US 20150143126A1

(19) United States

(12) Patent Application Publication

BLOM et al.

(10) Pub. No.: US 2015/0143126 A1

(43) Pub. Date: May 21, 2015

(54) **METHOD AND APPARATUS FOR ESTABLISHING A SECURITY ASSOCIATION**

(71) Applicant: **Telefonaktiebolaget LM Ericsson (publ)**, Stockholm (SE)

(72) Inventors: **Rolf BLOM**, Jarfalla (SE); **Karl Norrman**, Stockholm (SE)

(21) Appl. No.: **14/512,239**

(22) Filed: **Oct. 10, 2014**

Related U.S. Application Data

(63) Continuation of application No. 13/348,343, filed on Jan. 11, 2012, now Pat. No. 8,868,912, which is a continuation of application No. 11/305,329, filed on Dec. 19, 2005, now Pat. No. 8,122,240, which is a continuation-in-part of application No. 11/248,589, filed on Oct. 13, 2005, now abandoned.

Publication Classification

(51) **Int. Cl.**

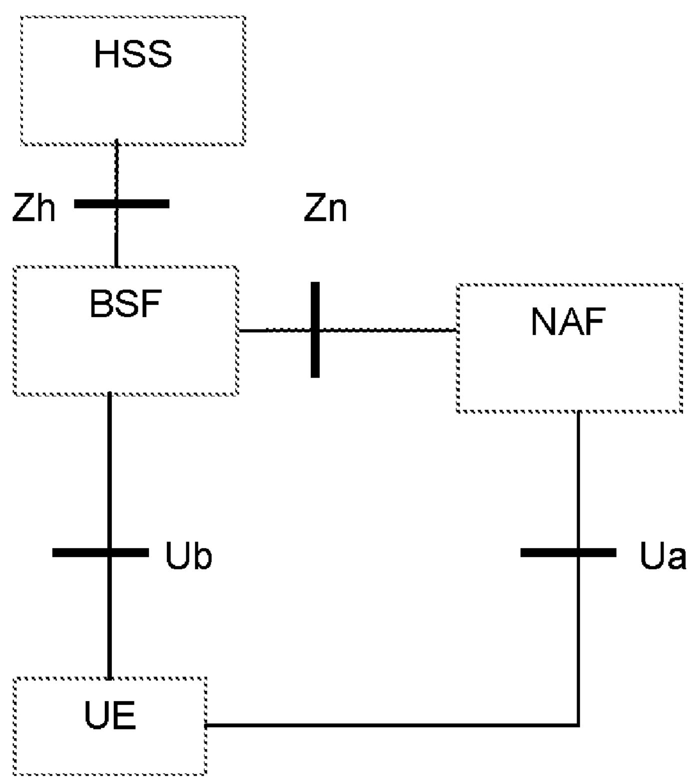
H04W 12/04 (2006.01)
H04L 29/08 (2006.01)
H04L 9/08 (2006.01)
H04L 29/06 (2006.01)

(52) **U.S. Cl.**

CPC *H04W 12/04* (2013.01); *H04L 63/0435* (2013.01); *H04L 63/062* (2013.01); *H04L 67/26* (2013.01); *H04L 9/0841* (2013.01)

(57) **ABSTRACT**

A method for establishing a security association between a client and a service node for the purpose of pushing information from the service node to the client, where the client and a key server share a base secret. The method comprises sending a request for generation and provision of a service key from the service node to a key server, the request identifying the client and the service node, generating a service key at the key server using the identities of the client and the service node, the base secret, and additional information, and sending the service key to the service node together with said additional information, forwarding said additional information from the service node to the client, and at the client, generating said service key using the received additional information and the base key. A similar approach may be used to provide p2p key management.



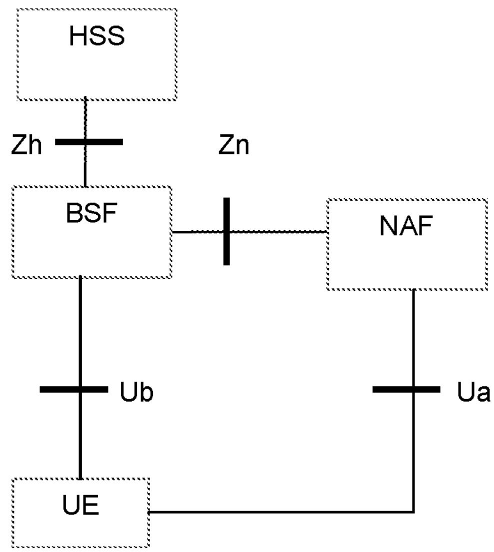


Figure 1

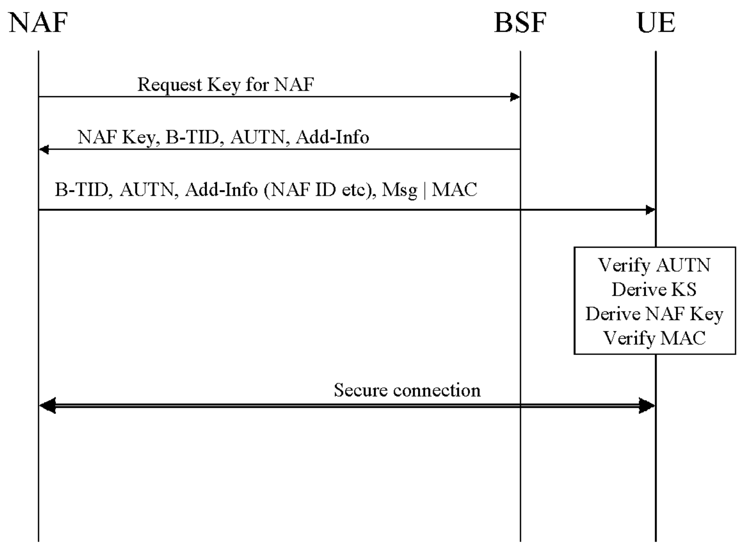


Figure 2

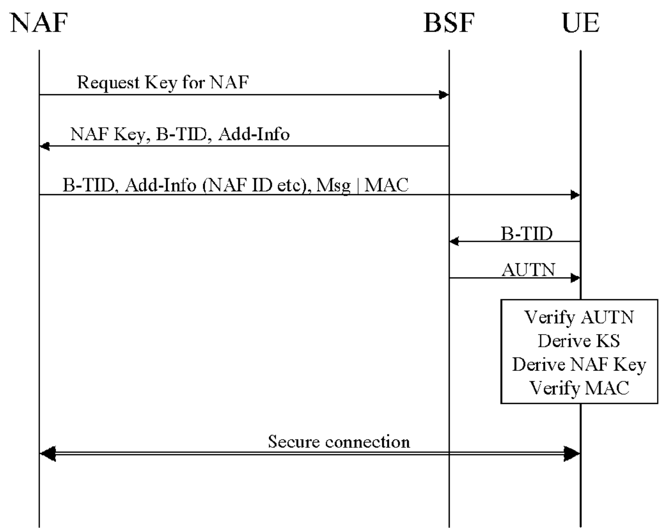


Figure 3

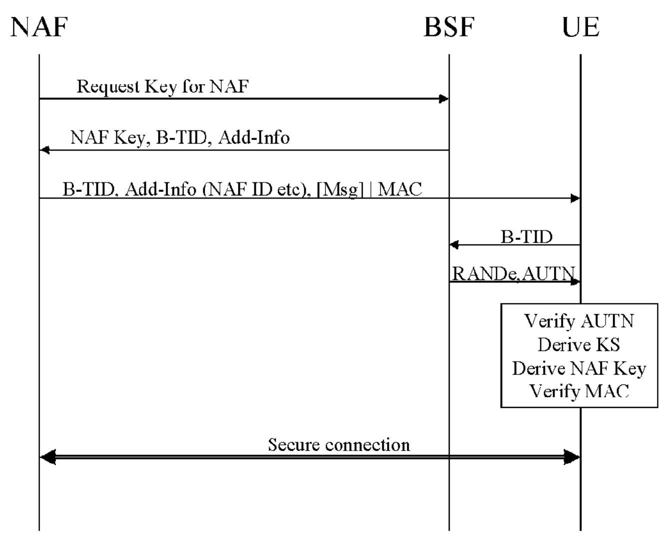


Figure 4

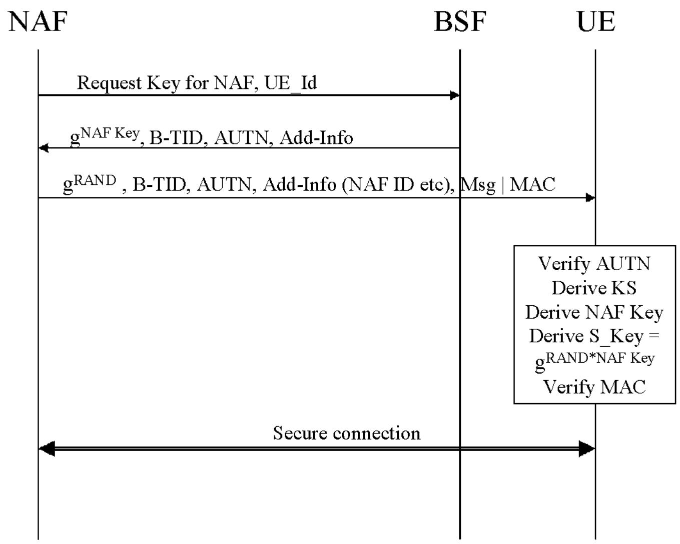


Figure 5

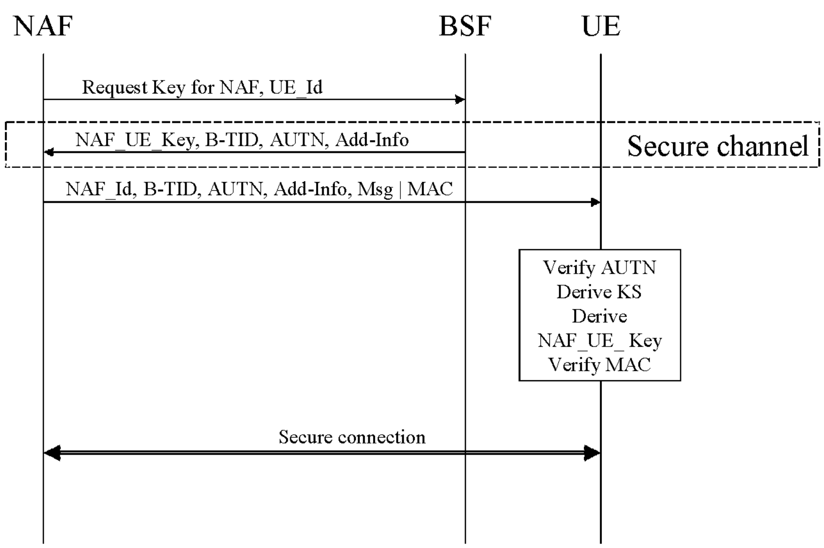


Figure 6

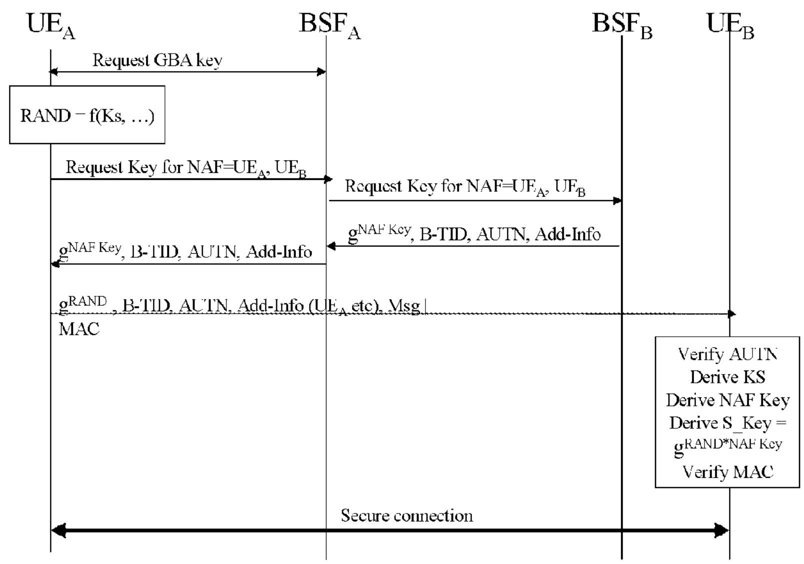


Figure 7

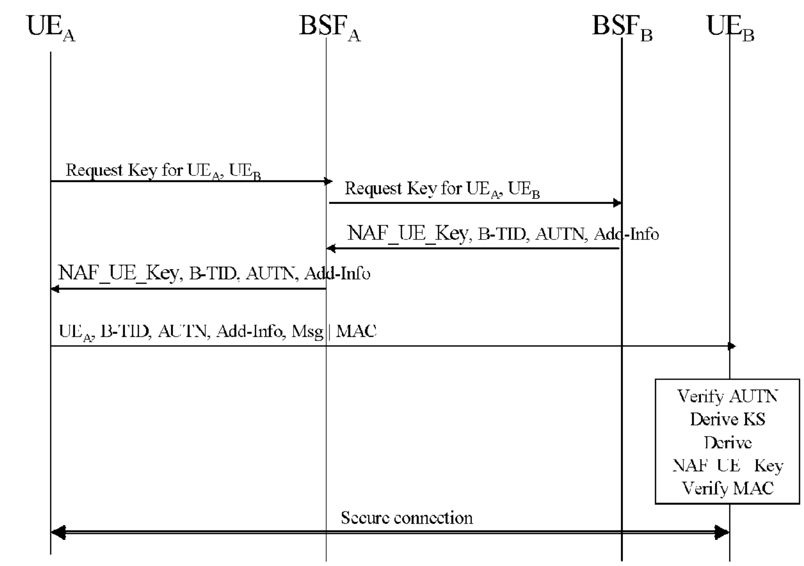


Figure 8

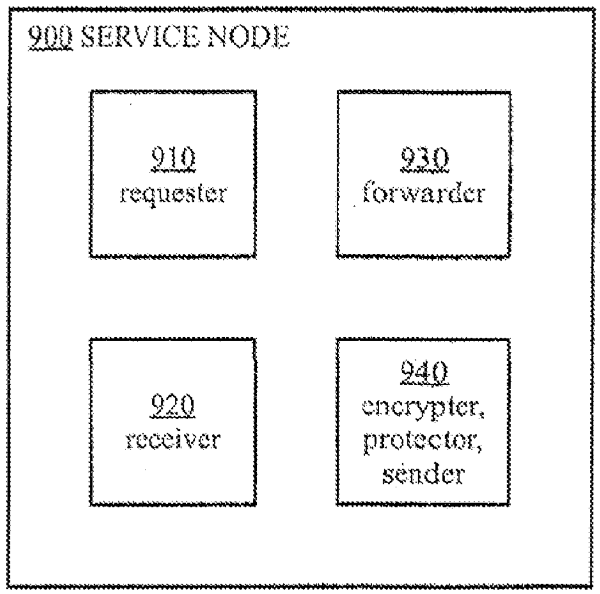


Figure 9

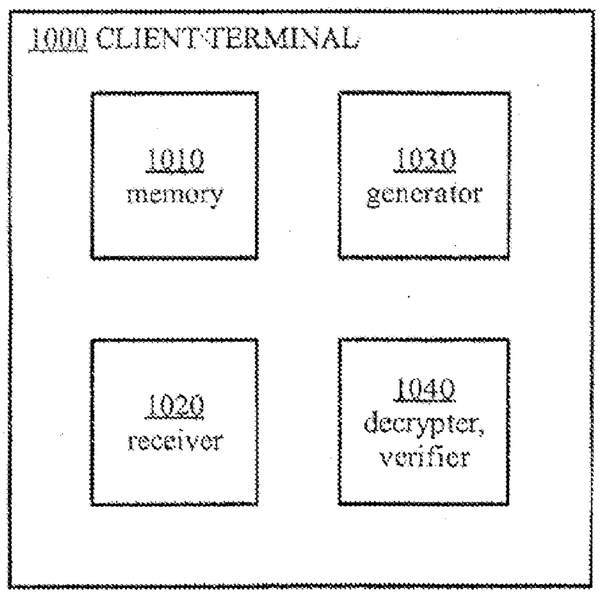


Figure 10

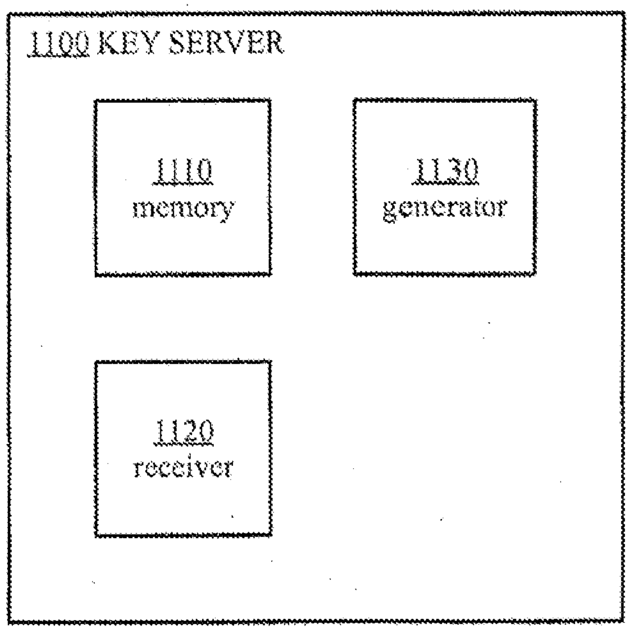


Figure 11

METHOD AND APPARATUS FOR ESTABLISHING A SECURITY ASSOCIATION

RELATED APPLICATIONS

[0001] This application is a Continuation of Ser. No. 13/348,343, filed Jan. 11, 2012, which is a Continuation of Ser. No. 11/305,329 filed Dec. 19, 2005, now U.S. Pat. No. 8,122,240, which is a Continuation-in-part of Ser. No. 11/248,589, filed Oct. 13, 2005, now abandoned, the entire contents of each of which are hereby incorporated by reference in this application.

FIELD OF THE INVENTION

[0002] The present invention relates to a method and apparatus for establishing a security association between a client terminal and a service node in order to deliver a push-type service and in particular, though not necessarily, to such a method and apparatus which employs a Generic Bootstrapping Architecture.

BACKGROUND TO THE INVENTION

[0003] In order to facilitate the provision of services to user terminals, a mobile network such as a 3G network will often require the establishment of a secure communication channel or "security association" between client terminals (i.e. mobile terminals) and the network-based service nodes which provide the services. The Generic Bootstrapping Architecture (GBA) is discussed in the 3GPP Technical Specification TS 33.220 and provides a mechanism whereby a client terminal (UE) can be authenticated to a Network Authentication Function (the service node), and secure session keys obtained for use between the client terminal and the Network Authentication Function. The simple network model for this architecture is illustrated in FIG. **1**. This mechanism bootstraps upon the known Authentication and Key Agreement (AKA) procedure [3GPP TS 33.102] which allows a client terminal to be authenticated to a Bootstrapping Server Function (BSF) of the client's home network on the basis of a secret K which is shared between the USIM of the client terminal and the Home Subscriber System (HSS) of the subscriber's home network. The AKA procedure further establishes session keys from which keys are derived that are afterwards applied between the client terminal and a Network Application Function (NAF). When a client terminal and NAF wish to obtain session keys from the BSF, the NAF sends a transaction identifier to the BSF, the transaction identifier containing an index which the BSF uses to identify the client terminal and appropriate keys which it forwards to the NAF.

[0004] According to the GBA mechanism, a UE initiates the key generation process by sending a request containing a user identity to the BSF. The request also contains the identity of the NAF. The BSF retrieves an authentication vector from the Home Subscriber System (HSS), each authentication vector consisting of a random number RAND, an expected response XRES, a cipher key CK, an integrity key IK and an authentication token AUTN. The BSF generates key material KS by concatenating CK and IK contained within the authentication vector. The BSF generates a key identifier B-TID in the format of a NAI by base64 encoding the RAND value and combining the encoded value with the BSF server name, i.e. as

base64encode(RAND)@BSF_servers_domain_name.

[0005] The BSF retains the key KS in association with the transaction identifier B-TID and the NAF identity. The B-TID and AUTN are sent by the BSF to the UE, the USIM of the client terminal verifying the value AUTN using the shared secret K and returning a digest of the expected result XRES to the BSF. The USIM also generates the key material KS using the secret K and the value RAND (recovered from the B-TID).

[0006] Following completion of this procedure, the UE communicates to the NAF, the received B-TID. The NAF and the BSF are authenticated to one another, and the NAF sends to the BSF the received B-TID together with its own identity. The BSF uses the B-TID and the identity of the NAF to locate the correct key KS, and uses KS to generate a NAF key. Other information such as the NAF identity is also used in the generation of the NAF key. The generated NAF key is returned to the NAF. The UE is similarly able to generate the NAF key using the key KS that it has already generated.

[0007] After the GBA mechanism has been run for the first time, subsequent requests to establish a security association between the UE and the same or a different NAF may use the already established key material KS, providing that key has not expired. However, this will still require that the UE initiate a request for establishment of a security association by sending its B-TID to the NAF.

SUMMARY OF THE INVENTION

[0008] There are occasions on which it is desirable to allow the NAF to initiate the establishment of a security association with the UE. For example, one might consider a push-type service, which delivers news, sports, and financial, etc information to users who have previously registered for a service. A typical operational procedure to achieve this might be for the service provider to send an SMS message to the UE which requests the user to open a secure connection. However, there are many threats related to this model as an SMS might be manipulated, sent by an unauthorized party, be replayed, etc. If a security association existed, or the service node could initiate one, before the actual service data is sent, security procedures could be based on this and most problems could be mitigated.

[0009] According to a first aspect of the present invention there is a provided method of establishing a security association between a first node and a second node for the purpose of pushing information from the first node to the second node, where the second node and a key generation function share a base secret, the method comprising:

- **[0010]** sending a request for generation and provision of a service key from the first node to the key generation function, the request containing identities of the first and second nodes;
- **[0011]** generating a service key at the key generation function using the identity of the first node, the base secret, and additional information, and sending the service key to the first node together with said additional information;
- **[0012]** forwarding said additional information and said identity of the first node from the first node to the second node; and
- **[0013]** at the second node, generating said service key using the received additional information, the first user identity, and the base secret.

[0014] It will be appreciated that the key generation function may be a stand-alone node or may be a distributed server.

In the case of a 3G network employing the Generic Bootstrapping Architecture, a Bootstrapping Server Function and a Home Subscriber Server may together provide the key generation function, where the Bootstrapping Server Function communicates with the service node and with the Home Subscriber Server. In the case of a 2G network, the key generation function may be a combination of a Bootstrapping Server Function and an AuC server.

[0015] In the case of a 3G network employing the Generic Bootstrapping Architecture, the service node comprises a Network Application Function. The step of generating a service key at the key generation function comprises the steps of:

- **[0016]** generating key material KS using said base secret; and
- **[0017]** generating the service key using said key material KS, the identity of the service node, and said additional information.

[0018] The step of generating the service key at the client also comprises these two steps.

[0019] Said step of generating a service key at the key server may utilise values other than those sent to the client by the service node. The client may obtain certain of those other values from the key server.

[0020] Said additional information may comprise one or more of:

- **[0021]** a random value;
- **[0022]** time stamp;
- **[0023]** sequence number;
- **[0024]** other identifiers

In the case of the Generic Bootstrapping Architecture, said random value is the RAND parameter and is carried within the B-TID.

[0025] Said additional information may comprise a transaction identifier in the format of an NAI, and comprising an encoded random value.

[0026] Said additional information may be forwarded from the service node to the client in a message also containing service data, the service data being encrypted with the service key, wherein the client can decrypt the encrypted data once it has generated the service key.

[0027] In one embodiment of the invention, the key generation function sends to the service node a network authentication value. The service node forwards this value to the client, together with said additional information. The client uses the base secret and the authentication value to authenticate the key generation function. Only if the key generation function is authenticated does the client generate and use the service key.

[0028] In an alternative embodiment of the invention, the client requests an authentication value from the key generation function after it has received said additional information from the service node. Only when the client has authenticated the key generation function is the service key generated and used.

[0029] The terminal may comprise means for receiving from the service node a message authentication code, the terminal comprising means for generating an authentication key or keys from at least a part of the key generation information, and using the authentication key(s) to authenticate the message authentication code. The generation means may be a USIM/ISIM.

[0030] Said service key may be a Diffie-Hellman key for the second node, the method further comprising the step of providing to the first node a Diffie-Hellman key for that first node, and sending the Diffie-hellman key for the first node to the second node, said security association being established on the basis of the two Diffie-Hellman keys.

[0031] According to a second aspect of the present invention there is provided a service node for delivering a push service to a client via a secure communication link, the service node comprising:

- **[0032]** means for sending a request for generation and provision of a service key to a key generation function, the request identifying the client and the service node;
- **[0033]** means for receiving from the key generation function a service key together with said additional information;
- **[0034]** means for forwarding said additional information to the client; and
- **[0035]** means for encrypting and/or integrity protecting service information using the service key and for sending the encrypted and/or protected information to the client.

[0036] In the case of the Generic Bootstrapping Architecture, said additional information comprises a B-TID containing the RAND value. Said means for forwarding is also arranged to forward to the client an identity of the service node.

[0037] According to a third aspect of the invention there is provided a client terminal for receiving a pushed service delivered by a service node, the client terminal comprising:

- **[0038]** memory means for storing a secret that is shared with a key generation function;
- **[0039]** means for receiving from said service node, key generation information;
- **[0040]** means for generating a service key using said shared secret and said key generation information; and
- **[0041]** means for using said service key to decrypt and/or verify the integrity of communications with the service node.

[0042] According to a fourth aspect of the present invention there is provided a key generation function for use in establishing a security association between a client and a service node for the purpose of pushing information from the service node to the client, the key server comprising:

- **[0043]** memory means for storing a secret that is shared with said client;
- **[0044]** means for receiving a request for generation and provision of a service key from said service node, the request identifying the client and the service node; and
- **[0045]** means for generating a service key using the identities of the client and the service node, the base secret, and additional information, and for sending the service key to the service node together with said additional information.

[0046] According to a fifth aspect of the present invention there is provided a method of establishing a security association between first and second clients for the purpose of pushing information from the first client to the second client, where the first and second clients have trust relationships with first and second key servers respectively and share a secret with their respective key servers, the method comprising:

- **[0047]** sending a request for generation and provision of a service key from the first client to said second key server via the first key server, the request identifying the first and second nodes;
- **[0048]** generating a service key at the second key server using the identity of the first node, the base secret, and

additional information, and sending the service key to the first node together with said additional information;

[0049] forwarding said additional information from the first node to the second node; and

[0050] at the second node, generating said service key using the received additional information and the base secret.

[0051] According to a sixth aspect of the present invention there is provided a method of protecting a node against replay attacks, the method comprising:

[0052] generating a service key at a bootstrapping server function;

[0053] providing the service key to a first node together with information required to generate the service key;

[0054] sending a key generation message from the first node to the second node, the message including said information, a replay prevention value, and a message authentication code calculated over the message body including the replay prevention value, the replay prevention value being incremented or decremented for each run of the procedure;

[0055] receiving said key generation message at said second node and storing the replay prevention value contained therein; and

[0056] at the second node, each time a key generation message is received, verifying said message authentication code, determining whether or not the replay prevention value contained in the message has already been stored at the second node, and, if yes, rejecting the message.

[0057] Embodiments of this aspect of the invention allow the second node to reject replay attacks based upon messages previously sent to the second node in respect of a valid GBA procedure. If the attacker were to merely increment that replay prevention value to a previously unused value, the second node would detect this change based upon the incorrect MAC value, and would hence detect the attack. Again, the first node may be a NAF server, with the second node being a client, or both the first and second nodes may be clients. It will be appreciated that features of the first to fifth aspects of the present invention may be combined with those of the sixth aspect, and vice versa.

BRIEF DESCRIPTION OF THE DRAWINGS

[0058] FIG. **1** illustrates a simple network model for the Generic Bootstrapping Architecture in accordance with an embodiment of the present invention;

[0059] FIG. **2** depicts signaling flows associated with respective procedures for establishing a security association between a client (UE) and NAF in accordance with an embodiment of the present invention;

[0060] FIG. **3** illustrates additional signaling flows associated with respective procedures for establishing a security association between a client (UE) and NAF in accordance with an embodiment of the present invention;

[0061] FIG. **4** depicts additional signaling flows associated with respective procedures for establishing a security association between a client (UE) and NAF in accordance with an embodiment of the present invention;

[0062] FIG. **5** illustrates additional signaling flows associated with respective procedures for establishing a security association between a client (UE) and NAF in accordance with an embodiment of the present invention;

[0063] FIG. **6** depicts additional signaling flows associated with respective procedures for establishing a security association between a client (UE) and NAF in accordance with an embodiment of the present invention;

[0064] FIG. **7** depicts additional signaling flows associated with respective procedures for establishing a security association between a client (UE) and NAF in accordance with an embodiment of the present invention;

[0065] FIG. **8** illustrates signaling flows associated with respective procedures for establishing a security association between a pair of clients (UE_A and UE_B) in accordance with an embodiment of the present invention;

[0066] FIG. **9** depicts a service node according to an embodiment of the present invention;

[0067] FIG. **10** illustrates a client terminal according to an embodiment of the present invention; and

[0068] FIG. **11** depicts a key server according to an embodiment of the present invention.

DETAILED DESCRIPTION OF CERTAIN EMBODIMENTS

[0069] The general Generic Bootstrapping Architecture (GBA) for 3G networks has been described with reference to FIG. **1**, which illustrates the interfaces (Ua, Ub, Zn, and Zh) between the various entities. It should be borne in mind that the description is on a relatively high level and actual implementations may "look" different whilst employing the same general functionality. For example, it is possible that when a BSF receives a service key request from a NAF (as will be described below), the receiving BSF must perform an address resolution step to identify a "serving" BSF for the NAF or client (UE) and, if the receiving BSF is not the serving BSF, the request is forwarded on to the serving BSF.

[0070] This discussion concerns the provision of a push service to a client. Typically, the client will have pre-registered with the service provider, but the initiative to push particular information is taken by the service provider. In such a situation, the service provider and the client will not already have a security association established with each other (security associations are typically short-lived), and one must be established.

[0071] A first solution proposed here takes the approach that the NAF asks the BSF for a NAF (or service) key. The BSF returns to the NAF, the NAF key together with the client transaction identifier (B-TID) and the corresponding network authentication value (AUTN). As has been stated above, the B-TID contains the encoded RAND value (as the NAI prefix), which can be used by the client to derive the base key (KS). The NAF can now compose a message containing the B-TID, AUTN, and further data including the NAF identity that the client requires in order to derive the NAF key, and send this message to the client. This message can be a message that only triggers the set-up of a SA (i.e. sharing of a service key) or it could contain service data (i.e. payload data) encrypted with the service key. In both cases, the values B-TID, AUTN, and other data required by the client to generate KS are sent in plain text but are "signed" with a Message Authentication Code. Note that the key(s) in the SA are derived using the key shared between the HSS and the UE, and that the AUTN is included in the message. It is therefore not possible to "spoof" messages even though the key used for integrity protecting the message is derived from the very SA it is intended to establish.

[0072] When the client receives the message, it retrieves the RAND part of the B-TID (by reversing the encoding) and the AUTN and applies them to the USIM/ISIM to derive the base key Ks. Then it uses the further data to derive the NAF key, and verifies the received message using the MAC.

[0073] The signalling exchanges associated with this procedure are illustrated in FIG. **2**.

[0074] In order to prevent the manipulation of the further data (required by the client) by the NAF, the BSF may sign that data using a derivative of KS. This may be important, for example, to prevent the NAF from extending the lifetime of a key.

[0075] The solution presented above allows the NAF to push to the client the information required to establish a security association between the two parties. Thus the client does not have to set up a connection with the BSF to perform these tasks. This represents an extremely time efficient solution. However, it requires that the NAF relay all key related information (key lifetime, Add-info, etc) in a protected form from the BSF to the UE. The B-TID and the other data might then comprise quit a large data structure. This might be problematic in the case where the volume of data that can be incorporated into the message structure used between the client and the NAF, e.g. where this structure is SMS.

[0076] In order to reduce the required data volume exchanged between the NAF and the client to establish the security association, the above solution may be modified by omitting the AUTN value from the data sent by the BSF to the NAF. The NAF now composes a message containing the B-TID and other necessary data (including the NAF identity) that the terminal needs to derive the NAF key and sends it to the client. Again, this message could be a message which only triggers the set-up of a security association, or it could contain encrypted payload data.

[0077] When the client receives the message from the NAF, it connects to the BSF transmitting the B-TID thereto, authenticates itself, and requests the remaining information necessary to derive the keying material associated with the B-TID, i.e. e.g. AUTN. After having received this information it derives the service (NAF) key and verifies the integrity of the message. As the client has to connect to the BSF, it can at the same time get all the information related to the keying material, i.e. Add-Info, key life time etc, thus reducing the amount of "administrative" information that has to be transmitted from the NAF to client.

[0078] The signalling exchange associated with this procedure, assuming the Ks generation scenario (i.e. analogous to FIG. **2**), is shown in FIG. **3**.

[0079] It may be undesirable in some circumstances to reveal the value RAND to the NAF. This may be avoided by forming the B-TID using a reference to the actual RAND value (or the effective RAND, RANDe), so that the NAF sees only the reference value. The effective RAND (RANDe) would then have to be signalled together with the AUTN from the BSF to the client. This modified procedure is illustrated in FIG. **4**.

[0080] The main advantage of the solutions described with reference to FIGS. **3** and **4** is that the BSF will have a further opportunity to control the key generation in the client. The client needs the AUTN to derive the key. On the other hand, the client will have to connect to the BSF and authenticate itself towards the BSF requiring a new variant of the GBA protocol over the Ub interface.

[0081] One threat to the solutions of FIGS. **3** and **4** is that an attacker might generate a batch of messages (purporting to contain a valid B-TID) and send them to different clients to launch a Denial-of-Service (DoS) attack. As the clients have no means to authenticate the messages (i.e. a AUTN), they will connect to the BSF in an attempt to authenticate the received messages. Such an attack will, if not resisted, consume considerable resources on the part of the BSF. To make such a DoS attack more difficult, it would be desirable to enable the client to immediately check the MAC of the message pushed by the NAF in order to validate the message without having to connect to the BSF. To achieve this, the client has to be able to derive the key that is used for the MACing of the message. As the AUTN is not sent to the client in the pushed message, this derivation has to be based only on the RAND (or derived value, FIG. **4**) in the B-TID.

[0082] A solution is to use the RAND (or derived value) in the B-TID to derive two keys Ck' and Ik' at the BSF. The BSF then derives a MAC key using these keys, and sends the MAC key to the NAF. This integrity key should preferably also depend on the NAF identity. Using a "fingerprint" of the other necessary information needed to derive the NAF key in the derivation of the integrity key would be one way to achieve this without having to send all the information to the UE. The NAF computes a second (short) MAC over at least a part of the data to be sent to the client, and includes the MAC in the message sent to the client. At the client, the USIM/ISIM uses the AKA algorithms to generate Ck' and Ik' and hence the second MAC key, and the client can then verify the message. Alternatively, the BSF can provide the keys Ck' and Ik' to the NAF to enable the NAF to generate the second MAC key itself. This doesn't stop replay of old message (although this could be addressed with the use of timestamps), but it does stop attackers from generating random messages.

[0083] In an alternative solution, illustrated in the signalling diagram of FIG. **5**, the BSF does not generate and send the NAF key itself to the NAF in response to the NAF request for a PUSH key for a given user. Rather, the BSF sends a Diffie-Hellman public value $g^{NAF\ Key}$ based on the NAF-Key (or on some other value based on the associated shared secret Ks) and data related to the identity of the involved parties and intended use of the key. The NAF may now chose a secret value RAND of its own, and append the corresponding public Diffie-Hellman value g^{RAND} for that secret value to the info sent to the UE. Both parties can then derive a common shared key, S_Key=$g^{RAND*NAF\ Key}$ The S_Key is used to key the MAC. It is noted that Diffie-Hellman schemes can be implemented over different types of groups. Here we use the standard notation when the group is Zp and the generating g element used is denoted g.

[0084] According to a still further alternative solution, illustrated in the signalling diagram of FIG. **6**, when the NAF requests a PUSH key for a given user, the BSF does not include a standard NAF key but rather derives a key which relies additionally on both the UE_identity and the NAF_identity (in addition to any further data). Such a key is be denoted "NAF_UE_Key" in the Figure. In order to secure the delivery of the delivery of the key to the NAF from the BSF, the BSF includes in the message to the BSF a MAC calculated using the NAF_UE Key.

[0085] The above discussion has considered the application of the invention to the provision of service related keys to users and service nodes. Another application of the present invention relates to the provision of keys to client terminals to allow one client terminal to push messages to a peer client terminal in a secure manner, that is to say peer-to-peer (p2p) key management.

[0086] According to one solution, an initiating UE, i.e. UE_A, employs the method illustrated generally in FIG. **7**. This approach relies upon an explicit trust relationship between BSF_A and BSF_B. The initiating party first performs a standard GBA procedure with the BSF_A of its home network in order to obtain a base key, Ks_A. UE_A then uses the base key to derive a RAND tied to the other party UE_B to which UE_A wishes to push a message. This can be done in the same way as NAF keys are derived. The second action performed by UE_A is to request key information for UE_B. This request, containing the identities of both clients, is sent to BSF_A, which forwards the request to the BSF within the home network of UE_B, i.e. BSF_B.

[0087] The BSF_B returns to UE_A, via BSF_A, a Diffie-Hellman public value for UE_B, namely $g^{NAF\ Key}$. It also returns the B-TID (containing the RAND' value used to generate the NAF Key), AUTN, and required further data. The initiating party UE_A then forms a message containing its public Diffie-Hellman value, g^{RAND}, and the information needed by the receiver to derive the KS_B, the related NAF_Key, and hence the session key $g^{RAND*NAF\text{-}Key}$, UE_A can of course derive the same session key.

[0088] An alternative p2p key management solution is illustrated in FIG. **8** and requires that BSF_B generate the key to be shared by the peers. The first action by the initiating party UE_A is to request a key for the other party UE_B. This request is sent to the initiating party's BSF_A, which forwards the request to the receiving party's BSF_B. The initiating party includes its identity as well as that of the receiving party's in the request, and the BSF_B derives the key to be shared, i.e. NAF_UE_Key. The derived key together with the B-TID, AUTN, etc is then delivered to UE_A.

[0089] With this scheme, the receiving party does receive an implicit verification of the sender's claimed identity as this identity is used in the NAF_UE_Key derivation. The receiving party could also get an explicit authentication if BSF_B includes a MAC based on a "NAF_Key" covering all data, as described above.

[0090] It will be appreciated by the person of skill in the art that various modifications may be made to the above described embodiments without departing from the scope of the present invention. For example, whilst the solutions presented above have been concerned with GBA, the invention has general applicability to architectures where information is to be pushed from a service provider and where the service provider and the client do not share a common secret. In another modification, where multiple solutions are implemented in parallel, the authentication request sent to the BSF contains a selector indicating which solution the NAF/UE shall employ.

1.-**28**. (canceled)

29. A method of establishing a security association between a first node and a second node for pushing information from the first node to the second node, wherein the second node and a key generation function share a secret, the method comprising:

sending, by the first node, a request for generation and provision of a service key from the first node to the key generation function, the request comprising an identity of the first node and an identity of the second node;

generating, by the key generation function, the service key using the identity of the first node, the secret, and additional information;

signing, by the key generation function, the additional information using a derivative of the secret;

sending, by the key generation function, the service key to the first node together with the additional information;

forwarding, by the first node, the additional information and the identity of the first node from the first node to the second node;

verifying, by the second node, the additional information; and

generating, by the second node, the service key using at least the additional information, the identity of the first node, and the secret.

30. The method according to claim **29**, wherein the first node is a service node and the second node is a client.

31. The method according to claim **30**, wherein the client is a client terminal of a 3G network employing a Generic Bootstrapping Architecture, and wherein the service node comprises a Network Application Function and the key generation function comprises a Bootstrapping Server Function.

32. The method according claim **31**, wherein the generating, by the key generation function, the service key further comprises:

generating, by the key generation function, key material using the secret; and

generating, by the key generation function, the service key using the key material , the identity of the service node, and the additional information.

33. The method according to claim **31**, wherein the generating, by the client, the service key, further comprises:

generating, by the client, key material using the secret; and

generating, by the client, the service key using the key material and the additional information.

34. The method according to claim **33**, wherein the secret is stored in an IP Multimedia Services Identity Module (ISIM) or Universal Subscriber Identity Module (USIM) of the client, and the generating, by the client, the key material is performed by the ISIM or the USIM.

35. The method according to claim **30**, wherein the generating, by the key generation function, the service key further comprises generating, by the key generation function, the service key utilizing values other than values sent to the client by the service node.

36. The method according to claim **35**, wherein at least some of the values other than values sent to the client by the service node are values obtained by the client from the key generation function.

37. The method according to claim **29**, wherein the additional information comprises at least one or more of:

a transaction identifier; and

a network authentication value.

38. The method according to claim **29**, wherein the additional information comprises a transaction identifier in a format of a Network Access Identifier (NAI), wherein the transaction identifier further comprises an encoded random value generated by the key generation function, the encoded random value being used, by the second node, to generate the service key.

39. The method according to claim **30**, wherein the additional information comprises a transaction identifier in a format of a Network Access Identifier (NAI), wherein the transaction identifier further comprises a pointer to a random value generated by and stored at the key generation function, the random value being used, by the client, to generate the service key, wherein the method further comprises:

sending, by the client, the pointer from the client to the key generation function; and

returning, by the key generation function, the random value to the client the random value being used, by the client, to generate the service key.

40. The method according to claim **30**, wherein the key generation function sends to the service node a network authentication value and the service node forwards the network authentication value to the client, together with the additional information, wherein the client uses the secret and the network authentication value to authenticate information received from the key generation function.

41. The method according to claim **30**, comprising:

sending, by the client, a request from the client to the key generation function for an authentication value after the client receives the additional information from the service node;

receiving, by the client, the authentication value; and

authorizing, by the client, the request received from the service node on the basis of the authentication value.

42. The method according to claim **30**, wherein the additional information is forwarded from the service node to the client in a message that further comprises service data, wherein the service data is encrypted or integrity protected with the service key, and wherein the client can decrypt the encrypted data once the client has generated the service key.

43. The method according to claim **29**, wherein the key generation function generates the service key further using the identity of the second node.

44. The method according to claim **29**, wherein the service key is a Diffie-Hellman key for the second node, and the method further comprises:

providing, to the first node, a Diffie-Hellman key for the first node; and

sending the Diffie-Hellman key for the first node to the second node, the security association being established on a basis of the Diffie-Hellman key for the first node and the Diffie-Hellman key for the second node.

45. The method according to claim **29**, wherein the first node is a first client and the second node is a second client.

46. The method according to claim **45**, wherein the key generation function is implemented on a key server having a trust relationship with the second client and the request for generation and provision of the service key is sent to the key server via a second key server having a trust relationship with the first client.

47. The method according to claim **46**, further comprising:

sending, by the first node, from the first node to the second node, the service key obtained by the first node, and deriving, at the first node and the second node, a session key using the service key obtained by the first node and the service key generated at the second node.

48. A service node arranged to deliver a push service to a client terminal via a secure communication link, the service node comprising:

a transmitter arranged to send a request for generating and provisioning of a service key to a key generation function, the request comprising an identity of the client terminal and an identity of the service node;

a receiver arranged to receive, from the key generation function, the service key together with additional information, the additional information having been signed using a derivative of a secret shared by the client terminal and the key generation function;

a forwarder arranged to forward the additional information to the client terminal; and

a sender arranged to encrypt or protect integrity of service information using the service key and to send the encrypted or protected information to the client terminal.

49. A client terminal arranged to receive a pushed service delivered by a service node, the client terminal comprising:

a storage medium for storing a secret that is shared with a key generation function;

a receiver arranged to receive, from the service node, key generation information, the key generation information having been signed using a derivative of the secret; and

a verifier arranged to verify the key generation information using the derivative of the secret; and

a generator arranged to generate a service key using the key generation information.

50. The client terminal according to claim **49**, wherein the verifier is further arranged to use the service key to decrypt or verify integrity of communications with the service node.

51. The client terminal according to claim **49**, wherein

the receiver is further arranged to receive from the service node a message authentication code; and

the generator is further arranged to generate at least one authentication key from at least a part of the key generation information, and to use the at least one authentication key to verify the message authentication code.

52. The client terminal according to claim **51**, wherein

the generator is further arranged to generate the at least one authentication key by sending the key generation information to an IP Multimedia Services Identity Module (ISIM) or Universal Subscriber Identity Module (USIM).

53. A key server implementing a key generation function arranged to establish a security association between a client terminal and a service node for pushing information from the service node to the client terminal, the key server comprising:

a storage medium for storing a secret that is shared with the client terminal;

a receiver to receive a request for generating and provisioning of a service key from the service node, the request comprising an identity of the client terminal and an identity of the service node; and

a generator arranged to generate the service key using the identity of the service node, the secret, and key generation information, to sign the key generation information using a derivative of the secret, and to send the service key to the service node along with the key generation information.

54. A method of establishing a security association between a first client and a second client for pushing information from the first client to the second client, wherein the first client has a trust relationship with a first key server and the second client has a trust relationship with a second key server, and wherein the first client shares a first secret with the first key server and the second client shares a second secret with the second key server, the method comprising:

sending, by the first key server, a request for generation and provision of a service key from the first client to the second key server via the first key server, the request comprising an identity of the first client and an identity of the second client;

generating, by the second key server, the service key using the identity of the first client, the secret, and key generation information;

signing, by the second key server, the key generation information, using a derivative of the secret;

sending, by the second key server, the service key to the first client together with the key generation information;

forwarding, by the first client, the key generation information from the first client to the second client;

verifying, by the second client, the key generation information using the derivative of the secret; and

generating, by the second client, using the service key using the key generation information and the secret.

55. A method in a client terminal for receiving a pushed service delivered by a service node, the method comprising:

storing a secret that is shared with a key generation function;

receiving, from the service node, key generation information, the key generation information having been signed using a derivative of the secret;

verifying the key generation information using the derivative of the secret; and

generating a service key using the key generation information and the secret.

56. The method according to claim **55**, further comprising:

using the service key to decrypt or verify integrity of communications with the service node.

57. The method according to claim **56**, further comprising:

receiving, from the service node, a message authentication code;

generating at least one authentication key from at least a part of the key generation information; and

using the at least one authentication key to verify the message authentication code.

58. The method according to claim **57**, wherein the generating the at least one authentication key is performed by sending the key generation information to an IP Multimedia Services Identity Module (ISIM) or Universal Subscriber Identity Module (USIM).

* * * * *