



US 20060062384A1

(19) **United States**

(12) **Patent Application Publication**
Dondeti

(10) **Pub. No.: US 2006/0062384 A1**

(43) **Pub. Date: Mar. 23, 2006**

(54) **METHOD AND APPARATUS FOR
GENERATING LARGE NUMBERS OF
ENCRYPTION KEYS**

(52) **U.S. Cl. 380/44; 380/28; 380/283**

(75) **Inventor: Lakshminath R. Dondeti, Chelmsford,
MA (US)**

Correspondence Address:
JOHN C. GORECKI, ESQ.
P.O BOX 553
CARLISLE, MA 01741 (US)

(73) **Assignee: Nortel Networks Limited, St. Laurent
(CA)**

(21) **Appl. No.: 10/946,322**

(22) **Filed: Sep. 21, 2004**

Publication Classification

(51) **Int. Cl.**
H04L 9/00 (2006.01)

(57) **ABSTRACT**

Entropy obtained from a series of key generation exchanges may be combined with entropy from a strong entropy source to allow the strong entropy to be stretched to generate a larger number of keys for use on a communication network, without requiring additional information from the group members and without requiring the entropy source to be increased in size or in number. In one embodiment, nonces exchanged during an initial key exchange are used to generate additional key material that is then fed, together with a fresh random secret, to another pseudo-random function to generate an additional key stream. The fresh random secret may be generated at the GCKS from a physical entropy source or other entropy source, and may be changed at will by the GCKS to further increase the strength of the keys. The methods are particularly useful for group key management where a large number of keys are required to be generated in a short time frame.

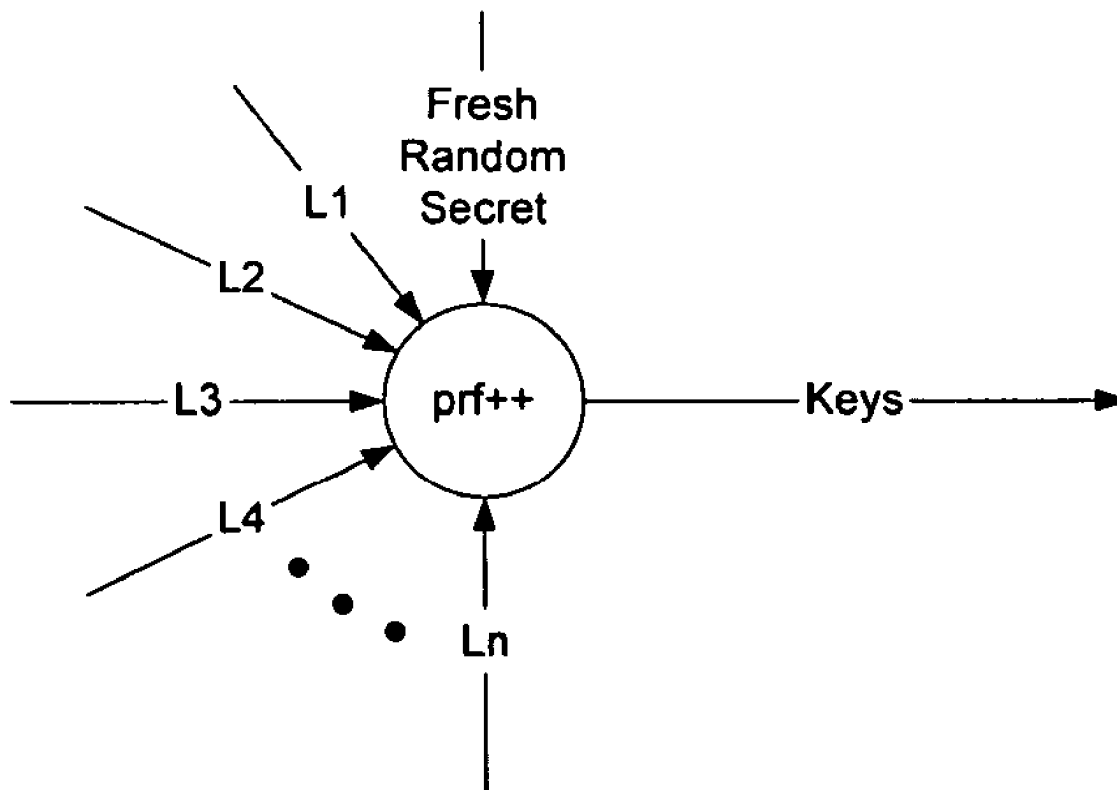


Figure 1

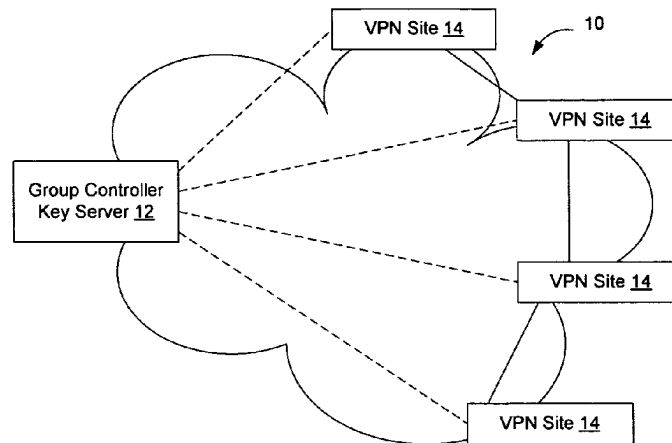


Figure 2

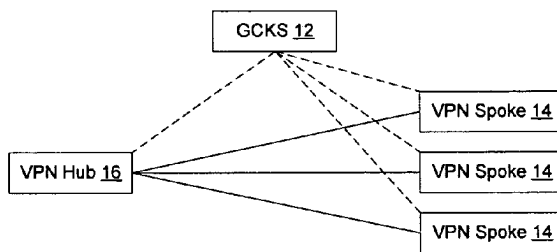


Figure 3

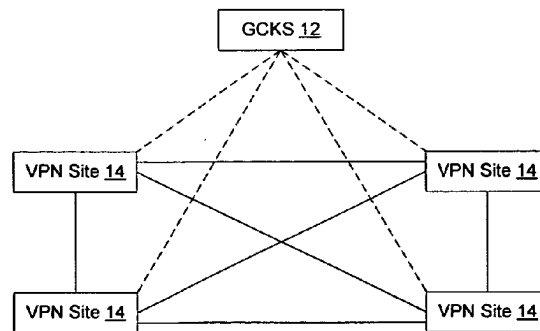


Figure 4A

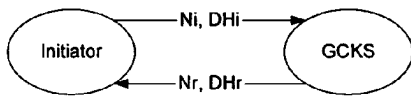


Figure 4B

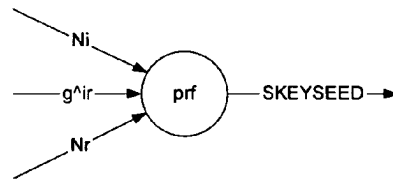


Figure 4C

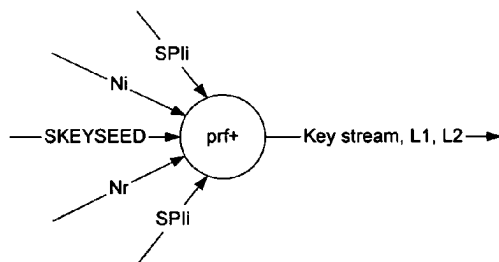


Figure 4D

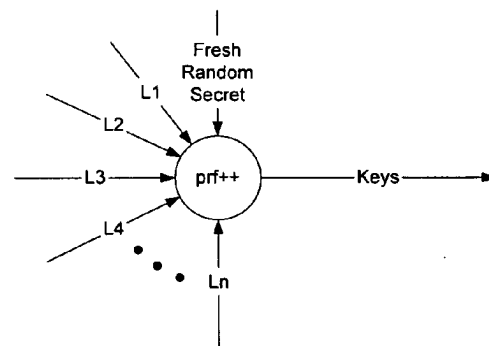
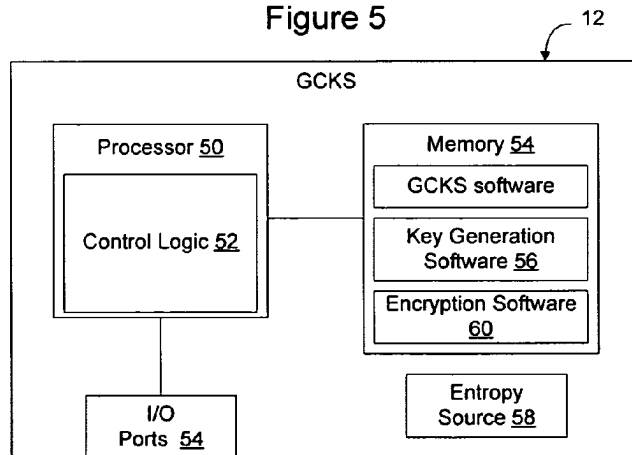


Figure 5



METHOD AND APPARATUS FOR GENERATING LARGE NUMBERS OF ENCRYPTION KEYS

BACKGROUND

[0001] 1. Field

[0002] This application relates to communication networks and, more particularly, to a method and apparatus for generating large numbers of encryption keys for use on a communication network.

[0003] 2. Description of the Related Art

[0004] Data communication networks may include various computers, servers, nodes, routers, switches, hubs, proxies, and other devices coupled to and configured to pass data to one another. These devices will be referred to herein as "network elements," and may provide a variety of network resources and services on the network. Conventionally, data has been communicated through data communication networks by passing protocol data units (such as packets, cells, frames, or segments) between the network elements by utilizing one or more communication links extending between the network elements. A particular protocol data unit may be handled by multiple network elements and cross multiple communication links as it travels between its source and its destination over the network.

[0005] To allow communications to take place in a secure manner on the communication network, it has become common to encrypt the communications, generally using encryption keys. Encryption keys are random strings of numbers that may have any number of characters. The longer the character string, and the more unpredictable the characters in the string, the stronger the key is said to be. As keys become stronger, the communications they encrypt become safer because it is harder to guess or infer the key that was used to encrypt the communication.

[0006] Since the strength of an encryption key depends in large part on the unpredictability of the characters in the key, generation of strong encryption keys requires the key generation program or network element to have access to a source of random numbers, referred to herein as an entropy source. If the entropy source is predictable, the values created by the entropy source will be easier to guess, and the strength of the keys created by the entropy source will be reduced accordingly.

[0007] Physical entropy sources may be used to generate random numbers for use in creating encryption keys, and may be considered somewhat superior to software entropy sources due to the unpredictability of the random information created. A few physical entropy sources include thermal noise, radioactive decay, oscillators, and disc drives, although other physical entropy sources exist as well. While physical entropy sources are able to generate truly random information, and hence may be used to create very strong encryption keys, they generally tend to be relatively slow. For example, the rotational speed of the disc in a computer disc drive may be used to generate random information. However, since the speed may not vary considerably in a short period of time, a disc drive-based physical entropy source may only generate sufficient random information to generate up to a few keys per second. As the keys become larger, the rate at which they may be generated is concomitantly reduced.

[0008] As networks have become larger, and the number of entities encrypting traffic on the networks has risen, the need for a source of strong encryption keys has increased. One driving force behind this need for encryption keys is the proliferation of Virtual Private Networks (VPNs). VPNs allow private traffic to be shared between VPN sites over a public network by securing the traffic in such a manner that other users of the communication network are not able to see the private traffic. There are several ways to do this, many of which involve encrypting the traffic before transmitting the traffic onto the communication network.

[0009] As VPNs increase in size and number, generation of keys by the group members has been replaced by centralized key management systems, referred to herein as Group Control Key Servers (GCKSs). A GCKS may provide many services on the network as well as key generation and management. While a large GCKS on the network may be required to generate thousands or tens of thousands of encryption keys for use by members of one or more Virtual Private Networks. Unfortunately, physical entropy sources are generally unable to provide a sufficient number of random values to generate strong keys for use in this environment, and software entropy sources may be too predictable to provide sufficiently strong encryption keys.

SUMMARY OF THE DISCLOSURE

[0010] According to an embodiment of the invention, a Group Controller Key Server (GCKS) generates additional random values from an initial exchange with group members that may be used to stretch a random value generated by the GCKS into multiple keys. In this manner, the GCKS is able to use a combination of physical and software entropy sources to allow entropy from a physical or other entropy source to be stretched to generate a large number of random keys for use on the network, without requiring additional information from the group members and without requiring the entropy source to be increased in size or in number.

[0011] According to an embodiment of the invention, group members exchange nonces and Diffie-Hellman values with GCKS according to an initial key exchange protocol such as Internet Key Exchange version two (IKE-v2). These values are fed into a pseudo-random function to create a seed key that will be used in a second pseudo-random function to create a string of keys for use by the group member for encrypting traffic on the network. The second pseudo-random function also creates additional random bits, in addition to the random bits used to create the key string, which is input to another third pseudo-random function along with a fresh random secret to generate additional keys for use on the network. The fresh random secret may be generated at the GCKS from a physical entropy source or other entropy source, and may be changed at will by the GCKS to further increase the strength of the keys. This allows a single strong entropy source, used to create the fresh random secret, to be stretched into a plurality of keys for use by group members by allowing it to be combined with other entropy information supplied by members of the group.

BRIEF DESCRIPTION OF THE DRAWINGS

[0012] Aspects of the present invention are pointed out with particularity in the claims. The following drawings

disclose one or more embodiments for purposes of illustration only and are not intended to limit the scope of the invention. In the following drawings, like references indicate similar elements. For purposes of clarity, not every element may be labeled in every figure. In the figures:

[0013] **FIG. 1** is a functional block diagram of an example of a communication network including a Group Controller Key Server (GCKS) according to an embodiment of the invention;

[0014] **FIG. 2** is a functional block diagram of hub and spoke VPN architecture configured to use a GCKS for distribution of keys according to an embodiment of the invention;

[0015] **FIG. 3** is a functional block diagram of mesh VPN architecture configured to use a GCKS for distribution of keys according to an embodiment of the invention;

[0016] **FIGS. 4A-4D** illustrate a process of generating keys according to an embodiment of the invention; and

[0017] **FIG. 5** is a functional block diagram of a GCKS according to an embodiment of the invention.

DETAILED DESCRIPTION

[0018] The following detailed description sets forth numerous specific details to provide a thorough understanding of the invention. However, those skilled in the art will appreciate that the invention may be practiced without these specific details. In other instances, well-known methods, procedures, components, protocols, algorithms, and circuits have not been described in detail so as not to obscure the invention.

[0019] **FIG. 1** illustrates an example communication network architecture **10** according to an embodiment of the invention in which a Group Controller Key Server (GCKS) **12** is configured to generate and distribute encryption keys for use by VPN sites **14** to allow the VPN sites to communicate securely over the network.

[0020] The GCKS may perform numerous functions on the network and the following description will focus on only a subset of those features. Specifically, the following description will focus on the GCKS' role in creating and distributing keys for use by VPN sites to allow the VPN sites to communicate securely over the network. The GCKS may also perform other functions for the VPN or for the network, such as providing access control on the VPN, defining VPN membership, overseeing route exchanges on the network, and otherwise configuring the topography of the VPN network. Thus, a GCKS may perform multiple functions on the network in addition to handling the key exchange aspects discussed herein and the invention is not limited to a network element or process that solely handles key generation and exchange.

[0021] As shown in **FIG. 1**, the GCKS **12** communicates with each of the VPN sites **14** requiring encryption services, and controls distribution of keys to the sites to control which sites can communicate with each other. By controlling which sites can communicate, the GCKS controls the topography of the VPN network. The keys may be used to secure communications between the GCKS and the network elements, or may be used to secure the data path between the various network elements. For example, a group of VPN

sites may use a common key while other VPN sites on the same VPN will use a different key. Since sites without access to a key can't decipher the communications, only those sites having access to a key may participate on that branch of the particular VPN. The architecture illustrated in **FIG. 1** has been simplified to highlight the interaction between the GCKS and the VPN sites. An actual implementation of a GCKS system may involve multiple GCKSs in a flat or tiered relationship and configured to interface thousands or more VPN sites on the communication network. Thus, the invention is not limited to the architecture illustrated in **FIG. 1** but rather extends to numerous network architectures.

[0022] **FIGS. 2 and 3** illustrate, in a simplified manner, several common VPN network topographies that may be created through the appropriate provisioning of keys to VPN members, and through other conventional mechanisms such as by controlling the dissemination of routing information between the VPN members. **FIG. 2** illustrates a hub-and-spoke Virtual Private Network (VPN) architecture in which VPN sites **16** are configured to be able to communicate with a VPN hub **18**, but not with each other. **FIG. 3** illustrates a meshed VPN architecture in which specific VPN sites **16** may be allowed to communicate with one or more other VPN sites in the meshed architecture. In either instance, secure keys must be generated and distributed to the various sites and hubs to allow the communications to take place. The GCKS **12**, according to an embodiment of the invention, is configured to generate large numbers of keys for use in these and other types of secure network architectures.

[0023] Keys may be used in many different ways on the network. For example, a given key may be used to communicate with members of a group to secure communications between the members and the GCKS. This key may be used, for example, to allow the GCKS to pass one or more traffic encryption keys to the group member so that that group member can communicate with a subset of the other group members. Keys that are used to communicate between group members and the GCKS are commonly referred to as Key-Exchange Keys (KEKs) and keys that are used to communicate between group members to encrypt traffic are generally referred to as Traffic Encryption Keys (TEKs). As used herein, the term "key" will be used in a generic sense to refer to KEKs, TEKs, and other forms of keys intended for other uses. The manner in which keys are distributed and used by the nodes is well documented and thus will not be explained in any greater detail herein.

[0024] According to an embodiment of the invention, a Group Controller Key Server (GCKS) generates additional random values from an initial exchange with group members that may be used to stretch a random value generated by the GCKS into multiple keys. In this manner, the GCKS is able to use a combination of physical and software entropy sources to allow entropy from a physical or other entropy source to be stretched to generate a large number of random keys for use on the network, without requiring additional information from the group members and without requiring the entropy source to be increased in size or in number.

[0025] According to an embodiment of the invention, group members exchange nonces and Diffie-Hellman values with GCKS according to an initial key exchange protocol such as Internet Key Exchange version two (IKE-v2). These values are fed into a pseudo-random function to create a seed

key that will be used in a second pseudo-random function to create a string of keys for use by the group member for encrypting traffic on the network. The second pseudo-random function also creates additional random bits, in addition to the random bits used to create the key string, which is input to another third pseudo-random function along with a fresh random secret to generate additional keys for use on the network. The fresh random secret may be generated at the GCKS from a physical entropy source or other entropy source, and may be changed at will by the GCKS to further increase the strength of the keys. This allows a single strong entropy source, used to create the fresh random secret, to be stretched into a plurality of keys for use by group members by allowing it to be combined with other entropy information supplied by members of the group.

[0026] Initially, the GCKS generate an initial key to be used as a seed key. This may be done using a physical or other source of entropy in a conventional manner. Alternatively, the GCKS may generate the initial key by starting with a fresh random secret generated from a resident or interfaced entropy source and collecting entropy from group members seeking keys and delaying providing keys to the group members until a sufficient amount of entropy has been collected. The seed key is thus formed from the native entropy optionally combined with other entropy provided by group members using a pseudo-random function.

[0027] If a new key is then required by a group member, the group member sends a request for a key to the GCKS and includes in the request a nonce generated by the group member. The nonce may be used directly for communications between the group member and the GCKS, and will be used to generate additional keys for use by the group member or other group members. Optionally, the GCKS may screen nonces generated by group members to discard nonces that are of weak value or do not contain sufficient entropy. For example, a group member that returns as a nonce the name of the group member may be deemed to be of insufficient randomness to be useable to generate a subsequent key.

[0028] Once a new key has been generated, the key is then returned to the group member. A given member only knows the nonce that it generated and sent to the server, and the one that it receives from the server. These nonces are used to protect the session and are used as inputs to a prf that may be used to generate additional keys. However, since the nonce is used with other secret information that the GCKS has or collects, no key for use with other group members is generated by nonces from a single session, so that the keys are not predictable given known inputs.

[0029] Occasionally, it may be desirable to update the initial source of entropy that was used to generate the seed key. This may be done by the GCKS by changing the initial fresh random secret that is used to generate subsequent keys. For example, the GCKS may obtain sufficient entropy from a physical entropy source to generate a new fresh random secret every 5 seconds. The GCKS may use this new fresh random secret to generate subsequent keys, while maintaining the extended source of entropy collected from generating keys for other group members. Other embodiments may be possible as well and the invention is not limited to this particular embodiment.

[0030] To help explain how the invention may be practiced in a practical context, an example will be provided in which an embodiment of the invention is used to modify a conventional key exchange protocol, referred to herein as Internet Key Exchange version two, (IKEv2), to allow an increased number of keys to be generated. Since the following example illustrates a modification to the IKEv2 protocol, some of the terminology that will be used to describe this example will be similar to the terminology used to describe that standard. The invention is not limited to use of the concepts within the IKEv2 protocol, however, as the principles of the invention may be used with many other types of key exchange protocols, and the manner in which the keys are generated may be used in many different contexts. Thus, for example, similar modifications may be made to key exchange protocols such as:

[0031] Internet Security and Key Management Protocol (ISAKMP), an Internet standard defined by Internet Engineering Task Force (IETF) Request For Comments (RFC) 2408, which describes a key management protocol for establishing security associations and cryptographic keys in the Internet environment

[0032] Internet Key Exchange (IKE),

[0033] Group Domain and Interpretation (GDOI) which provides encryption keys and cryptographic policy for groups of users;

[0034] Group Domain and Interpretation version two (GDOIv2), a group key distribution protocol:

and numerous other existing and to be developed protocols.

[0035] FIGS. 4A-4C illustrates a process of generating a large number of keys for use on a communication network according to one embodiment of the invention. Although the keys are described for use on a communication network, the keys may be used in other contexts as well and the invention is not limited to an embodiment that generates keys to be used to secure communication on a network.

[0036] As shown in FIG. 4A, when an initiator and responder wish to enter into a secure relationship, four cryptographic algorithms are negotiated: an encryption algorithm, an integrity protection algorithm, a Diffie-Hellman group, and a pseudo-random function (prf). Examples of pseudo-random functions that may be used include the Message Digest (MD) algorithms, such as MD-5, the Secure Hash Algorithms (SHA) such as SHA-1, and numerous other algorithms. The invention is not limited to use of a particular pseudo-random function. Once the process has been agreed, the initiator will generate a nonce (N_i =a one time used number) and its half of the Diffie-Hellman value (DH_i), and will transmit this information to the GCKS. The GCKS will similarly generate a nonce (N_r) and a Diffie-Hellman value (DH_r) and pass those values to the initiator.

[0037] As shown in FIG. 4B, the initiator's nonce, responder's nonce, and the Diffie-Hellman function (g^{ir}) are input to an agreed pseudo-random function (prf) to generate an initial random value SKEYSEED that will be used to generate a key stream. Generation of the value SKEYSEED may be performed in connection with the IKEv2 protocol, although the invention is not limited in this manner. If the IKEv2 protocol is used, the quantity SKEY-

SEED is calculated from the nonces exchanged during the IKE_SA_INIT exchange (discussed above in connection with FIG. 4A) and the Diffie-Hellman shared secret established during that exchange. SKEYSEED and its derivatives are computed as follows:

$$\text{SKEYSEED} = \text{prf}(\text{Ni} \parallel \text{Nr} \parallel \text{g}^{\text{ir}})$$

Where Ni is the initiator's nonce, Nr is the responder's nonce, and g^{ir} is the shared secret from the ephemeral Diffie-Hellman exchange. g^{ir} is represented as a string of octets in big-endian order padded with zeros if necessary to make it the length of the modulus. Ni and Nr are the nonces, stripped of any headers.

[0038] Keying material is derived as the output of the negotiated prf algorithm. Since the amount of keying material needed may be greater than the size of the output of the prf algorithm, the prf algorithm may be used iteratively. The term prf+ will be used to describe the function that outputs a pseudo-random stream based on the inputs to a prf as follows: (where $\parallel$ indicates concatenation)

$$\text{prf}^+(K, S) = T1 \parallel T2 \parallel T3 \parallel T4 \parallel$$

where:

$$[0039] \quad T1 = \text{prf}(K, S \parallel 0x01)$$

$$[0040] \quad T2 = \text{prf}(K, T1 \parallel S \parallel 0x02)$$

$$[0041] \quad T3 = \text{prf}(K, T2 \parallel S \parallel 0x03)$$

$$[0042] \quad T4 = \text{prf}(K, T3 \parallel S \parallel 0x04)$$

This function may continue to be used, as needed, to generate random information to generate a sufficient number of random bits to generate keys for the exchange. The keys are taken from the output string of the pseudo-random function.

[0043] The constant (0x01, 0x02, 0x03 . . .) concatenated to the end of each string feeding the prf is a single octet. The concatenation pseudo-random function may have a size limit, if desired. For example, in IKE-v2, prf+ is not defined beyond 255 times the size of the prf output. The invention is not limited in this manner, however, as other prf+ functions may be used as well. Particularly, since in one embodiment of the invention the value K is being changed in subsequent iterations, the repeated application of the prf is not weakened and the prf+ function may thus be used to extend beyond the 255 times size limit.

[0044] As shown in FIG. 4C, the prf+ pseudorandom function is used to generate a key stream including seven keys. Other numbers of keys may be used as well and the invention is not limited in this manner. For example, in IKEv2, the prf+ will use the SKEYSEED value generated using the process discussed above, the nonces from the initiator and responder, and two Security Parameter Indices, SPiI and SPiR which serve to identify the security association, to generate seven keys:

$$\{\text{SK}_d \parallel \text{SK}_{ai} \parallel \text{SK}_{ar} \parallel \text{SK}_{ei} \parallel \text{SK}_{er} \parallel \text{SK}_{pi} \parallel \text{SK}_{pr}\} = \text{prf}^+(\text{SKEYSEED}, \text{Ni} \parallel \text{Nr} \parallel \text{SPiI} \parallel \text{SPiR})$$

Where:

[0045] SK_d (Secret Key—derived) used for deriving new keys for the CHILD_SAs established with this IKE_SA;

[0046] SK_ai (Secret Key—authenticating initiator) and SK_ar (secret key—authenticating responder) used as a key to the integrity protection algorithm for authenticating the component messages of subsequent exchanges;

[0047] SK_ei (Secret Key—encryption initiator) and SK_er (Secret Key—encrypting responder) used for encrypting (and of course decrypting) all subsequent exchanges; and

[0048] SK_pi (Secret Key—used for the payload of initiator) and SK_pr (Secret Key—used for the payload of responder) which are used when generating an AUTH payload.

[0049] The two directions of traffic, in this embodiment, use different keys and thus the method described herein has calculated two keys. The keys used to protect messages from the original initiator are SK_ai and SK_ei. The keys used to protect messages in the other direction are SK_ar and SK_er. Each algorithm takes a fixed number of bits of keying material, which is specified as part of the algorithm. Where a different algorithm was to be used, the keys and the functions used to create the keys may be changed without departing from the scope of the invention.

[0050] According to an embodiment of the invention, in addition to calculating the key stream required for use by the initiator and responder, the SKEYSEED is also used to calculate two additional values SK_P_KEK and SK_S_KEK (referred to as L1 and L2 in FIGS. 4C and 4D) which may be used as inputs to another pseudorandom function to generate additional keys for use with other group members. Note that no new random information is needed from the GCKS to generate these additional values. Specifically,

$$\{\text{SK}_d \parallel \text{SK}_{ai} \parallel \text{SK}_{ar} \parallel \text{SK}_{ei} \parallel \text{SK}_{er} \parallel \text{SK}_{pi} \parallel \text{SK}_{pr} \parallel \text{SK}_P_KEK \parallel \text{SK}_S_KEK\} = \text{prf}(\text{SKEYSEED}, \text{Ni} \parallel \text{Nr} \parallel \text{SPiI} \parallel \text{SPiR})$$

[0051] As shown in FIG. 4D, these additional values SK_P_KEK and SK_S_KEK may be used to stretch entropy from another entropy source to allow the entropy source to create a large number of keys for use on the communication network. Specifically, these values may be used, as shown in FIG. 4D, to create a second string of key material, which may be used to create keys for use on the network. The stream of keys (SK_ks—Secret Key key stream) may be represented as:

$$\text{SK}_ks = \text{prf}(\text{fresh random secret}, \text{SK}_S_KEK1 \parallel \text{SK}_S_KEK2 \parallel \text{SK}_S_KEK3 \parallel \dots)$$

or, if SK_S_KEK1=L1 and SK_S_KEK2=L2, then:

$$\text{SK}_ks = \text{prf}(\text{fresh random secret}, L1 \parallel L2 \parallel L3 \parallel \dots \parallel Li), \quad 1 \leq i \leq n,$$

In this embodiment, the pseudorandom function used to create the new string of keys may be known only to the GCKS to increase security of the process. Alternatively, since the fresh random secret is only known to the GCKS and can be changed at will by the GCKS, the pseudorandom function may be publicly known.

[0052] The fresh random secret may contain random information generated by a physical entropy source or may contain random information generated in another manner. However obtained, since the entropy contained in the fresh random secret will be combined with random information

generated during other key exchanges, the entropy contained in the fresh random secret may be stretched to generate a large number of keys. This allows entropy from a physical entropy source to be combined with other sources of entropy to generate a large number of secure keys. Additionally, since the fresh random secret can be changed at will, the key generation process may continuously be changed to further increase the security associated with generating keys.

[0053] In the above description, the fresh random secret is a local secret at the GCKS, and is not known to the outside world. This is a source of randomness that makes it more difficult to decipher the seed key and makes it more difficult to deduce one key if access to the other keys is obtained. Additionally, the fresh random secret may be changed to make it less likely that a pattern may be exploited to obtain the keys. By using other strong random information supplied by group members and derived from the initial prf, this fresh random secret may be stretched to generate many secret keys where it could previously only be used to generate a limited number of secret keys.

[0054] Additionally, since multiple SK_S_KEs are created in each derivation, and because of the local secret, the fact that a nonce and SPI are supplied by a possible adversary (even one of the group members) is not a threat to the other group members. The strength of the algorithm is dependant on the fresh random secret and the pseudo-random function used. Several such pseudo-random functions that may be used in connection with generating the key stream include the message digest (MD) algorithms, such as MD-5, and the secure hash algorithms (SHA) such as SHA-1), although the invention is not limited to the use of one of these particular algorithms as many extant and to be developed algorithms may be used.

[0055] FIG. 5 illustrates an embodiment of a network element configured to implement the GCKS according to an embodiment of the invention. The GCKS may be embodied in a separate network element, as illustrated, or may be implemented as a process on another network element configured to perform other functions on the network. The invention is not limited to the manner in which the GCKS is embodied on the network or the manner in which the GCKS is configured to be implemented in a network element.

[0056] In the embodiment of the data management service illustrated in FIG. 5, the GCKS is configured to be implemented on a network element including a processor 50 having control logic 52 configured to implement the functions ascribed to the GCKS discussed herein in connection with FIGS. 1-4. Input/Output (I/O) ports may be provided to enable the network element to communicate with other network elements on the network.

[0057] The network element has a native or interfaced memory containing data and instructions to enable the processor to implement the functions ascribed to it herein. For example, the memory may contain a key generation software module configured to receive inputs from initiators and generate keys for use on the network, as discussed in greater detail above. A physical or software entropy source 58 may be included to provide the GCKS with a source of random information, for example for use in generating an initial seed key and to generate fresh random secrets.

[0058] The network element may also include one or more additional functional modules, such as an encryption module

60 to enable the GCKS to use one or more of the keys generated to secure communications with the group members during the communication process and in connection with transmission of keys on the network. The invention is not limited to a GCKS having only these particular modules as other modules may be implemented as well.

[0059] It should be understood that all functional statements made herein describing the functions to be performed by the methods of the invention may be performed by software programs implemented utilizing subroutines and other programming techniques known to those of ordinary skill in the art. Alternatively, these functions may be implemented in hardware, firmware, or a combination of hardware, software, and firmware. The invention is thus not limited to a particular implementation.

[0060] The control logic may be implemented as a set of program instructions that are stored in a computer readable memory within the network element and executed on a microprocessor, such as processor. However, in this embodiment as with the previous embodiments, it will be apparent to a skilled artisan that all logic described herein can be embodied using discrete components, integrated circuitry, programmable logic used in conjunction with a programmable logic device such as a Field Programmable Gate Array (FPGA) or microprocessor, or any other device including any combination thereof. Programmable logic can be fixed temporarily or permanently in a tangible medium such as a read-only memory chip, a computer memory, a disk, or other storage medium. Programmable logic can also be fixed in a computer data signal embodied in a carrier wave, allowing the programmable logic to be transmitted over an interface such as a computer bus or communication network. All such embodiments are intended to fall within the scope of the present invention.

[0061] It should be understood that various changes and modifications of the embodiments shown in the drawings and described herein may be made within the spirit and scope of the present invention. Accordingly, it is intended that all matter contained in the above description and shown in the accompanying drawings be interpreted in an illustrative and not in a limiting sense. The invention is limited only as defined in the following claims and the equivalents thereto.

What is claimed is:

1. A method of generating encryption keys, the method comprising the steps of:

obtaining information associated with key exchange sessions from group members;

generating keys for the group members from the information;

generating additional random information during the step of generating keys; and

combining the additional random information with a fresh random secret to stretch the fresh random secret to enable a large number of encryption keys to be generated from the fresh random secret.

2. A method of generating encryption keys, comprising:
combining, using a first pseudo-random function, a plurality of nonces and Diffie-Hellman values to create a first plurality of encryption keys and additional encryption material;
combining, using a second pseudo-random function, the additional encryption material with a fresh random secret to create a second plurality of encryption keys.
3. The method of claim 2, wherein the additional encryption material comprises a first string of random numbers and a second string of random numbers.
4. The method of claim 2, wherein a portion of the nonces and Diffie-Hellman values are obtained from a group member seeking encryption keys, and a portion of the nonces and Diffie-Hellman values are obtained from a Group Controller Key Server (GCKS).
5. A Group Controller Key Server (GCKS), comprising:
program logic code configured to participate in encryption key exchanges with group members;
program logic code configured to generate encryption keys for the group members;
program logic code configured to generate additional encryption key material in connection with generating encryption keys for the group members; and
program logic code configured to combine the additional encryption key material with a fresh random secret to generate additional keys from entropy contained in the additional encryption key material and entropy contained in the fresh random secret.
6. The GCKS of claim 5, wherein the program logic code configured to generate additional keys implements a pseudo-random function.
7. The GCKS of claim 5, further comprising a fresh random secret generator configured to generate the fresh random secret.
8. The GCKS of claim 7, wherein the fresh random secret generator is a physical entropy source.
9. The GCKS of claim 5, wherein the program logic code configured to generate additional keys is further configured to use entropy from other key exchanges with other group members.

10. A method of generating encryption keys for use on a communication network, the method comprising the steps of:

obtaining first entropy from key exchanges with group members; and

combining the first entropy with second entropy contained in a fresh random secret to generate additional keys for use on the communication network.

11. The method of claim 10, wherein the first entropy is obtained in connection with participating in a key exchange procedure, said key exchange procedure comprising exchanging nonces and Diffie-Hellman values, generating a seedkey from the nonces and Diffie-Hellman values, and using the seedkey to generate the encryption keys and the first entropy.

12. The method of claim 11, wherein the seedkey is used as input to a first pseudo-random function to generate a stream of pseudo-random information, a first portion of said pseudo-random information being used to form the encryption keys and a second portion of said pseudo-random information being used to form the first entropy.

13. The method of claim 10, wherein the step of combining the first entropy with second entropy comprises combining first entropy from a plurality of key exchanges with different group members.

14. The method of claim 10, further comprising the step of changing the fresh random secret from a first fresh random secret to a second fresh random secret.

15. The method of claim 10, wherein the fresh random secret is generated by a physical entropy source.

16. The method of claim 15, wherein step of combining the first entropy with the second entropy is performed using a pseudo-random function.

17. The method of claim 16, wherein the key exchanges are performed using a first pseudo-random function, and wherein the step of combining the first entropy with the second entropy is performed using a second pseudo-random function.

* * * * *