method terminates without establishing the secured connection and without sending the e-mail message. On the other hand, if the client can be authenticated, the method establishes the secured connection between the client and the MTA. Additionally, the method (optionally) determines whether the encryption employed for the secured connection meets a predefined minimum encryption strength. If the encryption employed does not meet the predefined minimum encryption strength, the method terminates (including terminating the secured connection without sending the e-mail message). However, if the encryption employed does meet the predefined minimum encryption strength, the MTA will send the e-mail message (for ultimate delivery at a target destination)."

'117 Claims	Exemplary Disclosures
	Holbrook at [0085]: "Portable messaging units 16 are usually disconnected from all other elements of messaging system 10, except when a user brings an individual portable messaging unit 40 to an individual messaging node 20, at which time temporary second communications link 72 may be established between portable messaging unit 40 and messaging node 20. Portable messaging unit 40 and messaging node 20 are arbitrary members of portable messaging units 16 and messaging nodes 14, respectively; any portable messaging unit may be connected at any messaging node. Individual users of messaging system 10 have user accounts, which may be associated with one or more distinct messaging address identifiers, such as e-mail addresses. An instance of portable messaging unit 40 may be configured for operation with one or more user accounts." Cooley at Abstract: "Systems and methods are provided that allow an application residing on a user device to intercept messages directed to that device and to extract control information and/or data from the intercepted messages for use by the application. In one embodiment, the intercepted message can cause the application to change its configuration or to present information to the user that came from a sending website. In another embodiment, a message from a device can trigger a website to send back a desired piece of data. The returned data may contain a flag, which directs the data to a formatting server. The formatting server then converts the data into a format that allows the data to be intercepted by the desired application on the user device. This then allows the desired data to be presented to the user in the context of an application resident on the user's device." Cooley at ¶[0005]: "Systems and methods are provided that allow an application residing on a user device to intercept messages directed to that device and to extract control

Headwater Research LLC
Ex. 2011, IPR2024-00003
Page 391 of 442

'117 Claims	Exemplary Disclosures
	information and/or data from the intercepted messages for use by the application. In one embodiment, the intercepted message can cause the application to change its configuration or to present information to the user that came from a sending website. In another embodiment, a message from a device can trigger a website (such as an airline website) to send back a desired piece of data. The returned data can, for example, contain a flag, which directs the data to a formatting server. The formatting server then converts the data into a format that allows the data to be intercepted by the desired application on the user device. This then allows the desired data to be presented to the user in the context of an application resident on the user's device." Cooley at ¶[0017]: "When the user touches tile 23-1 all the flights that the user has booked on, for example, AMERICAN™ airlines (AA) are displayed. Touching tile 24-1 then yields the screen shown in FIG. 2E which then displays the information that has been received from an external source. In this case this information would have come typically from the AA server in the manner discussed above. There could be any amount of information and tiles (or layers) to this information, all controlled by application 30 residing on the user's device with the updated information coming, from time to time, via messages from various external locations. Note that in FIG. 2F the gate and status of the flight have been updated. This update could have come as a result of a request from the user or could have come as a text message from the airline as part of an update service for customers. In any event, the message would have gone first to service center 12 and then via SMS broker 13 to the user's device where application 30 would have taken the data and placed it in the proper tile."

Headwater Research LLC
Ex. 2011, IPR2024-00003
Page 392 of 442

'117 Claims	Exemplary Disclosures
13. The network system of claim 1, at least one of the devices having a network stack in communication with the device messaging agent, wherein the secure connection between the network message server and that device is terminated within the network stack.	The references below disclose and/or render obvious this limitation. For example, see the following passages and/or figures, as well as related disclosures: Ozaki at 3:40-60: "The network layer (third layer) is a layer which defines a communication method between two arbitrary nodes. In the case of TCP/IP, the third layer corresponds to an IP layer. It is possible in the data-link layer to carry out a communication between nodes on the same network medium. The role of the network layer is to communicate by using routing between two arbitrary nodes existing on the network. Here, "routing" refers to transmitting IP packets by selecting an optimum path when packets are transmitted to a host in the TCP/IP network. For example, communication is carried out in the network layer between two Ethernet segments by routing packets. Also, it is possible to route packets not through physical network media but by routing to a dial-up PPP (Point to Point Protocol) line which connects a computer to a network (Ethernet) through a telephone line, also by routing to an exclusive line using an optical fiber and the like. For this purpose, addresses (IP addresses in case of TCP/IP) which are not dependent on the physical medium are ordinarily allotted to respective nodes and the routing is carried out based thereupon. IPsec encrypts every communication in this network layer." *See also* Ozaki at Figs. 25-26. Santamaria at [0034]: "The persistent notification service connections (e.g., push notification services 110 and 120) are periodically refreshed in order to maintain the persistency of the connections in some embodiments of the invention. For example, the push notification services 110 and 120 may timeout a connection to the mobile device 160 (thus stopping push notification services for the user) if the mobile device does not actively use that connection. Thus, in some embodiments of the invention, the mobile device 160 is required to communicate with the push notification services 110 and 120 in order to maintain its persistent connections. For example, at some periodic interval, the mobile device 160 transmits a notification service connection

'117 Claims	Exemplary Disclosures
	message to the push notification services 110 and 120 to refresh the connections. The notification service connection message informs the push notification services 110 and 120 that the mobile device 160 is still connected (and presumably wishes to continue to receive items of interest from the push notification services). It should be understood that refreshing the push notification connections informs the push notification service's servers that the mobile device wishes to remain connected to the service. In addition, refreshing the push notification connections informs each network element along the route to the push notification service's servers to maintain the connection (otherwise, e.g., a network element may clear the network resources associated with that connection). The notification service connection message may be a ping message or other keep alive message. The notification services may respond to a notification service connection message to notify the mobile device 160 that the connection is active. Thus, if the mobile device 160 does not receive a response from a notification service, the mobile device 160 may be required to re-establish the session with the notification service. It should be understood that in some embodiments of the invention, receipt of an item of interest from a notification service also refreshes the persistent notification service connection." Santamaria at [0036]: "In addition, in some embodiments of the invention, the mobile device 160 is required to periodically refresh the connection with the network data access element(s) 150 in order to maintain network data connectivity (e.g. such as NAT routers in the carrier's network, firewalls, or other network elements in the network). For example, since the data context consumes network resources, the network data access element(s) 150 may tear down the data context if it is not actively being used and/or has not been used in a certain amount of time (e.g., the data context is removed and the network resources of that data context are reallocated). It should be understood that if the data context is torn down, the mobile device 160 is not connected with any notification services. Of course, it should also be understood that the data context may be torn down for other reasons

'117 Claims	Exemplary Disclosures
	besides period of inactivity (e.g., if the location of the mobile device is out of range of the network, if the mobile device 160 is turned off, etc.). In other embodiments of the invention, the data context does not have to be refreshed by the mobile device 160. Typically, the data context timeout interval is less than the notification service timeout interval. If the data context is up, a notification service connection message transmitted by the mobile device 160 to a notification service inherently refreshes the connection with the network data access element(s) 150." Assmann at 4:59-67: "The above method can be modified so that the MTA acts as the client. In that case, the method is repeated with the MTA attempting to establish a secured connection with another server (e.g., second MTA). If the server can be authenticated, the e-mail message is transmitted on the secured connection. Otherwise, the secured connection is terminated. The first MTA (client, here) can (optionally) terminate the connection if the encryption strength is inadequate." Bobo at 6:66-7:12: "When the call reaches the MSDS 10, the call is routed within the MSDS 10 in a manner that will be described in more detail below with reference to FIG. 13. At step 46, the MSDS 10 answers the telephone call and receives the address signal from the DID trunk 15. Next, at step 48, the call is established between the MSDS 10 and the third party and, at step 50, the MSDS 10 receives the message transmitted over the telephone line. The message is stored at step 52, a database within the MSDS 10 is updated at step 54, and the intended recipient of the message is notified at step 56. The intended recipient of the message uses the services provided by the MSDS 10 and will hereinafter be referred to as a user. At step 58, the message is converted into hyper-text mark-up language (HTML)." Bobo at 7:39-50: "After the user gains access to the mailbox at step 72, the user can request information stored within the MSDS 10. The MSDS 10 receives the request at step

'117 Claims	Exemplary Disclosures
	76 and, at step 78, determines whether the information exists. As is common practice, the MSDS 10 also determines the validity of the request at step 78. The request from the user will include the mailbox number for the user, the message identifier, display preferences, and, if the message is a facsimile message, a page identifier. If for any reason the request is invalid, such as when a hacker is attempting to gain access to privileged information, the request for the information will be terminated." Bobo at cl. 31: "31. A network message storage and delivery system, comprising: a central processor for receiving an incoming call, for detecting an address signal on said incoming call, for detecting a message on said incoming call, and for placing said message in a storage area, said address signal being associated with a user of said network message storage and delivery system; a network server for receiving said message from said storage area, for converting said message into a mixed media page layout language, and for placing said message in said storage area; wherein when said network server receives a request from said user over said network, said network server transmits at least a portion of said message over said network to said user and wherein said network comprises the Internet and said network server comprises an Internet server."
14. The network system of claim 1, wherein at least one of the applications on at least one of the devices and the network application server corresponding to that application authenticate with each other prior to passing application data	The references below disclose and/or render obvious this limitation. For example, see the following passages and/or figures, as well as related disclosures: Chou at 10:22-29: "As noted previously, certain prior approaches may permit an authenticated user direct access to information content associated with the provision of enterprise communication services. Security issues can arise in these approaches, for example,

'117 Claims	Exemplary Disclosures
via the device messaging agent on that device and the network message server.	if the authenticated user is a remote user located outside an enterprise firewall and a direct pipe is established through the firewall to connect the remote user to the enterprise communication services." Chou at 2:27-37: "The mobile client device may register with the enterprise application through interaction with the wireless secure server. For example, the wireless secure server may comprise a password-protected register connector through which registration information associated with the mobile client device is supplied from the wireless secure server to the enterprise application. The wireless secure server may be operative to obtain user and device profile information from the mobile client device, and to store the profile information temporarily until the profile information is extracted from the wireless secure server by the enterprise application." Chou at 4:14-21: "The WSS 106 further includes a secure connector 120 which illustratively includes a password-protected register connector 122 and an application connector 124. Security can be provided in these elements using conventional encryption or other cryptographic operations of a type known in the art. The application connector 124 communicates alert and/or notification information to the mobile client 110 via PI 116 and PPG 114 as shown." Rakic at [0041]-[0051]: "For the client device, using the codes to validate the secure push message may protect the client device in a number of ways. For example, using the codes, the client device may authenticate the sender. In another example, using the codes may verify that an intended recipient of the secure push message is the client device, and that the client device includes a correct component. For instance, using the codes may

'117 Claims	Exemplary Disclosures
	verify that the secure push message is intended for a device that includes a specific Subscriber Identity Module (SIM) card. "In the above, for the client device, using the codes in the electronic signature of the secure push message may validate the secure push message in a number of ways. In this manner, the secure push message may provide a high level of security. The format of the secure push message may be simple and flexible, and may be easily modified to accommodate different types of applications/devices that receive/send secure push messages. "Exemplary Network and Devices "FIG. 1 illustrates the concepts described herein. As shown, network 100 may include a client device 102, an administrator device 104, a gateway 106, and a database device 108. Depending on the implementation, network 100 may include additional, fewer, or different devices than those illustrated in FIG. 1. For example, network 100 may include multiple client devices 102. "Device 102 may include any of the following devices that have the ability to or are adapted to communicate and interact with another device, such as a radiotelephone or a mobile telephone with ultra wide band or Bluetooth communication capability; a personal communications system (PCS) terminal that may combine a cellular radiotelephone with data processing, facsimile, and/or data communications capabilities; an electronic notepad, a laptop, and/or a personal computer that communicate with wireless peripherals (e.g., a wireless keyboard, speakers, etc.); a personal digital assistant (PDA) that can include a telephone; a Global Positioning System device and/or another type of positioning device; a gaming device or console; a peripheral (e.g., wireless headphone); a digital camera; or another type of computational or communication device.

'117 Claims	Exemplary Disclosures
	"Administrator device 104 may include any device that may be used to administer and/or control client device 102. Examples of administrator device 104 include a server, a personal computer, a laptop, and/or any other type of device with sufficient memory, processing speed, and/or bandwidth to administer/control one or more client devices 102. "Gateway 106 may include a device via which client device 102 may communicate with devices in network 100. In one implementation, gateway 106 may receive push messages from administrator device 104 in accordance with the push access protocol, as specified by wireless application protocol (WAP) specification 164 (WAP-164) or WAP-247. In addition, gateway 106 may send the push messages from administrator device 104 to client device 102 in accordance with push over-the-air (OTA) protocol, as specified by WAP specification 189 (WAP-189) or WAP-235. "Database device 108 may include a database of information about client device 102. In some implementations, the database may be included in administrator device 104, and in such instances, network 100 may not include a separate database device 108. "In the above, administrator device 104 may configure client device 102 via a cable 110. Alternatively, administrator device 104 may configure client device 102 via wireless communications. During the configuration, administrator device 104 may obtain information that is specific to client device 102, store the information in a database on database device 108, and store an administrator code in client device 102. "Once client device 102 is configured, administrator device 104 may retrieve the information related to client device 102 from database device 108, generate an electronic signature by signing a message based on the information, and combine the message with the electronic signature to produce a secure push message.

'117 Claims	Exemplary Disclosures
	Subsequently, administrator device 104 may send the secure push message to client device 102. "When client device 102 receives the secure push message, client device 102 may validate the secure push message based on the electronic signature. If client device 102 is able to validate the secure push message, client device 102 may send a response to administrator device 104 to indicate that client device 102 has successfully received and/or validated the secure push message. "Depending on the implementation, client device 102 may perform a specific task in accordance with contents of the secure push message. For example, client device 102 may set parameters that are related to an application in client device 102, update firmware, reset parameters that are associated with subscription services, set security parameters, etc." Rakic at [0065]-[0067]: "Secure push message server 504 may send secure push messages to client device 102 via a wireless network or gateway 106. Given a message or a payload from an application, a user, or an administrator, secure push message server 504 may generate a key, an electronic signature, and a secure push message. "Secure push message server 504 may generate the key based on codes (e.g., IMEI) that are obtained from client device 102, a component in client device 102 (e.g., removable memory 218), and an administrator. Depending on the implementation, additional codes or information may be used to generate the key. Once the key is generated, secure push message server 504 may generate an electronic signature based on the key and the message. "In possession of the electronic signature and the key, secure push message server 504 may generate and send the secure push message to client device 102. In one implementation, secure push message server 504 may generate the secure push

'117 Claims	Exemplary Disclosures
	message by appending the electronic signature to the message and/or other information (e.g., a timestamp)." Rakic at [0086]-[0088]: "The subscriber information, the client device identifier, and the administrator code may be stored in a database (block 910). For example, as illustrated in FIG. 10, administrator device 104 may store the subscriber information (e.g., IMSI, PUK, the phone number, etc.), the IMEI, and the administrator code in database 602 hosted on database device 108. In a different implementation, administrator device 104 may store the subscriber information, the IMEI, and the administrator code in a database that is hosted on administrator device 104. "FIG. 11 is a flow diagram of an exemplary process 1102 for enabling a client device to receive a push message, and FIG. 12 illustrates an example associated with process 1102. Assume that a component from which the subscribed information is obtained in process 902 is installed in client device 102 (e.g., SIM card 1002 is installed in client device 102). Client device 102 and administrator device 104 may communicate with one another via a wireless or wired communication link. "Process 1102 may begin at block 1104, where the client device identifier and a portion of the subscriber information may be used to retrieve the administrator code and other portions of the subscriber information (block 1104). For example, as shown in FIG. 12, administrator device 104 may send a database query that includes the portion of the subscriber information (e.g., phone number) and the client device identifier (e.g., IMEI), IMSI, etc. As further shown, administrator device 104 may receive other portions of the subscriber information (e.g., PUK) and the administrator code from database device 108 in response to the query." Rakic at [0109]-[0112]:

'117 Claims	Exemplary Disclosures
	"Process 1402 may begin at 1404, where the secure push message may be received (block 1404). For example, client device 102 may receive a secure push message that is sent from administrator device 104 at block 1318. "Information for constructing a key may be obtained from within client device 102 (block 1406). In one implementation, secure push message client 404 may retrieve the administrator code, which has been stored in client device 102 at block 1108 (FIG. 11), an IMEI (e.g., a client device identifier) of client device 102, and the subscriber information from a component (e.g., SIM card 218 or 1002) within client device 102. "The key may be constructed (block 1408). In some implementations, secure push message client 404 may construct the key by combining the administrator code, the subscriber information, and the client identifier. In one implementation, secure push message client 404 may construct the key in accordance with expression (3). "It may be determined whether the secure push message is valid (block 1410). For example, to validate the secure push message, secure push message client 404 may obtain data block 804 and signature block 806 from the secure push message that is received at block 1404, sign obtained data block 804 based on the key constructed at block 1408, and compare the value of signature field 826 in signature block 804 to the signed data block 804. If the signed data block 804 is identical to the value of signature field 826, secure push message client 404 may determine that the secure push message is valid. Otherwise, secure push message client 404 may determine that the secure push message is invalid." *See also* Rakic at Figs 1, 4, 5, 8. Anderson at 669-670: "Another approach is to do encryption and/or authentication at the IP layer using a protocol suite known as Ipsec. IPsec defines a *security association* as the

'117 Claims	Exemplary Disclosures
	combination of keys, algorithms and parameters used to protect a particular packet stream. Protected packets are either encrypted or authenticated (or both); in the latter case, an authentication header is added that protects data integrity using HMAC-SHA1, while in the former the packet is encrypted and encapsulated in other packets. (The use of encryption without authentication is discouraged as it's insecure [151].) There's also an *Internet Key Exchange* (IKE) protocol to set up keys and negotiate parameters. IKE has been through a number of versions (some of the bugs that were fixed are discussed in [465]). "IPsec is widely used by firewall vendors who offer a *virtual private network* facility with their products; that is, by installing one of their boxes in each branch between the local LAN and the router, all the internal traffic can pass encrypted over the Internet. Individual PCs, such as workers' laptops and home PCs, can in theory join a VPN given a firewall that supports IPsec, but this is harder than it looks. Compatibility has been a major problem with different manufacturers' offerings just not working with each other; although firewall-to-firewall compatibility has improved recently, getting random PCs to work with a given VPN is still very much a hit-or-miss affair." Anderson at 670-71: "Recall that when discussing public key encryption, I remarked that a server could publish a public key *KS* and any web browser could then send a message *M* containing a credit card number to it encrypted using *KS*: $\{M\}_{KS}$. This is in essence what the TLS protocol (formerly known as SSL) does, although in practice it is more complicated. It was developed to support encryption and authentication in both directions, so that both http requests and responses can be protected against both eavesdropping and manipulation. It's the protocol that's activated when you see the padlock on your browser toolbar." "Here is a simplified description of the version as used to protect web pages that solicit credit card numbers:

'117 Claims	Exemplary Disclosures
	1. the client sends the server a *client hello* message that contains its name *C*, a transaction serial number *C#*, and a random nonce N_C; 2. the server replies with a *server hello* message that contains its name *S*, a transaction serial number *S#*, a random nonce N_S, and a certificate *CS* containing its public key *KS*. The client now checks the certificate *CS* back to a root certificate issued by a company such as Verisign and stored in the browser; 3. the client sends a *key exchange* message containing a *pre-master-secret* key, K_0, encrypted under the server public key *KS*. It also sends a *finished* message with a message authentication code (MAC)computed on all the messages to date. The key for this MAC is the *master-secret*, K_1. This key is computed by hashing the pre-master-secret key with the nonces sent by the client and server: $K_1 = h(K_{CS},N_C,N_S)$. From this point onward, all the traffic is encrypted; we'll write this as $\{\ldots\}_{KCS}$ in the client-server direction and $\{\ldots\}_{KSC}$ from the server to the client. These keys are generated in turn by hashing the nonces with K_1. 4. The server also sends a finished message with a MAC computed on all the messages to date. It then finally starts sending the data. $C \rightarrow S : C,C\#, N_C$ $S \rightarrow C : S, S\#,N_S,CS$ $C \rightarrow S : [K0]_{KS}$ $C \rightarrow S : [finished,MAC(K_1,everythingtodate)]_{KCS}$ $S \rightarrow C : [finished,MAC(K_1,everythingtodate)\}_{KSC},[data]_{KSC}$ "The design goals included minimizing the load on the browser, and then minimizing the load on the server. Thus the public key encryption operation is done by the client, and the decryption by the server; the standard encryption method (*ciphersuite*) uses RSA for which encryption can be arranged to be very much faster than decryption. (This was a wrong design decision as browsers generally have a lot more compute cycles to spare than servers; it has created a

'117 Claims	Exemplary Disclosures
	brisk aftermarket for crypto accelerator boards for web servers.) Also, once a client and server have established a pre-master secret, no more public key operations are needed as further master secrets can be obtained by hashing it with new nonces." *See generally* Anderson at Ch. 21 ("Network Attack and Defense"); § 21.4.5.5 ("IPsec"); § 21.4.5.6 ("TLS"). Shenfield at [0143]-[0146]: "Reference is now made to Figure 16. When a content provider 110 registers with a push proxy 410, various steps are performed. A first step 1610 includes analyzing required client settings for content storage and delivery. This can be used, for example, for service advertisement in order to identify push clients 510 on devices capable of consuming content from content provider 110. "A second step 1620 allows push proxy 410 to set up the environment, including proxy storage, delivery options, transformation options, among others. "In step 1630, push proxy 410 can check whether the application is already registered to obtain content from a content provider 110. If this is the case, the application is ready to receive content and a notification from push proxy 410 to content provider 110 that the delivery channel is established and the application is ready for content can be sent. "Step 1630 can occur, for example, if an application is pre-installed on a device prior to content provider 110 coming on-line. Thus, the application is waiting for content provider 110 to become available or the application is of generic type (e.g. a browser or RSS Viewer) and is capable of consuming information from multiple content providers. In an alternative setting, if content provider 110 is already available before the application is installed, the notification step 1530 in Figure 15 can be used to initiate the content starting to flow from content provider 110 to a client application 150." *See also* Shenfield at [0147]-[0165].

'117 Claims	Exemplary Disclosures
	Hwang at [0102]-[0103]: "In step 803, using the information acquired from Table 11, the Terminal 800 generates a Roaming Request message for making a roaming registration request to the Visited SP 810, and sends the message to the BSM 813 of the Visited SP 810. The contents of the Roaming Request message generated by the Terminal 800 in step 803 are shown in Table 16 below. Table 16 Request ID Terminal ID Home SP ID Home SP BSM ID Purchase Item ID "Request ID, a first item of Table 16, is an identifier given to consistently identify and manage one roaming registration request procedure. The Request ID is an identifier generated for allowing a terminal to uniquely identify its own roaming registration request. The Request ID is constant from the time it is generated in FIG. 8 until the time the roaming registration request procedure of FIG. 8 ends. Terminal ID is a unique ID for the terminal to uniquely identify a terminal. The Terminal ID is used for identifying who makes the roaming registration request. Home SP ID is an identifier used for allowing the roaming terminal to inform the Visited SP 810 who it's Home SP 820 is. Using this information, the Visited SP 810 can determine to which entity the roaming-requested Terminal 800 originally belongs, and can also determine a roaming relationship with the corresponding entity. Here, when the BSM 813 of the Visited SP 810 determines from the Home SP ID that there is no roaming agreement made with the BSM 813 itself, the BSM 813 directly proceeds to step 806 where it can notify a roaming unavailability to the Terminal 800."

Headwater Research LLC
Ex. 2011, IPR2024-00003
Page 406 of 442

'117 Claims	Exemplary Disclosures
	Hwang at [0105]-[0107]: "In step 804, for the roaming registration request received from the Terminal 800 in step 803, the BSM 813 of the Visited SP 810 sends a roaming registration request to the BSM 823 of the Home SP 820 of the Terminal 800. The contents of the message sent from the BSM 813 of the Visited SP 810 to the BSM 823 of the Home SP 820 in step 804 are shown in Table 17 below. Table 17 Request ID Terminal ID Visited SP ID Visited SP BSM ID "Request ID, a first item of Table 17, is an identifier given to consistently identify and manage one roaming registration request procedure. The Request ID is an identifier generated for allowing a terminal to uniquely identify its own roaming registration request. The Request ID is constant from the time it is generated in FIG. 8 until the time the roaming registration request procedure of FIG. 8 ends. Terminal ID is a unique ID for the terminal to uniquely identify a terminal. The Terminal ID is used for identifying who makes the roaming registration request. Visited SP ID is used by the Visited SP 810 to provide its own information to the BSM 823 of the Home SP 820 of the roaming-requested terminal. Visited SP BSM ID is used for providing the BSM 823 of the Home SP 820 with the information indicating the entity with which the Visited SP 810 will make a roaming-related negotiation. This is because the Visited SP 810 can have several BSMs. "In step 805, the BSM 823 of the Home SP 820 sends a response to the roaming registration request received in step 804. Table 18 below shows the contents of the

'117 Claims	Exemplary Disclosures
	message sent from the BSM 823 of the Home SP 820 to the BSM 813 of the Visited SP 810. Table 18 Request ID Roaming Authorization Status Terminal Subscription Type Isomursu at 12:29-46: "The definition menu-item-token can be an optional command to be used by the server for authorization purposes if security is needed, if not then it can be omitted. Once a menu item has been sent to a terminal with a menu-item-token set, no command can be given to change or remove the existing menu-item without the same menu-item-token. This kind of authorization feature can be used in connection to other applications described here as well, not just in relation to the Menu application. For example a ring tone would be updated only if it is accompanied by the correct authorization word or code, like the menu-item-token. The definition menu-group-name specifies the menu group into which the menu item with name menu-item-name is put. If a menu item with the same name in the same group already exists (and the server is authorized to update the menu item by the correct menu-item-token), then the existing menu-item is replaced with the new one. The name of the menu group is placed in apostrophes in context with the definition." Isomursu at 13:48-14:20: "Authorisation to change menus may defined more closely by authorisation primitives. There are three authorization primitives which are add, remove, and list. Authorization lists may especially be used for operator and manufacturer menus. It is implementation specific how many entries the terminal is capable of storing in the authorization list of each menu. The authorization primitives and commands for

Headwater Research LLC
Ex. 2011, IPR2024-00003
Page 408 of 442

'117 Claims	Exemplary Disclosures
	entering them into the terminal can be implemented in a similar manner as was described above for menu-item primitives. The authorization commands are privileged in such sense that they will be accepted only for a menu with authorization list active, and when sent from an already authorized server. The first primitive authorization command, authorization-add, can be used to add one or more new authorized servers to the list of authorized servers for the given menu (menu-group-name). The menu-authorization-server-list comprises one or more lines of information, each defining one authorized server. For GSM/SMS, an authorized server is defined by a pair defining identity of the short message service centre and of the server, e.g. (MS-ISDN of a SMSC which is the mobile subscriber international ISDN number for the SM-SC; MS-ISDN of the server, which is the mobile subscriber international ISDN number for the authorized server). The command sequence can be following: <authorization-add>menu-group-name menu-authorization-server-list The second primitive authorization command, authorization-remove, can be used to remove one or more authorized servers from the list of authorized servers for the given menu (menu-group-name). The command sequence can be following: <authorization-remove>menu-group-flame menu-authorization-server-list The third primitive authorization command, authorization-list, can be used to request the handset to send the list of privileged servers for a requested menu (menu-group-name). The availability of this command depends on the handset implementation. The command sequence can be following: <authorization-list>menu-group-name." Okonnen at [0013]-[0014]:

Headwater Research LLC
Ex. 2011, IPR2024-00003
Page 409 of 442

'117 Claims	Exemplary Disclosures
	"In an embodiment according to the present invention, the number assignment module programming activity may comprise at least one of over-the-air service provisioning (OTASP) activity and over-the-air parameter administration (OTAPA) activity. "In an embodiment according to the present invention, the provisioned data unit may be adapted to store at least one of update agent related provisioning information, a universal resource locator of a server used to retrieve updates, and a security key used to authenticate server messages." Okonnen at [0021]: "In an embodiment according to the present invention, the index of provisioned update agents may provide an index value used to compute an address location of a provisioned update agent. The index of provisioned update agents may be an index to a table containing an address for an update agent in non-volatile memory the electronic device." Okonnen at [0052]: "The provisioned data 129 may also be used to store update agent related provisioning information, such as a universal resource locator (URL) of a device server or a management server that may be used to retrieve updates, a security key(s) that may be used to authenticate messages to/from the device server, and selective encryption/decryption of the messages, etc. Each update agent, such as update agent A 113 and update agent B 115, may have corresponding entries in provisioned data 129." Okonnen at [0065]: "In another embodiment according to the present invention, the index of provisioned update agent 209 may be an index to a table containing an address for an update agent in storage, such as for example, non-volatile memory 111 in mobile handset 107. The security key(s) 213 may be used to authenticate updates, for example, to authenticate during download of updates or during update activity."

'117 Claims	Exemplary Disclosures
	Fahrny at [0023]: "Keylist: A list of decoder addresses and respective decoder keys in ordered pairs. Keylists may be used by the Uplink Control System (UCS) for generation of authorization messages that are addressed to the diagnostic circuit that is embedded in decoders that are specific to the encoder system." Fahrny at [0041]: "Referring to FIG. 1 a, a diagram illustrating a media decoder (i.e., controller, processor, apparatus, circuit, device, etc.) 100 of the present invention is shown. The decoder 100 may be implemented in connection with a digital media stream distribution system (described in more detail in connection with FIGS. 2(a-b)). The controller 100 is generally implemented as a security processor (or processing system) that provides at least one security feature (e.g., encryption, decryption, authentication, security key management, copy protection, digital rights management, etc.) to at least one digital media input/output stream. The decoder 100 generally has an input 102 that receives at least one signal (e.g., VIDIN and PID), an input/output 104 that receives/presents a signal (e.g., TFHE) as well as additional data, signals, messages, and the like, an input 106 that receives a working key modifier and application function signal (e.g., WKM), and an output 108 that presents a signal (e.g., VIDOUT)." Assmann at 4:25-36: "An electronic mail ("e-mail") system is described that provides methodology to enforce authentication or encryption to/from Mail Transfer Agents and from Mail User Agents. In accordance with the present invention, support is added to enforce certain restrictions on the connections between two hosts (a server and a client), depending on whether sendmail (Message Transfer Agent) acts as a server (receiving e-mail) or as a client (sending e-mail). In this manner, it is possible for each type of connection to enforce an authentication of the other side or at least a certain key length of the symmetric cipher used for encryption."

'117 Claims	Exemplary Disclosures
	Assmann at 4:37-58: "A method of the present invention for sending an e-mail message using a secured connection that employs encryption includes the following method steps. A client's request (e.g., Mail User Agent's request) is received at a Message Transfer Agent (MTA) for establishing a secured connection with the MTA for sending an e-mail message. The method attempts to authenticate the client, through use of a certificate. If the client cannot be authenticated, the method terminates without establishing the secured connection and without sending the e-mail message. On the other hand, if the client can be authenticated, the method establishes the secured connection between the client and the MTA. Additionally, the method (optionally) determines whether the encryption employed for the secured connection meets a predefined minimum encryption strength. If the encryption employed does not meet the predefined minimum encryption strength, the method terminates (including terminating the secured connection without sending the e-mail message). However, if the encryption employed does meet the predefined minimum encryption strength, the MTA will send the e-mail message (for ultimate delivery at a target destination)." Assmann at 7:62-8:19: "Methodology to Enforce Authentication or Encryption to/from an MTA or from a MUA "A. Including TLS Support within the Sendmail MTA "STARTTLS has been implemented in the sendmail MTA for Sendmail Secure Switch 2.0. It is based on existing solutions, including OpenSSL (see e.g., OpenSSL: The open source toolkit for ssl/tls, currently available online at http://www.openssl.org/) and example programs and a patch for Postfix (see e.g., Lutz Jaenicke, Postix, 2000, currently available at http://www.aet.tu-cottbus.de/personen/jaenicke/pfixtls/). X.509 certificates tie a Distinguished Name (DN) together with a public key and those data structures are digitally signed by a Certificate Authority (CA). According to the X.509 standard, a Distinguished Name consists of several named fields, such as Country (C), State (ST), Location (L), Organization (O), and a Common Name (CN). The DN of the CA is called the cert

'117 Claims	Exemplary Disclosures
	issuer, and the DN of the cert itself is called the cert subject. The sendmail binary (i.e., "sendmail") uses a configuration file (sendmail.cf) to define several options, routing of e-mail, rewriting of addresses, and rejection or acceptance of connections and relaying attempts. The interaction of sendmail with its configuration file is defined by options, macros, classes, rulesets, and mailer definitions. For example, the decision whether an e-mail is relayed is based on a ruleset called check rcpt." Assmann at 8:34-45: "TLS only allows one to control the connection between two hosts: a server and a client. In accordance with the present invention, therefore, support is added to enforce certain restrictions on these connections, depending on whether sendmail acts as a server (receiving e-mail) or as a client (sending e-mail). In this manner, it is possible for each type of connection to enforce an authentication of the other side or at least a certain key length of the symmetric cipher used for encryption. In order to understand how that may be done in accordance with the present invention, it is first helpful to briefly review (E)SMTP, the (Extended) Simple Mail Transfer Protocol." Assmann at 15:23-33: "As illustrated by the above source code and also shown in accompanying FIGS. 4A–I, the system's authentication methodology for supporting remote connections to/from a Message Transfer Agent is as follows. At line 1, the method defines a map called "arith" to perform arithmetic operations. At line 3, the method next defines a class {tls} which contains VERIFY and ENCR as elements. At line 15, the method defines a ruleset called "LookUpDomain" which has four parameters all of which are included in angle brackets:" Lee II at 11: "Figure 6a shows how a user connects to the web server function (321) of a mail server (170) and logs in according to a good embodiment of the present invention. In other words, the user (150) accesses the mail server (170) using a web browser,

'117 Claims	Exemplary Disclosures
	and the user (1 50) presents the user ID and password to the mail server (170), and the mail server (170) verifies the user's identity and allows the user to log in. "In the way a user logs in to the web server (170), which can send and receive multiple types of messages (160), the user (1 50) accesses the web browser and enters an ID (identification) and password, and the web interface block (640), which is an extension of the web server block (630), requires the user to check the valid usage value in the data access block (600). the data access block (600) requires information about the user (150) to the service data function (650) with the ID received from the web interface block (640) the previous month, and the data access block (600) determines whether the user is a valid user with the user information returned from the data function (150), and the access block (600) provides an operation method that includes returning a valid user to the web interface (640) as true or false." Holbrook at [0085]: "Portable messaging units 16 are usually disconnected from all other elements of messaging system 10, except when a user brings an individual portable messaging unit 40 to an individual messaging node 20, at which time temporary second communications link 72 may be established between portable messaging unit 40 and messaging node 20. Portable messaging unit 40 and messaging node 20 are arbitrary members of portable messaging units 16 and messaging nodes 14, respectively; any portable messaging unit may be connected at any messaging node. Individual users of messaging system 10 have user accounts, which may be associated with one or more distinct messaging address identifiers, such as e-mail addresses. An instance of portable messaging unit 40 may be configured for operation with one or more user accounts." Holbrook at [0173]: "At the start of data exchange 90, at step 91, CPU 28 detects the placement of a portable messaging unit 40 in one of docking ports 26. This detection may comprise noticing an infrared signal from portable messaging unit 40, or a physical contact

'117 Claims	Exemplary Disclosures
	with a microswitch. In step 92, CPU 28 establishes the identity of user accounts associated with portable messaging unit 40, and performs account verification functions. In step 93, portable messaging unit 40 transfers outgoing messages to messaging node 20. This is done at an early stage to free non-volatile memory for the storage of new incoming messages subsequently received. In step 94, portable messaging unit 40 reports the amount of free non-volatile memory available for the storage of new incoming messages, and in step 95 messaging node 20 applies a prioritization algorithm, selecting messages for transfer if the memory is insufficient for the storage of all new messages. Then, in step 96, messaging node 20 transfers these selected messages to portable messaging unit 40. Finally, in step 97, status light 32 indicates the transaction completion of data exchange 90, at which time the user may remove portable messaging unit 40 from docking ports 26." Bobo at 7:39-50: "After the user gains access to the mailbox at step 72, the user can request information stored within the MSDS 10. The MSDS 10 receives the request at step 76 and, at step 78, determines whether the information exists. As is common practice, the MSDS 10 also determines the validity of the request at step 78. The request from the user will include the mailbox number for the user, the message identifier, display preferences, and, if the message is a facsimile message, a page identifier. If for any reason the request is invalid, such as when a hacker is attempting to gain access to privileged information, the request for the information will be terminated." XMPP at 9: "1. Stream Authentication – a client MUST complete stream authentication as documented in [XMPP-CORE] before attempting to establish a session or send any XML stanzas." SIPE at §11: "In normal usage, most SIP requests are used to setup and modify communication sessions. The actually communication between participants happens in the media

Headwater Research LLC
Ex. 2011, IPR2024-00003
Page 415 of 442

'117 Claims	Exemplary Disclosures
	sessions, not in the SIP requests themselves. The MESSAGE method changes this assumption; MESSAGE requests normally carry the actual communication between participants as payload. This implies that MESSAGE requests have a greater need for security than most other SIP requests. In particular, UAs that support the MESSAGE request SHOULD support end-to-end authentication, body integrity, and body confidentiality. and integrity mechanisms."
16. The network system of claim 2, wherein the network message server transmits the collected and buffered requests to the particular device upon the occurrence of a transmission trigger.	The references below disclose and/or render obvious this limitation. For example, see the following passages and/or figures, as well as related disclosures: Hämäläinen at [0007]-[0008]: "The service profile prepared by the system administrator on the console using the web-based interface can specify Information Push message delivery formats, such as Short Messaging Service (SMS) messages, Multimedia Messaging Service (MMS) messages, emails, and the like. The service profile can specify scheduled delivery or delivery triggered by certain specified events. The service profile can specify delivery to the mobile device using a short-range wireless access point or optionally a GSM, GPRS, or 3 G telephone network. Additionally, the service profile can specify backup delivery channels if the specified primary channel fails to send the user the push message. "In another aspect of the invention, a service manager server manages the delivery of push messages to mobile wireless devices having diverse routing, scheduling, and context or event-triggering requirements. The service manager server includes a service profile server for storing a plurality of service profiles of content delivery characteristics. The profiles each specify a bearer, using a specified delivery mode, for pushing a message to one of a plurality of mobile wireless devices of a specified recipient satisfying one of the plurality of service profiles when the one mobile wireless device is located within a coverage area of one of a plurality of access points. The service manager server also includes a content aggregation logic coupled to the service profile server, for polling web servers identified by a content

'117 Claims	Exemplary Disclosures
	characteristic in a service profile, accessing content from the web servers, and storing the content in a content queue. The service manager server also includes a service logic coupled to the content aggregation logic, the content queue, and the service profile server, for receiving a notification that content has been accessed and handling the accessed content in accordance with the service profiles in the service profile server. The service manager server also includes a context state logic coupled to the service logic and to the plurality of access points, for maintaining context information about the plurality of mobile wireless devices, including whether a specified mobile device can be reached over a communications link. The service manager server further includes a messaging logic coupled to the context state logic for selecting a specified primary bearer to send the push message to the specified mobile device if the specified mobile device can be reached over the primary bearer." Hämäläinen at [0021]-[0024]: "The Service Manager 170 pushes content, e.g. the messages 12A and 12B to the user's wireless device 100 based on the characteristics of the context of the device 100 at the respective access points 140A and 140B. The characteristics of the contexts that trigger the Service Manager 170 to push messages 12A and 12B to device 100 are specified by a service profile prepared by the system administrator on the console 175 using e.g. the web-based interface 400, 400′, or 400″ shown in FIGS. 4, 5, and 6, respectively. "The service profile prepared by the system administrator on the console 175 using the web-based interface 400, 400′, or 400″ can specify Information Push content and message delivery formats e.g. SMS messages including text and binary messages (for example WAP Push messages), MMS messages, emails, IP data stream or files such as audio, video and/or data content e.g. via FTP or DVB-H, etc. The service profile can specify scheduled delivery or delivery triggered by certain specified events. The service profile can specify delivery to the mobile device 100 using the short-range wireless access point 140 or optionally the wireless telecommunication, such as GSM, GPRS, or 3G telephone network. Additionally,

'117 Claims	Exemplary Disclosures
	the service profile can specify backup delivery channels if the specified primary channel fails to send the user the push content or message. Further, the system administrator can itself create ad hoc Information Push content and messages. "The Service Manager 170 in FIG. 1 includes Content Aggregation Logic 172, Context State Logic 174, Service Logic 176, and Messaging Logic 178. A more detailed system diagram of the components of the Service Manager 170 is shown in FIG. 2. "The Content Aggregation Logic 172, also referred to herein as the Content Fetcher 172 in FIG. 2, in one embodiment of the present invention, periodically polls content servers 190, such as web servers, and checks if content specified by the administrator's web-based console 175 has changed. The web servers 190 and specification of content are defined by the service profile in the Service Profile Server 204. If the content has changed, it will be stored to the Content Queue 205 and a content changed notification will be sent to the Service Logic 176, also referred to as the InfoPush Service Handler 176." *See also* Hämäläinen at Figs. 1, 2. Shenfield at [0176]-[0178]: "As described with reference to Figure 4 above, content may be provided that includes more information than the user may currently needs. For example, if the user requests location information for restaurants within his area, a service provider may wish to add advertising such as other services available in the area. However, the service provider may not wish to push this additional content immediately to the user, but instead provide a primer such as a headline or a table of contents showing the additional content. "In other situations, the content may be too large to send to the user, and the user may receive only the first part of the content and the remainder of the content is stored in a deferred retrieval message store 452.

'117 Claims	Exemplary Disclosures
	"Thereafter, the stored content can be passed to push client 510 either by push proxy 410 or when asked for my push client 510." Shenfield at [0182]-[0184]: "Alternatively, deferred retrieval manager 552 could also monitor network conditions as sent from wireless network 130 and if the data rate is below a certain rate can ask content pull broker 554 to pull content from deferred retrieval message store 452. "Alternatively, deferred retrieval manager 552 could see that the network status is favorable for pulling larger amounts of data, such as if the mobile device has connected with a WiFi network, and ask content pull broker 554 to pull the data from deferred retrieval message store 452. "As will be further appreciated, a user can always request to have the content pulled. Thus user request 1940 could also be used to trigger content pull broker 554 to pull the data from deferred retrieval message store 452." *See also* Shenfield at [0179]-[0181]. Santamaria at Abstract: "A method and apparatus to manage notification service connections. In one embodiment of the invention, a mobile device schedules a notification service connection message transmission interval for each of a plurality of notification service connections. The mobile device also associates a notification service connection message transmission window with each notification service connection message transmission interval. The mobile device synchronizes transmission of notification service connection messages based on overlapping notification service connection message transmission windows." Santamaria at [0005]: "Users of a mobile device (e.g., laptop, palmtop, portable media player, smartphone, multimedia mobile phone, mobile gaming systems such as a

'117 Claims	Exemplary Disclosures
	"Gameboy", etc.) may subscribe to one or more notification services. For example, users may subscribe to one or more "push" email services such as .Mac, Microsoft Exchange ActiveSync, push-IMAP, Yahoo! Push, etc. In the case of a push email service, for example, the email server may automatically transmit email messages and/or calendar updates to the email client on the mobile device for the user without the user requesting the item. In other words, the user does not request (poll) the email server for email messages or other items of interest. Thus, a push notification service is a persistent notification service. In addition, other types of services may use a similar push architecture (e.g., update/upgrade services, news services, weblog services, podcast services, etc.). In order to maintain a push notification service, the mobile device periodically refreshes the connection to the push notification service (e.g., by transmitting a ping message to the push server)." Santamaria at [0058]: "It should be understood that battery resources are conserved by synchronizing transmission of notification service connection messages based on overlapping notification service connection message transmission windows. For example, consolidating transmission of notification service connection messages for multiple notification services during a same wake period conserves battery resources. For example, waking a mobile device uses a significant amount of battery resources. Additionally, more battery resources are used when waking the device as compared with transmitting notification service connection messages. Thus, transmitting a notification service connection message to multiple notification services during the same wake period conserves battery resources (e.g., the mobile device is not independently awakened from sleep mode for each notification service connection message transmission). Thus, in some embodiments of the invention, synchronizing transmission of notification service connection messages based on overlapping notification service connection message transmission windows reduces the number of mobile device wakes needed, which reduces power consumption. It should be understood that the battery resource savings increase as the number of notification service clients increases."

'117 Claims	Exemplary Disclosures
	Okonnen at [0055]-[0056]: "In an embodiment according to the present invention, mobile handset 107 may display a list of available update agents to an end-user and solicit selection of an update agent to be used to update at least one of software and firmware. "In an embodiment according to the present invention, the mobile handset 107 may invoke an update agent based upon an update currently being processed, for example, based upon the update contents and/or characteristics, provided that the update agent is provisioned in the mobile handset 107. The update agent may also be capable of rebooting the mobile handset 107 after update completion. In an embodiment according to the present invention, a list of available updates may initially be displayed to the end-user to prompt a selection of an update by the end-user." Okonnen at [0058]: "On power up, the mobile handset 107 may execute the update application (UA) loader 127. The UA loader 127 may invoke a boot initialization code before determining to update the mobile handset 107 and determining the list of available update agents already provisioned (such as, during an initial provisioning operation initiated by the electronic device network 105) in the mobile handset 107. A list of provisioned update agents containing references to update agent A 113 and the update agent B 115 may be displayed to the end-user. When the end-user selects one of the provisioned update agents displayed, the UA loader 127 may invoke the selected provisioned update agent and transfer control to the selected update agent." Staack at [0009]: "In accordance with one aspect of the invention, the instant messaging service is enhanced by providing a storing capability for messages. When the intended receiver of the message is presently unable to receive the message because he is e.g. not on-line, busy and/or not reachable by the network, e.g. by the proxy server of the receiving user, because of any other reason, the message may be stored. This

'117 Claims	Exemplary Disclosures
	saving of the message enables its later delivery to the receiving user when this user is able to receive the message, e.g. after re-attachment to the network. No connection for bi-directional communication needs to be established." Staack at [0021]: "FIG. 1 shows a first embodiment of the invention and illustrates a case where a message is to be sent from a first network element 1 (user A) to a second network element 3 (user B). The network elements 1, 3 are, in the 25 present embodiment, client or user equipments such as terminals. In the present example, the network element 1 (user A) is an equipment trying to send a message (e.g. "MESSAGE user b@sonera.com" addressed to user b@sonera.com) to the receiving element 3 (user B) which is presently out 30 of reach, e.g. switched-off, busy, or located in a nonsupported area, or the like. The connection request of network element 1 is handled by a network element 2 which may be a server (such as a proxy server) which provide e.g. CSCF (Call Server Control Function), and/or is a home location server which contains a database storing information on the present locations of network element 3 and further network elements, reachability thereof, and the like." Staack at [0022]: "As shown in FIG. 1, the server 2 stores parameters for 5 several users (user equipments) to be served by server 2. These parameters define the users profiles, network capabilities, and status of the users and terminals. For user S, the server 2 stores the information "out-of-reach"; "store-and-forw'ard: notify"; "accepts: jpeg, gif", etc. This information may be updated by the server 2 or equipment 3 e.g. when re-entering the serving area of server 2, or when equipment 3 wants to change or supplement the types of acceptable messages. The "accepts" field defines the types of acceptable messages. The field "store-and-forward" can 15 be set by equipment 3, or by the operator or service provider of the network to "NO", "YES", "NOTIFY" (when the sending user is to be notified after successful delivery of the message to the user B)", "Forwarding Address or Service for forwarding messages", and the like. The operator or 20 service provider may

'117 Claims	Exemplary Disclosures
	provide different storing services for different subscribers, such as no storing possibility for normal subscribers, and storing possibility for premium subscribers." Staack at [0026]: "FIG. 2 shows the embodiment of FIG. 1 in a condition where the user equipment 3 (user B) can be reached again. When the user equipment 3 can be reached again, it will usually send a message signaling its present state or condition, e.g. its intention to receive access to the network. Such a message is shown in FIG. 2 as step 1.) and may consist in a request "register", "PDP context activation", or the like, depending on the type of network and the like. Such a request is addressed to server 2 which therefore recognizes the reachability of equipment 3. When detecting this situation, the server 2 sends, in step 2.) of FIG. 2, a message "notify: user B is on-line" to server 4. The server 4 checks its database with regard to any waiting message or messages stored for user B. When detecting such messages, the server 4 sends this message or messages directly to user equipment 3 as shown in step 3.), "MESSAGE USER B@sonera.com". The server 4 may also be adapted to send a confirmation to server 2 after successful transmission of the stored messages to user equipment 3. The server 2 then preferably sends, in step 4.), a message to user equipment 1 informing the latter on successful delivery of the message to user equipment 3. This message is shown in FIG. 2 as "NOTIFY A: contents has been 10 received"." *See also* Staack at Figs. 1, 2. Fan at 12: The matching conditions in the set Push message control policy may include the following: user identifier, maximum message length threshold, service control, content control, and control of the trusted PI. The following five Push message control strategies are introduced separately. Fan at 13:

'117 Claims	Exemplary Disclosures
	First, the user identifier The purpose of setting the matching condition is to determine, according to the user identifier, that the terminal to which the Push message is to be sent is the setting terminal of the Push message control policy when the Push sending agent performs the Push message control. Fan at 13-14: FIG. 9 is a flowchart of processing a terminal according to a Push message control policy. In this process, it is assumed that a service type of a Push message is a multimedia multimedia message, and the process includes: Step 901: After receiving the Push message sent by the PI, the Push sending proxy sends the Push message to the Push receiving proxy by using a NOTIFY or MESSAGE message in the SIP message. Step 902: After receiving the Push message, the Push receiving proxy returns a 200 OK message to the Push sending proxy to notify the Push sending proxy that the Push message has been successfully received. Step 903: The Push receiving agent checks the service type carried in the Accept-Contact header field in the SIP message carrying the Push message, and determines that the Push message is a multimedia multimedia message. If the terminal does not allow to receive the message of the multimedia multimedia message service in the Push message control policy, the Push receiving agent discards the Push message without sending it to the application module, and the user does not know that a message arrives; if the Push message controls the policy, the terminal If the multimedia MMS service is not allowed to be received, the Push receiving agent temporarily stores the Push message; if the Push message control policy, the terminal allows to receive the message of the multimedia MMS service, and then sends the message to the application module for corresponding service processing. White at [0066]: "The present invention also includes features enabling users to customize the delivery of messages using transmission rules so that, for example, messages are

Headwater Research LLC
Ex. 2011, IPR2024-00003
Page 424 of 442

'117 Claims	Exemplary Disclosures
	transmitted at the most convenient times. Users can create rules that specify when message transmission should occur, as well as customize the persistence of messages in the message stores. Transmission rules provide the ability to configure the messaging system to control what messages are sent and the timing of when to send them. The ability to control which messages can be sent during a transmission is important in an environment in which mobile devices are occasionally connected through different types of network connections (e.g. wireless and wireline), which have varying degrees of reliability and throughput (bandwidth). Transmission rules provide the ability to configure the messaging system so that a user can specify when messages are to be transmitted depending upon, for example, the type of network connectivity that is available from time to time. A user can, for instance, create a transmission which specifies sending high priority messages whenever connectivity is available, but reserves transmission of lower priority messages to "off hours" or when a better, more reliable connection is available." Ross at 6:35-48: "In the subscribers table 72, a field such as a "mkt seg id" is included for each subscriber to associate that subscriber with a given market segment ID. This segmentation of subscribers may be done using many different grouping methods. For example, subscribers may be associated by their physical locations, usage characteristics, or the network to which they are connected. This allows an SMSC administrator, for example, to optimally set the storage and delivery characteristics based upon the subscriber's profiles, network capacity, and SMSC capacity, switch capabilities, current system load, etc. The variables stored in the market segment table 70 may then be used to affect the storage and delivery characteristics for each subscriber based upon the market segment ID for that subscriber." *See also* Ross, 3:15-37, 3:52-59, 6:18-34, 8:35-54, 9:5-38.

'117 Claims	Exemplary Disclosures
17. The network system of claim 16, wherein the transmission trigger is the expiration of a periodic timer.	The references below disclose and/or render obvious this limitation. For example, see the following passages and/or figures, as well as related disclosures: Hämäläinen at [0034]: "The Scheduler 206 in FIG. 2 describes scheduled notifications or action requests that are based on a programmable timer or a calendar." Hämäläinen at [0052]: "In the Service Logic 176, also referred to as the InfoPush Service Handler 176 in FIG. 3, step 308 resolves recipients, delivery mode, and bearers for changed or new content. Step 310 provides input from the context state handler 174. Step 312 provides input from Service Profile Server 204. Step 314 provides input from the scheduler 206. Step 308 flows to the decision step 316 to determine the delivery mode. If the mode is an event, then the process flows to step 318 to store the content until a triggering event is received (the trigger can be a context state change or a timer event.) Step 320 then forwards the information push message to the message sender 178 when a triggering event is received. The triggering event may be immediate delivery." Hämäläinen at [0056]: "Back in the InfoPush Service Handler 176 of FIG. 3, if the decision step 316 determines that the delivery mode is a content update, then the process flows to step 330 to resolve the delivery time. Then step 330 flows to either step 332 or to step 334. Step 332 schedules the timer if content is specified by the service profile to be delivered later. When a scheduled timer event occurs, then the information push message is forwarded. Step 314′ provides input from scheduler 206. Instead, if step 330 flows to step 334, the information push message is forwarded if the content is specified by the service profile to be delivered immediately. Both step 332 and step

'117 Claims	Exemplary Disclosures
	334 flow to step 322 to put the information push message into the message queue 208." Shenfield at [0022]: "Architecture 100 includes a content provider 110. Content provider 110 is arranged to provide dynamic content to users that are subscribed with content provider 110. Examples can include, for example, a website selling books. A user may register with content provider 110 to obtain a list of newly released books within specified genres. Other examples could include news sites which might provide headlines to users on a periodic basis, traffic sites which might provide up-to-date traffic information to users during certain periods of the day, stock market sites which could provide updated stock quotes or currency exchange rates to users, among others." Shenfield at [0078]: "b. A second way that update notification block 562 can notify applications 512 of new content is to store the content in content storage 580 and to optionally notify applications 512 that content is waiting. Notification in this case is optional. Specifically, if an application 512 knows that information destined for it is stored within a specific memory block, one option for the application discovering that is has new data is to periodically poll the memory location to see whether there has been something written to it. Alternatively update notification block 562 can send a message to application 512 indicating that it has new data an possibly the location that the data is stored." *See also* Shenfield at [0176]-[0184]. Santamaria at [0005]: "Users of a mobile device (e.g., laptop, palmtop, portable media player, smartphone, multimedia mobile phone, mobile gaming systems such as a "Gameboy", etc.) may

Headwater Research LLC
Ex. 2011, IPR2024-00003
Page 427 of 442

'117 Claims	Exemplary Disclosures
	subscribe to one or more notification services. For example, users may subscribe to one or more "push" email services such as .Mac, Microsoft Exchange ActiveSync, push-IMAP, Yahoo! Push, etc. In the case of a push email service, for example, the email server may automatically transmit email messages and/or calendar updates to the email client on the mobile device for the user without the user requesting the item. In other words, the user does not request (poll) the email server for email messages or other items of interest. Thus, a push notification service is a persistent notification service. In addition, other types of services may use a similar push architecture (e.g., update/upgrade services, news services, weblog services, podcast services, etc.). In order to maintain a push notification service, the mobile device periodically refreshes the connection to the push notification service (e.g., by transmitting a ping message to the push server)." Santamaria at [0023]: "A method and apparatus for managing notification service connections is described. In one embodiment of the invention, a mobile device (e.g., laptop, palmtop, portable media player, smartphone, multimedia mobile phone, mobile gaming systems such as a "Gameboy", etc.) includes one or more notification service clients (e.g., push notification service clients (e.g., push email clients such as .Mac, Microsoft Exchange, push-IMAP, Yahoo! Push, etc., update/upgrade services, news services, weblog services, podcast services, etc.) and/or pull notification service clients (e.g., pull email clients such as POP3, IMAP, etc.)). For each notification service client, the mobile device independently schedules a notification service connection message transmission interval and sets a notification service connection message transmission timer. In addition, a notification service connection message transmission window is associated with each notification service connection message transmission interval and is based on the value of the notification service connection message transmission interval. Upon a notification service connection message transmission timer expiring for a particular notification service client, the mobile device transmits a notification

'117 Claims	Exemplary Disclosures
	service connection message for that notification service, and also transmits a notification service connection message for each other notification service whose notification service connection message transmission window overlaps that expiring transmission timer. The mobile device resets the notification service connection message transmission timer for each notification service upon transmitting a notification service connection message." Santamaria at [0044]: "According to one embodiment of the invention, after a timer for a notification service connection has elapsed, the service connection manager 320 instructs the notification service client 310 to transmit a notification service connection message, and also notifies the notification service client 310 of any re-try events, as indicated by the number 364 (e.g., a re-try event may include re-establishing a dropped notification service connection). The notification service client 310, with use of the network stack 330, transmits notification service connection messages, as indicated by numeral 374. In addition, the notification service client 310 may instruct the service connection manager 320 to calibrate the notification service connection message transmission interval, as indicated by number 362. Calibrating the notification service connection message transmission interval is discussed in greater detail with reference to FIG. 6." Santamaria at [0054]: "Referring to FIG. 5, the notification service connection message transmission timer associated with the push notification service 120 elapses at time 15. Thus, at time 15, the mobile device 160 is wakened (represented by the dashed line wake 560) and a notification service connection message is transmitted to the push notification service 120 in order to refresh the notification service connection (represented by the beat 570). It should be understood that the mobile device 160 does not transmit a notification service connection message to the push notification service 110 or the

'117 Claims	Exemplary Disclosures
	pull notification service 130 during the wake time 560. For example, the push notification service 110 has a beat interval of 30 minutes. Thus to refresh the connection to the push notification service 110, a beat may be transmitted up to 30 minutes (i.e., the connection needs only be refreshed once every 30 minutes). If the mobile device 110 transmitted a beat to the push notification service 110 each and every time a beat was transmitted to the push notification service 120, which has a beat interval of 15 minutes, effectively twice as many beat messages as necessary would be transmitted. Thus, according to one embodiment of the invention, the mobile device transmits notification service connection messages to a particular notification service only if it is awake during a notification service connection message transmission window for that particular notification service. Since the wake 560 does not overlap a notification service connection message transmission window for the push notification service 110 or the pull notification service 130, the mobile device does not transmit notification service connection messages for those notification services in order to conserve bandwidth." Staack at [0024]: "The network additionally contains a network element such as a server 4 used for storing any SIP message not promptly deliverable to the intended recipient. This server 4 is, in the present embodiment, not only used as a storing server but also as a forwarding server for actively forwarding any stored message to the recipient, e.g. periodically or when receiving information that the recipient is reachable again." White at [0031]: "Policy: In this document, the term "policy" refers to a predefined set of rules for transmitting/receiving messages (e.g., on a mobile device or a server). Standard transmission policies which are provided in the currently preferred embodiment of the present invention include "scheduled", "automatic", and "ondemand" policies. With a "scheduled" policy, message transfers are done at regular, specified time

Headwater Research LLC
Ex. 2011, IPR2024-00003
Page 430 of 442

'117 Claims	Exemplary Disclosures
	intervals. With an "automatic" policy, the QAnywhere messaging system attempts to perform message transmissions exactly when needed. With an "ondemand" policy, the messaging system leaves it to the messaging application to control when message transfers are performed. Users may also provide a custom policy for message delivery by creating transmission rules as described herein." White at [0066]: "The present invention also includes features enabling users to customize the delivery of messages using transmission rules so that, for example, messages are transmitted at the most convenient times. Users can create rules that specify when message transmission should occur, as well as customize the persistence of messages in the message stores. Transmission rules provide the ability to configure the messaging system to control what messages are sent and the timing of when to send them. The ability to control which messages can be sent during a transmission is important in an environment in which mobile devices are occasionally connected through different types of network connections (e.g. wireless and wireline), which have varying degrees of reliability and throughput (bandwidth). Transmission rules provide the ability to configure the messaging system so that a user can specify when messages are to be transmitted depending upon, for example, the type of network connectivity that is available from time to time. A user can, for instance, create a transmission which specifies sending high priority messages whenever connectivity is available, but reserves transmission of lower priority messages to "off hours" or when a better, more reliable connection is available." Holbrook at [0106]: "Messaging node 20 establishes occasional communications with central server 12, and thereby participates in the full messaging system 10. Via first communications link 70, and probably other intermediate network elements known to persons skilled in the art, node computer 22 establishes an Internet connection to central server 12

'117 Claims	Exemplary Disclosures
	and exchanges various data, such as message traffic and administrative information, as discussed below. Such exchanges may occur on a regular schedule, at regular intervals, at a message volume threshold, or by user request. For example, it may be desirable to initiate a communications session only when message transfer time is significantly longer than the expected time for establishing communications (such as handshaking and similar transactions), to increase the efficiency of connection time. Most preferably, any of several triggers may be responsible for initiating a communications session, such as a large volume of outgoing messages, a time interval sufficiently long for a large volume of incoming messages to be expected, or for a further delay in messaging to have an adverse impact upon the performance of messaging system 10."
18. The network system of claim 16, wherein the transmission trigger is the receipt of a transmission from the device messaging agent of the particular device.	The references below disclose and/or render obvious this limitation. For example, see the following passages and/or figures, as well as related disclosures: Mulligan at 10:54-59: "Another example of a user preference includes network preferences. This type of preference designation relates to the type of network the user is connected to at a given time. For example, the user can designate that high bandwidth messages are sent only while the user is connected to 3G networks, or that messages are not to be sent while roaming, etc." Mulligan at 10:60-11:4: "Still another example of a user preference is a presence preference. These are preferences relating to the users activity at a particular time. For example, the user may designate that audio messages are not to be sent while in a meeting, when sleeping, or at any time in which an audio message would be disruptive or otherwise undesirable. The presence preferences may also be used to forward messages to another terminal, person, etc. when the particular terminal is offline.

'117 Claims	Exemplary Disclosures
	Further, unsolicited messages from certain message origins may be ignored, messages with large attachments may be deferred, etc. The number and type of user preferences that can be implemented in this fashion is virtually endless." Mulligan at 11:31-49: "In one embodiment, the push router obtains 602 the presence information described above from the presence agent. Similarly, the push router may obtain the user preferences from the user preference agent, and the bearer information from the push adaptation layer, as illustrated in blocks 604 and 606 respectively. When the desired information has been collected, the push router processes 608 this information to determine the appropriate bearer in which to direct the push message. When the push router has determined which bearer will be the recipient of the push message, it may be decided to buffer 610 the message. For example, if the collected information indicates that it is not an appropriate time for the user to receive the push message on the mobile terminal, the push message may be buffered 610. At a suitable time, or immediately if no buffering is required, the push message is forwarded 612 to the appropriate bearer. In one embodiment of the invention, the push router may deliver reports back to the push message initiator after the push message has been delivered to the appropriate bearer, as illustrated at block 614." Mulligan at 12:51-63: "If the push message is not to be sent immediately, the message may be buffered until a more suitable time arises to send the message. A decision whether to buffer the message is made depending on the information collected by the push router, as shown at decision block 720. For example, if the user is in a meeting or otherwise unavailable, or is in a low bandwidth area, the push router will know this from the collected information, and a decision can be made to postpone delivery of the push message. In such a case, the message is buffered 722 or otherwise stored until a more suitable delivery time arises. In the context of the above example, this more

'117 Claims	Exemplary Disclosures
	suitable time may be when the user is out of the meeting, or is in a higher bandwidth area." Mulligan at 12:64-13:6: "At a suitable time, or immediately if no buffering is required, the push router forwards the push message to the appropriate bearer as shown at block 724. In one embodiment of the invention, the push router provides delivery reports back to the push message initiator. More particularly, when the push router has completed its task of delivering the push message, the push router may compose a return message, and pass this return message to the Web Services endpoint for formatting and forwarding to the push initiator, as shown at block 726." Hämäläinen at [0026]-[0034]: "The following states are examples of availability/reachability information handled by the Context State Handler 174: LOST—indicates that the device 100 is not in the coverage area of any access point 140. FOUND—indicates that the device 100 is found. ASSIGNED—indicates that the device 100 is in the process of joining the coverage area of an access point 140. JOINED—indicates that the device 100 has joined an access point 140 and is reachable. ROAMING—indicates that the device 100 is temporarily out of coverage by any access point. ERROR—indicates that the connection to the mobile device 100 has failed after several attempts. The connection to the device 100 will retried after a predetermined timeout has occurred.

Headwater Research LLC
Ex. 2011, IPR2024-00003
Page 434 of 442

'117 Claims	Exemplary Disclosures
	"The Service Logic 176, also referred to as the InfoPush Service Handler 176 in FIG. 2, listens to notifications from Context State Handler 174, the Scheduler 206, and the Content Fetcher 172. The InfoPush Service Handler 176 sends information push messages based on notifications and Service Profile Server 204. "The Scheduler 206 in FIG. 2 describes scheduled notifications or action requests that are based on a programmable timer or a calendar." Hämäläinen at [0052]: "In the Service Logic 176, also referred to as the InfoPush Service Handler 176 in FIG. 3, step 308 resolves recipients, delivery mode, and bearers for changed or new content. Step 310 provides input from the context state handler 174. Step 312 provides input from Service Profile Server 204. Step 314 provides input from the scheduler 206. Step 308 flows to the decision step 316 to determine the delivery mode. If the mode is an event, then the process flows to step 318 to store the content until a triggering event is received (the trigger can be a context state change or a timer event.) Step 320 then forwards the information push message to the message sender 178 when a triggering event is received. The triggering event may be immediate delivery." Santamaria at [0005]: "Users of a mobile device (e.g., laptop, palmtop, portable media player, smartphone, multimedia mobile phone, mobile gaming systems such as a "Gameboy", etc.) may subscribe to one or more notification services. For example, users may subscribe to one or more "push" email services such as .Mac, Microsoft Exchange ActiveSync, push-IMAP, Yahoo! Push, etc. In the case of a push email service, for example, the email server may automatically transmit email messages and/or calendar updates to the email client on the mobile device for the user without the user requesting the item. In other words, the user does not request (poll) the email server for email messages or other items of interest. Thus, a push notification service is a persistent notification service. In addition, other types of services may use a similar push architecture (e.g., update/upgrade services, news

'117 Claims	Exemplary Disclosures
	services, weblog services, podcast services, etc.). In order to maintain a push notification service, the mobile device periodically refreshes the connection to the push notification service (e.g., by transmitting a ping message to the push server)." Okonnen at [0017]: "In an embodiment according to the present invention, the electronic device may be adapted to display a list of available update agents to an end-user and solicit selection of an update agent to be used to update at least one of software and firmware." Okonnen at [0055]-[0056]: "In an embodiment according to the present invention, mobile handset 107 may display a list of available update agents to an end-user and solicit selection of an update agent to be used to update at least one of software and firmware. "In an embodiment according to the present invention, the mobile handset 107 may invoke an update agent based upon an update currently being processed, for example, based upon the update contents and/or characteristics, provided that the update agent is provisioned in the mobile handset 107. The update agent may also be capable of rebooting the mobile handset 107 after update completion. In an embodiment according to the present invention, a list of available updates may initially be displayed to the end-user to prompt a selection of an update by the end-user." Okonnen at [0058]: "On power up, the mobile handset 107 may execute the update application (UA) loader 127. The UA loader 127 may invoke a boot initialization code before determining to update the mobile handset 107 and determining the list of available update agents already provisioned (such as, during an initial provisioning operation initiated by the electronic device network 105) in the mobile handset 107. A list of provisioned update agents containing references to update agent A 113 and the update agent B 115 may be displayed to the end-user. When the end-user selects

'117 Claims	Exemplary Disclosures
	one of the provisioned update agents displayed, the UA loader 127 may invoke the selected provisioned update agent and transfer control to the selected update agent." Staack at [0009]: "In accordance with one aspect of the invention, the instant messaging service is enhanced by providing a storing capability for messages. When the intended receiver of the message is presently unable to receive the message because he is e.g. not on-line, busy and/or not reachable by the network, e.g. by the proxy server of the receiving user, because of any other reason, the message may be stored. This saving of the message enables its later delivery to the receiving user when this user is able to receive the message, e.g. after re-attachment to the network. No connection for bi-directional communication needs to be established." Staack at [0021]: "FIG. 1 shows a first embodiment of the invention and illustrates a case where a message is to be sent from a first network element 1 (user A) to a second network element 3 (user B). The network elements 1, 3 are, in the 25 present embodiment, client or user equipments such as terminals. In the present example, the network element 1 (user A) is an equipment trying to send a message (e.g. "MESSAGE user b@sonera.com" addressed to user b@sonera.com) to the receiving element 3 (user B) which is presently out 30 of reach, e.g. switched-off, busy, or located in a nonsupported area, or the like. The connection request of network element 1 is handled by a network element 2 which may be a server (such as a proxy server) which provide e.g. CSCF (Call Server Control Function), and/or is a home location server which contains a database storing information on the present locations of network element 3 and further network elements, reachability thereof, and the like." Staack at [0022]: "As shown in FIG. 1, the server 2 stores parameters for 5 several users (user equipments) to be served by server 2. These parameters define the users profiles, network capabilities, and status of the users and terminals. For user S, the server 2

'117 Claims	Exemplary Disclosures
	stores the information "out-of-reach"; "store-and-forw'ard: notify"; "accepts: jpeg, gif", etc. This information may be updated by the server 2 or equipment 3 e.g. when re-entering the serving area of server 2, or when equipment 3 wants to change or supplement the types of acceptable messages. The "accepts" field defines the types of acceptable messages. The field "store-and-forward" can 15 be set by equipment 3, or by the operator or service provider of the network to "NO", "YES", "NOTIFY" (when the sending user is to be notified after successful delivery of the message to the user B)", "Forwarding Address or Service for forwarding messages", and the like. The operator or 20 service provider may provide different storing services for different subscribers, such as no storing possibility for normal subscribers, and storing possibility for premium subscribers." Staack at [0024]: "The network additionally contains a network element such as a server 4 used for storing any SIP message not promptly deliverable to the intended recipient. This server 4 is, in the present embodiment, not only used as a storing server but also as a forwarding server for actively forwarding any stored message to the recipient, e.g. periodically or when receiving information that the recipient is reachable again." Staack at [0026]: "FIG. 2 shows the embodiment of FIG. 1 in a condition where the user equipment 3 (user B) can be reached again. When the user equipment 3 can be reached again, it will usually send a message signaling its present state or condition, e.g. its intention to receive access to the network. Such a message is shown in FIG. 2 as step 1.) and may consist in a request "register", "PDP context activation", or the like, depending on the type of network and the like. Such a request is addressed to server 2 which therefore recognizes the reachability of equipment 3. When detecting this situation, the server 2 sends, in step 2.) of FIG. 2, a message "notify: user B is on-line" to server 4. The server 4 checks its database with regard to any waiting message or messages stored for user B. When detecting such messages, the server 4 sends this message or messages directly to user equipment 3 as shown in step 3.),

Headwater Research LLC
Ex. 2011, IPR2024-00003
Page 438 of 442

'117 Claims	Exemplary Disclosures
	"MESSAGE USER B@sonera.com". The server 4 may also be adapted to send a confirmation to server 2 after successful transmission of the stored messages to user equipment 3. The server 2 then preferably sends, in step 4.), a message to user equipment 1 informing the latter on successful delivery of the message to user equipment 3. This message is shown in FIG. 2 as "NOTIFY A: contents has been 10 received"." *See also* Staack at Figs. 1, 2. Holbrook at [0106]: "Messaging node 20 establishes occasional communications with central server 12, and thereby participates in the full messaging system 10. Via first communications link 70, and probably other intermediate network elements known to persons skilled in the art, node computer 22 establishes an Internet connection to central server 12 and exchanges various data, such as message traffic and administrative information, as discussed below. Such exchanges may occur on a regular schedule, at regular intervals, at a message volume threshold, or by user request. For example, it may be desirable to initiate a communications session only when message transfer time is significantly longer than the expected time for establishing communications (such as handshaking and similar transactions), to increase the efficiency of connection time. Most preferably, any of several triggers may be responsible for initiating a communications session, such as a large volume of outgoing messages, a time interval sufficiently long for a large volume of incoming messages to be expected, or for a further delay in messaging to have an adverse impact upon the performance of messaging system 10." Holbrook at [0173]: "At the start of data exchange 90, at step 91, CPU 28 detects the placement of a portable messaging unit 40 in one of docking ports 26. This detection may comprise noticing an infrared signal from portable messaging unit 40, or a physical contact with a microswitch. In step 92, CPU 28 establishes the identity of user accounts associated with portable messaging unit 40, and performs account

Headwater Research LLC
Ex. 2011, IPR2024-00003
Page 439 of 442

'117 Claims	Exemplary Disclosures
	verification functions. In step 93, portable messaging unit 40 transfers outgoing messages to messaging node 20. This is done at an early stage to free non-volatile memory for the storage of new incoming messages subsequently received. In step 94, portable messaging unit 40 reports the amount of free non-volatile memory available for the storage of new incoming messages, and in step 95 messaging node 20 applies a prioritization algorithm, selecting messages for transfer if the memory is insufficient for the storage of all new messages. Then, in step 96, messaging node 20 transfers these selected messages to portable messaging unit 40. Finally, in step 97, status light 32 indicates the transaction completion of data exchange 90, at which time the user may remove portable messaging unit 40 from docking ports 26." Bobo at Abstract: "A Message Storage and Deliver System (MSDS) is connected to a plurality of DID phone lines and receives facsimile messages, voice messages, and data messages. The MSDS assigns a separate telephone number for each user of the system and can simultaneously receive more than one message for a single user. The messages are stored in memory and are also converted into appropriate hyper-text mark-up language (HTML) files. The MSDS is connected to the Internet and notifies the users with an E-mail message each time a message is received. The MSDS can also page the user so that the user receives almost instantaneous notice of the message. The users can then connect to the MSDS through the Internet and have the messages downloaded to their computers or the users can simply preview the messages stored in the MSDS. The users of the MSDS therefore have the advantage of being able to receive their messages at any time and at any location at a reasonable cost. The MSDS offers a number of options on how the messages may be sent to the user, such as several preview options available with facsimile messages. The user can also telephone the MSDS to listen to messages or to alter the service provided by the MSDS." Bobo at 7:13-25:

'117 Claims	Exemplary Disclosures
	"After the MSDS 10 receives a message for one of its users, the user can then communicate with the MSDS 10 at any time and at any location by connecting to the Internet World Wide Web 30 and retrieving the message stored within the MSDS 10. With reference to FIG. 3, at step 60 the user first connects to the Internet 30, such as through a personal computer 32 which may be connected to the Internet 30 in any suitable manner, such as through its own portal or node or through some intermediate access provider. The personal computer 32 is not limited to a single computer but may instead comprise a network of computers, such as a local area network within an office." Bobo at 7:39-50: "After the user gains access to the mailbox at step 72, the user can request information stored within the MSDS 10. The MSDS 10 receives the request at step 76 and, at step 78, determines whether the information exists. As is common practice, the MSDS 10 also determines the validity of the request at step 78. The request from the user will include the mailbox number for the user, the message identifier, display preferences, and, if the message is a facsimile message, a page identifier. If for any reason the request is invalid, such as when a hacker is attempting to gain access to privileged information, the request for the information will be terminated." Bobo at cl. 31: "31. A network message storage and delivery system, comprising: a central processor for receiving an incoming call, for detecting an address signal on said incoming call, for detecting a message on said incoming call, and for placing said message in a storage area, said address signal being associated with a user of said network message storage and delivery system; a network server for receiving said message from said storage area, for converting said message into a mixed media page layout language, and for placing said message in said storage area;

'117 Claims	Exemplary Disclosures
	wherein when said network server receives a request from said user over said network, said network server transmits at least a portion of said message over said network to said user and wherein said network comprises the Internet and said network server comprises an Internet server."

Headwater Research LLC
Ex. 2011, IPR2024-00003
Page 442 of 442