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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
22 March 2018 (22.03.2018)

WIPO | PCT

(10) International Publication Number
WO 2018/053271 A1

(51) International Patent Classification:
H04L 9/00 (2006.01) *H04L 29/06* (2006.01)
H04L 9/08 (2006.01) *H04L 9/32* (2006.01)

(21) International Application Number:
PCT/US2017/051788

(22) International Filing Date:
15 September 2017 (15.09.2017)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
62/395,889 16 September 2016 (16.09.2016) US

(71) Applicant: **IDAC HOLDINGS, INC.** [US/US]; 200 Bellevue Parkway, Suite 300, Wilmington, DE 19809 (US).

(72) Inventors: **SEHRAWAT, Vipin, Singh**; 2400 Waterview Parkway, Apt. 522, Richardson, TX 75080 (US). **BRUSILOVSKY, Alec**; 200 Bellevue Parkway, Suite 300, Wilmington, DE 19809 (US). **CHOYI, Vinod, Kumar**; 200 Bellevue Parkway, Suite 300, Wilmington, DE 19809 (US). **SHAH, Yogendra, C.**; 200 Bellevue Parkway, Suite 300, Wilmington, DE 19809 (US). **FERDI, Samir**; 200 Bellevue Parkway, Suite 300, Wilmington, DE 19809 (US).

(74) Agent: **SAMUELS, Steven, B.** et al.; Baker & Hostetler LLP, Cira Centre, 12th Floor, 2929 Arch Street, Philadelphia, PA 19104-2891 (US).

(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DJ, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IR, IS, JO, JP, KE, KG, KH, KN, KP, KR, KW, KZ, LA, LC, LK, LR, LS, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SA, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(54) Title: UNIFIED AUTHENTICATION FRAMEWORK

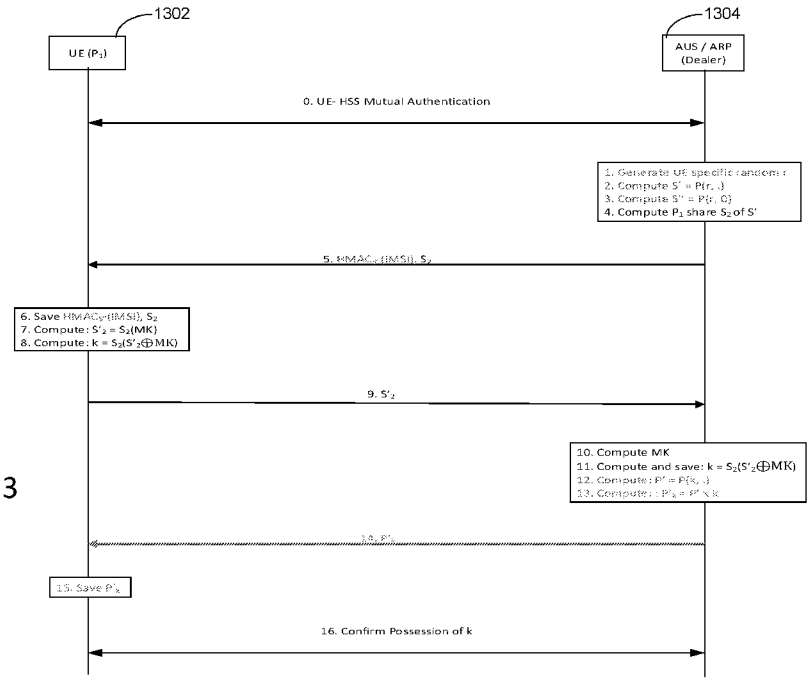


Fig. 13

(57) Abstract: A unified authentication framework for NextGen (5G) systems is described herein. In an example, static Security Associations between an AAA Server (HLR HSS in EPS) and Authenticator (MME in EPS) are replaced with flexible Security Associations that are based on cryptographic properties, which allow for better adaptability to 5G use cases. Further, the flexible security associations provide enhanced resistance to attacks that target shared secrets, authentication vectors, or the like. Further still, the flexible security associations provide an enhanced ability to recover from such attacks.

WO 2018/053271 A1

[Continued on next page]

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

Published:

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

UNIFIED AUTHENTICATION FRAMEWORK

CROSS REFERENCE TO RELATED APPLICATIONS

[0001] This application claims the benefit of U.S. Provisional Patent Application Serial No. 62/395,889, filed September 16, 2016, the contents of which are hereby incorporated by reference in its entirety.

BACKGROUND

[0002] Security of data shared between two parties typically relies on the use of cryptographic secrets – either shared between two parties using symmetric key cryptography or through a proof of possession of independent secrets using asymmetric key cryptography. The security of the data then relies on the security of the secrets or keys that protect the data. These keys may be compromised through side-channel or direct attack on the devices, captured during communications, or weak protection measures, which results in also compromising the security of the data. When these keys are used as long term keys, a compromise of the keys may result in a disastrous outcome and sometimes recovery may not be possible. It is therefore recognized herein that existing approaches to security architectures may be too rigid and may lack robust recovery capabilities.

[0003] An example current 3GPP authentication architecture 99 is shown in Fig. 1. The authentication architecture 99 is sufficient for the uses cases in a typical 2G to 4G 3GPP system. The Next Generation (NextGen) or 5G system is under development in 3GPP. New and enhanced use cases may be realized by 5G systems, which may require a flexible security architecture and authentication framework. Current security architecture and authentication framework proposals follow the Extensible Authentication Protocol (EAP) model, which is based on the example security architecture 200 shown in Fig. 2. In some cases, as shown in Fig. 3, current approaches require passing root keys/keying material from an AAA server 302 to an EAP authenticator 304.

SUMMARY

[0004] A unified authentication framework for NextGen (5G) systems is described herein. By way of an example, various security solutions for robust and secure recovery from compromised

keys used for data protection for the emerging 5G wireless security systems are described herein. In an example, static Security Associations between an AAA Server (HLR HSS in EPS) and Authenticator (MME in EPS) are replaced with flexible Security Associations that are based on cryptographic properties, which allow for better adaptability to 5G use cases. Further, the flexible security associations provide enhanced resistance to attacks that target shared secrets, authentication vectors, or the like. Further still, the flexible security associations provide an enhanced ability to recover from such attacks.

[0005] In an example, an apparatus, for instance a user equipment (UE), receives a first cryptographic function computed by a network entity, and a first cryptographic parameter from the network entity. The UE computes a third cryptographic parameter using the first cryptographic function and a second cryptographic parameter that is locally generated at the UE. The UE sends the third cryptographic parameter to the network entity. The UE also computes a master key using the first cryptographic parameter and the second or third cryptographic parameter. The UE may verify that the network entity also correctly derived the master key. In some cases, the master key master key is isolated from any intermediary party between the UE and the network entity.

BRIEF DESCRIPTION OF THE DRAWINGS

[0006] Fig. 1 depicts an example of a current 3GPP authentication architecture.

[0007] Fig. 2 depicts an example of an Extensible Authentication Protocol (EAP) security architecture that allows for various EAP authentication methods.

[0008] Fig. 3 depicts an example of a generalized 3GPP security architecture, which shows examples of key transfers and key derivations.

[0009] Fig. 4 depicts an example of a (e.g., 5G) security architecture, which shows loose coupling between function elements, and a key derivation that replaces a key transfer, in accordance with an example embodiment.

[0010] Fig. 5 depicts an example of a high-level security architecture for NextGen systems.

[0011] Figs. 6A and 6B depict an example of a 5G network architecture.

[0012] Fig. 7 shows an example list of Common Service Functions (CSFs).

[0013] Figs. 8A-C show an example of a distributed security architecture with slices.

[0014] Fig. 9 shows an example of a NextGen Authentication Reference Framework.

[0015] Fig. 10 is a high level flow diagram that depicts an authentication, security context transfer, and key generation process in accordance with an example embodiment.

[0016] Fig. 11 is a call flow that shows credentials being generated and key possession being verified in accordance with an example embodiment.

[0017] Fig. 12 is a call flow for providing moderate isolation of credentials in accordance with an example embodiment.

[0018] Fig. 13 is a call flow for providing complete isolation of credentials in accordance with an example embodiment.

[0019] Fig. 14 is a call flow for security anchor function (SEA) provisioning in accordance with an example embodiment.

[0020] Fig. 15 is a call flow for computing an SEA key (K_{SEA}) in accordance with an example embodiment.

[0021] Fig. 16 is a call flow for pre-provisioning for complete isolation and recovery in accordance with an example embodiment.

[0022] Fig. 17 is a call flow for recovery and verifying key possession in accordance with an example embodiment.

[0023] Fig. 18 is a call flow for recovery and SEA provisioning in accordance with an example embodiment.

[0024] Fig. 19 is a call flow for the computation of the K_{SEA} in accordance with an example embodiment.

[0025] Figs. 20A and 20B is a call flow for a recovery from a K_{SEA} hack in accordance with an example embodiment.

[0026] Fig. 21 illustrates an example of a non-roaming scenario with various example slice functions.

[0027] Fig. 22 shows a hierarchical credential generation in accordance with an example embodiment.

[0028] Fig. 23 shows a partially hierarchical credential generation in accordance with an example embodiment.

[0029] Fig. 24 shows another partially hierarchical credential generation with separate access network (AN) and core network (CN) credential generation anchors in accordance with an example embodiment.

[0030] Fig. 25 is call flow for a tracking update handover in accordance with an example embodiment.

[0031] Fig. 26 depicts an example multi-slice access or handoff between slices.

[0032] Fig. 27 is a call flow for handoff between slices or accesses to slices simultaneously, in accordance with an example embodiment.

[0033] Fig. 28 depicts an example over-the-top (OTT) scenario.

[0034] Fig. 29 depicts various example levels of isolation and their relationship to each other.

[0035] Fig. 30 shows an example of a generation of keying material when handing-off between slices or during instantiation of a new slice, in accordance with an example embodiment.

[0036] Fig. 31A is a system diagram illustrating an example communications system in which one or more disclosed embodiments may be implemented.

[0037] Fig. 31B is a system diagram illustrating an example wireless transmit/receive unit (WTRU) that may be used within the communications system illustrated in Fig. 31A according to an embodiment.

[0038] Fig. 31C is a system diagram illustrating an example radio access network (RAN) and an example core network (CN) that may be used within the communications system illustrated in Fig. 31A according to an embodiment.

[0039] Fig. 31D is a system diagram illustrating a further example RAN and a further example CN that may be used within the communications system illustrated in Fig. 31A according to an embodiment.

[0040] Fig. 32 is a call flow for long-term key provisioning in accordance with an example embodiment.

[0041] Fig. 33 is a call flow for provisioning an SEA/MME with appropriate credentials in accordance with an example embodiment.

[0042] Fig. 34 is a call flow for a computation of K_{MME} / K_{SEA} in accordance with an example embodiment.

[0043] Fig. 35 is a call flow for provisioning certain parameters so that recovery can be performed after attacks where credential or keys are stolen, in accordance with an example embodiment.

[0044] Fig. 36 is a call flow for recovering from an AUS/ ARP (HSS) side hack in accordance with an example embodiment.

[0045] Fig. 37 is a call flow for of MME/SEA provisioning in accordance with an example embodiment.

[0046] Fig. 38 is a call flow a computation of K_{MME} / K_{SEA} for in accordance with an example embodiment.

[0047] Fig. 39 is a call flow for recovering from a key hack in accordance with an example embodiment.

DETAILED DESCRIPTION OF ILLUSTRATIVE EMBODIMENTS

[0048] Referring initially to Figs. 2 and 3, a Security Association (SA) established between the AAA Server 302 (e.g., home location register (HLR)/home subscriber server (HSS) in an evolved packet system (EPS)) and Authenticator 304 (e.g., mobile management entity (MME) in an EPS) is static, and based on contractual agreements between roaming mobile network operators (MNOs) or may be set up as a default in a non-roaming use case. It is recognized herein that such a rigid security architecture makes the communication system vulnerable to attacks targeting the shared secrets, authentication vectors, or the like. Its inelasticity also makes recovery from such attacks cumbersome. In addition, it is further recognized herein that extension of such an inflexible architecture into 5G use cases (e.g., exposing the security architecture to third parties for authentication of a user equipment (UE)) is difficult and likely to lead to an impractical 5G system.

[0049] It is recognized herein that replacement of the static Security Association (SA) between the AAA Server (e.g., HLR HSS in EPS) and Authenticator (e.g., MME in EPS) with a flexible SA based on cryptographic properties allows for better adaptability to 5G use cases, enhanced resistance to attacks targeting the shared secrets, authentication vectors, etc., and the ability to recover from such attacks.

[0050] Fig. 3 depicts an example current (2G through 4G) security architecture or system 300. At 301, a root key (e.g., K_{ASME} in an EPS) may be derived between a universal subscriber identity module (USIM) 306 of a UE 308 and the authenticator 304 (e.g., HSS/MME) without transferring the key over an air interface (e.g., Uu in EPS). Other keys may be transferred over dedicated secure channels. For example, at 303, the K_{ASME} may be sent from the AAA server 302 (e.g., HSS) to the authenticator 304 (e.g., MME) as part of an authentication vector. At 305, a root key for over-the-air interface (K_{ENB}) may be transferred from the authenticator 304 to a NodeB 310 (e.g., eNB). The K_{ENB} may also be transferred from a source eNB to a target eNB during S1 (vertical) or X2 (horizontal) handover. Handover key material for additional freshness may be transferred from the authenticator 304 (e.g., MME) to the target eNB at S1 (vertical) or X2 (horizontal) handover. In addition to the security issues presented by transferred keys, it is recognized herein that the system 300 defines rigid interfaces between entities that transfer keys between each other, thereby making such architectures inflexible. In some cases, rigid coupling between the AAA 302 (e.g., HSS) and the authenticator 304 (e.g., MME), and rigid coupling between the authenticator 304 (e.g., MME) and the NodeB 310 (e.g., eNB) makes the security architecture 300 inflexible and introduces security vulnerabilities related to transferring the keys and key material. It is recognized herein that the same rigidity of couplings can complicate recovery from security breaches caused by transferring the keys and key material.

[0051] Fig. 4 depicts an example next generation (NextGen or NG) or 5G security architecture or system 400, which is flexible to allow an on-demand association between the core network (CN) and the radio access network (RAN), in accordance with an example embodiment. The terms NextGen, NG, 5G, and new radio (NR) can be used interchangeably herein, without limitation, unless otherwise specified. The example security architecture 400 include a supplicant or NG-UE 402, a NodeB or gNB 404, an authenticator or security anchor 406, and an AAA server 408. It will be appreciated that the example system 400 is simplified to facilitate description of the disclosed subject matter and is not intended to limit the scope of this disclosure. Other devices, systems, and configurations may be used to implement the embodiments disclosed herein in addition to, or instead of, a system such as the system shown in Fig. 4, and all such embodiments are contemplated as within the scope of the present disclosure.

[0052] The system 400 provides for key derivation instead of transferring keys and key material between the AAA 408 and the Authenticator/Security Anchor 406, between the Authenticator/Security Anchor and NodeB/gNB, and during gNB/Security Anchor handovers.

[0053] Referring now to Fig. 5, various entities may be included in a NextGen security architecture, such as the example architecture 400. For example, the architecture may include an Authentication Credential Repository and Processing Function (ARP) 502 (which may also be referred to as an Unified Data Management (UDM)), an Authentication Server Function (AUSF) 504, a Security Anchor Function (SEAF) 506, and a Security Context Management (SCM) Function 508. In an example, the SCM 508 may be co-located with the SEAF 506. The terms MME and SEAF are used interchangeably herein, without limitation, and the associated keys K_{ASME} and K_{SEA} may be used interchangeably as well, without limitation, unless otherwise specified. The terms UDM and ARP/ARPF may also be used interchangeably herein, without limitation, unless otherwise specified.

[0054] Referring also to Fig. 8, the Authentication Credential Repository and Processing Function (ARP) 502 (also referred to as the UDM) may store long-term security credentials used in authentication, and may execute any cryptographic algorithms that use the long-term security credentials as input. The ARP/UDM may also store a subscriber profile 501, for instance the security related portion of the subscriber profile 501. The ARP 502 may reside in a secure environment in an operator's Home Network or a third party system. The ARP 502 may interact with the AUSF 504.

[0055] In an example, the ARP 502 may store long term-security credentials that include shared permanent secrets such as the key K in EPS AKA or EAP-AKA, EPS-AKA*, EAP-AKA* as well as public/private key pairs (e.g., in EAP-TLS). In an example case of AKA-based

authentication (e.g., EPS AKA or EAP-AKA or EAP-AKA'), the ARP 502 generates the authentication vectors. The 4G-equivalent of the reference point between an access point (AP) and the AUS in EPS may be SWx for non-3GPP access to the EPC. In some cases, the reference point is undefined for 3GPP access to the evolved packet core (EPC) as it may be internal to the HSS.

[0056] The Authentication Server Function (AUS) 504 may interact with the ARP 502 and the SEA 506. The AUS 504 may reside on a location in an operator's Home Network or a third party system that is not exposed to physical access by an attacker. In an example in which EAP-based authentication is performed, the AUS 504 may perform the function of the EAP server. In the case of EPS AKA, the AUS 504 may be part of the HSS and may forward messages between the SEA 506 and the ARP 502.

[0057] In some cases, the Security Anchor Function (SEA) 506 is an authentication function in the core network that interacts with the AUS 504 and the NG-UE. The SEA 506 may receive, from the AUS 504, the intermediate key (e.g., session keys) that was established as a result of the NG-UE authentication process. The SEA 506 may also interact with the Mobility Management (MM) function (also referred to as an Access and Mobility Management Function (AMF)) (e.g., during initial Attach) and with the SCM 508. The SEA 506 may reside at a location in an operator's network, which is not exposed to physical access by an attacker. In the roaming case, the SEA 506 may reside in the visited network. If the SEA 506 resides in the visited network, in some cases, the intermediate key is specific to the visited network.

[0058] In an example of EPS AKA, the SEA 506 receives the intermediate key K_{ASME} from the AUS 504. In an example case of EAP-based authentication, the SEA 506 takes the authenticator role from the point of view of the UE and the AUS 504, and receives the intermediate key MSK (which may be similar to K_{ASME} or K_{SEA}) from the AUS 504.

[0059] The Security Context Management Function (SCM) 508 may receive a key from the SEA 506 that it uses to derive further (e.g., access-network specific) keys. The SCM 508 may reside at a location in an operator's network, which is not exposed to physical access by an attacker. In an example roaming case, the SCM 508 resides in the visited network.

[0060] Description of cryptographic mechanisms is now presented. Compartmented Threshold Secret Sharing (CTSS) is a secret sharing scheme that divides a secret into multiple shares by a dealer. The shares are shared among shareholders in such a way that any authorized subset of share-holders can reconstruct the secret; whereas any un-authorized subset of share-holders cannot recover the secret. If the maximal length of shares is equal to the length of the secret in a secret sharing scheme, the scheme is called ideal. If the shares corresponding to each

unauthorized subset provide no information, in the information-theoretic sense, the scheme is called perfect. Shamir proposed the first (t, n) threshold secret sharing scheme, which is ideal and perfect.

[0061] Hierarchical threshold secret sharing scheme is a generalization of simple threshold secret sharing scheme, and it has been studied extensively. In a hierarchical threshold secret sharing scheme, the shareholders play different roles; while in a simple threshold secret sharing scheme the shareholders play the same role. The proposed solutions described herein may use a variation of Hierarchical threshold secret sharing called Compartmented Threshold Secret Sharing (CTSS). In the CTSS scheme, when there are: $t \geq \{\sum_{mj=1} t_j\}$ shareholders in total and these shareholders satisfy the requirement that there are at least t_i shareholders for each compartment C_i, for i = 1, …, m. Such combination of shareholders can reconstruct the secret. There are multiple efficient and powerful CTSS constructions available. Depending on the requirements any CTSS can be used in embodiments described herein.

[0062] Turning now to key homomorphic PRFs, a pseudorandom function $F : K \times X \rightarrow Y$ is said to be key homomorphic if given F(k1, x) and F(k2, x) there is an efficient algorithm to compute $F(k1 \oplus k2, x)$, where $\oplus$ denotes a group operation on k1 and k2 such as XOR. Key homomorphic PRFs are natural objects to study and have a number of interesting applications: they can simplify the process of rotating encryption keys for encrypted data stored in the cloud, they give one round distributed PRFs, and they can be the basis of a symmetric-key proxy re-encryption scheme. In an embodiment, the low-depth key homomorphic PRF construction is used.

[0063] Turning now to a secret key sharing scheme, a secret sharing scheme divides a secret into multiple shares by a dealer and shared among shareholders in such a way that any authorized subset of share-holders can reconstruct the secret; whereas any un-authorized subset of share-holders cannot recover the secret. If the maximal length of shares is equal to the length of the secret in a secret sharing scheme, the scheme is called ideal. If the shares corresponding to each unauthorized subset provide absolutely no information, in the information-theoretic sense, the scheme is called perfect. Shamir proposed the first (t, n) threshold secret sharing scheme, which is ideal and perfect. The proposed solutions use Compartmented Threshold Secret Sharing (CTSS) scheme. There are multiple efficient and powerful CTSS constructions available. Depending on the requirements, any CTSS scheme can be used in embodiments described herein.

[0064] A new CTSS scheme is described herein. The scheme can be referred to as 2-CTSS, and is based on one time pad's principle. In some cases, the scheme allows the dealer to efficiently compute two shares of polynomials and secrets. For example, assume the desired secret to be shared is an univariate polynomial, S' = P(x, ·). The dealer generates a random polynomial, S1 = P'(x, ·) with nonzero, random coefficients in place of the nonzero coefficients in S'. The two

shares of the secret S' are: S1 and S2 = S' – S1. Thus, 2-CTSS may allow the dealer to compute shares for at least two levels of hierarchy, which can be iterated to distribute shares members divided into any number of hierarchical structure.

[0065] Various embodiments described herein may be characterized as having limited isolation, intermediate isolation, or complete isolation. In a limited isolation with authorization scenario, when deriving security parameters (e.g., encryption keys) between two entities, no intermediate entity has enough information to derive the parameters without authorization or without an active and/or passive attack. In an intermediate isolation scenario, when deriving security parameters (e.g., encryption keys) between two entities, no intermediate party has enough information to derive the parameters without an active and/or passive attack. In a complete isolation scenario, when deriving security parameters (e.g., encryption keys) between two entities, no entity can impersonate any other entity and/or glean any information that it does not know or is not authorized to know (even by launching an active Man In-The-Middle (MITM) attack on the communication between any two parties).

[0066] Referring now to Figs. 6A and 6B, an example architecture or system 600 is shown, which includes example core security functions 601 that may be part of the 5G Core Security functions shared by the sub-network, services, or slices. As used herein, the terms sub-network, Network Slice Instance (NSI), and slices may be used interchangeably, without limitation, unless otherwise specified. Services may be realized by means of a NSI or by one or more slices 602. In some cases, the core functions may be shared among slices. In some cases, functions may be over-ridden by using a dedicated function within a slice, and in some cases, replicated within a sub-network or enhanced with an additional function within the slice. As shown in Figs. 6A and 6B, one or more UEs 604 may connect to a 5G network 603 initially using a Basic Connectivity Slice (BCS) 602a that provides the UEs with appropriate radio access (via a RAN 605) and functions in a core network 607. The connection provided by the BCS 602a may allow the UEs 604 to be identified and/or authenticated by the operator, and authorized by the operator. The BCS 602a may also allow the operator to provide one or more emergency services 608 to the UEs 604.

[0067] In some cases, based upon a combination of identification, authentication, and/or authorization procedures that may have been performed, a Slice Selection Function (SSF) 606 may provide a given UE 604 with access to one or more slices 601. The request for one or more slices 601 may be made by the UE 604 or may be based upon the profile of the UE 604 that is stored within the operator network. As used herein, the term UE may also refer to applications within the UE that may be used and triggered by a human user or by an automated software/hardware process within the UE. In some cases, the process by which the appropriate selection of the service/slice is

performed may be based upon a profile of the service/application/slice, which is referred to as the Service Description Document (SDD). The SDD may describe the characteristics of the slice. The characteristics may include, but not limited to, security and privacy offered by the slice, quality of service, duration of the service and certain cryptographic parameters that provides integrity, and authenticity and confidentiality of the SDD.

[0068] Still referring to Figs. 6A and 6B, slice 0 is the Default Slice (DS) 601a. As part of the DS 601a, a given UE 604 may be provided with the ability to access messaging, voice, and web services. Slices 601b, 601c, and 601d (Slices 1-3) are slices that may correspond to the use-cases for supporting eMBB, IoT (aka MIoT), and URLL (aka URLLC), respectively. As shown, slices 601a-d (Slices 0-3) may use the services of Common Control Functions (CCF) 610. The CCF 610 may consist of various functions, such as Mobility Management (MM) and Session Management (SM), which are offered by the operator. In accordance with the illustrated example, slices 601e and 601f (Slices 4 and 5) do not use the functions within the CCF 610. Similarly, slice 601f (Slice 5) does not use the Common Security Functions (CSF) 612. Custom security function(s) (CuSF) may be implemented by slice 601f. The CuSF may use a completely independent security function 601 that is not available within the CSF 612. The independent security function 601 may be managed and operated within the slice 601, or it may be a completely independent instance of a function within the CSF 612. In some cases, the Access Network (AN) 605 may be shared by the slices 601, for instance all the slices 601, and the CSF 612. In certain cases, the AN 605 may be shared while the CN 607 may be sliced and isolated from other CN slices. In other cases, the CN 607 may be shared while the AN 605 is sliced and isolated from other AN slices. Described herein are examples in which both the CN and AN are sliced, though it will be understood that that the embodiments described herein may apply to alternative slicing arrangements.

[0069] An example list of Common Security Functions 612 that may be available to the entire 5G network or to a subset of Security Functions that are provided by the operator and/or per slice is depicted in Fig. 7. Fig. 7 provides an example list of CSFs 612. As discussed earlier, a subset or entirely new security function may be instantiated within a slice that may depend upon the functions within the CSF 612, or that may be completely independent of the functionalities within the CSF 612.

[0070] In an example, the CSFs 612 are split into functions that may be distributed within the 5G network based upon the nature of the deployment (e.g., roaming in case of mobile while non-roaming in case of certain IoT use-cases) and for the sake of optimization and performance (e.g., reducing handoff latencies), certain functions may be implemented closer to the AN, while certain critical functions (e.g., authentication functions) may be deployed within the CN in a highly

secure manner. Using the high-level deployment model depicted in TR 23.799, Figs. 8A-C shows that some of the security functions may be deployed at the same time incorporating the notion of slices.

[0071] Referring to Figs. 8A-C, various example security related functions from Fig. 7 are described, and example logical placements within an example network 800 are shown. Figs. 8A-C depict the architecture shown in Fig. 5, with the notion of slices. Referring to Figs. 8A-C, in addition to slices being a sub-network in itself, some of the slicing may be performed across the CN, AN, Control Plane (CP), and the User Plane (UP). In an example embodiment, a slice is made up of CN, AN, CP, and UP functions that forms a logical self-contained network. Similar principles may be applied for the following scenarios as well, presented by way of example and without limitation. In example Scenario 1a, slice 801a is self-contained consisting of CP, UP functions in the CN and AN. Security Context (SC) transfer is hierarchical. In example Scenario 1b, slice 801b is self-contained and SC transfer is flat. In example Scenario 2, the CN is sliced in slice 801c, and the AN is shared. In example Scenario 3, the CN is shared, and the AN is sliced in slice 801d. In example Scenario 4, the CP is sliced in slice 801e, and the UP is shared. In example Scenario 5, the CP is shared and the UP is sliced in slice 801f.

[0072] As shown, the Security Context (SC) can be transferred from a higher entity to a lower entity in the hierarchy. In some cases, after an authentication is carried out by the AUS 504, a primary SC is created that describes the security association between the given UE and the AUS 504. The primary SC may also include the various security parameters (e.g., protocols, algorithms, key length etc.) that may be used in order to secure the UE communications with each of the different network/slice entities. The SC may be transferred in a hierarchical manner from the AUS 504 to the SEA 506 to the SCM 508 to the various network functions within the slice. At each of the functions within the hierarchy, in some examples, only the relevant information within the SC is transferred in order to maintain security of the UE with its home network, maintain security in general, and to maintain the privacy of the UE when roaming outside the home network. The following illustrated security contexts are defined herein:

- SC_P: This is the primary SC that may be created based upon an identification and authentication of a UE by the AUS 504. The SC_P may consist of the relevant information associated with the UE. In some cases, the primary SC is only shared between the AUS 504 and the UE.
- SC_{SEA}: The SC_{SEA} may describe the security parameters that the AUS 504 deemed to be necessary in order that a security association between the UE and the SEA 506 can be

established. In addition, it may also contain parameters that can be used in order to create security associations between the UE and other security functions within the hierarchy.

- SC_{SCM}: Similar to the SC_{SEA}, the SC_{SCM} may contain security parameters that may be used in order to create the security association between the UE and the SCM 508, and therefore, in some cases, the SC_{SCM} may contain only the relevant security parameters and information transferred by the SEA 506 to the SCM 508.
- Similar to the aforementioned security contexts, $SC_{CP\text{-}CN}$
- $SC_{CP\text{-}AN}$
- $SC_{UP\text{-}CN}$
- $SC_{UP\text{-}AN}$:may contain only the relevant security parameters and information transferred from their respective associated higher-up entity in the hierarchy.

[0073] Table 1 below depicts an example SC that may be created based upon an identification, authentication, and an authorization procedure carried out between the UE and the AUS 504 using any one of the authentication protocols, such as EPS-AKA, EAP-AKA', EPS-AKA*, EAP-AKA* Open-Id etc. The example in Table 1 focuses on EAP-AKA' for the purposes of illustration, though it will be understood that alternative authentication protocols may be used.

Table 1: An Example Security Context Primary

Security Context Primary (SC_P)			
Parameters	**Detailed params**	**Value**	**Description**
UE-Id			- Identifies the UE, more particularly the IMEI
IMSI			- Identifies the subscription (optional to preserve privacy)
TMSI			- Identifies the temporary Id, which is derived
Slice-Id(s)			- Identifies the slice(s) assigned to the UE / application
Auth Protocol		EAP	- Identifies the main authentication protocol that was used to authenticate the UE (e.g. EAP, EPS, AKA*, TLS)
Access Network Info			- Stores the identity of the AN, used in the EAP-AKA', EAP-AKA* etc.
Authentication Method(s)		AKA'	- The Authentication method (e.g. AKA', TLS); also identifies the credential-type (e.g. symmetric key, public key, password etc.)
Credential-Id		34352...	- Randomly generated credential identity, that uniquely identifies the

			CK|IK. The Credential itself is stored in a secure element (e.g. TEE)
Security Protection(s) per Slice	Isol_val	Low	- Determines the amount of credential isolation that is required (e.g. None, Low, Med, High)
	Recovery	No	- Indicates if LTK leak recovery capability should be enabled or not
	CP Protection - Authenticity	High	- Authenticity protection for Control Plane (None, Low-High)
	CP Protection - Confidentiality	Low	- Confidentiality protection for Control Plane (None, Low-High)
	AN-UP Protection - Authenticity	High	- Authenticity protection for AN User Plane (None, Low-High)
	AN-UP Protection - Confidentiality	High	- Confidentiality protection for AN User Plane (None, Low-High)
	CN-UP Protection - Authenticity	Low	- Authenticity protection for CN User Plane (None, Low-High)
	CN-UP Protection - Confidentiality	High	- Confidentiality protection for CN User Plane (None, Low-High)
	Non-Repudiation	None	- Indicates if Non-repudiation is provided
	Multi-factor Authentication	Yes	- Indicates whether a user authentication has to be carried out in addition to subscriber authentication
	Bootstrapping	No	- Indicates whether Bootstrapping of credentials is done using: - Network Credentials (e.g. USIM subscription credentials) to derive Application / Service-specific credentials - Application / Service-level user or subscriber authentication credentials are used to derive Network-level credentials
	Binding of Authentication	N/A	- Identifies if the authentication and associated credentials are bound together
	Credential Generation Details[]		Details the key generation process and is a vector of parameters that are to be used: algorithm (e.g. PRF), Credential-Id, Nonce, Label etc..
	Privacy / slice / domain	Yes	- Provides guidance on the level of privacy that is required of the subscriber and User at CN-level, AN-

			level; Roaming and Non-roaming scenario
SEAF-Id		SEAF@op1.com	- Provides the identity of the SEAF
SCM-Id		SCM@op2.com	- Provides the identity of the SCM (it could be in a roaming network)
CN-Anchor-Id		CN-Anc@op2.com	- Provides the identity of the anchor within the CN
AN-Anchor-Id		AN-Anc@op2.com	- Provides the identity of the anchor within the AN
VNFs within the slice	Optional	{VNF1@op1.com, …, VNFn@op1.com}	- Lists the network functions that form part of the slice

[0074] Another example NextGen authentication reference architecture or system 900 is shown in Fig. 9. In an example authentication, the SEA 506 authenticates a UE 902 by exchanging AU1 messages via an NG1 interface. In order to authenticate the UE 902, the SEA 506 interacts with the AUS 504 to obtaining or derive keys and authentication material. Upon successful authentication, the SEA 506 may provide the SCM 508 with access-independent Security Context (SC) over an SC1 interface. In some cases, the SCM 508 derives access specific SC and sends it to the AN for CP and/or UP protection (e.g., confidentiality, integrity). The SC may be sent via SC4.x messages over an NG2 interface. The SC may be provided to CP and/or UP functions in the AN 905, depending on the access network (3GPP, WLAN, etc.) for example. In an example case where the SCM 508 is only able to send to the CP-AN (e.g. due to AN design), the CP-AN may derive key material in the SC and send it to UP-AN instead. The SCM 508 may send SC via SC3 messages to UP-CN function for UP data protection in the core network. SC3 messages may be sent over the NG4 interface. In an example, the SCM 508 may send the SC to an NAS Proxy 904 via SC2 as a single point for all NAS messaging protection establishment. The SCM 508 may be co-located with SEA 506.

[0075] In an example, an NAS Proxy function 904 serves as a common front end for NAS messages from the UE 902 sent via NG1 interface. These messages may be authentication (AU1), MM, or SM related and may be routed respectively to the appropriate SEA 506, CP-MM 906, or CP-SM 908 functions. In some case, the NAS Proxy 904 may be co-located with the CP-MM 906.

[0076] Referring now to Fig. 10, the EPS-AKA process is enhanced by including the ability for intermediary entities (e.g., SEA, SCM, CN, AN) to be able to generate the credentials autonomously using keying material provided by a higher-level function in the hierarchy. In Fig. 10, an example system 1000 includes a UE 1002, an AN 1004, a CN 1006, the SCM 508, the SEA 506, the AUS 504, and the ARP 502. It will be appreciated that the example system 1000 is

simplified to facilitate description of the disclosed subject matter and is not intended to limit the scope of this disclosure. Other devices, systems, and configurations may be used to implement the embodiments disclosed herein in addition to, or instead of, a system such as the system shown in Fig. 10, and all such embodiments are contemplated as within the scope of the present disclosure.

[0077] At 1, in accordance with the illustrated example, the UE 1002 requests access to a slice/service by optionally including an SDD in a request that is sent to the CN 1006. At 2, the UE 1002 and the AUS 1004 may perform an authentication process using an enhanced EPS-AKA process (e.g., EPS-AKA', EPS-AKA* etc.). In some cases, the authentication may be skipped based upon the slice that was requested or if the profile of the UE 1002 does not require the UE 1002 to be authenticated. The authentication process may also be carried out using EAP-AKA', Open-Id, IPSec, TLS, by way of examples. The SEA 506 performs the role of an authenticator in the home visited network or as an authentication proxy (e.g., Authentication Authorization and Accounting in Visited network: AAAv) within the visited network (at 2a). At 2b, the subscription profile is fetched from the ARP 502. The subscriber identity may be an IMSI or a Subscriber Permanent Identifier (SUPI) that is used to identify a symmetric key or a public key identifier or certificate of the UE 1002. In some cases, the subscriber identity may be replaced by a user identity that is used to identify a user. Once a successful authentication is carried out, or is generated based on the risk associated with the slice that was requested and generates an SCP, which may be then communicated to the UE 1002. Alternatively, a UE may be able to generate an SCP on its own based on the negotiations and parameters generated based on the identification and authentication process that was carried out at 2.

[0078] At 3, in accordance with the illustrated example, the AUS 504 generates keying material using mechanisms that may have been pre-agreed with the UE 1002. The AUS 504 may also generate the SC_{SEA}, and communicate the keying material and the SC_{SEA} to the SEA 506. Keying material may be the credential itself, or preferably in some cases, cryptographic material which can then be used to generate required credential(s) (Key(s)). At 4, using the SC_{SEA}, the SEA 506 and the UE 1002 may perform an authentication or verify key possession, as further described herein. A further set of credentials may be generated.

[0079] At 5, the SEA 506 may provide appropriate keying material and appropriate context (SC_{SCM}) to the SCM 508. At 6, using the SC_{SCM}, the SCM 508 and the UE 1002 may perform an authentication or verify key possession, as described further herein. A further set of credentials may be generated, and the SCM 508 may generate the appropriate context for the CN 1006 (SC_{CN}). Additionally, the UE 1002 may be provisioned with an IP address based upon an implicit or an explicit request from the UE 1002. An explicit IP address request may be performed

by means of a DHCP request issued by the UE 1002. In certain cases, the DHCP request may be piggybacked in the initial message from the UE1 1002 to the SCM 508 / SEA 506. More preferably, in other cases, the request may be an implicit one, wherein the SCM 508 / SEA 506, upon indication of successful authentication of the UE 1002, may obtain an IP address on behalf of the UE 1002, which is then provisioned to the UE1 1002 at step 6. Alternatively, the security context from SCM may be provided to a Session Management Function (SMF), which may be in control of the DHCP IP address allocation procedure. At 7, the SEA 506 may provide appropriate keying material and appropriate context (SC_{CN}) to the CN 1006. Optionally, the allocated IP address may be transported during Step 7. At 8, appropriate entities within the CN 1006 and the UE 1002 may generate one or more key(s) to protect the link between the UE 1002 and the CN 1006. The keys that may be generated may be used for message authentication/integrity, message confidentiality. Other keys (e.g., key encryption key, etc.) may also be generated. The keys may be generated based on the keying material received by the CN 1006 from the SCM 508. At 9, the SCM 508 may provide keying material and context to the AN 1004. At 10, based on the keying material received from the SCM at 9, the AN and the UE may derive additional keys for protecting the communications between the UE 1002 and the AN 1004. The credentials generated at 10 may be used for message authenticity, message encryption, and key encryption, for example.

[0080] By way of further introduction, in various example embodiments, credentials (e.g., keys) are derived instead of transported. Further, credentails can be recovered from an attack that involves stealing (or compromise) of keys from a database or memory. Various attacks include an attack on a device database or memory, side-channel attack or attack during transit or communications, or the like. In particular various embodiments described herein leverage the capabilities of hardware anchored security mechanisms, such as secure and trusted execution environments, secure boot, immutable roots of trust, platform attestation, secure hardware keys, physically unclonable functions, and the like. The security mechanisms extend to recovering from such a compromise of the derived keys.

[0081] In an example, the HSS can provision information to the parties so that each party can use their own secret values to compute the keys. Isolation can thus be provided. For example, in some cases, if a key is required to be known only to two parties, then no other party (including the dealer) can gain any information about the key. Each shareholder party may contain its own (possibly random) Master Secret securely stored (e.g., in hardware), which enables key derivation and isolation. Pseudo Random Permutation (e.g., AES), Key Homomorphic (secure) Pseudo Random Function (sPRF) and Hierarchical Secret Sharing is used in accordance with various embodiments. Hierarchical secret sharing is an extension of secret sharing schemes. Multiple,

provably secure hierarchical secret sharing constructions have been developed. Some proposed schemes may use a specific class of Hierarchical secret sharing, called Compartmented Threshold Secret Sharing (CTSS). CTSS scheme may require at least ti parties from level i to participate in secret reconstruction. Key Homomorphic PRFs may allow outputs of multiple PRFs to be combined such that the result reveals no extra information, other than the final output. At least two provably secure Key Homomorphic PRF constructions are known.

[0082] With respect to recovering from a database compromise, an extended version of the scheme described herein uses the device specific Master Secrets to enable recovery from dealer database hacks. In an example, it is assumed that keys and information in a dealer database are vulnerable to a database hack. In may also be assumed, in some cases, that each shareholder party master key is securely stored and cannot be stolen. In various examples, values derived from device specific Master Secrets are used in conjunction with a secure hash function to perform authentication and secure communication, even after the database has been compromised.

[0083] In some examples, the cryptographic schemes used herein rely on the use of Z*5 (Abelian group of integers modulo 5) as finite field Fp, but it will be understood that it can be a large Galois field, and that Z*5 may be used to simplify the examples. The degree, t-1, of the polynomial used in secret sharing scheme governs the threshold number (t) of parties required to reconstruct the secret. As used herein, unless otherwise specified, the Dealer/AUSF/ARPF/HSS may refer to the same or similar functionality, and can exist on a single entity. In an example, the Dealer Function may be a third-party.

[0084] Referring generally to Figs. 11-20, 25, 27, and 32-29, example call flows are shown. Various functions, devices, network entities, systems, and configurations are shown that perform the example call flows. It will be appreciated that the example call flows are simplified to facilitate description of the disclosed subject matter and is not intended to limit the scope of this disclosure. Other functions, devices, network entities, systems, and configurations may be used to implement the embodiments disclosed herein in addition to, or instead of, a system such as the system shown in Figs. 11-20, 25, 27, and 32-29, and all such embodiments are contemplated as within the scope of the present disclosure.

[0085] Turning now to Fig. 11, credentials are generated and key possession is verified. With reference to Fig. 11, in accordance with the illustrated embodiment, at 0, a first function 1106 (Function A) (e.g., AUS, SEA, SCM, CN) provisions keying material to a second function 1104 (Function B) based on a prior authentication between a UE 1102 and the Function A. As described above, keying material may include one or more credentials or parameters that may be used to derive one or more credentials. In addition to keying material, one or more security contexts may

also be transferred. In some cases, it may be assumed that the UE 1102 and the Function A may use security context that provides information on the appropriate cryptographic algorithm, same key length, and other cryptographic parameters in order to be able to derive intermediate keys. Function B may be, for example, the SEA 506, the SCM 508, an anchor function in the CN, or an anchor function within the AN. The credential generation process may be initiated by the UE 1102 or the Function B. In this example, the process is initiated by the UE 1102. At 1, the UE 1102 may generate a random value (Nonce1), and using an intermediate key based on certain key material, the UE 1102 may generate a Message Authentication Code (MAC) on the message that it intends to send to Function B. At 2, in accordance with the illustrated example, the UE 1102 includes the message, Nonce1, and the MAC and sends it to Function B.

[0086] At 3, the Function B, using the Nonce1, the message, and the intermediate key generated based on the keying material, authenticates the MAC. If the MAC is a match, then the Function B may generate a second random value (Nonce2). Function B may use the Nonce2 and optionally Nonce1 along with the intermediate key to generate a full set of keys used for integrity protection (MAC key), confidentiality protection for both uplink and downlink traffic, key encryption key (KEK - optional), group key (GK - optional), and an Extended Master Session Key (EMSK - optional). A pseudo-random generation function (PRF: e.g., HMAC-SHA-256) may be used for key generation and key expansion. Function B may then generate a message that includes the Nonce2, and generate a MAC using the message and the Nonce2. At 4, Function B sends the message along with Nonce2 and the MAC to the UE 1102. At 5, the UE 1102 may verify the MAC2 generated by the Function B and authenticate the Function B and then may follow a similar process as Function B performs in order to generate the full key set, which may include Nonce1, Nonce2, and Intermediate key to generate integrity and confidentiality protection keys for both uplink and downlink, as well as optional KEK, GK and EMSK.

[0087] In an example intermediate isolation embodiment, referring to Fig. 12, it is assumed that a supplicant (S) 1202 and an authentication server (AS) 1206 have a pre-decided secret K. One example of a supplicant is a user equipment having connection manager software that enables connection with a network. It may also be assumed that each entity can generate *n* bit long random numbers.

[0088] Referring to Fig. 12, in accordance with the illustrated embodiment, at 1, the AS 1206 computes: $K1 = PRP_K(AP_{id} \oplus K)$. Here, PRP may be any Pseudo Random Permutation (for e.g., AES), and AP_{id} is the public id of an access point (AP) 1204. The AP 1204 may be any entity that performs an access control function to the network. Examples of AP 1204 may include a 5G SEAF/AMF, 4G MME, WiFi AP, Radio access network controller (e.g., eNB). One of the primary

roles of an AP may be to perform policy enforcement and more specifically enforcement of access control policies. The policies may have been pre-provisioned or provisioned as part of an authentication process by the AS/AUSF. At 2, the session key generation is bound to the public IDs of the AP 1204 and the supplicant 1202. The AS 1206 computes: $K2 = Prf_{K1}(AP_{id} \oplus S_{id})$. Here, S_{id} is the public ID of the supplicant 1202. At 3, the S 1202 sends K2 to the AP 1204. At 4, the AP 1204 saves the K2. At 5, the AP 1204 generates a random integer r1. At 6, the AP 1204 computes: $K' = K2 \oplus r1$. At 7, in accordance with the illustrated example, the AP 1204 sends K' to S 1202. At 8, because S 1202 possesses the shared secret K, it is able to compute: $K1 = PRP_K(AP_{id} \oplus K)$. At 9, S 1202 uses K1 and K to compute: $K3 = K1 \oplus K$. At 10, the S 1202 computes K2 by using K1 as: $K2 = Prf_{K1}(AP_{id} \oplus S_{id})$. At 11, the S 1202 verifies the K2 value sent by the AP 1204. At 12, in accordance with the illustrated example, S 1202 computes the random number generated by the AP 1204 as: $r1 = K' \oplus K2$. At 13, the S 1202 generates an *n* bits long random integer r2. At 14, the S 1202 combines its random number r2 with the AP's random number r1 as: $R = PRP_{r1}(K2 \oplus r2)$. At 15, the S 1292 binds its pubic ID with the public ID of the AP 1204, along with the shared secret K to session key generation by computing: $K^* = Prf_{K3}(AP_{id} \oplus S_{id}) \oplus r2$. At 16, the S 1202 computes the session key as: $SessKey = Prf_K(AP_{id} \oplus S_{id}) \oplus (r2 + r1)$. At 17, the S 1202 sends (R, K*) to the AP 1204. At 18, the AP 1204 uses its random number r1 to invert R and compute: $K2 \oplus r2 = PRP^{-1}_{r1}(K2 \oplus r2)$. At 19, the AP 1204 computes r2 as: $r2 = K2 \oplus r2 \oplus K2$. At 20, the AP 1204 computes the pivotal key generation element as: $Prf_{K3}(AP_{id} \oplus S_{id}) = K^* \oplus r2$. At 21, the AP 1204 computes: $Prf_{K3}(AP_{id} \oplus S_{id}) + K2 = Prf_{K \oplus K1}(AP_{id} \oplus S_{id}) = Prf_K(AP_{id} \oplus S_{id})$. At 22, the AP 1204 computes the session key as: $SessKey = Prf_K(AP_{id} \oplus S_{id}) \oplus (r2 + r1)$. At 23, the AP 1204 and the S 1202 may confirm that both parties have computed the correct session key (SessKey) value.

[0089] Described now are processes for complete isolation of credentials, in order that none of the functions are able to impersonate the other functions or the UE. The steps involved may include UE/AUSF/HSS provisioning and SEA/MME provisioning. In an example, each entity is allowed to embed a secret value into the key computation and mutual authentication process. Thus, it may become virtually impossible for a polynomial adversary to deduce any information about the keys.

[0090] Referring to Fig. 32, an example long-term key provisioning process is shown between a UE 3202 and an AUS/HSS/ARP 3204. It may be assumed that the UE 3202 has an immutable master secret, MS, which is securely stored. A derivative of MS called MK may be computed as: $MK = PRNG(MS \oplus T)$, where T is the current value of the UE's counter. The Dealer 3204 (AUS/ARP/HSS) may possess a secret bivariate polynomial (polynomial with two variables),

P(x1, x2), which is securely stored. For example, let P(x1, x2) = 2x12 + x22 + 3x1x2 + 4. Example parameters and their characteristics are provided in the table below:

Symbol	Security Parameters	Characteristics	Size	Description	Step
r S'	-Enables generation of UE specific polynomial -UE specific polynomial required to compute LTK and secondary key	-Pseudorandom integer -Pseudorandom share of original polynomial P	-|p| bits -|p| bits	-Derived by feeding UE's IMSI and a random element to PRNG -Derived by setting a variable in P to be equal to the UE specific pseudorandom integer r	1
S'_2	-To securely send MK to HSS	-Pseudorandom integer	-|p| bits	-Generated by using S_2 (UE's share of S') and MK (a pseudorandom integer derived from UE's master secret, MS)	5
P'_k	-Information to compute secondary key(s)	-Pseudorandom polynomial	-|p| bits	-Derived by using polynomial P and LTK k	7

[0091] Referring to Fig. 32, at 0, in accordance with the illustrated example, the UE 3202 and the dealer/HSS 3204 perform mutual authentication. At 1, the Dealer/HSS 3204 may generate UE specific security parameters. In an example, the HSS 3204 generates a UE specific random number r by using some private function f (e.g., a hash function), which belongs to a public function family F. For example, AUS may have a counter, C, initialized as a random integer and it may compute r as: $r = \text{Hash}_{C \oplus IMSI}(IMSI) \pmod p$. Let r be 3 for this example. The HSS may evaluate its secret bivariate polynomial $P(x_1, x_2)$ at $x_1 = r$. It hence computes: $S' = P(r, \cdot)$, which is a polynomial in x_2. For example, $P(x_1=3, x_2) = 18 + x_2^2 + 9x_2 + 4 \equiv x_2^2 + 4x_2 + 2 \pmod 5$. The HSS sets $x_2 = 0$ in S' to compute: $S'' = S'(0) = P(r, 0)$. For example, $P(x_1 = 3, x_2 = 0) \equiv 2 \pmod 5$. At 2, the HSS 3204 uses a hierarchal secret sharing scheme to compute two CTSS shares of S'. In an example, $t_1 = t_2 = 1$. The UEs, for instance the UE 3202, are placed at the second level for the hierarchal secret sharing, while the SEA/MMEs are placed at the first level. This may ensure that any number of malicious UEs cannot combine their shares to retrieve S' or any relevant information. The set containing the first and second level parties are called P_0 and P_1, respectively. HSS computes share S_2 of S' for the second level party, i.e., the UE 3202. For example, the Dealer 3204 splits $P(x_1, x_2) = 2x_1^2 + x_2^2 + 3x_1x_2 + 4$ into $P_1(x_1, x_2) = 4x_1^2 + 3x_2^2 + 4x_1x_2 + 1$ and $P_2(x_1, x_2) = 3x_1^2 + 3x_2^2 + 4x_1x_2 + 3$. The Dealer assigns P_1 to level 1 and P_2 to level 2. It then computes level 2 share of S' as: $S_2 = P_2(x_1=3, x_2) = 27 + 3x_2^2 + 12x_2 + 3 = 3x_2^2 + 12x_2 + 30 \equiv 3x_2^2 + 2x_2 \pmod 5$.

[0092] At 3, in accordance with the illustrated example, the Dealer 3204 (AUS/ARP/HSS) sends ($HMAC_{S''}$(IMSI), S_2) to the UE 3202. $HMAC_{S''}$(IMSI) is the HMAC of IMSI, computed by using S'' as the key, and may be required in order for the UE 3202 to authenticate itself to the MME and AUS/ARP. At 4, the UE 3202 may compute the LTK, k. The UE 3202 may save ($HMAC_{S''}$(IMSI), S_2). The UE 3202 may evaluate S_2 on MK, i.e. it sets x_2=MK to get: $S'_2 = S_2(MK)$. For example, if UE's MK is 2 then $S'_2 = 3*4 + 2*2 = 16 \equiv 1 \pmod 5$. The UE 3202 may compute its LTK k as: $k = S_2(S'_2 \oplus MK) = S_2(3) = 3*9 + 2*3 = 33 \equiv 3 \pmod 5$. At 5, the UE may sends S'_2 to the Dealer/AUS/ARP/HSS 3204. At 6, the Dealer/HSS 3204 computes the LTK, k. For example, the Dealer/AUS/ARP/HSS 3204 may receive S'_2 and, as it knows the secret polynomial P, it may compute MK from S'_2. The Dealer/HSS 3204 may compute and saves long term key/secret as: $k = S_2(S'_2 \oplus MK)$. At 7, the Dealer/HSS 3204 computes parameters required to compute the secondary key. The Dealer/HSS 3204 may evaluates its secret polynomial by setting x_1 = k to get: P' = P(k, ·). P' will be used for k_{MME} (secondary key) computation. For example, $P' = P(k, \cdot) = 2*9 + x_2^2 + 3*3x_2 + 4 \equiv x_2^2 + 4x_2 + 2 \pmod 5$. At 8, the Dealer/HSS 3204 may compute P'_k = P' x k. The computation may be performed in finite field F_p. For example, P'_k for $P' \equiv x_2^2 + 4x_2 + 2 \pmod 5$, and k = 3 will be: $3x_2^2 + 2x_2 + 1 \pmod 5$. At 7, the Dealer/HSS 3204 sends P'_k to the UE 3202, which saves P'_k. The UE 3202 can compute P' from P'_k by multiplying by k^{-1} which for k = 3 is 2. At 8, the UE 3202 and the Dealer/HSS 3204 may confirm that the other party has successfully computed the correct LTK.

[0093] Referring to Fig. 13, it is assumed that a UE 1302 has an immutable master secret (MS) that is securely stored. The UE 1302 may generate a master key (MK) from the immutable master secret. It is also assumed that an AUS/ARP 1304 (e.g., HSS) possesses a secret bivariate polynomial, P(x1, x2), which is securely stored.

[0094] In accordance with the illustrated embodiment, at 0, the UE 1302 and the AUS/ARP 1304 (HSS) perform mutual authentication. Thus, UE 1302 may perform an authentication with a network entity, such as the AUS/ARP 1304. In some cases, the UE 1302 may perform an authentication of the AUS/ARP 1304 based on a pre-provisioned certificate or public key associated with the AUS/ARP 1304. The AUS/AR may also transport its certificate or public key at 0. In certain cases, only a hash of the public key associated with the AUS/ARP 1304 may be computed and transported. The authentication may be performed by using AUS / ARP 1304 (HSS)'s raw public key or certificate which may have been pre-provisioned to the UE 1302. The UE 1302 may be authenticated by the AUS/ARP 1304 (HSS) by means of public credential (e.g. a device certificate associated with the UE 1302) or a pre-provisioned shared secret between the UE 1302 and the ASU / ARP 1304 to the which the AUS/ARP 1304 may have direct access.

Alternatively, the UE 1302 may be authenticated with a trusted third party (TTP), using public key or shared secret mechanisms and after which, the TTP generates and provides an assertion to the AUS / ARP 1304. The AUS/ARP 1304 may then verify the assertion (digitally signed) using appropriate credentials (e.g. shared secret between AUS/ARP 1304 or verifying the signature generated using the private key of the TTP). On successful verification of the assertion, the AUS/ARP 1304 deems the UE to have been successfully authenticated by the TTP with which it has a trust relationship. At 1, the AUS/ARP 1304 (HSS) may generate a UE specific random number r by using a private cryptographic function f (e.g., a hash function or a good pseudo-random function), which may belong to a public function family F. At 2, the AUS/ARP 1304 (HSS) evaluates its secret bivariate polynomial P(x1, x2) at x1 = r. It can compute: S' = P(r, ·), which is a polynomial in x2. At 3, in accordance with the illustrated example, the AUS/ARP 1304 (HSS) sets x2 = 0 in S' to compute: S'' = S'(0) = P(r, 0). At 4, the AUS/ARP 1304 (HSS) may use the above described 2-CTSS scheme to compute second level share of S'. UEs may form part of the second level share. This level placement may ensure that any number of malicious UEs cannot combine their shares to retrieve S' or any relevant information. The set containing the first and second level parties are called P0 and P1 herein, respectively. The AUS/ARP 1304 (HSS) computes share S_2 of S' for the second level party (e.g., the UE 1302).

[0095] Alternatively, the AUS/ARP 1304 (HSS) may use a hierarchal secret sharing scheme to compute a second level share of S' as described above and use it to derive the shares instead of the 2-CTSS, which can be thought of as a specialized form of secret sharing. In an example, UEs, for instance all UEs, are placed at the second level for the hierarchal secret sharing, while SEAs, for instance all SEAs, are placed at the first level.

[0096] At 5, in accordance with the illustrated example, the AUS/ARP 1304 (HSS) sends (HMACS''(IMSI), S_2) to the UE 1302. Thus, the UE 1302 may receive a first cryptographic function computed by the network entity and a first cryptographic parameter. In an example, the first cryptographic function, S_2, which is a share of the second level party, is used to generate a third cryptographic parameter. The HMACS''(IMSI) is a hash message authentication code (HMAC) of the IMSI of the UE 1302, computed by using S'', a cryptographic parameter supplied by the AUS/HSS as the key. At 6, the UE 1302 saves (HMACS''(IMSI), S_2). Thus, the UE 1302 may save the first cryptographic parameter. At 7, the UE 1302 evaluates S_2 on its master key (MK) to get: $S'_2 = S_2(MK)$. The MK may be derived by combining the MS with a randomly generated random number, r1. The MK is computed as MK = PRNG (MS$\oplus$ r1). Thus, the UE 1302 may compute a third cryptographic parameter (e.g., S'_2) using the cryptographic function and a second cryptographic parameter (e.g., MK) generated at the UE 1302. At 8, the UE 1302 computes its

LTK k, which may be referred to as a master key, as: $k = S_2(S'_2 \oplus MK)$. Thus, the UE 1302 may compute the master key using the first cryptographic parameter and the second or third cryptographic parameter. At 9, the UE 1302 sends S'_2 to the AUS/ARP 1304 (HSS). At 10, the AUS/ARP 1304 (HSS) receives S'_2 and, because it knows the secret polynomial P, it computes MK from S'_2. At 11, the AUS/ARP 1304 (HSS) computes and saves a long term key/secret as: $k = S_2(S'_2 \oplus MK)$. At 12, the AUS/ARP 1304 (HSS) evaluates its secret polynomial by setting x1 = k to get: $P' = P(k, \cdot)$. P' may then be used for K_{SEA} / K_{MME} / K_{ASME} computation. At 13, the AUS/ARP 1304 (HSS) computes: P'k = P' x k. The computation may be performed in finite field Fp. At 14, the AUS/ARP 1304 (HSS) sends P'k to the UE 1302, and the UE may save P'k (at 15). Thus, the UE 1302 may receive a second cryptographic parameter that is computed based on the third cryptographic parameter. At 16, the UE 1302 and the AUS/ARP 1304 (HSS) may confirm that the other party has successfully computed the correct LTK k. Thus, the UE 1302 may verify that the network entity also correctly derived the master key.

[0097] Confirming the possession of a key may entail that both the UE and the AUS perform a challenge response message, which may involve the generation of at least one message authentication code (MAC) or digital signatures (DS) independently by the UE and the AUS for their respective messages. The UE may verify the MAC/DS generated by the AUS for the message originated from the AUS, and the AUS may verify the MAC/DS generated by the UE for a message originating from the UE. In some examples, the only values saved by the UE 1302 are P'k, HMACS''(IMSI) and S_2. The UE may delete the values r1, r2 once the LTK has been computed.

[0098] Referring now to Fig. 33, an example of provisioning an SEA/MME 3302 with appropriate credentials is shown. It may be assumed that the MME 3302 has an immutable master secret, MMEMS, which is securely stored. A derivative of MMEMS called MMEMK is computed as: MMEMK = PRNG(MMEMS⊕T), where T is the current value of MME's counter. It may also be assumed that an AUS/ARP 3304 (HSS) has a secret bivariate polynomial, P(x1, x2), which is securely stored. Example parameters and their characteristics for SEA/MME provisioning are provided in the table below:

Symbol	Security Parameters	Characteristics	Size	Description	Step
-r	-Enables generation of UE specific polynomial	-Pseudorandom integer	-|p| bits	-Derived by feeding UE's IMSI and a random element to PRNG	
-S'	-UE specific polynomial required to compute LTK and secondary key	-Pseudorandom share of original polynomial P	-|p| bits	-Derived by setting a variable in P to be equal to the UE specific pseudorandom integer r	2
-S''	-Allows UE, SEA to perform mutual authentication	-Pseudorandom integer	-|p| bits	-Derived by evaluating S' at zero	
$-S'_1$	-To securely send the current pseudorandom	-Pseudorandom integer	-|p| bits	-Generated by using S_1 (SEA's share of S') and SEAMK (a pseudorandom	5

	integer, SEAMK, derived from SEA's master secret			integer derived from SEA's master secret)	
-P* -Q	-Parameter to compute secondary key(s) -Allows SEA to securely send secondary key computation parameter(s) to the UE. Also performs an implicit authentication of UE.	-Pseudorandom polynomial -Pseudorandom integer	- |p| bits -|p| bits	-Derived by using polynomial P, LTK and SEAMK -Computed by embedding data derived from MK, SEAMK, IMSI, SEAID and LTK	6

[0099] At 0, in accordance with the illustrated example, the MME 3302 and the HSS 3304 perform mutual authentication. At 1, the MME 3302 may send an authentication request to the HSS by sending the IMSI of a given UE. At 2, the HSS 3304 computes security parameters for the UE with the provided IMSI. The HSS 3304 may computes UE specific random number r by using some private function f, which belongs to a public function family F. For example, let r = 3. The HSS 3304 may use its secret polynomial to compute: S' = P(r, ·). For example, P(x1=3, x2) = 18 + x22 + 9x2 + 4 ≡ x22 + 4x2 + 2(mod 5). The HSS may computes S'' from S' as: S'' = S'(0) = P(r, 0). For example, P(x1=3, x2=0) ≡ 2(mod 5). At 3, in some cases, because the MME 3302 is placed on the first level of the hierarchal secret sharing, AUS may compute level 1 share S1 of S' for the MME 3302. For example, S1 = P1(x1=3, x2) = 4*9 + 3x22 + 4*3x2 + 1 = 3x22 + 12x2 + 37 ≡ 3x22 + 2x2 + 2(mod 5). At 4, in accordance with the illustrated example, the AUS sends (HMACS''(MMEid), S1) to the MME 3302. HMACS''(MMEid) is called Auth token and may be saved by the MME 3302. At 5, the MME 3303 may evaluate its polynomial share S1 at MME_{MK} to get: S'1 = S1(MME_{MK}) and may send S'1 to the AUS / ARP (HSS). For example, if MME_{MK} = 1 then: S'1 = S1(MME_{MK}) = 3 + 2 + 2 ≡ 4(mod 5). At 6, the HSS 3304 computes MME specific security parameters for secondary key computation. For example, the HSS 3304 may use its knowledge of the secret polynomial P to compute MME_{MK} from S'1. The HSS may compute a special value Q, which allows the MME 3302 to compute the same kMME as the UE, but without leaking any extra information to either party. Q may be computed as: Q = $HMAC_{MK}$(IMSI)⊕$HMACMME_{MK}$(MMEid)⊕k, where k is the current LTK. For example, if $HMAC_{MK}$(IMSI) ≡ 4(mod 5), $HMACMME_{MK}$(MMEid) ≡ 3(mod 5) and k = 1 then Q = 4⊕2⊕1 = 7 ≡ 2(mod 5). HSS computes: P' = P(k, ·) = 2 + x22 + 3x2 + 4 ≡ x22 + 3x2 + 1 (mod 5). HSS computes: P* = P'*MME_{MK}. The computation is performed in finite field Fp. HSS sends (P*, Q) to the MME, which saves Q and P*.

[00100] Referring to Fig. 34, an example computation of K_{MME} / K_{SEA} is described. Example security parameters and their characteristics relative to Fig. 34 are described in the table provided below:

Symbol	Security Parameters	Characteristics	Size	Description	Step
-Q'	- Allows SEA to securely send secondary key computation parameter(s) to the UE. Also performs an implicit authentication of UE.	- Pseudorandom integer	-|p| bits	-Generated by using Q, S*(a value generated by using a key homomorphic PRF) and S''	3

[00101] At 1, in accordance with the example, the MME 3404 evaluates its polynomial share S_1 at 0 to compute: $S'_1 = S_1(0)$. For example, for $S_1 = 3x_2^2 + 2x_2 + 2$, $S'_1 = 2$. The MME 3404 sends S'_1 to the UE 3402. The UE 3402 may use S'_1 to authenticate the MME 3404. For example, the UE 3402 may evaluates its share S_2 at 0 to get: $S'_2 = S_2(0)$. For example, for $S_2 = 3x_2^2 + 2x_2$, $S'_2 = 0$. Using S'_2, S'_1 UE performs perform polynomial interpolation and computes $S'' = P(r, 0)$. For example, for $P = P(x_1, x_2) = 2x_1^2 + x_2^2 + 3x_1x_2 + 4$, and $r = 3$, $S'' \equiv 2 \pmod 5$. It verifies the value of S'' by computing: $c' = HMAC_{S''}$(IMSI) and comparing it with $HMAC_{S''}$(IMSI) value stored in its database. The UE 3402 computes its authentication parameters. For example, the UE 3402 computes: $H' = HMAC_{S''}$(MMEid). This value may be already provided to the MME 3404 by the HSSHSS. The UE 3402 may compute: $Sr = S'_2 \oplus H'$. This may ensure that only the MME 3404, which is in possession of H', will be able to compute UE's evaluated share, S'_2. In an example, the UE 3402 computes: r1 = PRNG(MK). Let r1 = 1. The UE 3402 computes: $S_2^* = Prf_{MK \oplus r1}(S'')$. For example, S'' is fed to a key homomorphic pseudorandom function with the key being $MK \oplus r1$. For example, $S_2^* = Prf_2(2) \equiv 4 \pmod 5$. The UE 3402 sends (Sr, S_2^*), to the MME. 3404. The MME 3404 authenticates the UE 3402. For example, the MME 3404 may recover S'_2 as: $S'_2 = Sr \oplus H' = Sr \oplus HMAC_{S''}$(MMEid). The MME 3404 may use S'_2, S'_1 to perform polynomial interpolation and computes $S'' = P(r, 0)$. As this is a 2 party computation, the MME 3404 may only needs to compute: $S'_1 + S'_2$. The MME 3404 verifies the value of S'' by computing: $d' = HMAC_{S''}$(MMEid) and comparing it with $HMAC_{S''}$(MMEid) value, stored in its database. The MME 3404 computes: Mr1 = PRNG(MMEMK). Let Mr1 = 4. The MME 3404 computes: $S_1^* = Prf_{MMEMK \oplus Mr1}(S'')$. For example, if MMEMK = 1 then $S_1^* = Prf_0(2) \equiv 2 \pmod 5$. The MME 3404 computes: $S^* = S_1^* + S_2^* = Prf_{MMEMK \oplus Mr1 \oplus r1 \oplus MK}(S'')$. For example, if $S_1^* \equiv 2 \pmod 5$ and $S_2^* \equiv 4 \pmod 5$ then $S^* \equiv 1 \pmod 5$.

[00102] Continuing with the example illustrated in Fig. 34, at 2, the MME 3404 computes and saves $k_{MME} = P'(HMAC_{MMEMK}(MMEid) \oplus S^*)$. For example, if $P' \equiv x_2^2 + 3x_2 + 1 \pmod 5$,

MMEMK = 1, $HMAC_{MMEMK}$(MMEid) = 2 and S* ≡ 1(mod 5) then k_{MME} = P'(3) ≡ 9 + 9 + 1(mod 5) ≡ 4. The MME 3404 computes: Q' = Q⊕S*⊕S''. At 3, the MME 3404 may send Q to the UE 3402. For example, for S* ≡ 1(mod 5), S'' ≡ 2(mod 5) and Q ≡ 2(mod 5) then the Q' value becomes: 2⊕2⊕1 = 1. At 4, the UE 3402 may use Q to compute secondary key, k_{MME}. For example, the UE 3402 may compute: $HMAC_{r1}$(IMSI)⊕k⊕Q' = $HMAC_{MMEMK}$(MMEid)⊕S*⊕S''. The UE 3402 may save ($HMAC_{MMEMK}$(MMEid)⊕S*⊕S'', MMEid) in its database. UE 3402 may compute and save: k_{MME} = P'($HMAC_{MMEMK}$(MMEid)⊕S*). For example, if P' ≡ x_2^2 + 3x_2 + 1(mod 5), MMEMK = 1, $HMAC_{MMEMK}$(MMEid) = 2 and S* ≡ 1(mod 5) then k_{MME} = P'(3) ≡ 9 + 9 + 1(mod 5) ≡ 4. At 5, the UE 3402 and the MME 3404 may confirm that both parties have computed the correct k_{MME} value.

[00103] Referring now to Fig. 14, an example of provisioning an SEA 1402 with appropriate credentials is shown. It may be assumed that the SEA 1402 has an immutable master secret, *seaMK*, which is securely stored. It may also be assumed that an AUS/ARP 1404 (HSS) has a secret bivariate polynomial, P(x1, x2), which is securely stored.

[00104] In accordance with the illustrated embodiment, at 0, the SEA 1402 and the AUS/ARP 1404 (e.g., HSS) perform mutual authentication. At 1, the SEA 1402 sends an authentication request to the AUS/ARP 1404 (HSS) by sending the IMSI of the UE. At 2, the AUS/ARP 1404 (HSS) computes a UE-specific random number r by using a private function f, which may belong to a public function family F. At 3, the AUS/ARP 1404 (HSS) uses its secret polynomial to compute: S' = P(r, ·). At 4, the AUS/ARP 1404 (HSS) computes S'' from S' as: S'' = S'(0) = P(r, 0). At 5, because the SEA 1402 is placed on the first level of the 2-CTSS, AUS / ARPARP (HSS) computes level 1 share S_1 of S' for the SEA. Alternatively, the SEA 1402 may use hierarchal secret sharing mechanism, and thus the AUS/ARP 1404 (HSS) may compute level 1 share S1 of S' for the SEA 1402. At 6, the AUS/ARP 1404 (HSS) sends (HMACS''(SEAid), S1) to the SEA 1402. At 7, the SEA 1402 saves HMACS''(SEAid, S1). At 8, the SEA 1402 evaluates its polynomial share S1 at SeaMK to get: S'1 = S1(SeaMK). At 9, the SEA 1402 sends S'1 to the AUS/ARP 1404 (HSS). At 10, the AUS/ARP 1404 (HSS) uses its knowledge of the secret polynomial P to compute seaMK from S'1. At 11, the AUS/ARP 1404 (HSS) computes a special value Q, which may allow the SEA 1402 to compute the same Ksea / K_{asme} as the UE, but without leaking any extra information to either party. Q may be computed as: Q = $HMAC_{MK}$(IMSI)⊕$HMAC_{seaMK}$(SEAid)⊕k, where k is the current LTK of the UE. At 12, the AUS/ARP 1404 (HSS) computes: P' = P(k, ·). At 13, the AUS/ARP 1404 (HSS) computes: P* = P'

x seaMK. The computation may be performed in finite field Fp. At 14, the AUS/ARP 1404 (HSS) sends (P*, Q) to the SEA 1402. At 15, the SEA saves Q and P*.

[00105] Referring now to Fig. 15, in accordance with the illustrated embodiment, an example K_{SEA} is computed. In accordance with the illustrated example, at 1, an SEA 1504 evaluates its polynomial share S_1 at 0 to compute: $S'_1 = S_1(0)$. At 2, the SEA sends S'_1 to the UE. In some cases, an eavesdropper may be able to gain some information if it gets hold of multiple evaluated shares from more than level of 2-CTSS. Thus, in an example, the UE is not required to send its evaluated share. At 3, the UE 1502 evaluates its share S_2 at 0 to get: $S'_2 = S_2(0)$. At 4, the UE 1502 uses S'_2, S'_1, and its knowledge x2 being zero to perform polynomial interpolation and compute $S'' = P(r, 0)$. The UE 1502 may verify the value of S'' by computing: $c' = HMAC_{S''}(IMSI)$ and comparing it with $HMAC_{S''}(IMSI)$ value stored in its database. At 5, the UE 1502 computes: $H' = HMAC_{S''}(SEAid)$. This value may be already provided to the SEA 1504 by the AUS/ARP (HSS). Therefore, it may be used for mutual authentication. At 6, the UE 1502 computes: $Sr = S'_2 \oplus H'$. This may ensure that only the SEA 1504, which is in possession of H', is able to compute the UE's evaluated share S'_2. At 7, the UE 1502 computes: r1 = PRNG(MK). At 8, the UE 1502 computes: $S_2^* = Prf_{MK \oplus r1}(S'')$. At 9, the UE 1502 sends (Sr, S_2^*) to the SEA 1504.

[00106] Still referring to Fig. 15, at 10, in accordance with the illustrated example, the SEA 1504 recovers S'_2 as: $S'_2 = Sr \oplus H' = Sr \oplus HMAC_{S''}(SEAid)$. At 11, the SEA 1504 uses S'_2, S'_1, and its knowledge of x2 being zero to perform polynomial interpolation and compute $S'' = P(r, 0)$. Because this is a two party computation, in some cases, the SEA 1504 only needs to compute: $S'_1 \oplus S'_2$. At 12, the SEA 1504 verifies the value of S'' by computing: $d' = HMAC_{S''}(SEAid)$ and comparing it with $HMAC_{S''}(SEAid)$, stored in its database. At 13, the SEA 1504 computes: Mr1 = PRNG(seaMK). At 14, the SEA 1504 computes: $S_1^* = Prf_{seaMK \oplus Mr1}(S'')$. At 15, the SEA 1504 computes: $S^* = S_1^* + S_2^* = Prf_{seaMK \oplus Mr1 \oplus r1 \oplus MK}(S'')$. At 16, the SEA 1504 computes and saves $k_{SEA} = P'(HMAC_{seaMK}(SEAid) \oplus S^*)$. At 17, the SEA 1504 computes: $Q = Q \oplus S^* \oplus S''$. At 18, the SEA 1504 sends Q to the UE 1502. At 19, the UE 1502 computes: $HMAC_{r1}(IMSI) \oplus k \oplus Q = HMAC_{seaMK}(SEAid) \oplus S^* \oplus S''$. At 20, the UE 1502 may save $(HMAC_{seaMK}(SEAid) \oplus S^* \oplus S'', SEAid)$ in its database. At 21, the UE 1502 computes and saves: $k_{SEA} = P'(HMAC_{seaMK}(SEAid) \oplus S^*)$. The UE 1502 and the SEA 1504 may confirm that the other party has successfully computed the correct k_{SEA} value, at 22.

[00107] In an example, when using the 2-CTSS scheme, the placement of SEAs is on one level of the 2-CTSS and the eNodeBs is at another level. When using Hierarchical secret sharing, SEAs may be placed on one level of hierarchal secret sharing scheme and the enodeBs may be at

another level. Because the key computation involves the secrets of the specific parties, in some cases, it may be ensured that the parties at the same level cannot compute the keys of the other parties at the same level.

[00108] Turning now to examples of complete isolation and recovery, complete isolation provides acceptable security, but, as an example, a database of one of the parties may be hacked and data may still be stolen. Thus, described herein are schemes which, along with providing complete isolation, also provide the ability to recover from a database hack of any of the parties.

[00109] Described herein are embodiments that not only enable complete isolation, but also enable recovery from malicious or non-malicious attacks. The recovery may involve the UE and AUSF security association (long-term). The recovery may also include the recovery of the security association of the session between the UE and MME /SEA as well as AUS and MME / SEA. The provisioning of security parameters and functions are described in order that UE and AUSF long-term security association recovery can occur. Referring to Fig. 35, an example is shown in which certain parameters may be provisioned so that recovery can be performed after attacks where credential or key(s) are stolen. In the example, it is assumed that a UE 3502 has an immutable master secret, *MK*, which is securely stored. It is also assumed that an AUS/ARP 3504 (HSS) possesses a secret bivariate polynomial, P(x1, x2), which is securely stored. The table below describes example parameters used for the provisioning process relative to Fig. 35.

Symbol	Security Parameters	Characteristics	Size	Description	Step
-r	-Enables generation of UE specific polynomial	-Pseudorandom integer	-|p| bits	-Derived by feeding UE's IMSI and a random element to PRNG	
-S'	-UE specific polynomial required to compute LTK and secondary key	-Pseudorandom share of original polynomial P	-|p| bits	-Derived by setting a variable in P to be equal to the UE specific pseudorandom integer r	1
-S''	-Allows UE, SEA to perform mutual authentication	-Pseudorandom integer	-|p| bits	-Derived by evaluating S' at zero	
$-S'_2$	-To securely send the current pseudorandom integer, MK, derived from UE's master secret	-Pseudorandom integer	-|p| bits	-Generated by using S_2 (UE's share of S') and MK (a pseudorandom integer derived from UE's master secret)	5
$-P'_k$	-Information to compute secondary key(s)	-Pseudorandom polynomial	-|p| bits	-Derived by using polynomial P and LTK	7

[00110] Referring to Fig. 35, in some cases, complete isolation provides acceptable security but not if the database of one of the parties gets hacked. In some examples, the schemes that are described herein are robust enough to recover from a database hack of any of the three parties. At 0, in accordance with the illustrated example, the UE 3502 and the HSS 3504 perform mutual authentication. At 1, the HSS 3504 computes UE specific security parameters. The HSS 3504 may generate a UE specific random number r by using some private function f (e.g., a keyed hash function), which belongs to a public function family F. The HSS 3504 may evaluate its secret

bivariate polynomial P(x1, x2) at x1 = r. It hence computes: S' = P(r, ·), which is a polynomial in x2. The HSS 3504 may set x2 = 0 in S' to compute: S'' = S'(0) = P(r, 0). At 1, the HSS 3504 uses hierarchal secret sharing scheme to compute second level share of S'. In an example, all UEs are placed at the second level for the hierarchal secret sharing, while all MMEs are placed at the first level. This ensures that any number of malicious UEs cannot combine their shares to retrieve S' or any relevant information. HSS 3504 computes share S2 of S' for the second level party (e.g., the UE 3502). At 2, the HSS 3504 sends (HMACS''(IMSI), S2) to the UE 3502. At 3, upon receiving (HMACS''(IMSI), S2), the UE 3502 computes and saves recovery related parameters. For example, in some cases, the UE 3502 may save HMACS''(IMSI) as it is received but S2 is saved only after (pseudo)randomization. The UE 3502 generates and saves a random number r0. In an example, the UE 3502 uses r0 and MK to compute a pseudo random number r1 as: r1 = PRNG(MK⊕r0). r1 is directly involved in the Long Term Key (LTK) computation, and hence is dubbed key generation random factor. In order to randomize its share S2 UE computes share randomization factor as: r2 = PRNG(MK⊕r1). The share S2 is saved in the form of: Sr2 = S2 x r2. The computation is performed in finite field Fp. In some cases, the only values saved by the UE 3502 are r0 and Sr2. The UE 3502 evaluates S2 on x2 = r1 to get: S'2 = S2(r1). At 4, the UE 3502 computes LTK as: k = S2(S'2⊕r1).

[00111] At 5, in accordance with the illustrated example, the UE 3502 sends recovery parameters, S'2, to the HSS 3504. At 6, the HSS computes the LTK, k. In the example, the HSS receives S'2 and as it knows the secret polynomial and the UE specific random r, it computes r1 from S'2. The HSS computes and saves: (HMACr1(IMSI), IMSI). This value will be used in recovering from a database hack. In the example, the HSS 3504 does not save r1 itself but a pseudorandom value that is generated using r1. The HSS computes and saves long term key/secret as: k = S2(S'2⊕r1). The HSS computes UE specific secondary key parameters. For example, the HSS evaluates its secret polynomial by setting x1 = k to get: P' = P(k, ·). P' is used for kMME computation. HSS computes: P'k = P' x k. The computation is performed in finite field Fp. At 7, the HSS 3504 sends P'k to the UE 3502, which saves P'k. At 8, the UE 3502 and HSS 3504 confirm that the other party has successfully computed the correct LTK k.

[00112] Referring to Fig. 16, an example embodiment for the provisioning of cryptographic functions and parameters, which may enable recovery procedures that utilize saved or provisioned cryptographic functions and parameters is illustrated. In the example, it is assumed that a UE 1602 has an immutable master secret, *MK*, which is securely stored. It is also assumed that an AUS/ARP 1604 (HSS) possesses a secret bivariate polynomial, P(x1, x2), which is securely stored,

possibly separately from other derived cryptographic material including functions and keys, and possibly held to higher level of security and access control than the derived cryptographic functions and parameters. It is also an objective of the example, in some cases, to only derive critical security and cryptographic functions and parameters, for transient use when needed and to then delete these after use. In some examples, all parameters are deleted after use and if any functions or parameters are saved, then they are saved in a protected manner whereby they protect the core of the security architecture. In order to obtain the secure functions and parameters required to recover a compromised cryptographic function or key, the underlying hardware anchored key and possibly the intermediate stored functions and parameters may be required. The hardware anchored key may remain protected by way of a variety of security mechanisms available in the industry to provide security from hardware anchored security mechanisms, and these may be exposed, in some examples, to only correctly authenticated and authorized parties, including the hardware on and the software that may run on a secured platform.

[00113] In accordance with the illustrated embodiment, at 0, the UE 1602 and the AUS/ARP 1604 (HSS) perform mutual authentication using similar mechanisms that have been described for Fig. 13. At 1, the AUS/ARP 1604 (HSS) generates a UE specific random number r by using a private function f (e.g., a keyed hash function), which belongs to a public function family F. At 2, the AUS/ARP 1604 (HSS) evaluates its secret bivariate polynomial P(x1, x2) at x1 = r. The AUS/ARP 1604 may compute: S' = P(r, ·), which is a polynomial in x2. At 3, the AUS/ARP 1604 (HSS) sets x2 = 0 in S' to compute: S'' = S'(0) = P(r, 0). At 4, the AUS/ARP 1604 (HSS) uses 2-CTSS scheme to compute second level share of S'. All UEs are placed at the second level for the 2-CTSS, while all SEAs are placed at the first level. This ensures that any number of malicious UEs cannot combine their shares to retrieve S' or any relevant information. The set containing the first and second level parties are called P_0 and P_1, respectively. AUS / ARP (HSS) computes share S_2 of S' for the second level party i.e. the UE.

[00114] In an alternative embodiment, a hierarchal secret sharing scheme to compute a second level share of S' may be done instead of 2-CTSS. In an example, the UEs, for instance all the UEs, are placed at the second level for the hierarchal secret sharing, while the SEAs are placed at the first level. This may ensure that any number of malicious UEs cannot combine their shares to retrieve S' or any relevant information. The set containing the first and second level parties may be referred to herein as P_0 and P_1, respectively. The AUS/ARP 1604 (HSS) computes share S_2 of S' for the second level party (e.g., the UE 1602).

[00115] With continuing reference to Fig. 16, in accordance with the illustrated example, at 5, the AUS/ARP (HSS) 1604 sends ($HMAC_{S''}$(IMSI), S_2) to the UE 1602. In an example, the

$HMAC_{S''}$(IMSI) is the HMAC of the IMSI of the UE 1602, computed by using S'' as the key. At 6, in response to receiving ($HMAC_{S''}$(IMSI), S_2), the UE 1602 saves HMACS''(IMSI) as it is received, but S_2 is saved after (pseudo)randomization, in some examples. At 7, the UE 1602 generates and saves a random number r0. At 8, the UE 1602 uses r0 and its master secret MK to compute a pseudo random number r1 as: r1 = PRNG(MK⊕r0). Random number r1 may be directly involved in the Long Term Key (LTK) computation, and hence it may be dubbed as a key generation random factor. At 9, in order to randomize its share S_2, the UE 1602 may compute a share randomization factor as: r2 = PRNG(MK⊕r0). At 10, the share S_2 may be saved in the form of: $S_{r2} = S_2$ x r2. The computation may be performed in finite field Fp. In some examples, the only values saved by the UE 1602 are r0 and S_{r2}. At 11, the UE 1602 evaluates S_2 on x2 = r1 to get: $S'_2 = S_2(r1)$. At 12, the UE 1602 computes LTK as: $k = S_2(S'_2 \oplus r1)$. At 13, the UE 1602 sends S'_2 to the AUS/ARP 1604 (HSS). In some examples, the value r1 may be deleted after the generation of the LTK, k.

[00116] At 14, in accordance with the illustrated example, the AUS/ARP 1604 (HSS) receives S'_2 and, because it knows the secret polynomial and the UE specific random r, it can compute r1 from S'_2. At 15, the AUS/ARP 1604 (HSS) computes and saves: ($HMAC_{r1}$(IMSI), IMSI). This value may be used in recovering from a database hack. In the example, AUS/ARP 1604 (HSS) does not save r1 itself, but instead saves a pseudorandom value that is generated using r1. At 16, the AUS/ARP 1604 (HSS) computes and saves a long term key/secret as: $k = S_2(S'_2 \oplus r1)$. At 17, the AUS/ARP 1604 (HSS) evaluates its secret polynomial by setting x1 = k to get: P' = P(k, ·). P' may be used for k_{SEA} / K_{MME} / K_{ASME} computation. At 18, the AUS/ARP 1604 (HSS) computes: P'k = P' x k. The computation may be performed in a finite field Fp. At 19, the AUS/ARP 1504 (HSS) sends the P'k to the UE 1602. At 20, the UE 1602 may save the P'k. At 21, the UE 1602 and the AUS/ARP 1604 (HSS) may confirm that the other party has successfully computed the correct LTK k. Similar mechanisms described for Fig. 13 may be used to confirm possession of LTK, k at the UE 1602 and the AUS / ARP 1604.

[00117] Other than the functions and parameters indicated as being saved, in some cases, the other functions and parameters used to derive the LTK k may be deleted. These may be considered as transient and safe deletion reinforces the security of the LTK k.

[00118] Referring now to Fig. 36, it may be assumed that, when recovering from an AUS/ ARP (HSS) side hack, an UE 3602 has an immutable master secret, MS, which is securely stored. A derivative of MS called MK is computed as: MK = PRNG(MS⊕T), where T is the current value of UE's counter. As the Long Term Key/Secret k is stored in HSSHSS's database it may be leaked due

to the hack. The (HMACrandom(Ax), Ax) pairs stored in the HSS's database are also vulnerable to hacks. The secret bivariate polynomial P(x1, x2) is securely stored and cannot be hacked. The table provided below summarizes example parameters and their descriptions relative to Fig. 36.

Symbol	Security Parameters	Characteristics	Size	Description	Step
-r'_0, r'_1, r'_2	-Chain of three pseudorandom integers	-Pseudorandom integers	-|p| bits	-Derived by using UE's MK	
-S_{r2}	-UE's polynomial share randomized with r_2	-Pseudorandom share of original polynomial P	-|p| bits	-Derived by multiplying S_2 with r_2	2
-S'_2	-Allows UE to securely send parameters to compute fresh LTK	-Pseudorandom integer	-|p| bits	-Generated by using S_2 (UE's share of S') and r'_1	4
-P'_k	-Information to compute secondary key(s)	-Pseudorandom polynomial	- |p| bits	-Derived by using polynomial P and LTK k	6

[00119] At 1, in accorance with the illuttrated example, the HSS 3604 discovers that there has been a hack and sends an alert to the UE 3602. At 2, upon receiving the alert, the UE 3602 begins the recovery procedure by computing the recovery related information. In the example, the UE uses the saved random number r0 to compute *key generation random factor* r1 as: r1 = PRNG(MK⊕r0). The UE computes *share randomization factor* as: r2 = PRNG(MK⊕r1). The UE uses the saved value S_{r2} to compute its polynomial share S_2 as: $S_2 = S_{r2}/r2$. The computation is performed in finite field F_p. The UE updates the recovery parameters. In the example, the UE 3602 deletes r0 from its database. The UE generates and saves a fresh random number r0'. The UE computes fresh *key generation random factor* using r0' as: r1' = PRNG(MK⊕r0'). In order to randomize its share S_2, the UE 3602 may compute fresh *share randomization factor* as: r2' = PRNG(MK⊕r1'). The UE 3602 randomizes its polynomial share as: $S_{r2'} = S_2 \times r2'$, and saves $S_{r2'}$. The UE 3602 evaluates it polynomial share at r1' to get: $S'_2 = S_2(r1')$. The UE 3602 uses a fixed Pseudo Random Permutation (PRP) algorithm (e.g., AES) to compute: $r1'' = PRP_{r1'}(r1)$.

[00120] Still referring to Fig. 36, in accordance with the example, at 3, the UE 3602 computes and saves fresh LTK k' as: k' = S2(S'2⊕r1'). At 4, the UE sends the recovery parameters, (S'2, r1''), to the HSS. At 5, upon receiving S'2 and r1'', the HSS 3604 first performs a verification to authenticate the UE 3602 and then remove the stale information. In the example, the HSS 3604 computes r1' from S'_2 by using its knowledge of the secret polynomial P and the UE specific random r. The HSS inverts the PRP by using r1' as the key to recover the plaintext i.e. r1. The HSS computes $HMAC_{r1}$(IMSI) and compares it with the $HMAC_{r1}$(IMSI) value stored in its database. Successful verification implies that the UE has knowledge of r1, and hence authenticates it. The HSS removes ($HMAC_{r1}$(IMSI), IMSI) from its database. The HSS computes and saves ($HMAC_{r1'}$(IMSI), IMSI). The HSS computes and saves the fresh LTK k' as: $k' = S_2(S'_2 \oplus r1')$. The

HSS computes fresh, UE specific secondary key computation parameters. In the example, the HSS computes fresh P' = P(k', ·). Recall that P' is used in k_{MME} computation. The HSS computes: $P'_{k'}$ = P' * k'. The computation is done in finite field F_p. At 6, in accordance with the illustrated example, the HSS 3604 sends $P'_{k'}$ to the UE 3602, which saves it. At 7, the UE 3602 and the HSS 3604confirm that both parties have computed the correct LTK k'.

[00121] Referring now to Fig. 17, it may be assumed that, when recovering from an AUS/ ARP (HSS) side hack, an UE 1702 has an immutable master secret, MK, which is securely stored. It may also be assumed that a Long Term Key/Secret k that is stored in a database of a AUS/ARP 1704 (HSS) may be leaked due to the hack. In an example, the (HMACrandom(Ax), Ax) pairs stored in the AUS/ARP (HSS)'s database are also vulnerable to hacks, but a secret bivariate polynomial P(x1, x2) is securely stored and cannot be hacked.

[00122] In accordance with the illustrated example, at 1, the AUS/ARP 1704 (HSS) discovers that there has been a hack and sends an alert to the UE 1702. Upon receiving the alert, the UE 1702 begins the recovery procedure. The UE 1702 may use the saved random number r0 to compute key generation random factor r1 as: r1 = PRNG(MK⊕r0). At 3, the UE 1702 may compute a share randomization factor as: r2 = PRNG(MK⊕r1). At 4, the UE 1702 uses the saved value S_{r2} to compute its polynomial share S_2 as: $S_2 = S_{r2}/r2$. The computation may be performed in finite field Fp. At 5, the UE 1702 deletes r0 from its database. At 6, the UE 1702 generates and saves a fresh random number r0'. At 7, the UE 1702 computes fresh key generation random factor using r0' as: r1' = PRNG(MK⊕r0'). At 8, in order to randomize its share S_2, the UE 1702 may compute is fresh share randomization factor as: r2' = PRNG(MK⊕r1'). At 9, the UE 1702 randomizes its polynomial share as: $S_{r2'} = S_2$ x r2', and saves $S_{r2'}$. At 10, the UE 1702 evaluates its polynomial share at r1 to get: $S'_2 = S_2(r1')$. At 11, the UE 1702 uses a fixed Pseudo Random Permutation (PRP) algorithm (e.g., AES) to compute: r1'' = PRPr1'(r1). At 12, the UE 1702 computes and saves fresh LTK k' as: $k' = S_2(S'_2 \oplus r1')$. At 13, the UE 1702 sends S'_2 and r1'' to the AUS/ARP 1704 (HSS).

[00123] With continuing reference to Fig. 17, at 14, in accordance with the illustrated example, upon receiving S'_2 and r1'', the AUS/ARP 1704 (HSS) computes r1' from S'_2 by using its knowledge of the secret polynomial P and the UE specific random r. At 15, the AUS/ARP 1704 (HSS) inverts the PRP by using r1' as the key to recover the plaintext (e.g., r1). At 16, the AUS/ARP 1704 (HSS) computes HMACr1(IMSI) and compares it with the $HMAC_{r1}$(IMSI) value stored in its database. In some examples, successful verification implies that the UE 1702 has knowledge of r1, and hence authenticates it. At 17, the AUS/ARP 1704 (HSS) removes

($HMAC_{r1}$(IMSI), IMSI) from its database. At 18, the AUS/ARP 1704 (HSS) computes and saves ($HMAC_{r1'}$(IMSI), IMSI). At 19, the AUS/ARP 1704 (HSS) computes and saves the fresh LTK k' as: $k' = S_2(S'_2 \oplus r1')$. At 20, the AUS/ARP 1704 (HSS) computes fresh $P' = P(k', \cdot)$. As mentioned above, the P' may be used in the k_{SEA} computation. At 21, the AUS/ARP 1704 (HSS) computes $P'k' = P' \times k'$. The computation may be done in finite field Fp. At 22, the AUS/ARP 1704 (HSS) sends P'k' to the UE 1702. At 23, the UE 1702 saves P'k'. At 24, in accordance with the illustrated example, the UE 1702 and the AUS/ARP 1704 (HSS) confirm that both parties have computed the correct LTK k'.

[00124] Turning now to an example of MME/SEA provisioning as shown in Fig. 37, it is assumed that an MME 3702 has an immutable master secret, MME_{MS}, which is securely stored. A derivative of MME_{MS} called MME_{MK} is computed as: $MME_{MK} = PRNG(MME_{MS} \oplus T)$, where T is the current value of MME's counter. An HSS 3704 has a secret bivariate polynomial, P(x1, x2), which is securely stored. The table provided below lists example parameters and their descriptions relative to Fig. 37.

Symbol	Security Parameters	Characteristics	Size	Description	Step
-r	-Enables generation of UE specific polynomial	-Pseudorandom integer	-|p| bits	-Derived by feeding UE's IMSI and a random element to PRNG	
-S'	-UE specific polynomial required to compute LTK and secondary key	-Pseudorandom share of P	-|p| bits	-Derived by setting a variable in P to be equal to the UE specific pseudorandom integer r	2
-S''	-Allows UE, SEA to perform mutual authentication	-Pseudorandom integer	-|p| bits	-Derived by evaluating S' at zero	
-Mr_0, Mr_1, Mr_2	-Chain of three pseudorandom integers	-Pseudorandom integers	-|p| bits	-Derived by using UE's MK	
-S_{Mr2}	-UE's polynomial share randomized with r_2 -Allows SEA to send Mr_1 to the AUS	-Pseudorandom share of original polynomial P	-|p| bits	-Derived by multiplying S_2 with r_2	5
-S''		-Pseudorandom integer	-|p| bits	-Generated from SEA's polynomial share, S_1, and Mr_1	
-P*	-Allows computation of secondary key(s)	-Pseudorandom polynomial	-|p| bits	-Derived by using polynomial P, LTK and Mr_1	
-Q	-Allows SEA to securely send secondary key computation parameter(s) to the UE. Also performs an implicit authentication of UE.	-Pseudorandom integer	-|p| bits	-Computed by embedding data derived from r_1, Mr_1, IMSI, SEAID and LTK	6

[00125] At 0, in accordance with the illustrated example, the MME 3702 and the HSS 3704 perform mutual authentication. At 1, the MME 3702 may send an authentication request to the HSS 3704 by sending the IMSI of the UE. At 2, the HSS 3704 computes the security parameters specific to the UE having the provided IMSI value. In the example, the HSS computes UE specific random number r by using the private function f. The HSS uses its secret polynomial to compute: $S' = P(r, \cdot)$. The HSS computes S'' from S' as: $S'' = S'(0) = P(r, 0)$. At 3, as the MME

3702 is placed on the first level of the 2-CTSSS or the hierarchal secret sharing, the HSS 3704 computes level 1 share S1 of S' for the MME 3702. At 4, the HSS 3704 sends (HMACS''(MMEid), S1) to the MME 3702. At 5, the MME 3702 computes recovery parameters. In the example, the MME saves HMACS''(MMEid), but S1 is saved after randomization. The MME generates and saves a random number Mr0. The MME uses the random number Mr0 and MMEMK to compute key generation random factor as: Mr1 = PRNG(MMEMK⊕Mr0). The MME uses Mr1 and MMEMK to compute share randomization factor as: Mr2 = PRNG(MMEMK⊕Mr1). The Mr2 is used to randomize MME's share S1 as: SMr2 = Mr2 x S1. The computation is performed in finite field Fp. The MME evaluates its polynomial share S1 at Mr1 to get: S'1 = S1(Mr1). The MME sends recovery parameter, S'1, to the AUS / ARP (HSS) 3704 (at 5). At 6, the HSS 3704 computes MME specific parameters for secondary key computation. In the example, the HSS uses its knowledge of the secret polynomial P and the UE specific random r to compute Mr1 from S'1. The HSS computes and saves: (HMACMr1(MMEid), MMEid). The HSS computes a special value Q, which allows MME to compute kMME without leaking any secret information to the UE. Q is computed as: Q = HMACr1(IMSI)⊕HMACMr1(MMEid)⊕k. Here k is the current LTK of the UE, and r1 is the current key generation random factor of the UE. In some cases, (HMACr1(IMSI), IMSI) is saved in AUS's database. The HSS computes: P' = P(k, ·). The HSS computes: P* = P' x MR1. The computation is performed in finite field Fp. The HSS 3704 may send (P*, Q) to the MME 3702, which may save P* and Q.

[00126] Referring to Fig. 18, an alternative embodiment is described, wherein, a SEA 1802 may have an immutable master secret, SEA_{MK}, which is securely stored. Further, an AUS/ARP 1804 (HSS) may have a secret bivariate polynomial, P(x1, x2), which is securely stored. In accordance with the illustrated example, at 0, the SEA 1802 and the AUS/ARP 1804 (HSS) perform mutual authentication. At 1, the SEA 1802 sends an authentication request to the AUS/ARP 1804 (HSS), for example, by sending the IMSI of the UE. At 2, the AUS/ARP 1804 (HSS) computes a UE specific random number r by using the private function f. At 3, the AUS/ARP 1804 (HSS) uses its secret polynomial to compute: S' = P(r, ·). At 4, the AUS/ARP 1804 (HSS) computes S'' from S' as: S'' = S'(0) = P(r, 0). In an example, because the SEA 1802 is placed on the first level of the hierarchal secret sharing, the AUS/ARP 1804 (HSS) computes level 1 share S_1 of S' for the SEA 1802. At 6, the AUS/ARP 1804 (HSS) sends ($HMAC_{S''}$(SEAid), S1) to the SEA 1802.

[00127] At 7, in accordance with the illustrated example, the SEA 1802 saves $HMAC_{S''}$(SEAid), but S1 may be saved after randomization. At 8, the SEA 1802 generates and

saves a random number Mr0. At 9, the SEA 1802 uses the random number Mr0 and its master secret, seaMK, to compute key generation random factor as: Mr1 = PRNG(seaMK⊕Mr0). At 10, the SEA 1802 uses Mr1 and seaMK to compute share randomization factor as: Mr2 = PRNG(seaMK⊕Mr1). At 11, Mr2 is used to randomize SEA's share S1 as: S_{Mr2} = Mr2 x S_1. The computation is performed in finite field Fp. At 12, the SEA 1802 evaluates its polynomial share S_1 at Mr1 to get: $S'_1 = S_1(Mr1)$. At 13, the SEA 1802 sends S'_1 to the AUS/ARP 1804 (HSS). The AUS/ARP 1804 (HSS) may use its knowledge of the secret polynomial P and the UE specific random r to compute Mr1 from S'1. At 15, the AUS/ARP 1804 (HSS) computes and saves: ($HMAC_{Mr1}$(SEAid), SEAid). At 16, the AUS/ARP 1804 (HSS) may compute a special value Q, which may allow the SEA 1802 to compute k_{SEA} without leaking any secret information to the UE. For example, the value Q may be computed as: Q = $HMAC_{r1}$(IMSI)⊕$HMAC_{Mr1}$(SEAid)⊕k. In this example, k is the current LTK of the UE, and r1 is the current key generation random factor of the UE. In some cases, the ($HMAC_{r1}$(IMSI), IMSI) is saved in the AUS's database. At 17, the AUS/ARP 1804 (HSS) computes: P' = P(k, ·). At 18, the AUS/ARP 1804 (HSS) computes: P* = P' x MR1. The computation may be performed in finite field Fp. At 19, the AUS/ARP 1804 (HSS) sends (P*, Q) to the SEA 1802. At 20, the SEA 1802 may save P* and Q.

[00128] Referring now to Fig. 38, an example embodiment of a computation of K_{MME} / K_{SEA} is shown. The table provided below describes the example parameters used:

Symbol	Security Parameters	Characteristics		Size	Description	Step
- S_2*	-Enables SEA to compute the secondary key	-Pseudorandom integer		-|p| bits	-Derived by using r_1, r_2, S'', and key homomorphic PRF	2
-Q'	-Allows UE to compute secondary key	-Pseudorandom integer		-|p| bits	-Derived by using Q, S* and S''	
-Ct	-Allows UE to authenticate the information sent by the SEA	-Pseudorandom integer		-|p| bits	-Computed by encrypting S'' using S8 as the key	4

[00129] At 1, in accordance with the illustrated example, mutual authentication occurs between UE 3802 and the MME 3804, wherein the MME 3804 computes its secret share. The authentication process may entail one or more back and forth messaging between the UE 3802 and the MME 3804. In the example, the MME uses the saved random number Mr0 and MMEMK to compute: Mr1 = PRNG(MMEMK⊕Mr0). The MME uses MR1 to compute share randomization factor as: Mr2 = PRNG(MMEMK⊕Mr1). The MME computes its polynomial share S1 as: S1 = SMr2/Mr2. The computation is done in finite field Fp. The MME computes: S'1 = S1(0). In the illustrated example, the MME 3804 sends S'_1 to the UE 3802. The UE 3802 authenticates the MME 3804. In the example, the UE uses its saved random number r0 and MK to compute r1 = PRNG(MK⊕r0). The UE uses r1 to compute the next random number in the chain as: r2 =

PRNG(MK⊕r1). The UE uses the saved Sr2 value to compute its polynomial share S2 of S' as: S2 = Sr2/r2. The computation is performed in finite field Fp. The UE evaluates its share S2 at 0 to get: S'2 = S2(0). The UE uses S'2, S'1, and its knowledge x2 being zero to perform polynomial interpolation and compute S'' = P(r, 0). It verifies the value of S'' by computing: c' = HMACS''(IMSI) and comparing it with HMACS''(IMSI) value stored in its database.

[00130] Still referring to Fig. 38, the UE 3802 may compute secondary key computation parameters. In the example, the UE computes: H' = HMACS''(MMEid). This value is already provided to MME by the HSS. Therefore, it may be used for UE authentication. The UE computes: Sr = S'2⊕H'. This may ensure that only the MME, which is in possession of H', will be able to compute UE's evaluated share S'2. The UE computes: S2* = Prfr2⊕r1(S''). The UE sends Sr to the MME. The MME 3804 authenticates the UE 3802. In the example, the MME recovers S'_2 as: S'_2 = Sr⊕$HMAC_{S''}$(MMEid). The MME uses S'_2, S'_1 to perform polynomial interpolation and compute: S'' = P(r, 0). The MME verifies the value of S'' by computing: d' = $HMAC_{S''}$(MMEid), and comparing it with $HMAC_{S''}$(MMEid) value stored in its database. At 2, in accordance with the illustrated example, the UE 3802 sends S_2* to the MME 3804. At 3, the MME 3804 saves S_2*. In the example, the MME 3804 computes: S1* = PrfMr2⊕Mr1(S''). The MME computes: S* = S1* + S2* = PrfMr2 ⊕Mr1⊕r1⊕r2(S''). The MME computes and saves kMME = P'(HMACMr1(MMEid)⊕S*). At 4, the MME 3804 computes: Q' = Q ⊕S*⊕S'', and sends (Q', Ct = PRP_{S*}(S'')) to UE 3802. At 5, the UE 3802 computes recovery parameters. In the example, the UE computes: $HMAC_{r1}$(IMSI)⊕k⊕Q' = $HMAC_{Mr1}$(MMEid)⊕S*⊕S''. The UE saves ($HMAC_{Mr1}$(MMEid)⊕S*⊕S'', MMEid) in its database. The UE computes and saves: k_{MME} = P'($HMAC_{Mr1}$(MMEid) ⊕S*). At 6, the UE 3802 and the MME 3804 confirm that the both parties have successfully computed the correct k_{MME} value.

[00131] Referring now to Fig. 19, an example embodiment of a computation of K_{SEA} is shown. At 1, in accordance with the illustrated example, an SEA 1904 uses the saved random number Mr0 and its master secret seaMK to compute: Mr1 = PRNG(seaMK⊕Mr0). At 2, the SEA 1904 uses MR1 to compute a share randomization factor as: Mr2 = PRNG(seaMK⊕Mr1). At 3, the SEA 1904 computes its polynomial share S_1 as: S_1 = S_{Mr2}/Mr2. The computation is done in finite field Fp. At 4, the SEA computes: S'_1 = S_1(0). At 5, the SEA 1904 sends S'_1 to the UE 1902. In some cases, an eavesdropper may be able to gain some information if it gets hold of multiple evaluated shares from more than one level of hierarchal secret sharing or share from the 2-CTSS scheme. Thus, in an example embodiment, the UE 1902 does not sends its evaluated share.

[00132] At 6, in accordance with the illustrated example, the UE 1902 uses its saved random number r0 and master secret MK to compute r1 = PRNG(MK⊕r0). At 7, the UE 1902 uses r1 to compute the next random number in the chain as: r2 = PRNG(MK⊕r1). At 8, the UE 1902 uses the saved S_{r2} value to compute its polynomial share S_2 of S' as: $S_2 = S_{r2}/r2$. In an example, the computation is performed in finite field Fp. At 9, the UE 1902 evaluates its share S_2 at 0 to get: $S'_2 = S_2(0)$. At 10, in accordance with the example, the UE 1902 uses S'_2, S'_1, and its knowledge x2 being zero to perform polynomial interpolation and compute S'' = P(r, 0). The UE 1902 may verify the value of S'' by computing: c' = $HMAC_{S''}$(IMSI) and comparing it with HMACS''(IMSI) value stored in its database. At 11, the UE 1902 computes: H' = $HMAC_{S''}$(SEAid). This value may be already provided to the SEA 1904 by the AUS/ARP (HSS). Therefore, in some examples, it may be used for UE authentication. At 12, the UE 1902 computes: Sr = S'_2⊕H'. In some cases, this may ensure that only the SEA 1904, which is in possession of H', is able to compute the UE's evaluated share S'_2. At 13, the UE 1902 computes: $S_2^* = Prf_{r2⊕r1}(S'')$. At 14, the UE 1902 sends (Sr, S2*) to the SEA 1904.

[00133] Still referring to Fig. 19, at 15, in accordance with the illustrated example, the SEA 1904 recovers S'_2 as: S'2 = Sr⊕H' = Sr⊕$HMAC_{S''}$(SEAid). At 16, the SEA 1904 uses S'_2, S'_1 to perform polynomial interpolation and compute: S'' = P(r, 0). At 17, the SEA 1904 verifies the value of S'' by computing: d' = $HMAC_{S''}$(SEAid), and comparing it with $HMAC_{S''}$(SEAid) value stored in its database. At 18, the SEA 1904 saves S2*. At 19, the SEA 1904 computes: S1* = $Prf_{Mr2⊕Mr1}(S'')$. At 20, the SEA 1904 computes: S* = S1* + S2* = $Prf_{Mr2⊕Mr1⊕r1⊕r2}(S'')$. At 21, the SEA 1904 computes and saves k_{SEA} = P'(HMAC$_{Mr1}$(SEAid)⊕S*). At 22, the SEA 1904 computes: Q = Q ⊕S*⊕S''. At 23, the SEA 1904 sends (Q, Ct = PRPS*(S'')) to the UE 1902. At 24, the UE 1902 may compute HMAC$_{r1}$(IMSI)⊕k⊕Q = HMAC$_{Mr1}$(SEAid)⊕S*⊕S''. At 25, the UE 1902 may save (HMAC$_{Mr1}$(SEAid)⊕S*⊕S'', SEAid) in its database, for example. At 26, the UE 1902 computes and saves: k_{SEA} = P'(HMAC$_{Mr1}$(SEAid) ⊕S*). At 27, the UE 1902 and the SEA 1904 may confirm that the both parties have successfully computed the correct k_{SEA} value.

[00134] Referring now to Figs. 39, 20A, and 20B, an example MME-side k_{MME} / SEA-side k_{SEA} hack recovery is shown. Referring to Fig. 39, it may be assumed that the data stored by an example MME / SEA 3904 is vulnerable to a hack, except for its master secret, MMEMK. The table provided below summarizes the example parameters used and their description relative to Fig. 39.

Symbol	Security Parameters	Characteristics	Size	Description	Step
-Q^, S'_1	-Allows UE to authenticate the SEA	-Pseudorandom integers	-|p| bits each	-Derived from SEA's polynomial share, Mr_1, Q and SEAID	

- Dt', Lt	-Allows SEA to authenticate UE and compute recovery parameters	-Pseudorandom share of original polynomial P	-|p| bits each	-Computed by using both parties' evaluated polynomial shares and Mr_1	1
-Qt	-Enables the UE to compute secondary key and generate recovery parameters	-Pseudorandom integer	-|p| bits	-Computed using Q, S* and S# (computed using Mr'_1, Mr'_2, r_1, r_2, S'', S* and key homomorphic PRF)	3

[00135] In accordance with the illustrated example, at 1, the MME 3904 computes its share and the Authentication Token. In the example, the MME uses the saved random number Mr0 and MMEMK to compute: Mr1 = PRNG(MMEMK⊕Mr0). The MME computes: Mr2 = PRNG(MMEMK⊕Mr1). The MME computes: S1 = SMr2/Mr2. The MME computes the Auth. token as: S'1 = S1(0). The MME computes secondary key recovery parameters as: Q^ = Q⊕HMACMr1(MMEid)⊕S'1. The MME sends (S'1, Q^) to the UE. The UE 3904 computes and evaluates its share and Auth. Token. In the example, the UE locates Dt = HMACMr1(MMEid)⊕S*⊕S'' in the database. The UE uses its current key generation random factor, r1, by using the saved random number r0 and MK. UE computes: r1 = PRNG(MK⊕r0). The UE computes: r2 = PRNG(MK⊕r1). The UE computes: S2 = Sr2/r2. The UE computes: S'2 = S2(0). The UE computes: Gt = Q^⊕HMACr1(IMSI)⊕k = S''⊕S'1⊕S*. The UE computes: Ft = Q^⊕Dt⊕k⊕HMACr1(IMSI) = HMACMr1(MMEid)⊕S'1. The UE computes: Jt = Gt⊕S'2 = S''⊕S'1⊕S'2⊕S* = S''⊕S''⊕S* = S*. The UE has Ct = PRPS*(S''), which was saved during the kMME computation. UE uses S* to invert the encryption and retrieve the value of S''. The UE computes: S*2 = Prfr1⊕r2(S''). The UE computes: S*2⊕S* = S*1. The UE computes: Lt = HMACMr1(MMEid)⊕S*2⊕S'2. The UE computes secondary key recovery parameters. In the example, the UE computes: Wt = Dt⊕S*1⊕S*⊕S'' = HMACMr1(MMEid)⊕S*1. The UE computes: S2# = Prfr2⊕r1(Wt). The UE computes: Dt' = Dt⊕S''⊕S2# = HMACMr1(MMEid)⊕S*⊕S2#. The UE sends (Dt', Lt) to the MME. In the example, the MME authenticates the UE. For example, the MME may retrieve S'2 from Lt as: S'2 = Lt⊕HMACMr1(MMEid)⊕S*2. In the example, the MME computes S'' from S'1 and S'2. The value of S'' is verified by using the saved value of HMACS''(MMEid). The MME computes: S1* = PrfMr2⊕Mr1(S''). The MME computes: Wt = HMACMr1(MMEid)⊕S*1. The MME computes: S* = S1* + S2* = PrfMr2⊕Mr1⊕r2⊕r1(S''). The MME computes: S2# = S*⊕Dt'⊕HMACMr1(MMEid).

[00136] Still referring to Fig. 39, in accordance with the illustrated example, at 2, the MME 3904 computes fresh secondary key k'MME. In the example, the MME deletes Mr0 and S2*. The MME generates and saves fresh Mr0'. The MME uses the fresh Mr0' to compute: Mr1' = PRNG(MMEMK⊕Mr0'). The MME computes: Mr2' = PRNG(MMEMK⊕Mr1'). The MME

updates SMr2' using the fresh Mr2' value, e.g., randomized the share S1 by using the Mr2'. The computation is performed in finite field Fp. The MME computes: S1# = PrfMr2'⊕Mr1'(Wt). the MME computes: S# = S1# + S2# = PrfMr2'⊕Mr1'⊕r2⊕r1(Wt). The MME computes and saves fresh key as: k'MME = P'(HMACMr1'(MMEid)⊕S#). At 3, the MME 3904 computes: Qt = Q⊕S*⊕S#, and sends Qt to the UE 3902. At 4, the UE 3902 computes fresh secondary key recovery parameters. In the example, the UE computes: Rt = Qt⊕k⊕HMACr1(IMSI) = HMACMr1'(MMEid)⊕S#⊕S''. The UE replaces MME's record with (Rt, MMEid). At 5, the UE 3902 computes fresh key as: k'MME = P'(HMACMr1'(MMEid)⊕S#). At 6, the MME 3904 and UE 3902 confirm that both parties computed the correct k'MME value.

[00137] Referring now to Figs. 20A and 20B, it may be assumed that the data stored by an example SEA 2004 is vulnerable to a hack, except for its master secret, seaMK. In accordance with the illustrated embodiment, at 1, the SEA 2004 uses the saved random number Mr0 and its master secret seaMK to compute: Mr1 = PRNG(seaMK⊕Mr0). At 2, the SEA 2004 computes: Mr2 = PRNG(seaMK⊕Mr1). At 3, the SEA 2004 computes: S1 = SMr2/Mr2. At 4, the SEA 2004 computes: $S'_1 = S_1(0)$. At 5, the SEA 2004 computes: $Q^\wedge = Q \oplus HMAC_{Mr1}(SEAid) \oplus S'_1$. At 6, the SEA 2004 sends (Q^) to the UE 2002.

[00138] In accordance with the illustrated example, at 7, the UE 2002 locates $Dt = HMAC_{Mr1}(SEAid) \oplus S^* \oplus S''$ in the database. The UE 2002 may use its current key generation random factor, r1, by using the saved random number r0 and master secret MK. The UE 2002 may compute: r1 = PRNG(MK⊕r0). At 9, the UE 2002 computes: r2 = PRNG(MK⊕r1). At 10, the UE 2002 computes: $S_2 = S_{r2}/r2$. At 11, the UE 2002 computes: $S'_2 = S2(0)$. At 12, the UE 2002 compute: $Gt = Q^\wedge \oplus HMAC_{r1}(IMSI) \oplus k = S'' \oplus S'_1 \oplus S^*$. At 13, the UE 2002 computes: $Ft = Q^\wedge \oplus Dt \oplus k \oplus HMACr1(IMSI) = HMAC_{Mr1}(SEAid) \oplus S'_1$. At 14, the UE 2002 computes: $Jt = Gt \oplus S'_2 = S'' \oplus S'_1 \oplus S'_2 \oplus S^* = S'' \oplus S'' \oplus S^* = S^*$. At 15, the UE 2002 may have Ct = PRPS*(S''), which may have been saved during the k_{SEA} computation. The UE 2002 may use S* to invert the encryption and retrieve the value of S''. At 16, in accordance with the example, the UE 2002 computes: $S^*_2 = Prf_{r1 \oplus r2}(S'')$. At 17, the UE 2002 computes: $S^*_2 \oplus S^* = S^*_1$. At 16, the UE 2002 computes: $Lt = HMAC_{Mr1}(SEAid) \oplus S^*_2 \oplus S'_2$. At 19, the UE 2002 computes $Wt = Dt \oplus S^*_1 \oplus S^* \oplus S'' = HMAC_{Mr1}(SEAid) \oplus S^*_1$. At 20, the UE 2002 computes: $S_2\# = Prf_{r2 \oplus r1}(Wt)$. At 21, the UE 2002 computes: $Dt' = Dt \oplus S'' \oplus S_2\# = HMAC_{Mr1}(SEAid) \oplus S^* \oplus S_2\#$. At 22, the UE 2002 sends (Dt', Lt) to the SEA 2004.

[00139] Referring to Fig. 20B, in accordance with the illustrated example, at 23, the SEA 2004 retrieves S'_2 from Lt as: $S'2 = Lt \oplus HMAC_{Mr1}(SEAid) \oplus S^*_2$. At 24, the SEA 2004 computes

S'' from S'1 and S'2. The value of S'' may be verified by using the saved value of HMACS''(SEAid). At 25, the SEA 2004 computes: S1* = $Prf_{Mr2\oplus Mr1}$(S''). At 26, the SEA 2004 computes: Wt = $HMAC_{Mr1}$(SEAid)$\oplus$S*1. At 27, the SEA 2004 computes: S* = S_1* + S_2* = $Prf_{Mr2\oplus Mr1\oplus r2\oplus r1}$(S''). At 28, the SEA 2004 computes: $S_2^{\#}$ = S*$\oplus$Dt'$\oplus$$HMAC_{Mr1}$(SEAid). At 29, the SEA 2004 deletes Mr0 and S_2*. At 30, the SEA 2004 generates and saves fresh Mr0'. At 31, the SEA 2004 uses the fresh Mr0' to compute: Mr1' = PRNG(seaMK$\oplus$Mr0'). At 32, the SEA 2004 computes: Mr2' = PRNG(seaMK$\oplus$Mr1'). At 33, the SEA 2004 may update $S_{Mr2'}$ using the fresh Mr2' value (e.g., randomized the share S_1 by using the Mr2'. The computation may be performed in finite field Fp. At 34, the SEA 2004 computes: S_1# = $Prf_{Mr2'\oplus Mr1'}$(Wt). At 35, the SEA 2004 computes: $S^{\#}$ = $S_1^{\#}$ + $S_2^{\#}$ = $Prf_{Mr2'\oplus Mr1'\oplus r2\oplus r1}$(Wt). At 36, the SEA 2004 computes and saves a fresh key as: k'$_{SEA}$ = P'($HMAC_{Mr1'}$(SEAid)$\oplus$$S^{\#}$). At 37, the SEA 2004 computes: Qt = Q$\oplus$S*$\oplus$$S^{\#}$. At 38, the SEA 2004 sends Qt to the UE 2002. At 39, in accordance with the illustrated example, the UE 2002 computes: Rt = Qt$\oplus$k$\oplus$$HMAC_{r1}$(IMSI) = $HMAC_{Mr1'}$(SEAid)$\oplus$$S^{\#}$$\oplus$S''. At 40, the UE 2002 may replace the SEA's record with (Rt, SEAid). At 41, the UE 2002 computes a fresh key as: k'$_{SEA}$ = P'($HMAC_{Mr1'}$(SEAid)$\oplus$$S^{\#}$). At 42, the SEA 2004 and the UE 2002 may confirm that that both parties computed the correct k'$_{SEA}$ value.

[00140] As described above, in various examples, SEA is placed on the first level of the 2-CTSS, AUS / ARP (HSS) computes level 1 share S_1 of S' for SEA. Alternatively, the SEAs are at one level of a hierarchal secret sharing scheme, and the CN and AN functions are at another level that is different than the level of the SEAs. Because the key computations involve the secrets of the specific parties, it is ensured that even the parties at the same level cannot compute the keys of the other parties at the same level. Thus, in accordance with various embodiments, all parties are loosely coupled.

[00141] Different deployment models for NextGen systems are now described. The deployment models are based upon trust relationships between a NextGen operator and other operators in a roaming scenario, and also between a NextGen Operator and an Over-The-Top (OTT) provider (e.g. Google). In an example non-roaming scenario, the core security functions ARP, AUS, SEA and the SCM are located with the security domain of the NextGen operator.

[00142] In some cases, all SEAs at a level of 2-CTSS scheme and the CN and AN functions are all at another level. As the key computation involves the secrets of the specific parties it is ensured that even the parties at the same level cannot compute the keys of the other parties at the same level. To revoke LTK k of UE, in some examples, the HSS must revoke or update the UE specific random number r as it's the basis of the key generation as well as authentication. Recall

that r may be computed by using a secret function by inputing some UE specific information like IMSI. The HSS maintains a list of the UEs whose keys have been revoked. When a UE's LTK is revoked its UE specific random number is computed as: $r' = f_k(f_k(IMSI)) = f_k(r)$. Next, based on the r' value fresh S' and S'' values are computed and new share of S' is sent to the SEA. HSS removes ($HMAC_{r1}(IMSI)$, IMSI) from its database, where r1 is the *key generation random factor*. SEA removes the UE specific k_{SEA}, and updates its UE specific polynomial share as well as the authentication information ($HMAC_{S''}(SEAid)$). This ensures that the UE can no longer authenticate itself to SEA. Thus, all parties may be loosely coupled, in accordance with various embodiments.

[00143] Fig. 21 illustrates an example high-level NextGen architecture for a non-roaming scenario. As shown in the example, the SEA 2102 and SCM 2104 form part of the CP-CN functions 2101. The credential generation process used for securing the various interfaces (e.g., NG1) between the UE 2106 and the CN functions 2101, as well as the UE and AN functions may vary depending upon the deployment model.

[00144] With respect to credential generation, in some examples, credential generation mechanisms are anchored on different sets of credentials, such as Hierarchical, Partially Hierarchical and, Partially Flat key derivation. With respect to an example Hierarchical Credential Generation, referring to Fig. 22, the credentials may be generated in a hierarchical manner. At an AUS 2402, a first credential 2404 (K_{AUS}) and a second credential 2406 (K'_{AUS}) may be generated. The second credential 2404 (K'_{AUS}) may be used for a re-authentication procedure or for a credential re-generation procedure periodically. Similarly, at an SEA/SCM 2406, a first credential 2410 (K_{SEA}) may be generated, and a second credential 2412 (K'_{SEA}) may be generated and used for re-authentication, verifying key possession (e.g., in handoff, tracking area update, handing off between slices, or instantiation of a new slice), or key re-generation at the SEA/SCM domain level. Each of the credentials at each of the levels may have an associated expiration timer. In some cases, if a credential at a higher level expires then the entire hierarchy of credentials at the lower layers may have to be generated. An example key generation process is provided below:

KCP-CN = PRF (KSEA, Nonce1 || Nonce2||"CP-CN Key Generation Process"||iter=0)

KUP-CN = PRF (KSEA, Nonce1 || Nonce2||"UP-CN Key Generation Process"||iter=1)

[00145] With respect to partially hierarchical/flat credential generation, as shown in Fig. 23, variations may be applied to the credential variation shown in Fig. 22. In accordance with the illustrated example, within a home operator network, the SEA/SCM 2408 is able to directly communicate with an anchor function within the AN 2414, and both the SEA/SCM 2408 and the anchor function within the AN 2414 are able to mutually authenticate and trust one another.

[00146] In the example credential generation process depicted in Fig. 24, an anchor function within the CN 2416 is used to derive credentials (integrity/authenticity and confidentiality credentials) for both CP and UP, while an anchor function within the AN 2414 is used to derive credentials for CP, UP, Key Encryption Key (KEK), and Group Key (GK).

[00147] Referring now to Fig. 25, an example for handoff is shown, in which an SEA 2514 is leveraged, and an SCM 2512 is used for context transfer. Referring to Fig. 25, a change of CP-CN anchor (MM or SM) may be necessary, in some cases, which may be because there is no direct link between a Source AN 2504 and a Target AN 2506. In accordance with the illustrated example, at 1, the source AN 2504 sends a Handover Required message to the source CP-CN 2508. At 2, the source CP-CN 2508 sends an Anchoring Request to the target CP-CN 2510. At 3, the target CP-CN 2510 may initiate a key material exchange or derivation procedure with the SEA 2514. As part of the procedure, the SEA 2514 may update security context (SC) of the UE 2502 with target CP-CN and target AN information. The SEA 2514 may push the SC to the SCM 2512, which derives/exchanges with the target CP-CN 2510 a specific security context. At 4, the SCM 2512 may derive/exchange with the target AN 2506, a specific security context for the AN 2506. At 5, the target CP-CN 2510 may send an Anchoring Response to the source CP-CN 2508. At 6, the source CP-CN 2508 may send a handover command to the source AN 2504, which propagates to the UE 2502. The command may carry target CP-CN/AN information. At 7, in accordance with the example, the UE 2502 derives keys for the target CP-CN 2510 and the AN 2506. At 8, using derived keys, the UE 2502 may send a protected handover confirmation message to the target AN 2506.

[00148] It is recognized herein that tracking area update is another example scenario that may impact SEA procedures. A difference in an example tracking area update as compared to the procedure described with reference to Fig. 25 is that, from the SEA/SCM point of view, in an example, only a security context update towards the target CP-CN 2510 is required. The CP-CN 2510 may then be responsible for propagating updates of the security parameters toward the UE 2502 (e.g., algorithms, trigger a new key derivation) using, for e.g., a NAS procedure such as LTE NAS Security Mode Command procedures.

[00149] Referring to Fig. 26, in an example, the Security Context is transported when roaming and/or is handed-off between slices in a roaming scenario. In an example roaming scenario, a first SEA 2602 (SEA_h) is in a home network 2604 (Operator 1), and a second SEA 2606 (SEA_v) is in a visited network 2608 (Operator 2). After a UE 2610 has performed a full authentication with the AUS 2612, any further authentications may be performed by the SEA 2606 (SEA_v), which is used to perform local authentications within the visited network 2608. In some

cases, the SEA 2602 (SEA_h) may be involved only if the security context (SCSEAv) has expired. It is recognized herein that performing authentications within the home operator network 2604 may become expensive, and may also increase the latencies because the connection between the home network 2604 and the visited network 2608 may be connected using a wide-area network (e.g. public Internet). In an example, communications between the first SEA 2602 (SEA_h) and the second SEA 2606 (SEA_v) are carried over a secure communications channel, which may not be necessary when the authentications are performed locally within the visited network 2608, thus avoiding expensive encryption and decryption processes. Optionally, in some cases, the SEA may remain within the home network 2604, while the SCM 2614 can be deployed within the visited network 2608. In some examples, any re-authentication with the SEA is still handled remotely within the home network 2604. However, the SCM 2614 that is deployed in the visited network 2608 can be used to manage the security contexts locally.

[00150] Referring to Fig. 27, an example scenario in which a UE 2702 may request for simultaneous access to multiple slices at the same time is described. The method may leverage the same primary SCP that was established as part of an initial authentication. Fig. 27 illustrates handoff between a first slice (Slice 1) and a second slice (Slice 2), or when the UE 2702 is provided to access to two or more slices simultaneously. At 1, in accordance with the illustrated example, the UE 2702 requests access to Slice 1. At 2, the UE 2702 performs authentication with an AUS 2716 using any type of authentication mechanisms, such as EPS-AKA, EAP-AKA', EAP-TLS, EAP-TTLS, TLS, IPSec or Open-Id. The subscriber profile may be identified by an IMSI, a user id, a biometric-id, or a device id, for example. The UE 2702 or a user that uses the UE 2702 may perform mutual authentication with the AUS 2716. If a successful authentication was carried out, then resulting keying material that was generated may be used to create, for example: Security Context (SC_p), a Temporary-ID, Associated Session Keys that will be used as a Master Session Key, and an SC_{SEA}. At 3, the AUS 2716 may send the SC_{SEA} and keying material to the SEA 2714. At 4, in accordance with the illustrated example, the SEA 2714 uses the SC_{SEA} to generate credentials in conjunction with the UE 2702. The mechanisms by which the keys are generated may be based upon the level of isolation that is required and may use low, intermediate or complete isolation mechanisms to generate the credentials. It is assumed in this example that appropriate cryptographic material has been pre-provisioned for the desired level of isolation. Upon generating the keys and optionally verifying key possession, the SEA 2714 may generate appropriate security context, $SC_{SL1\text{-}CN}$ for the CN 2710 in Slice 1 and $SC_{SL1\text{-}CN}$ for the AN 2708 in Slice 1.

[00151] Still referring to Fig. 27, in accordance with the example, at 5, the SEA 2714 or SCM 2712 sends the $SC_{SL1\text{-}CN}$ and keying material to an anchor function in the CN 2710 of Slice 1.

At 6, the CN 2710 within Slice 1 and the UE 2702 generate appropriate credentials in conjunction with each other, using the keying material and the $SC_{SL1\text{-}CN}$. The UE 2702 may also be provisioned with an IP address that belongs to the same network as CN 2710. Similarly, at 7, the SEA 2714 or SCM 2712 may send the $SC_{SL1\text{-}AN}$ and keying material to an anchor function in the AN 2708 of Slice 1. At 8, the AN 2708 within Slice1 and the UE 2702 may generate appropriate credentials using the keying material and the $SC_{SL1\text{-}AN}$. At 9, due to physical mobility of the UE 2702, load-balancing reasons, or a request for another slice by an application within the UE 2702, for example, a handoff to another slice (e.g., Slice 2) or simultaneous access to another slice (e.g., Slice 2) is performed. If the access to a slice or handoff to the slice is of higher risk, then appropriate full authentication may have to be carried out. In this example, AUS-level authentication is required, and the new access request or handoff request is handled by means of bootstrapping from the full authentication that was performed at 2. At 10, the SEA 2714 or SCM 2712 may generate a new security context, $SC_{SL2\text{-}CN}$, and provide appropriate keying material to an anchor function in the CN 2706 within Slice 2. At 11, the UE 2702 and the CN 2706 within Slice 2 may credentials using keying material and $SC_{SL2\text{-}CN}$. At 12, the SEA 2714 or the SCM 2712 may generate a security context, $SC_{SL2\text{-}AN}$, and provide appropriate keying material to an anchor function in the AN 2704 within Slice 2. At 13, the UE 2702 and the AN 2704 within Slice 2 may generate credentials using keying material and $SC_{SL2\text{-}AN}$.

[00152] Referring also to Fig. 31, an example key hierarchy when the UE 2702 is either being handed-off to a new slice or when the UE 2702 would require the instantiation of a new slice is illustrated. At the SEA/SCM, a first key K_{SEA} or a second key K'_{SEA} may be used as the root credential for generation of the rest of credentials within the hierarchy. In certain cases, when the K_{SEA} is nearing expiration for example, then K'_{SEA} may be used as the root credential or else a new K_{SEA} and optionally a new K'_{SEA} may have to be generated using K_{AUS}. It will be understood that the credential generation process illustrated in Fig. 31 is an example key generation process, and the key hierarchy may be generated using hierarchical, partially hierarchical, or a combination thereof for AN/CN as described above for each slice. In addition, with respect to each slice, the mechanisms may be different for certain use cases, where access to an AN by SEA might not be possible, for example, because the AN may be tightly controlled by a radio network provider (e.g., 5G base station). In such cases, by way of example, it may be possible for Slice 1 to follow the hierarchical key generation process while the Slice 2 may follow the Partially Hierarchical key generation process.

[00153] Referring now to Fig. 28, a NextGen system may interoperate with an application/service provider or an IdM system. In some cases, an operator 2802 may off-load

certain authentication procedures to a third-party. For example, the operator 2802 may offload all authentication or some authentication procedures, particularly in the case of multi-factor authentication procedures.

[00154] As illustrated in Fig. 28, the operator 2802 may bind the authentications carried out by an OTT 2804 and the operator 2802, and the associated keying material may be bound at the SEA 2806. The SEA 2806 may communicate with the AUS 2808 in the OTT network 2804 using the NG5 interface. The UE 2810 may be mutually authenticated with the AUS 2808 in OTT 2804, which may follow similar high-level procedures as those described with reference to Fig. 27 (e.g., at 2). The authentication factors and methods however, may be different (e.g., user or device authentication, by means of biometrics, public key respectively, etc.) from those carried out between the AUS 2812 in the operator 2802 and the UE 2810. The authentication and the credentials (e.g., keys) that result from the authentication may be bound together procedurally or cryptographically. Any keying material and credentials that are generated by the SEA 2806 and other entities (e.g., CN, AN) may inherit the key hierarchy based on the bounded credentials.

[00155] Fig. 31A is a diagram illustrating an example communications system 100 in which one or more disclosed embodiments may be implemented. The communications system 100 may be a multiple access system that provides content, such as voice, data, video, messaging, broadcast, etc., to multiple wireless users. The communications system 100 may enable multiple wireless users to access such content through the sharing of system resources, including wireless bandwidth. For example, the communications systems 100 may employ one or more channel access methods, such as code division multiple access (CDMA), time division multiple access (TDMA), frequency division multiple access (FDMA), orthogonal FDMA (OFDMA), single-carrier FDMA (SC-FDMA), zero-tail unique-word DFT-Spread OFDM (ZT UW DTS-s OFDM), unique word OFDM (UW-OFDM), resource block-filtered OFDM, filter bank multicarrier (FBMC), and the like.

[00156] As shown in Fig. 31A, the communications system 100 may include wireless transmit/receive units (WTRUs) 102a, 102b, 102c, 102d, a RAN 104/113, a CN 106/115, a public switched telephone network (PSTN) 108, the Internet 110, and other networks 112, though it will be appreciated that the disclosed embodiments contemplate any number of WTRUs, base stations, networks, and/or network elements. Each of the WTRUs 102a, 102b, 102c, 102d may be any type of device configured to operate and/or communicate in a wireless environment. By way of example, the WTRUs 102a, 102b, 102c, 102d, any of which may be referred to as a "station" and/or a "STA", may be configured to transmit and/or receive wireless signals and may include a user equipment (UE), a mobile station, a fixed or mobile subscriber unit, a subscription-based unit, a pager, a cellular telephone, a personal digital assistant (PDA), a smartphone, a laptop, a netbook, a personal

computer, a wireless sensor, a hotspot or Mi-Fi device, an Internet of Things (IoT) device, a watch or other wearable, a head-mounted display (HMD), a vehicle, a drone, a medical device and applications (e.g., remote surgery), an industrial device and applications (e.g., a robot and/or other wireless devices operating in an industrial and/or an automated processing chain contexts), a consumer electronics device, a device operating on commercial and/or industrial wireless networks, and the like. Any of the WTRUs 102a, 102b, 102c and 102d may be interchangeably referred to as a UE.

[00157] The communications systems 100 may also include a base station 114a and/or a base station 114b. Each of the base stations 114a, 114b may be any type of device configured to wirelessly interface with at least one of the WTRUs 102a, 102b, 102c, 102d to facilitate access to one or more communication networks, such as the CN 106/115, the Internet 110, and/or the other networks 112. By way of example, the base stations 114a, 114b may be a base transceiver station (BTS), a Node-B, an eNode B, a Home Node B, a Home eNode B, a gNB, a NR NodeB, a site controller, an access point (AP), a wireless router, and the like. While the base stations 114a, 114b are each depicted as a single element, it will be appreciated that the base stations 114a, 114b may include any number of interconnected base stations and/or network elements.

[00158] The base station 114a may be part of the RAN 104/113, which may also include other base stations and/or network elements (not shown), such as a base station controller (BSC), a radio network controller (RNC), relay nodes, etc. The base station 114a and/or the base station 114b may be configured to transmit and/or receive wireless signals on one or more carrier frequencies, which may be referred to as a cell (not shown). These frequencies may be in licensed spectrum, unlicensed spectrum, or a combination of licensed and unlicensed spectrum. A cell may provide coverage for a wireless service to a specific geographical area that may be relatively fixed or that may change over time. The cell may further be divided into cell sectors. For example, the cell associated with the base station 114a may be divided into three sectors. Thus, in one embodiment, the base station 114a may include three transceivers, i.e., one for each sector of the cell. In an embodiment, the base station 114a may employ multiple-input multiple output (MIMO) technology and may utilize multiple transceivers for each sector of the cell. For example, beamforming may be used to transmit and/or receive signals in desired spatial directions.

[00159] The base stations 114a, 114b may communicate with one or more of the WTRUs 102a, 102b, 102c, 102d over an air interface 116, which may be any suitable wireless communication link (e.g., radio frequency (RF), microwave, centimeter wave, micrometer wave, infrared (IR), ultraviolet (UV), visible light, etc.). The air interface 116 may be established using any suitable radio access technology (RAT).

[00160] More specifically, as noted above, the communications system 100 may be a multiple access system and may employ one or more channel access schemes, such as CDMA, TDMA, FDMA, OFDMA, SC-FDMA, and the like. For example, the base station 114a in the RAN 104/113 and the WTRUs 102a, 102b, 102c may implement a radio technology such as Universal Mobile Telecommunications System (UMTS) Terrestrial Radio Access (UTRA), which may establish the air interface 115/116/117 using wideband CDMA (WCDMA). WCDMA may include communication protocols such as High-Speed Packet Access (HSPA) and/or Evolved HSPA (HSPA+). HSPA may include High-Speed Downlink (DL) Packet Access (HSDPA) and/or High-Speed UL Packet Access (HSUPA).

[00161] In an embodiment, the base station 114a and the WTRUs 102a, 102b, 102c may implement a radio technology such as Evolved UMTS Terrestrial Radio Access (E-UTRA), which may establish the air interface 116 using Long Term Evolution (LTE) and/or LTE-Advanced (LTE-A) and/or LTE-Advanced Pro (LTE-A Pro).

[00162] In an embodiment, the base station 114a and the WTRUs 102a, 102b, 102c may implement a radio technology such as NR Radio Access , which may establish the air interface 116 using New Radio (NR).

[00163] In an embodiment, the base station 114a and the WTRUs 102a, 102b, 102c may implement multiple radio access technologies. For example, the base station 114a and the WTRUs 102a, 102b, 102c may implement LTE radio access and NR radio access together, for instance using dual connectivity (DC) principles. Thus, the air interface utilized by WTRUs 102a, 102b, 102c may be characterized by multiple types of radio access technologies and/or transmissions sent to/from multiple types of base stations (e.g., a eNB and a gNB).

[00164] In other embodiments, the base station 114a and the WTRUs 102a, 102b, 102c may implement radio technologies such as IEEE 802.11 (i.e., Wireless Fidelity (WiFi), IEEE 802.16 (i.e., Worldwide Interoperability for Microwave Access (WiMAX)), CDMA2000, CDMA2000 1X, CDMA2000 EV-DO, Interim Standard 2000 (IS-2000), Interim Standard 95 (IS-95), Interim Standard 856 (IS-856), Global System for Mobile communications (GSM), Enhanced Data rates for GSM Evolution (EDGE), GSM EDGE (GERAN), and the like.

[00165] The base station 114b in Fig. 31A may be a wireless router, Home Node B, Home eNode B, or access point, for example, and may utilize any suitable RAT for facilitating wireless connectivity in a localized area, such as a place of business, a home, a vehicle, a campus, an industrial facility, an air corridor (e.g., for use by drones), a roadway, and the like. In one embodiment, the base station 114b and the WTRUs 102c, 102d may implement a radio technology such as IEEE 802.11 to establish a wireless local area network (WLAN). In an embodiment, the

base station 114b and the WTRUs 102c, 102d may implement a radio technology such as IEEE 802.15 to establish a wireless personal area network (WPAN). In yet another embodiment, the base station 114b and the WTRUs 102c, 102d may utilize a cellular-based RAT (e.g., WCDMA, CDMA2000, GSM, LTE, LTE-A, LTE-A Pro, NR etc.) to establish a picocell or femtocell. As shown in Fig. 31A, the base station 114b may have a direct connection to the Internet 110. Thus, the base station 114b may not be required to access the Internet 110 via the CN 106/115.

[00166] The RAN 104/113 may be in communication with the CN 106/115, which may be any type of network configured to provide voice, data, applications, and/or voice over internet protocol (VoIP) services to one or more of the WTRUs 102a, 102b, 102c, 102d. The data may have varying quality of service (QoS) requirements, such as differing throughput requirements, latency requirements, error tolerance requirements, reliability requirements, data throughput requirements, mobility requirements, and the like. The CN 106/115 may provide call control, billing services, mobile location-based services, pre-paid calling, Internet connectivity, video distribution, etc., and/or perform high-level security functions, such as user authentication. Although not shown in Fig. 31A, it will be appreciated that the RAN 104/113 and/or the CN 106/115 may be in direct or indirect communication with other RANs that employ the same RAT as the RAN 104/113 or a different RAT. For example, in addition to being connected to the RAN 104/113, which may be utilizing a NR radio technology, the CN 106/115 may also be in communication with another RAN (not shown) employing a GSM, UMTS, CDMA 2000, WiMAX, E-UTRA, or WiFi radio technology.

[00167] The CN 106/115 may also serve as a gateway for the WTRUs 102a, 102b, 102c, 102d to access the PSTN 108, the Internet 110, and/or the other networks 112. The PSTN 108 may include circuit-switched telephone networks that provide plain old telephone service (POTS). The Internet 110 may include a global system of interconnected computer networks and devices that use common communication protocols, such as the transmission control protocol (TCP), user datagram protocol (UDP) and/or the internet protocol (IP) in the TCP/IP internet protocol suite. The networks 112 may include wired and/or wireless communications networks owned and/or operated by other service providers. For example, the networks 112 may include another CN connected to one or more RANs, which may employ the same RAT as the RAN 104/113 or a different RAT.

[00168] Some or all of the WTRUs 102a, 102b, 102c, 102d in the communications system 100 may include multi-mode capabilities (e.g., the WTRUs 102a, 102b, 102c, 102d may include multiple transceivers for communicating with different wireless networks over different wireless links). For example, the WTRU 102c shown in Fig. 31A may be configured to

communicate with the base station 114a, which may employ a cellular-based radio technology, and with the base station 114b, which may employ an IEEE 802 radio technology.

[00169] Fig. 31B is a system diagram illustrating an example WTRU 102. As shown in Fig. 31B, the WTRU 102 may include a processor 118, a transceiver 120, a transmit/receive element 122, a speaker/microphone 124, a keypad 126, a display/touchpad 128, non-removable memory 130, removable memory 132, a power source 134, a global positioning system (GPS) chipset 136, and/or other peripherals 138, among others. It will be appreciated that the WTRU 102 may include any sub-combination of the foregoing elements while remaining consistent with an embodiment.

[00170] The processor 118 may be a general purpose processor, a special purpose processor, a conventional processor, a digital signal processor (DSP), a plurality of microprocessors, one or more microprocessors in association with a DSP core, a controller, a microcontroller, Application Specific Integrated Circuits (ASICs), Field Programmable Gate Arrays (FPGAs) circuits, any other type of integrated circuit (IC), a state machine, and the like. The processor 118 may perform signal coding, data processing, power control, input/output processing, and/or any other functionality that enables the WTRU 102 to operate in a wireless environment. The processor 118 may be coupled to the transceiver 120, which may be coupled to the transmit/receive element 122. While Fig. 31B depicts the processor 118 and the transceiver 120 as separate components, it will be appreciated that the processor 118 and the transceiver 120 may be integrated together in an electronic package or chip.

[00171] The transmit/receive element 122 may be configured to transmit signals to, or receive signals from, a base station (e.g., the base station 114a) over the air interface 116. For example, in one embodiment, the transmit/receive element 122 may be an antenna configured to transmit and/or receive RF signals. In an embodiment, the transmit/receive element 122 may be an emitter/detector configured to transmit and/or receive IR, UV, or visible light signals, for example. In yet another embodiment, the transmit/receive element 122 may be configured to transmit and/or receive both RF and light signals. It will be appreciated that the transmit/receive element 122 may be configured to transmit and/or receive any combination of wireless signals.

[00172] Although the transmit/receive element 122 is depicted in Fig. 31B as a single element, the WTRU 102 may include any number of transmit/receive elements 122. More specifically, the WTRU 102 may employ MIMO technology. Thus, in one embodiment, the WTRU 102 may include two or more transmit/receive elements 122 (e.g., multiple antennas) for transmitting and receiving wireless signals over the air interface 116.

[00173] The transceiver 120 may be configured to modulate the signals that are to be transmitted by the transmit/receive element 122 and to demodulate the signals that are received by the transmit/receive element 122. As noted above, the WTRU 102 may have multi-mode capabilities. Thus, the transceiver 120 may include multiple transceivers for enabling the WTRU 102 to communicate via multiple RATs, such as NR and IEEE 802.11, for example.

[00174] The processor 118 of the WTRU 102 may be coupled to, and may receive user input data from, the speaker/microphone 124, the keypad 126, and/or the display/touchpad 128 (e.g., a liquid crystal display (LCD) display unit or organic light-emitting diode (OLED) display unit). The processor 118 may also output user data to the speaker/microphone 124, the keypad 126, and/or the display/touchpad 128. In addition, the processor 118 may access information from, and store data in, any type of suitable memory, such as the non-removable memory 130 and/or the removable memory 132. The non-removable memory 130 may include random-access memory (RAM), read-only memory (ROM), a hard disk, or any other type of memory storage device. The removable memory 132 may include a subscriber identity module (SIM) card, a memory stick, a secure digital (SD) memory card, and the like. In other embodiments, the processor 118 may access information from, and store data in, memory that is not physically located on the WTRU 102, such as on a server or a home computer (not shown).

[00175] The processor 118 may receive power from the power source 134, and may be configured to distribute and/or control the power to the other components in the WTRU 102. The power source 134 may be any suitable device for powering the WTRU 102. For example, the power source 134 may include one or more dry cell batteries (e.g., nickel-cadmium (NiCd), nickel-zinc (NiZn), nickel metal hydride (NiMH), lithium-ion (Li-ion), etc.), solar cells, fuel cells, and the like.

[00176] The processor 118 may also be coupled to the GPS chipset 136, which may be configured to provide location information (e.g., longitude and latitude) regarding the current location of the WTRU 102. In addition to, or in lieu of, the information from the GPS chipset 136, the WTRU 102 may receive location information over the air interface 116 from a base station (e.g., base stations 114a, 114b) and/or determine its location based on the timing of the signals being received from two or more nearby base stations. It will be appreciated that the WTRU 102 may acquire location information by way of any suitable location-determination method while remaining consistent with an embodiment.

[00177] The processor 118 may further be coupled to other peripherals 138, which may include one or more software and/or hardware modules that provide additional features, functionality and/or wired or wireless connectivity. For example, the peripherals 138 may include an accelerometer, an e-compass, a satellite transceiver, a digital camera (for photographs and/or

video), a universal serial bus (USB) port, a vibration device, a television transceiver, a hands free headset, a Bluetooth® module, a frequency modulated (FM) radio unit, a digital music player, a media player, a video game player module, an Internet browser, a Virtual Reality and/or Augmented Reality (VR/AR) device, an activity tracker, and the like. The peripherals 138 may include one or more sensors, the sensors may be one or more of a gyroscope, an accelerometer, a hall effect sensor, a magnetometer, an orientation sensor, a proximity sensor, a temperature sensor, a time sensor; a geolocation sensor; an altimeter, a light sensor, a touch sensor, a magnetometer, a barometer, a gesture sensor, a biometric sensor, and/or a humidity sensor.

[00178] The WTRU 102 may include a full duplex radio for which transmission and reception of some or all of the signals (e.g., associated with particular subframes for both the UL (e.g., for transmission) and downlink (e.g., for reception) may be concurrent and/or simultaneous. The full duplex radio may include an interference management unit 139 to reduce and or substantially eliminate self-interference via either hardware (e.g., a choke) or signal processing via a processor (e.g., a separate processor (not shown) or via processor 118). In an embodiment, the WRTU 102 may include a half-duplex radio for which transmission and reception of some or all of the signals (e.g., associated with particular subframes for either the UL (e.g., for transmission) or the downlink (e.g., for reception)).

[00179] Fig. 31C is a system diagram illustrating the RAN 104 and the CN 106 according to an embodiment. As noted above, the RAN 104 may employ an E-UTRA radio technology to communicate with the WTRUs 102a, 102b, 102c over the air interface 116. The RAN 104 may also be in communication with the CN 106.

[00180] The RAN 104 may include eNode-Bs 160a, 160b, 160c, though it will be appreciated that the RAN 104 may include any number of eNode-Bs while remaining consistent with an embodiment. The eNode-Bs 160a, 160b, 160c may each include one or more transceivers for communicating with the WTRUs 102a, 102b, 102c over the air interface 116. In one embodiment, the eNode-Bs 160a, 160b, 160c may implement MIMO technology. Thus, the eNode-B 160a, for example, may use multiple antennas to transmit wireless signals to, and/or receive wireless signals from, the WTRU 102a.

[00181] Each of the eNode-Bs 160a, 160b, 160c may be associated with a particular cell (not shown) and may be configured to handle radio resource management decisions, handover decisions, scheduling of users in the UL and/or DL, and the like. As shown in Fig. 31C, the eNode-Bs 160a, 160b, 160c may communicate with one another over an X2 interface.

[00182] The CN 106 shown in Fig. 31C may include a mobility management entity (MME) 162, a serving gateway (SGW) 164, and a packet data network (PDN) gateway (or PGW)

166. While each of the foregoing elements are depicted as part of the CN 106, it will be appreciated that any of these elements may be owned and/or operated by an entity other than the CN operator.

[00183] The MME 162 may be connected to each of the eNode-Bs 162a, 162b, 162c in the RAN 104 via an S1 interface and may serve as a control node. For example, the MME 162 may be responsible for authenticating users of the WTRUs 102a, 102b, 102c, bearer activation/deactivation, selecting a particular serving gateway during an initial attach of the WTRUs 102a, 102b, 102c, and the like. The MME 162 may provide a control plane function for switching between the RAN 104 and other RANs (not shown) that employ other radio technologies, such as GSM and/or WCDMA.

[00184] The SGW 164 may be connected to each of the eNode Bs 160a, 160b, 160c in the RAN 104 via the S1 interface. The SGW 164 may generally route and forward user data packets to/from the WTRUs 102a, 102b, 102c. The SGW 164 may perform other functions, such as anchoring user planes during inter-eNode B handovers, triggering paging when DL data is available for the WTRUs 102a, 102b, 102c, managing and storing contexts of the WTRUs 102a, 102b, 102c, and the like.

[00185] The SGW 164 may be connected to the PGW 166, which may provide the WTRUs 102a, 102b, 102c with access to packet-switched networks, such as the Internet 110, to facilitate communications between the WTRUs 102a, 102b, 102c and IP-enabled devices.

[00186] The CN 106 may facilitate communications with other networks. For example, the CN 106 may provide the WTRUs 102a, 102b, 102c with access to circuit-switched networks, such as the PSTN 108, to facilitate communications between the WTRUs 102a, 102b, 102c and traditional land-line communications devices. For example, the CN 106 may include, or may communicate with, an IP gateway (e.g., an IP multimedia subsystem (IMS) server) that serves as an interface between the CN 106 and the PSTN 108. In addition, the CN 106 may provide the WTRUs 102a, 102b, 102c with access to the other networks 112, which may include other wired and/or wireless networks that are owned and/or operated by other service providers.

[00187] Although the WTRU is described in Fig. 31A-31D as a wireless terminal, it is contemplated that in certain representative embodiments that such a terminal may use (e.g., temporarily or permanently) wired communication interfaces with the communication network.

[00188] In representative embodiments, the other network 112 may be a WLAN.

[00189] A WLAN in Infrastructure Basic Service Set (BSS) mode may have an Access Point (AP) for the BSS and one or more stations (STAs) associated with the AP. The AP may have an access or an interface to a Distribution System (DS) or another type of wired/wireless network that carries traffic in to and/or out of the BSS. Traffic to STAs that originates from outside the BSS

may arrive through the AP and may be delivered to the STAs. Traffic originating from STAs to destinations outside the BSS may be sent to the AP to be delivered to respective destinations. Traffic between STAs within the BSS may be sent through the AP, for example, where the source STA may send traffic to the AP and the AP may deliver the traffic to the destination STA. The traffic between STAs within a BSS may be considered and/or referred to as peer-to-peer traffic. The peer-to-peer traffic may be sent between (e.g., directly between) the source and destination STAs with a direct link setup (DLS). In certain representative embodiments, the DLS may use an 802.11e DLS or an 802.11z tunneled DLS (TDLS). A WLAN using an Independent BSS (IBSS) mode may not have an AP, and the STAs (e.g., all of the STAs) within or using the IBSS may communicate directly with each other. The IBSS mode of communication may sometimes be referred to herein as an "ad-hoc" mode of communication.

[00190] When using the 802.11ac infrastructure mode of operation or a similar mode of operations, the AP may transmit a beacon on a fixed channel, such as a primary channel. The primary channel may be a fixed width (e.g., 20 MHz wide bandwidth) or a dynamically set width via signaling. The primary channel may be the operating channel of the BSS and may be used by the STAs to establish a connection with the AP. In certain representative embodiments, Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA) may be implemented, for example in in 802.11 systems. For CSMA/CA, the STAs (e.g., every STA), including the AP, may sense the primary channel. If the primary channel is sensed/detected and/or determined to be busy by a particular STA, the particular STA may back off. One STA (e.g., only one station) may transmit at any given time in a given BSS.

[00191] High Throughput (HT) STAs may use a 40 MHz wide channel for communication, for example, via a combination of the primary 20 MHz channel with an adjacent or nonadjacent 20 MHz channel to form a 40 MHz wide channel.

[00192] Very High Throughput (VHT) STAs may support 20MHz, 40 MHz, 80 MHz, and/or 160 MHz wide channels. The 40 MHz, and/or 80 MHz, channels may be formed by combining contiguous 20 MHz channels. A 160 MHz channel may be formed by combining 8 contiguous 20 MHz channels, or by combining two non-contiguous 80 MHz channels, which may be referred to as an 80+80 configuration. For the 80+80 configuration, the data, after channel encoding, may be passed through a segment parser that may divide the data into two streams. Inverse Fast Fourier Transform (IFFT) processing, and time domain processing, may be done on each stream separately. The streams may be mapped on to the two 80 MHz channels, and the data may be transmitted by a transmitting STA. At the receiver of the receiving STA, the above

described operation for the 80+80 configuration may be reversed, and the combined data may be sent to the Medium Access Control (MAC).

[00193] Sub 1 GHz modes of operation are supported by 802.11af and 802.11ah. The channel operating bandwidths, and carriers, are reduced in 802.11af and 802.11ah relative to those used in 802.11n, and 802.11ac. 802.11af supports 5 MHz, 10 MHz and 20 MHz bandwidths in the TV White Space (TVWS) spectrum, and 802.11ah supports 1 MHz, 2 MHz, 4 MHz, 8 MHz, and 16 MHz bandwidths using non-TVWS spectrum. According to a representative embodiment, 802.11ah may support Meter Type Control/Machine-Type Communications, such as MTC devices in a macro coverage area. MTC devices may have certain capabilities, for example, limited capabilities including support for (e.g., only support for) certain and/or limited bandwidths. The MTC devices may include a battery with a battery life above a threshold (e.g., to maintain a very long battery life).

[00194] WLAN systems, which may support multiple channels, and channel bandwidths, such as 802.11n, 802.11ac, 802.11af, and 802.11ah, include a channel which may be designated as the primary channel. The primary channel may have a bandwidth equal to the largest common operating bandwidth supported by all STAs in the BSS. The bandwidth of the primary channel may be set and/or limited by a STA, from among all STAs in operating in a BSS, which supports the smallest bandwidth operating mode. In the example of 802.11ah, the primary channel may be 1 MHz wide for STAs (e.g., MTC type devices) that support (e.g., only support) a 1 MHz mode, even if the AP, and other STAs in the BSS support 2 MHz, 4 MHz, 8 MHz, 16 MHz, and/or other channel bandwidth operating modes. Carrier sensing and/or Network Allocation Vector (NAV) settings may depend on the status of the primary channel. If the primary channel is busy, for example, due to a STA (which supports only a 1 MHz operating mode), transmitting to the AP, the entire available frequency bands may be considered busy even though a majority of the frequency bands remains idle and may be available.

[00195] In the United States, the available frequency bands, which may be used by 802.11ah, are from 902 MHz to 928 MHz. In Korea, the available frequency bands are from 917.5 MHz to 923.5 MHz. In Japan, the available frequency bands are from 916.5 MHz to 927.5 MHz. The total bandwidth available for 802.11ah is 6 MHz to 26 MHz depending on the country code.

[00196] Fig. 31D is a system diagram illustrating the RAN 113 and the CN 115 according to an embodiment. As noted above, the RAN 113 may employ an NR radio technology to communicate with the WTRUs 102a, 102b, 102c over the air interface 116. The RAN 113 may also be in communication with the CN 115.

[00197] The RAN 113 may include gNBs 180a, 180b, 180c, though it will be appreciated that the RAN 113 may include any number of gNBs while remaining consistent with an embodiment. The gNBs 180a, 180b, 180c may each include one or more transceivers for communicating with the WTRUs 102a, 102b, 102c over the air interface 116. In one embodiment, the gNBs 180a, 180b, 180c may implement MIMO technology. For example, gNBs 180a, 108b may utilize beamforming to transmit signals to and/or receive signals from the gNBs 180a, 180b, 180c. Thus, the gNB 180a, for example, may use multiple antennas to transmit wireless signals to, and/or receive wireless signals from, the WTRU 102a. In an embodiment, the gNBs 180a, 180b, 180c may implement carrier aggregation technology. For example, the gNB 180a may transmit multiple component carriers to the WTRU 102a (not shown). A subset of these component carriers may be on unlicensed spectrum while the remaining component carriers may be on licensed spectrum. In an embodiment, the gNBs 180a, 180b, 180c may implement Coordinated Multi-Point (CoMP) technology. For example, WTRU 102a may receive coordinated transmissions from gNB 180a and gNB 180b (and/or gNB 180c).

[00198] The WTRUs 102a, 102b, 102c may communicate with gNBs 180a, 180b, 180c using transmissions associated with a scalable numerology. For example, the OFDM symbol spacing and/or OFDM subcarrier spacing may vary for different transmissions, different cells, and/or different portions of the wireless transmission spectrum. The WTRUs 102a, 102b, 102c may communicate with gNBs 180a, 180b, 180c using subframe or transmission time intervals (TTIs) of various or scalable lengths (e.g., containing varying number of OFDM symbols and/or lasting varying lengths of absolute time).

[00199] The gNBs 180a, 180b, 180c may be configured to communicate with the WTRUs 102a, 102b, 102c in a standalone configuration and/or a non-standalone configuration. In the standalone configuration, WTRUs 102a, 102b, 102c may communicate with gNBs 180a, 180b, 180c without also accessing other RANs (e.g., such as eNode-Bs 160a, 160b, 160c). In the standalone configuration, WTRUs 102a, 102b, 102c may utilize one or more of gNBs 180a, 180b, 180c as a mobility anchor point. In the standalone configuration, WTRUs 102a, 102b, 102c may communicate with gNBs 180a, 180b, 180c using signals in an unlicensed band. In a non-standalone configuration WTRUs 102a, 102b, 102c may communicate with/connect to gNBs 180a, 180b, 180c while also communicating with/connecting to another RAN such as eNode-Bs 160a, 160b, 160c. For example, WTRUs 102a, 102b, 102c may implement DC principles to communicate with one or more gNBs 180a, 180b, 180c and one or more eNode-Bs 160a, 160b, 160c substantially simultaneously. In the non-standalone configuration, eNode-Bs 160a, 160b, 160c may serve as a

mobility anchor for WTRUs 102a, 102b, 102c and gNBs 180a, 180b, 180c may provide additional coverage and/or throughput for servicing WTRUs 102a, 102b, 102c.

[00200] Each of the gNBs 180a, 180b, 180c may be associated with a particular cell (not shown) and may be configured to handle radio resource management decisions, handover decisions, scheduling of users in the UL and/or DL, support of network slicing, dual connectivity, interworking between NR and E-UTRA, routing of user plane data towards User Plane Function (UPF) 184a, 184b, routing of control plane information towards Access and Mobility Management Function (AMF) 182a, 182b and the like. As shown in Fig. 31D, the gNBs 180a, 180b, 180c may communicate with one another over an Xn interface.

[00201] The CN 115 shown in Fig. 31D may include at least one AMF 182a, 182b, at least one UPF 184a, 184b, at least one Session Management Function (SMF) 183a, 183b, and possibly a Data Network (DN) 185a, 185b. While each of the foregoing elements are depicted as part of the CN 115, it will be appreciated that any of these elements may be owned and/or operated by an entity other than the CN operator.

[00202] The AMF 182a, 182b may be connected to one or more of the gNBs 180a, 180b, 180c in the RAN 113 via an N2 interface and may serve as a control node. For example, the AMF 182a, 182b may be responsible for authenticating users of the WTRUs 102a, 102b, 102c, support for network slicing (e.g., handling of different PDU sessions with different requirements), selecting a particular SMF 183a, 183b, management of the registration area, termination of NAS signaling, mobility management, and the like. Network slicing may be used by the AMF 182a, 182b in order to customize CN support for WTRUs 102a, 102b, 102c based on the types of services being utilized WTRUs 102a, 102b, 102c. For example, different network slices may be established for different use cases such as services relying on ultra-reliable low latency (URLLC) access, services relying on enhanced massive mobile broadband (eMBB) access, services for machine type communication (MTC) access, and/or the like. The AMF 162 may provide a control plane function for switching between the RAN 113 and other RANs (not shown) that employ other radio technologies, such as LTE, LTE-A, LTE-A Pro, and/or non-3GPP access technologies such as WiFi.

[00203] The SMF 183a, 183b may be connected to an AMF 182a, 182b in the CN 115 via an N11 interface. The SMF 183a, 183b may also be connected to a UPF 184a, 184b in the CN 115 via an N4 interface. The SMF 183a, 183b may select and control the UPF 184a, 184b and configure the routing of traffic through the UPF 184a, 184b. The SMF 183a, 183b may perform other functions, such as managing and allocating UE IP address, managing PDU sessions, controlling policy enforcement and QoS, providing downlink data notifications, and the like. A PDU session type may be IP-based, non-IP based, Ethernet-based, and the like.

[00204] The UPF 184a, 184b may be connected to one or more of the gNBs 180a, 180b, 180c in the RAN 113 via an N3 interface, which may provide the WTRUs 102a, 102b, 102c with access to packet-switched networks, such as the Internet 110, to facilitate communications between the WTRUs 102a, 102b, 102c and IP-enabled devices. The UPF 184, 184b may perform other functions, such as routing and forwarding packets, enforcing user plane policies, supporting multi-homed PDU sessions, handling user plane QoS, buffering downlink packets, providing mobility anchoring, and the like.

[00205] The CN 115 may facilitate communications with other networks. For example, the CN 115 may include, or may communicate with, an IP gateway (e.g., an IP multimedia subsystem (IMS) server) that serves as an interface between the CN 115 and the PSTN 108. In addition, the CN 115 may provide the WTRUs 102a, 102b, 102c with access to the other networks 112, which may include other wired and/or wireless networks that are owned and/or operated by other service providers. In one embodiment, the WTRUs 102a, 102b, 102c may be connected to a local Data Network (DN) 185a, 185b through the UPF 184a, 184b via the N3 interface to the UPF 184a, 184b and an N6 interface between the UPF 184a, 184b and the DN 185a, 185b.

[00206] In view of Figs. 31A-31D, and the corresponding description of Figs. 31A-31D, one or more, or all, of the functions described herein with regard to one or more of: WTRU 102a-d, Base Station 114a-b, eNode-B 160a-c, MME 162, SGW 164, PGW 166, gNB 180a-c, AMF 182a-ab, UPF 184a-b, SMF 183a-b, DN 185a-b, and/or any other device(s) described herein, may be performed by one or more emulation devices (not shown). The emulation devices may be one or more devices configured to emulate one or more, or all, of the functions described herein. For example, the emulation devices may be used to test other devices and/or to simulate network and/or WTRU functions.

[00207] The emulation devices may be designed to implement one or more tests of other devices in a lab environment and/or in an operator network environment. For example, the one or more emulation devices may perform the one or more, or all, functions while being fully or partially implemented and/or deployed as part of a wired and/or wireless communication network in order to test other devices within the communication network. The one or more emulation devices may perform the one or more, or all, functions while being temporarily implemented/deployed as part of a wired and/or wireless communication network. The emulation device may be directly coupled to another device for purposes of testing and/or may performing testing using over-the-air wireless communications.

[00208] The one or more emulation devices may perform the one or more, including all, functions while not being implemented/deployed as part of a wired and/or wireless communication

network. For example, the emulation devices may be utilized in a testing scenario in a testing laboratory and/or a non-deployed (e.g., testing) wired and/or wireless communication network in order to implement testing of one or more components. The one or more emulation devices may be test equipment. Direct RF coupling and/or wireless communications via RF circuitry (e.g., which may include one or more antennas) may be used by the emulation devices to transmit and/or receive data.

[00209] Although features and elements are described above in particular combinations, each feature or element can be used alone or in any combination with the other features and elements. Additionally, the embodiments described herein are provided for exemplary purposes only. Furthermore, the embodiments described herein may be implemented in a computer program, software, or firmware incorporated in a computer-readable medium for execution by a computer or processor. Examples of computer-readable media include electronic signals (transmitted over wired or wireless connections) and computer-readable storage media. Examples of computer-readable storage media include, but are not limited to, a read only memory (ROM), a random access memory (RAM), a register, cache memory, semiconductor memory devices, magnetic media such as internal hard disks and removable disks, magneto-optical media, and optical media such as CD-ROM disks, and digital versatile disks (DVDs). A processor in association with software may be used to implement a radio frequency transceiver for use in a WTRU, UE, terminal, base station, RNC, or any host computer.

What is Claimed:

1. An apparatus comprising a processor, a memory, and communication circuitry, the apparatus being connected to a network via its communication circuitry, the apparatus further comprising computer-executable instructions stored in the memory of the apparatus which, when executed by the processor of the apparatus, cause the apparatus to perform operations comprising:

receiving a first cryptographic function and a first cryptographic parameter from a network entity;

computing a third cryptographic parameter using the first cryptographic function and a second cryptographic parameter that is locally generated at the apparatus;

sending the third cryptographic parameter to the network entity;

computing a master key using at least one of the first cryptographic parameter, the second cryptographic parameter, or the third cryptographic parameter; and

verifying that the network entity also correctly derived the master key.

2. The apparatus as recited in claim 1, wherein the apparatus further comprises computer-executable instructions stored in the memory of the apparatus which, when executed by the processor of the apparatus, cause the apparatus to perform further operations comprising:

before receiving the first cryptographic function and the first cryptographic parameter from the network entity, authenticating with the network entity.

4. The apparatus as recited in any one of the preceding claims, wherein the apparatus further comprises computer-executable instructions stored in the memory of the apparatus which, when executed by the processor of the apparatus, cause the apparatus to perform further operations comprising:

storing at least one of the first cryptographic parameter and the first cryptographic function.

5. The apparatus as recited in any one of the preceding claims, wherein the apparatus further comprises computer-executable instructions stored in the memory of the apparatus which, when executed by the processor of the apparatus, cause the apparatus to perform further operations comprising:

receiving a second cryptographic function from the network entity.

6. The apparatus as recited in claim 1, wherein the apparatus further comprises computer-executable instructions stored in the memory of the apparatus which, when executed by the processor of the apparatus, cause the apparatus to perform further operations comprising:

generating and saving a fourth cryptographic parameter used with a master secret to derive the second cryptographic parameter; and

deriving and saving a third cryptographic function that is computed based on the second cryptographic parameter.

7. The apparatus as recited in any one of the preceding claims, wherein the apparatus further comprises computer-executable instructions stored in the memory of the apparatus which, when executed by the processor of the apparatus, cause the apparatus to perform further operations comprising:

deleting at least one of the first, second, or third cryptographic parameters, and the first cryptographic function.

8. The apparatus as recited in any one of the preceding claims, wherein the apparatus further comprises computer-executable instructions stored in the memory of the apparatus which, when executed by the processor of the apparatus, cause the apparatus to perform further operations comprising:

making an access request for one or more slices of the network; and

based on the access request, receiving the first cryptographic function and the first cryptographic parameter from the network entity.

9. The apparatus as recited in claim 6, wherein the apparatus further comprises computer-executable instructions stored in the memory of the apparatus which, when executed by the processor of the apparatus, cause the apparatus to perform further operations comprising:

receiving an alert indicating that a hack associated with the network entity has occurred; and

based on the alert, computing a fresh master key using the stored second cryptographic function, the fourth cryptographic parameter, and the master secret.

10. The apparatus as recited in any one of the preceding claims, wherein the master key is isolated from an intermediary party between the apparatus and the network entity.

11. The apparatus as recited in any one of the preceding claims, wherein the network entity comprises an authentication server function that resides on a home network of the apparatus.

12. The apparatus as recited in any one of claims 1 to 10, wherein the network entity comprises an authentication server function and an authentication credential repository and processing function.

13. A method performed by a user equipment in communication with a network entity via a network, the method comprising:

receiving a first cryptographic function and a first cryptographic parameter from the network entity;

computing a third cryptographic parameter using the first cryptographic function and a second cryptographic parameter that is locally generated at the apparatus;

sending the third cryptographic parameter to the network entity;

computing a master key using at least one of the first cryptographic parameter, the second cryptographic parameter, or the third cryptographic parameter; and

verifying that the network entity also correctly derived the master key.

14. The method as recited in claim 13, the method further comprising:

before receiving the first cryptographic function and the first cryptographic parameter from the network entity, authenticating with the network entity.

15. The method as recited in any one of claims 13 and 14, the method further comprising:

storing at least one of the first cryptographic parameter and the first cryptographic function.

16. The method as recited in any one of claims 13 to 15, the method further comprising:

receiving a second cryptographic function from the network entity.

17 The method as recited in any one of claims 13 to 16, the method further comprising:

generating and saving a fourth cryptographic parameter used with a master secret to derive the second cryptographic parameter; and

deriving and saving a third cryptographic function that is computed based on the second cryptographic parameter.

18. The method as recited in claim 17, the method further comprising:

receiving an alert indicating that a hack associated with the network entity has occurred; and

based on the alert, computing a fresh master key using the stored second cryptographic function, the fourth cryptographic parameter, and the master secret.

19. The method as recited in any one of claims 13 to 18, the method further comprising:

deleting at least one of the first, second, or third cryptographic parameters, and the first cryptographic function.

20. The method as recited in any one of claims 13 to 19, the method further comprising:

making an access request for one or more slices of the network; and

based on the access request, receiving the first cryptographic function and the first cryptographic parameter from the network entity.

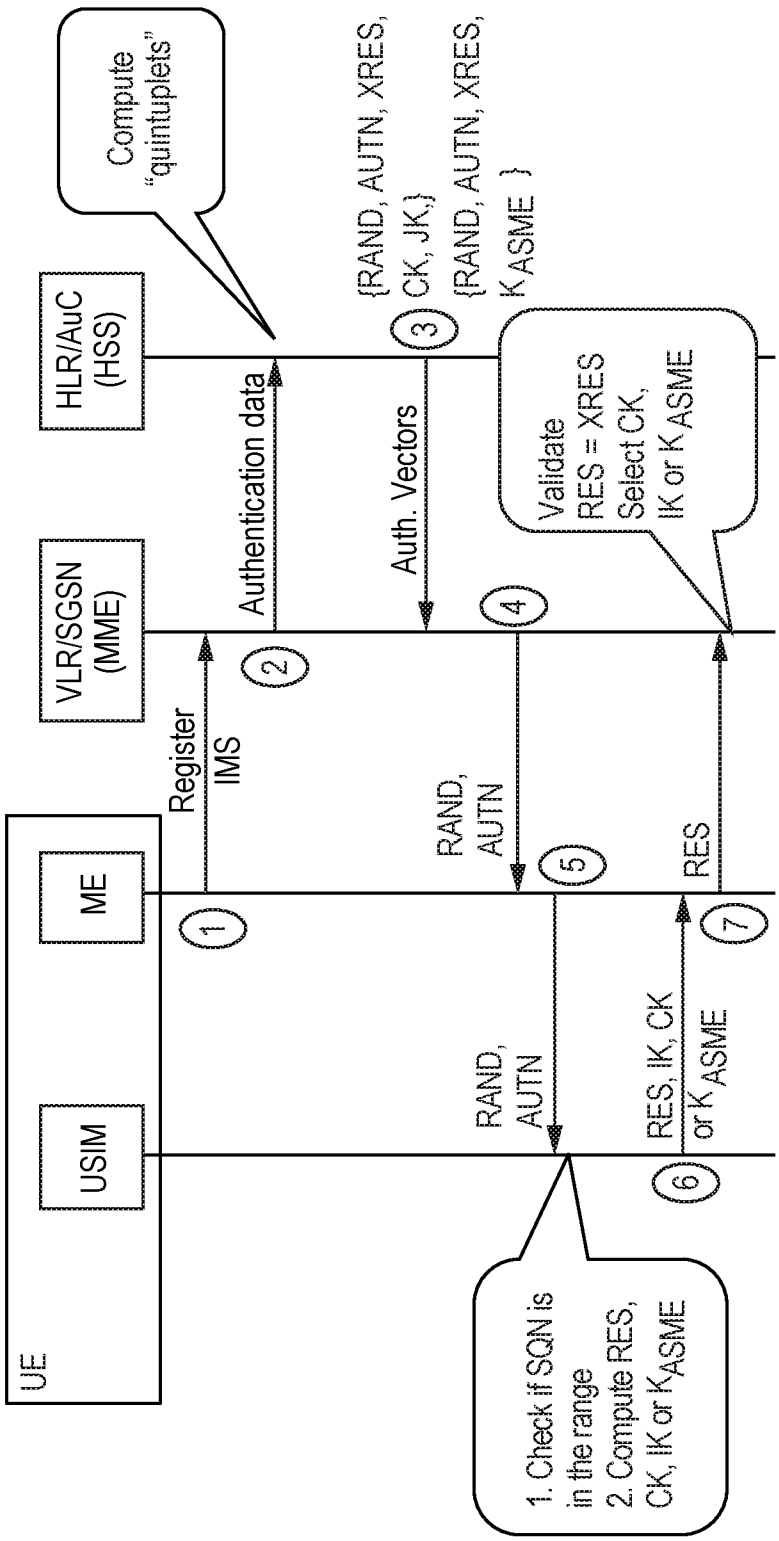

99
UE
USIM
ME
VLR/SGSN (MME)
HLR/AuC (HSS)
Compute "quintuplets"
Register IMS
Authentication data
{RAND, AUTN, XRES, CK, JK,} {RAND, AUTN, XRES, KASME }
Auth. Vectors
Validate RES = XRES Select CK, IK or KASME
RAND, AUTN
RES
RAND, AUTN
RES, IK, CK or KASME
1. Check if SQN is in the range
2. Compute RES, CK, IK or KASME
1
2
3
4
5
6
7
AuC - Authentication Center
AUTH - Authentication vector (quintuplet)
AUTN - Authentication Token
CK - Ciphering Key
HLR - Home location Register
IK - Integrity Key
IMSI - International Mobile Subscriber Identity
KASME - Root key of Access Security Management Entity - Authentication Token
ME - Mobile Equipment
RAND - Random challenge
RES - Response to an authentication challenge
VLR - Visited Location Register
XRES - eXpected Response

FIG. 1

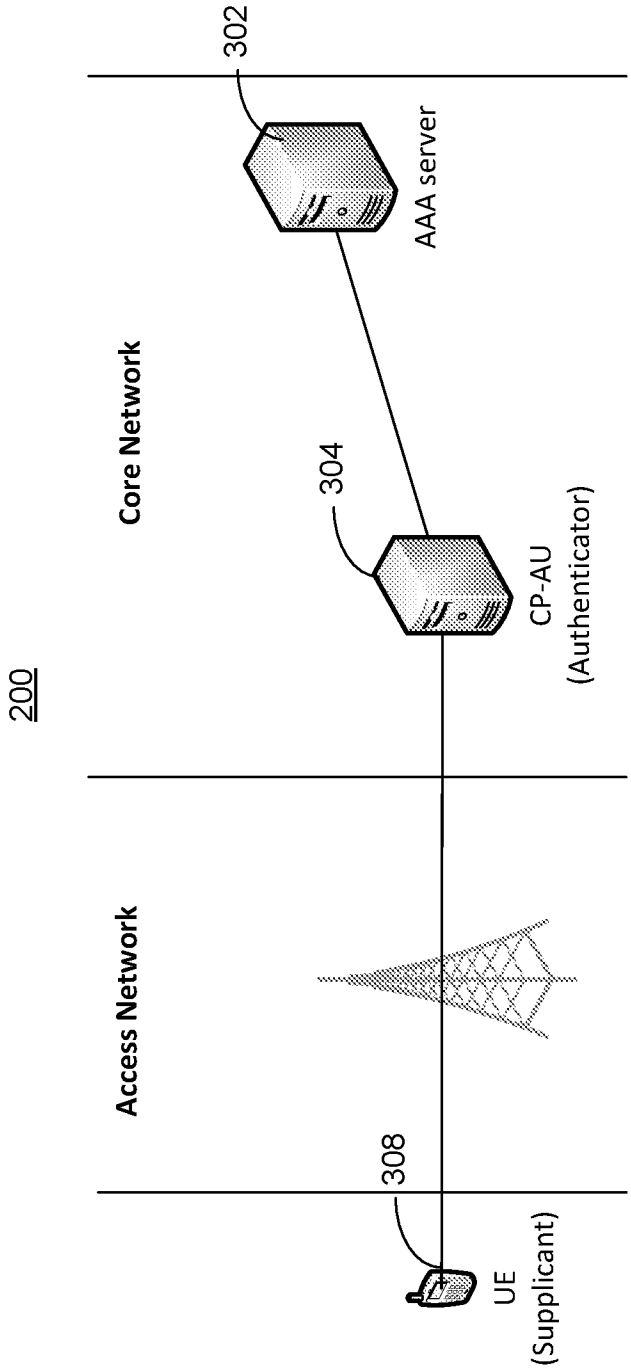

200
Core Network
Access Network
302
AAA server
304
CP-AU
(Authenticator)
308
UE
(Supplicant)

Fig. 2

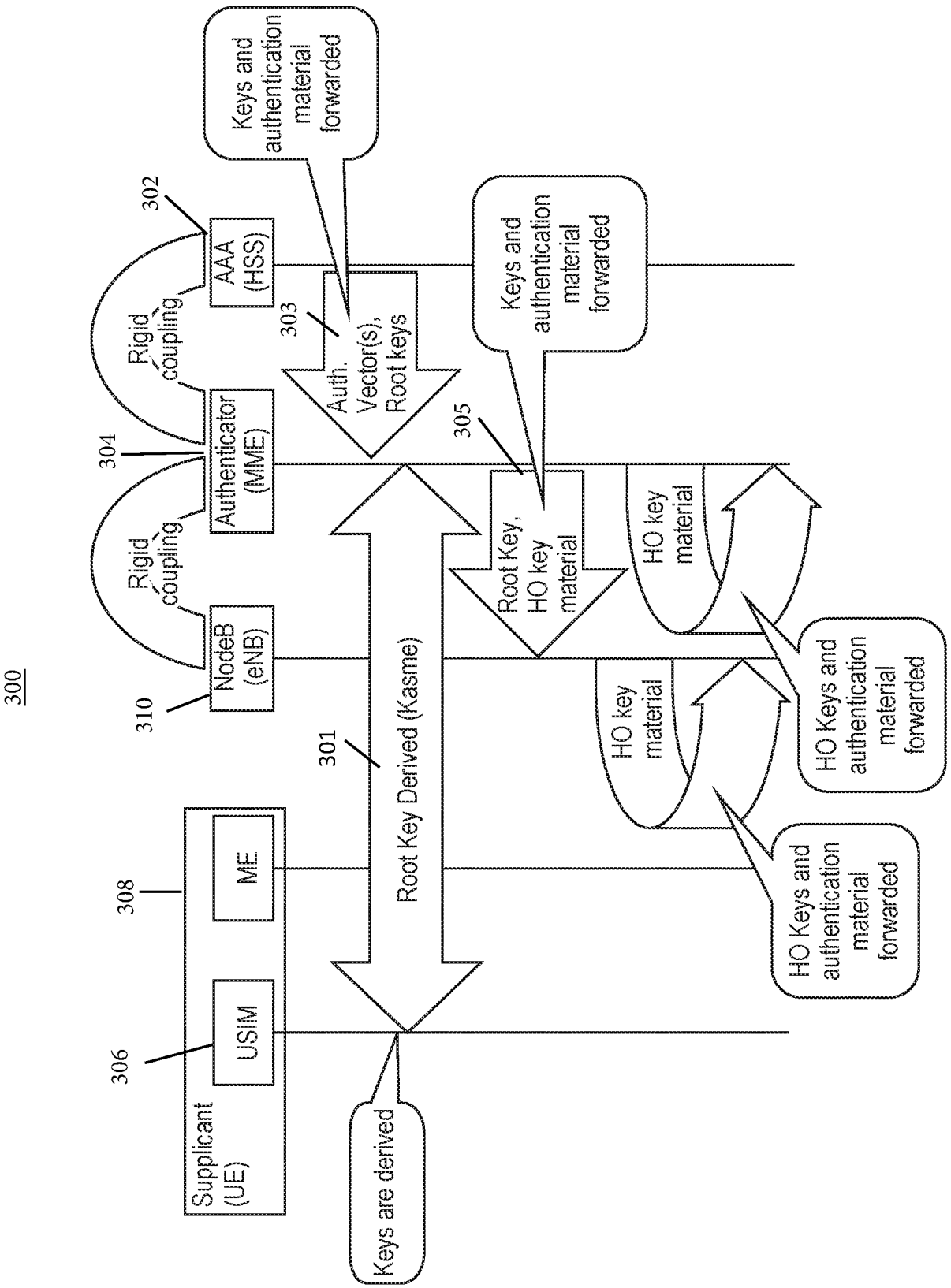

300
Supplicant (UE)
USIM
ME
306
308
NodeB (eNB)
Authenticator (MME)
AAA (HSS)
310
304
302
Rigid coupling
Rigid coupling
303
Auth. Vector(s), Root keys
Keys and authentication material forwarded
Keys and authentication material forwarded
305
Root Key, HO key material
HO key material
HO key material
HO Keys and authentication material forwarded
HO Keys and authentication material forwarded
301
Root Key Derived (Kasme)
Keys are derived

FIG. 3

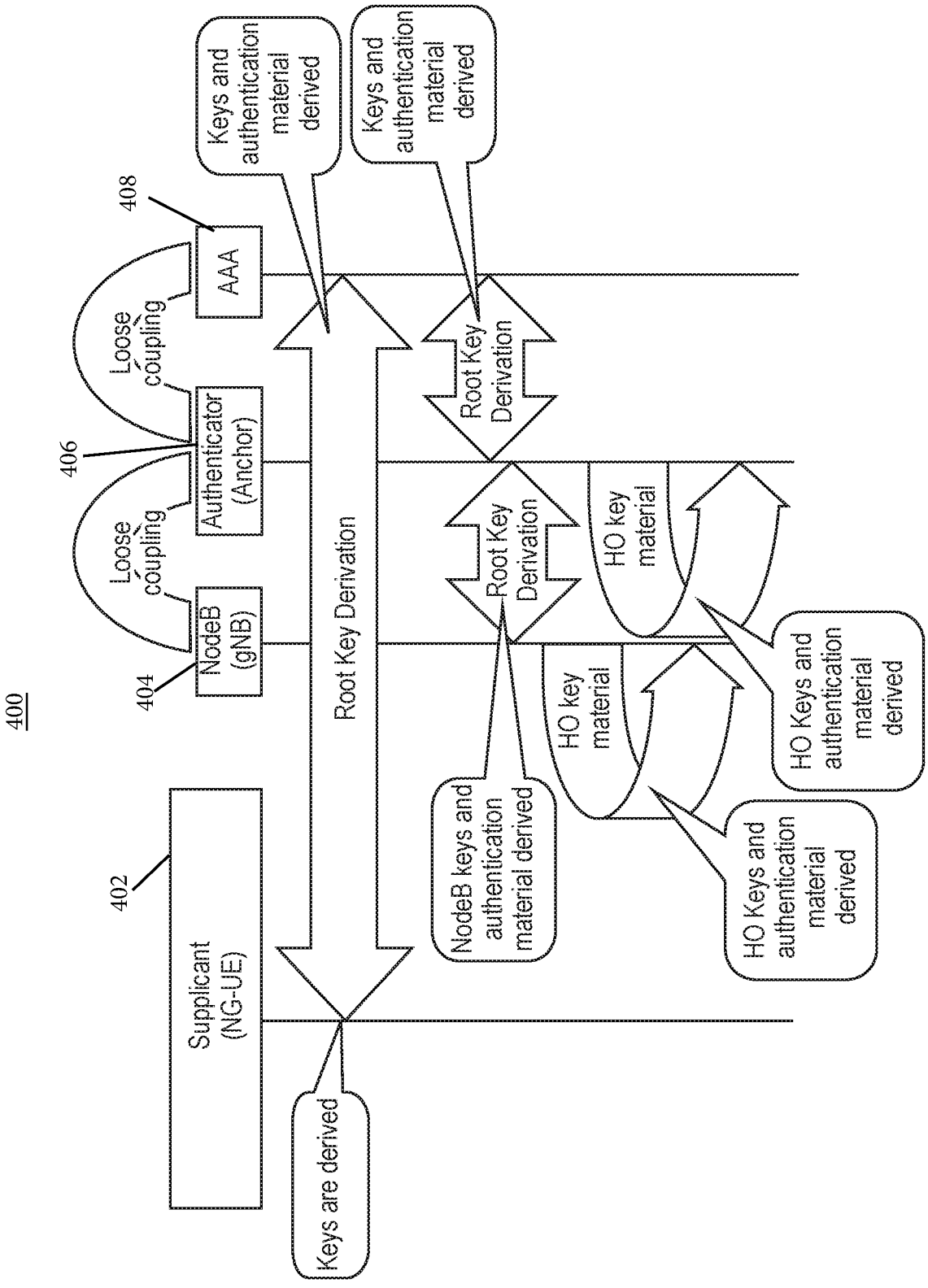

400
402
Supplicant (NG-UE)
404
NodeB (gNB)
406
Authenticator (Anchor)
408
AAA
Loose coupling
Loose coupling
Keys and authentication material derived
Keys and authentication material derived
Root Key Derivation
Root Key Derivation
Root Key Derivation
HO key material
HO key material
NodeB keys and authentication material derived
HO Keys and authentication material derived
HO Keys and authentication material derived
Keys are derived

FIG. 4

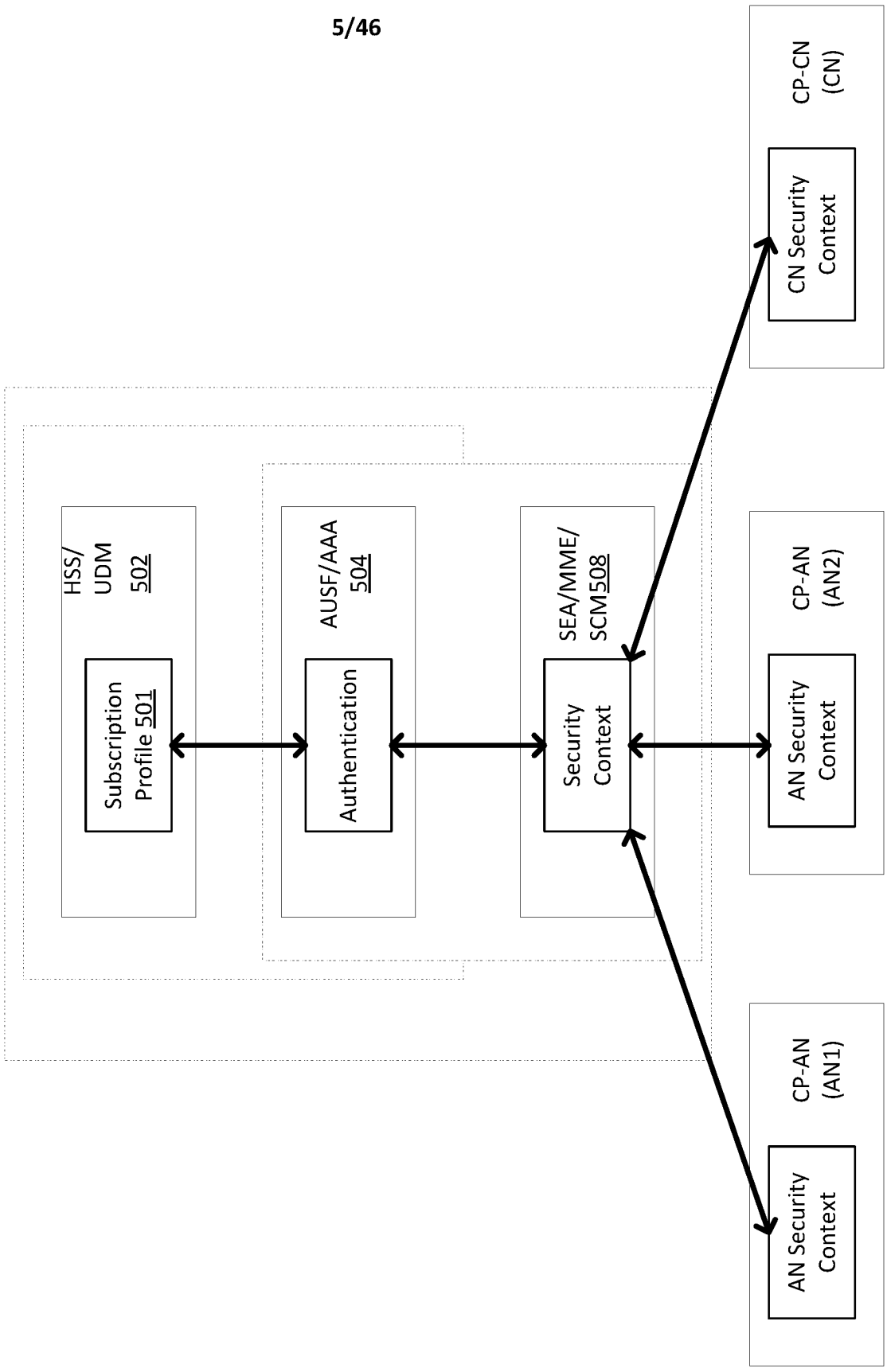

HSS/
UDM
502
Subscription
Profile 501
AUSF/AAA
504
Authentication
SEA/MME/
SCM508
Security
Context
CP-CN
(CN)
CN Security
Context
CP-AN
(AN2)
AN Security
Context
CP-AN
(AN1)
AN Security
Context

Fig. 5

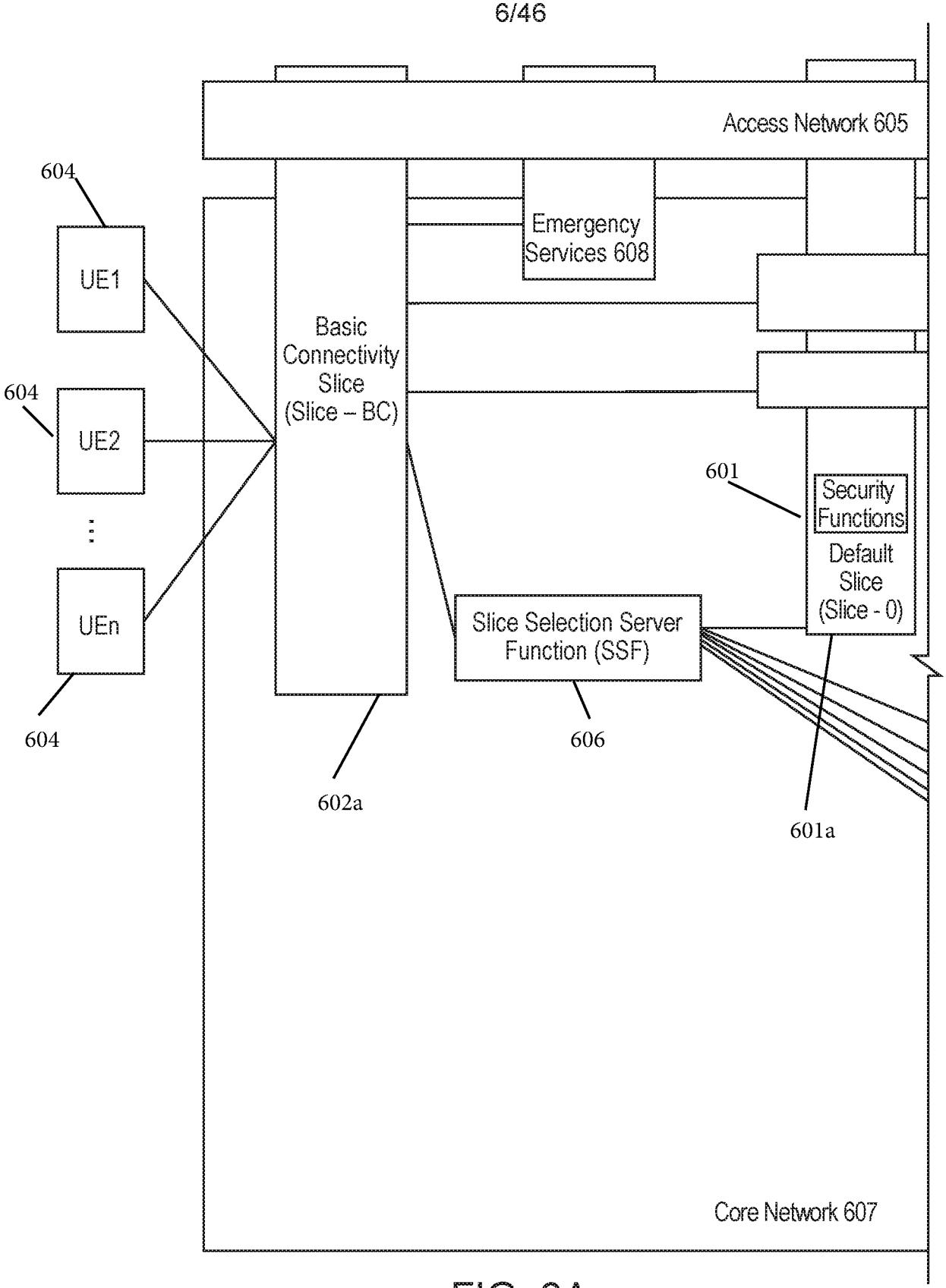

Access Network 605
604
UE1
604
UE2
UEn
604
Basic Connectivity Slice (Slice – BC)
Emergency Services 608
601
Security Functions
Default Slice (Slice - 0)
Slice Selection Server Function (SSF)
602a
606
601a
Core Network 607

FIG. 6A

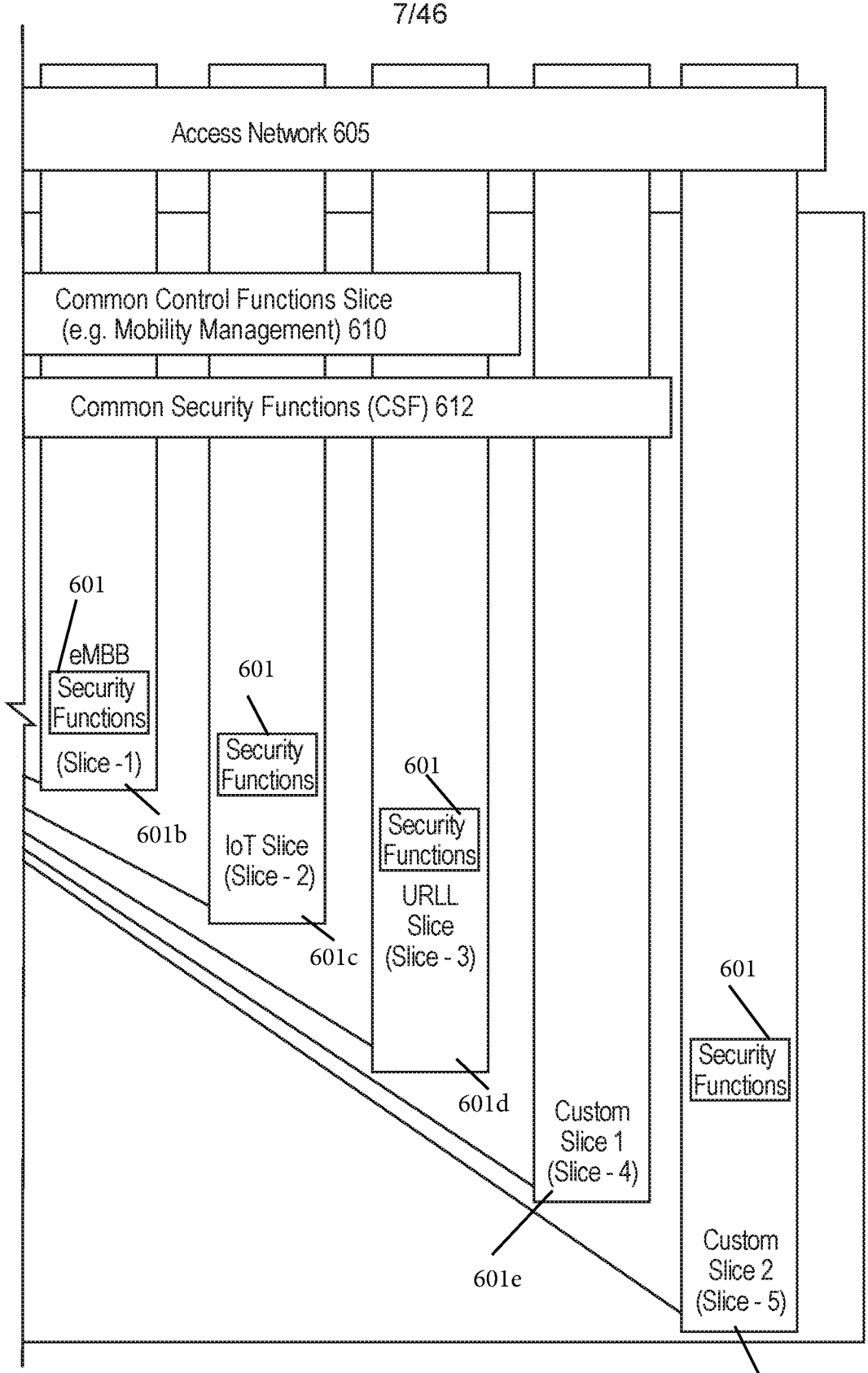

Access Network 605
Common Control Functions Slice
(e.g. Mobility Management) 610
Common Security Functions (CSF) 612
601
eMBB
Security Functions
(Slice -1)
601b
601
Security Functions
IoT Slice
(Slice - 2)
601c
601
Security Functions
URLL
Slice
(Slice - 3)
601d
601
Security Functions
Custom
Slice 1
(Slice - 4)
601e
601
Security Functions
Custom
Slice 2
(Slice - 5)
601f

FIG. 6B

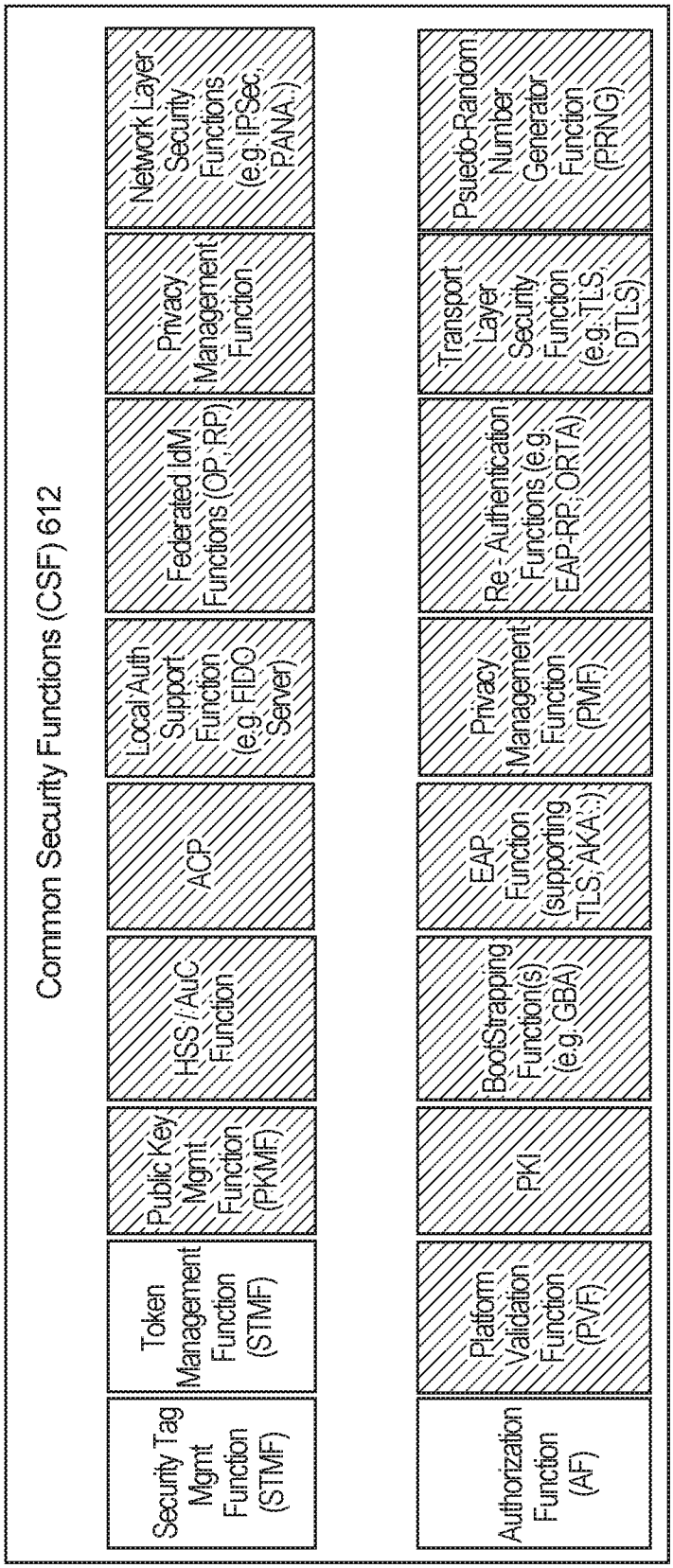

Common Security Functions (CSF) 612
Security Tag Mgmt Function (STMF)
Token Management Function (STMF)
Public Key Mgmt Function (PKMF)
HSS / AuC Function
ACP
Local Auth Support Function (e.g. FIDO Server)
Federated IdM Functions (OP, RP)
Privacy Management Function
Network Layer Security Functions (e.g. IPSec, PANA)
Authorization Function (AF)
Platform Validation Function (PVF)
PKI
BootStrapping Function(s) (e.g. GBA)
EAP Function (supporting TLS, AKA's)
Privacy Management Function (PMF)
Re - Authentication Functions (e.g. EAP-RP, ORTA)
Transport Layer Security Function (e.g. TLS, DTLS)
Psuedo-Random Number Generator Function (PRNG)

FIG. 7

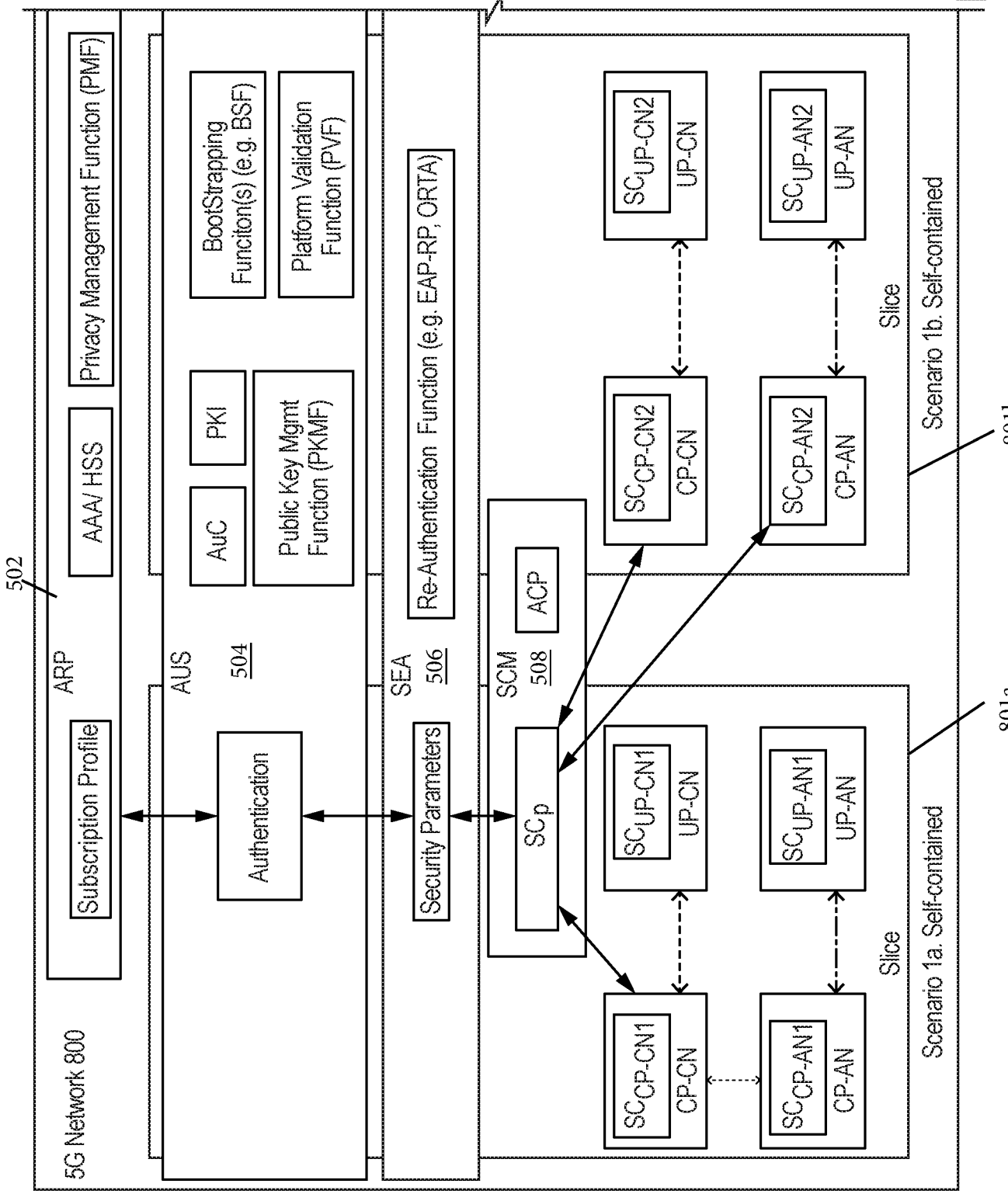

5G Network 800
502
ARP
Subscription Profile
AAA/ HSS
Privacy Management Function (PMF)
AUS
504
Authentication
AuC
PKI
Public Key Mgmt Function (PKMF)
BootStrapping Funciton(s) (e.g. BSF)
Platform Validation Function (PVF)
SEA
506
Security Parameters
Re-Authentication Function (e.g. EAP-RP, ORTA)
SCM
508
ACP
SCp
SCCP-CN1
CP-CN
SCUP-CN1
UP-CN
SCCP-AN1
CP-AN
SCUP-AN1
UP-AN
Slice
Scenario 1a. Self-contained
801a
SCCP-CN2
CP-CN
SCUP-CN2
UP-CN
SCCP-AN2
CP-AN
SCUP-AN2
UP-AN
Slice
Scenario 1b. Self-contained
801b

FIG. 8A

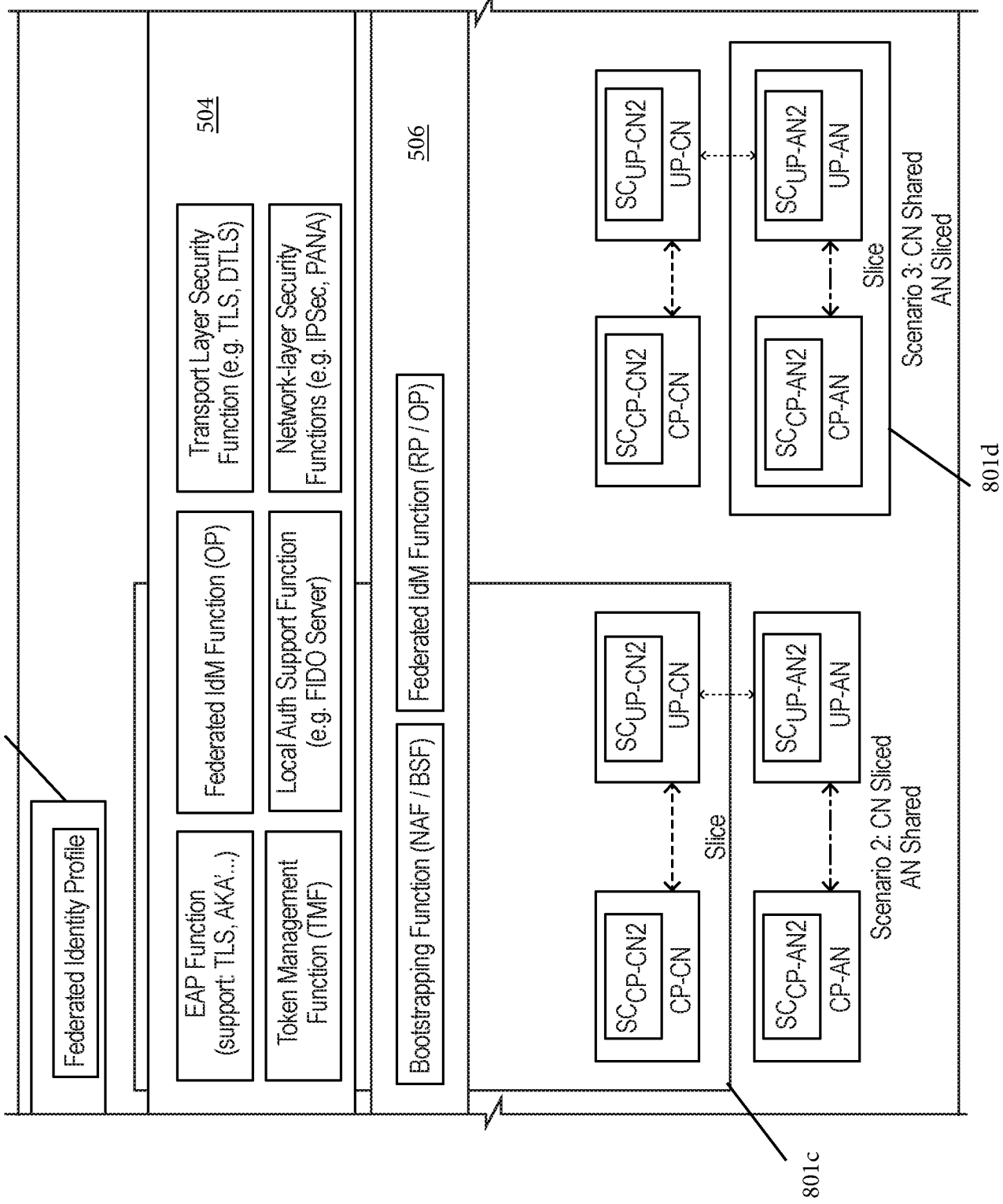
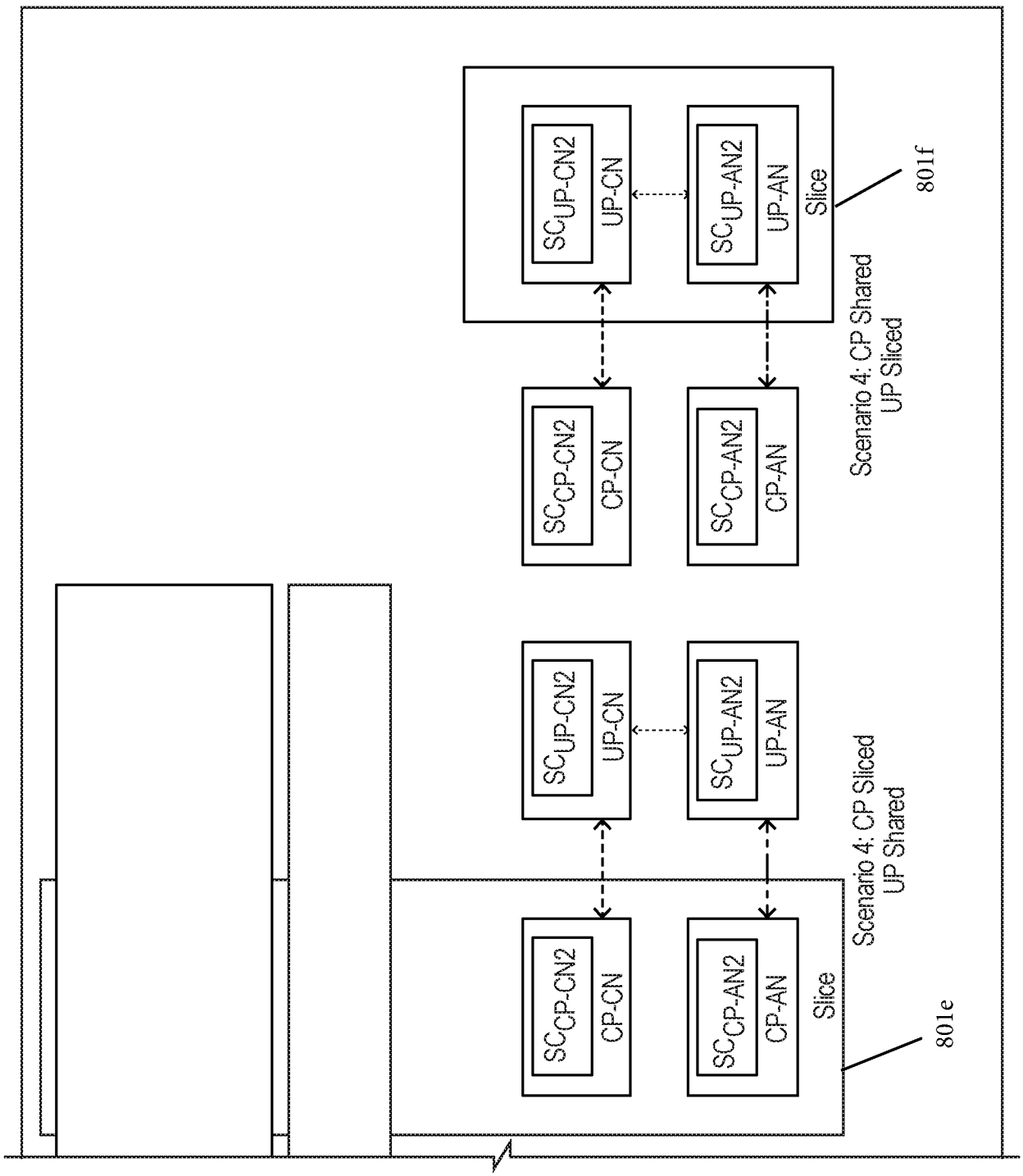

502
Federated Identity Profile
504
EAP Function (support: TLS, AKA'...)
Federated IdM Function (OP)
Transport Layer Security Function (e.g. TLS, DTLS)
Token Management Function (TMF)
Local Auth Support Function (e.g. FIDO Server)
Network-layer Security Functions (e.g. IPSec, PANA)
506
Bootstrapping Function (NAF / BSF)
Federated IdM Function (RP / OP)
SCUP-CN2
UP-CN
SCCP-CN2
CP-CN
SCUP-AN2
UP-AN
SCCP-AN2
CP-AN
Slice
Scenario 3: CN Shared AN Sliced
801d
Scenario 2: CN Sliced AN Shared
801c

FIG. 8B

FIG. 8C


Scenario 4: CP Shared
UP Sliced
801f
Slice
UP-CN
UP-AN
CP-CN
CP-AN
SCUP-CN2
SCUP-AN2
SCCP-CN2
SCCP-AN2
Scenario 4: CP Sliced
UP Shared
801e
Slice
UP-CN
UP-AN
CP-CN
CP-AN
SCUP-CN2
SCUP-AN2
SCCP-CN2
SCCP-AN2

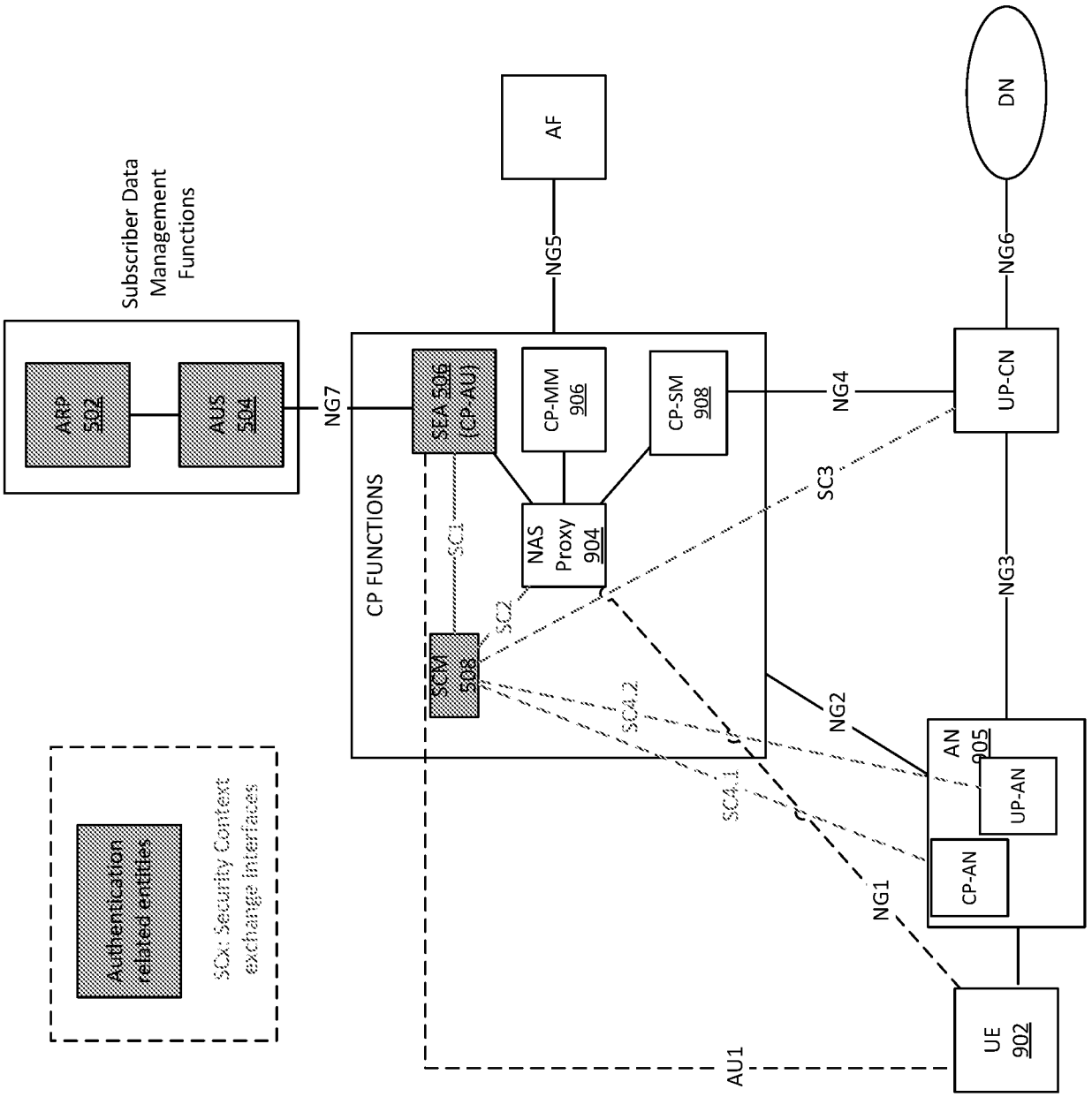

Subscriber Data Management Functions
ARP 502
AUS 504
NG7
AF
NG5
DN
NG6
SEA 506 (CP-AU)
CP-MM 906
CP-SM 908
NG4
UP-CN
CP FUNCTIONS
NAS Proxy 904
SC1
SC3
SC2
SCM 508
SC4.2
SC4.1
NG2
NG3
AN 905
UP-AN
CP-AN
NG1
UE 902
AU1
Authentication related entities
SCx Security Context exchange interfaces

Fig. 9

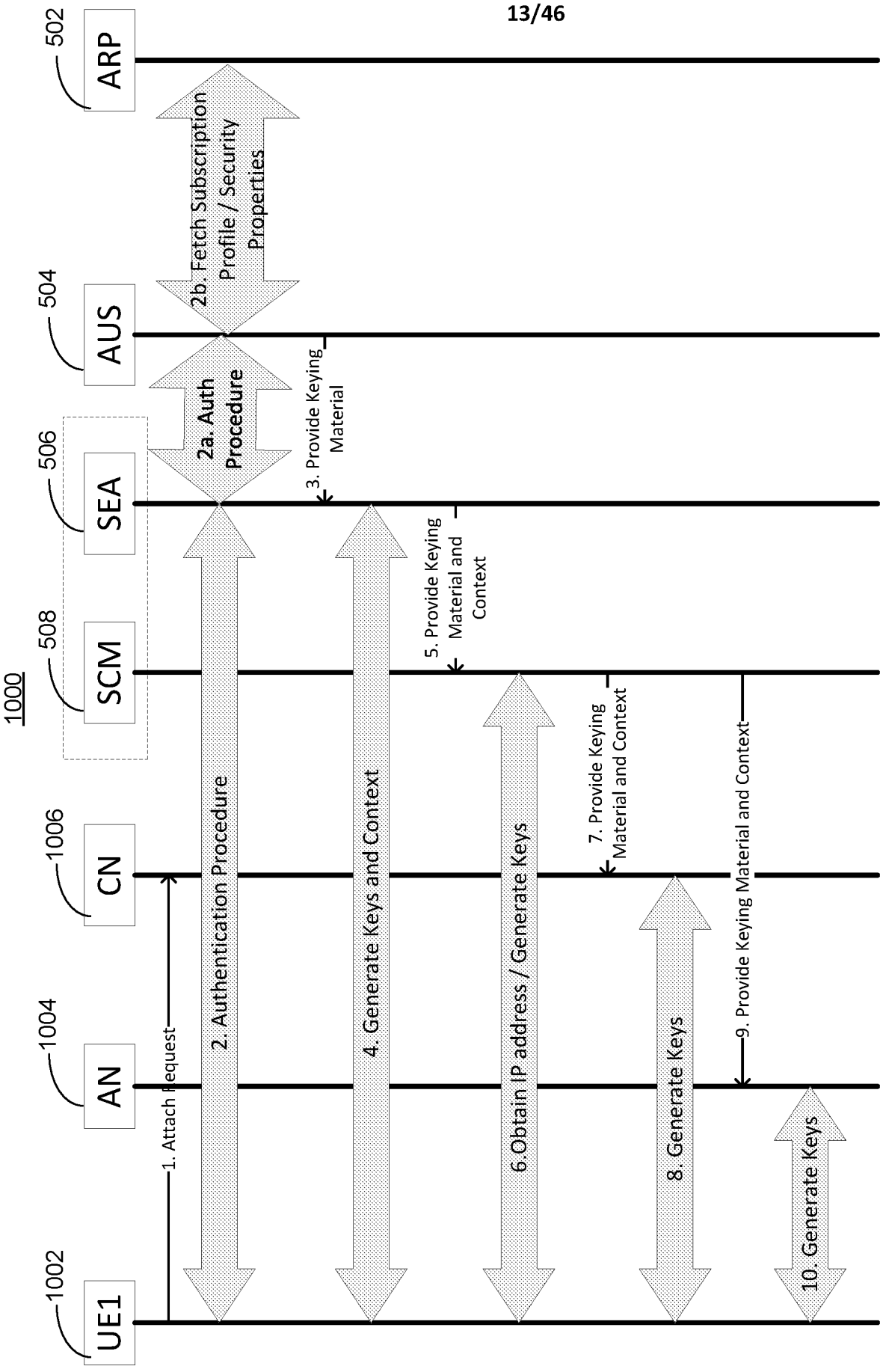

1000
1002
UE1
1004
AN
1006
CN
508
SCM
506
SEA
504
AUS
502
ARP
1. Attach request
2. Authentication Procedure
2a. Auth Procedure
2b. Fetch Subscription Profile / Security Properties
3. Provide Keying Material
4. Generate Keys and Context
5. Provide Keying Material and Context
6.Obtain IP address / Generate Keys
7. Provide Keying Material and Context
8. Generate Keys
9. Provide Keying Material and Context
10. Generate Keys

Fig. 10

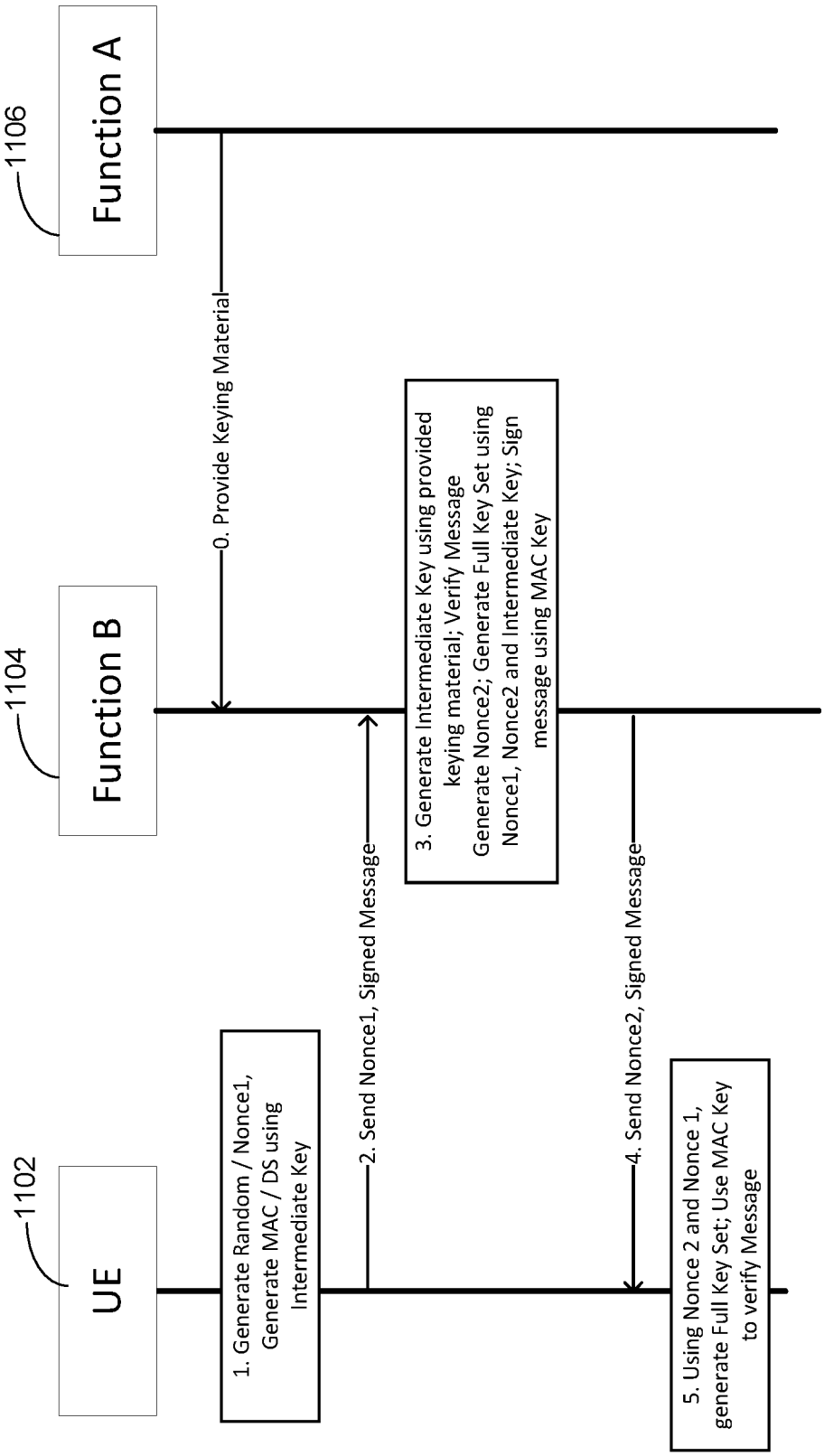

1106
Function A
0. Provide Keying Material
1104
Function B
3. Generate Intermediate Key using provided keying material; Verify Message
Generate Nonce2; Generate Full Key Set using Nonce1, Nonce2 and Intermediate Key; Sign message using MAC Key
2. Send Nonce1, Signed Message
4. Send Nonce2, Signed Message
1102
UE
1. Generate Random / Nonce1, Generate MAC / DS using Intermediate Key
5. Using Nonce 2 and Nonce 1, generate Full Key Set; Use MAC Key to verify Message

Fig. 11

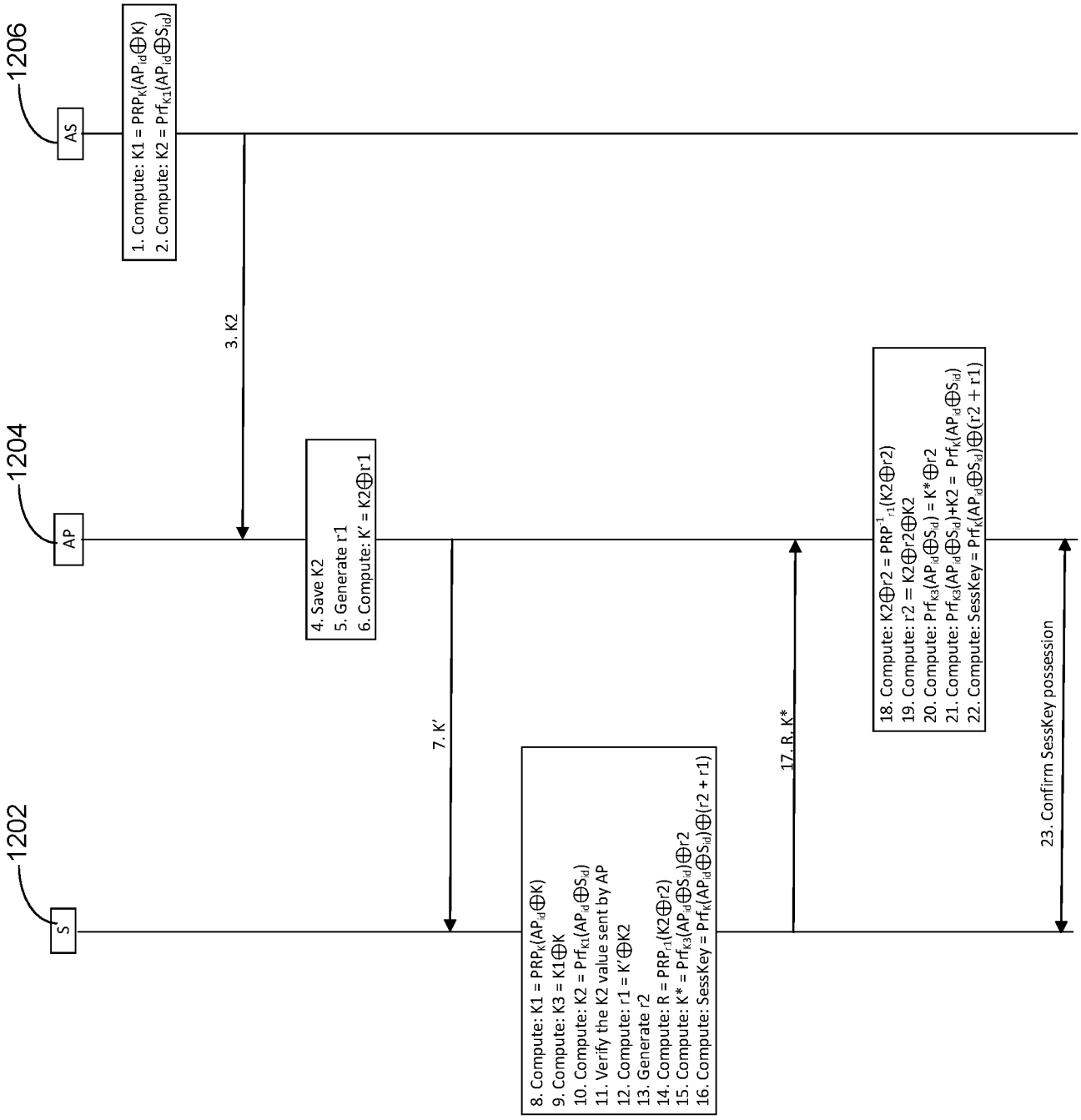

1206
AS
1. Compute: K1 = PRP_K(AP_id⊕K)
2. Compute: K2 = Prf_K1(AP_id⊕S_id)
3. K2
1204
AP
4. Save K2
5. Generate r1
6. Compute: K' = K2⊕r1
7. K'
1202
S
8. Compute: K1 = PRP_K(AP_id⊕K)
9. Compute: K3 = K1⊕K
10. Compute: K2 = Prf_K1(AP_id⊕S_id)
11. Verify the K2 value sent by AP
12. Compute: r1 = K'⊕K2
13. Generate r2
14. Compute: R = PRP_r1(K2⊕r2)
15. Compute: K* = Prf_K3(AP_id⊕S_id)⊕r2
16. Compute: SessKey = Prf_K(AP_id⊕S_id)⊕(r2 + r1)
17. R, K*
18. Compute: K2⊕r2 = PRP^-1_r1(K2⊕r2)
19. Compute: r'2 = K2⊕r2⊕K2
20. Compute: Prf_K3(AP_id⊕S_id) = K*⊕r2
21. Compute: Prf_K3(AP_id⊕S_id)+K2 = Prf_K(AP_id⊕S_id)
22. Compute: SessKey = Prf_K(AP_id⊕S_id)⊕(r2 + r1)
23. Confirm SessKey possession

Fig. 12

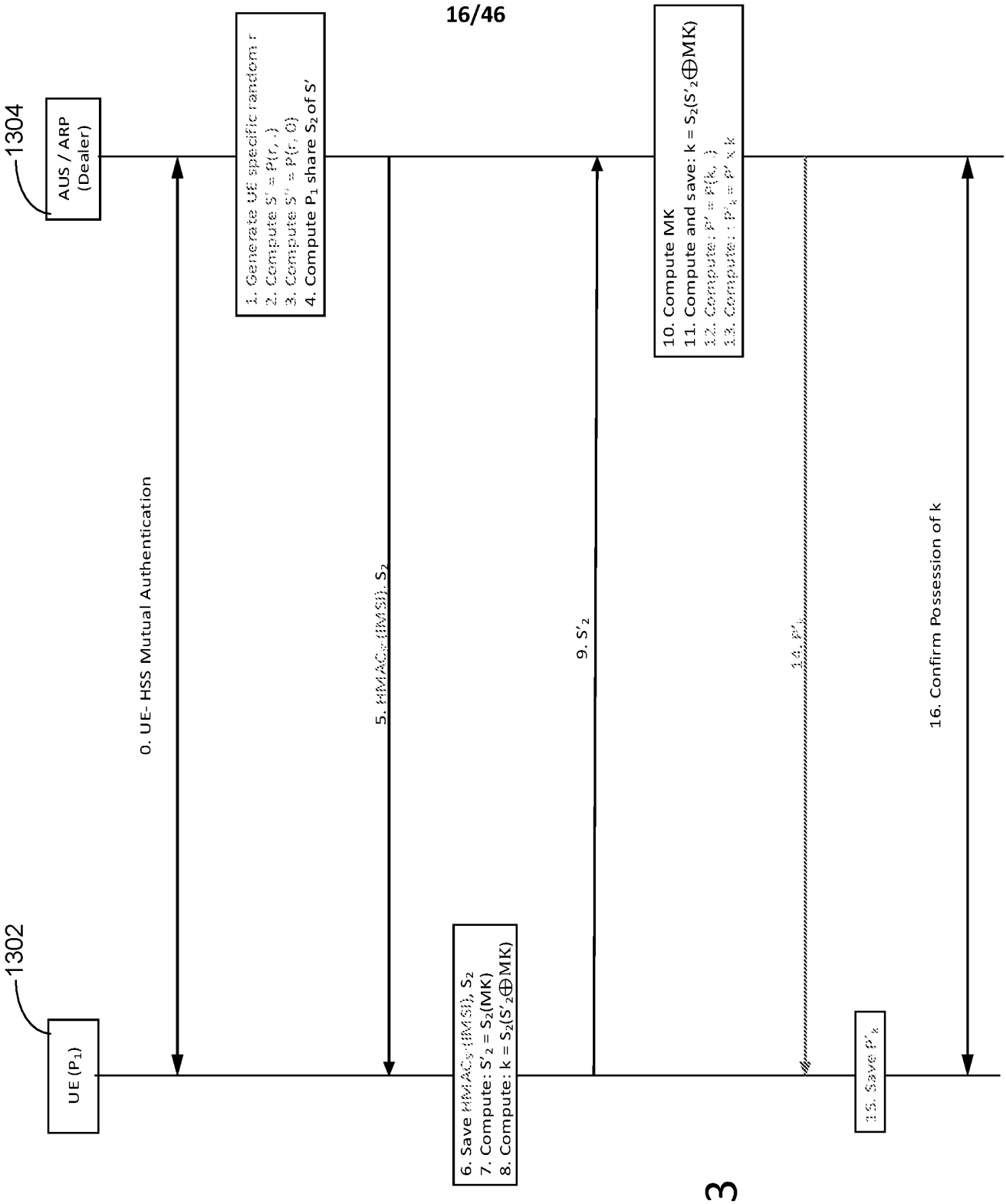

1304
AUS / ARP (Dealer)
1302
UE (P1)
1. Generate UE specific random r
2. Compute S' = P(r, .)
3. Compute S'' = P(r, 0)
4. Compute P1 share S2 of S'
0. UE- HSS Mutual Authentication
5. MMAC(IMSI), S2
6. Save MMAC(IMSI), S2
7. Compute: S'2 = S2(MK)
8. Compute: k = S2(S'2⊕MK)
9. S'2
10. Compute MK
11. Compute and save: k = S2(S'2⊕MK)
12. Compute: P' = P(k, .)
13. Compute: P'k = P' x k
14. P'k
15. Save P'k
16. Confirm Possession of k

Fig. 13

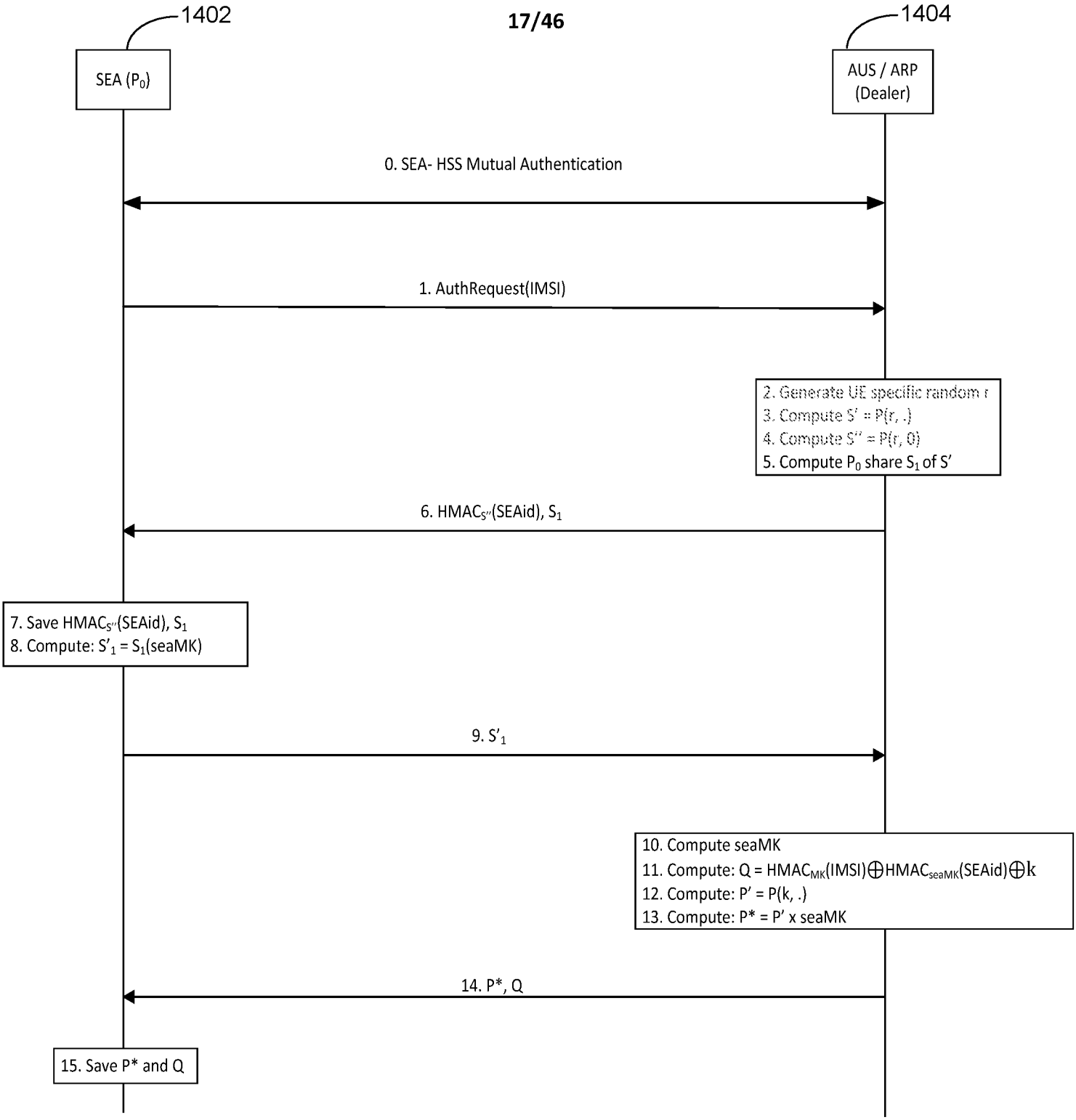

1402
SEA (P_0)
1404
AUS / ARP (Dealer)
0. SEA- HSS Mutual Authentication
1. AuthRequest(IMSI)
2. Generate UE specific random r
3. Compute S' = P(r, .)
4. Compute S'' = P(r, 0)
5. Compute P_0 share S_1 of S'
6. $HMAC_{S''}$(SEAid), S_1
7. Save $HMAC_{S''}$(SEAid), S_1
8. Compute: $S'_1 = S_1$(seaMK)
9. S'_1
10. Compute seaMK
11. Compute: Q = $HMAC_{MK}$(IMSI)⊕$HMAC_{seaMK}$(SEAid)⊕k
12. Compute: P' = P(k, .)
13. Compute: P* = P' x seaMK
14. P*, Q
15. Save P* and Q

Fig. 14

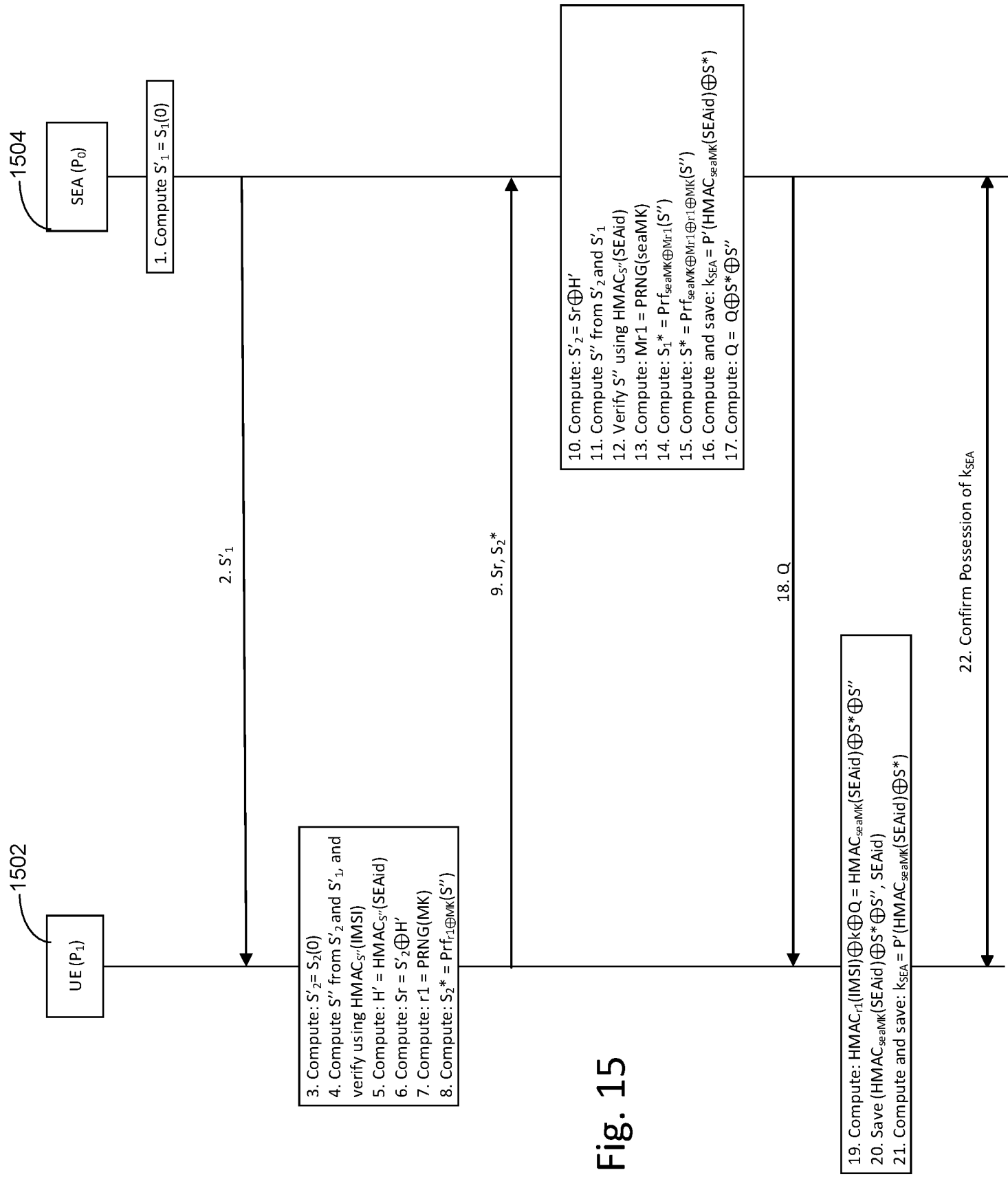

1504
SEA (P_0)
1. Compute $S'_1 = S_1(0)$
2. S'_1
9. Sr, S_2*
10. Compute: S'_2 = Sr⊕H'
11. Compute S'' from S'_2 and S'_1
12. Verify S'' using $HMAC_{S''}$(SEAid)
13. Compute: Mr1 = PRNG(seaMK)
14. Compute: S_1* = $Prf_{seaMK⊕Mr1}(S'')$
15. Compute: S* = $Prf_{seaMK⊕Mr1⊕r1⊕MK}(S'')$
16. Compute and save: k_{SEA} = P'($HMAC_{seaMK}$(SEAid)⊕S*)
17. Compute: Q = Q⊕S*⊕S''
18. Q
22. Confirm Possession of k_{SEA}
1502
UE (P_1)
3. Compute: S'_2 = $S_2(0)$
4. Compute S'' from S'_2 and S'_1, and verify using $HMAC_{S''}$(IMSI)
5. Compute: H' = $HMAC_{S''}$(SEAid)
6. Compute: Sr = S'_2⊕H'
7. Compute: r1 = PRNG(MK)
8. Compute: S_2* = $Prf_{r1⊕MK}(S'')$
19. Compute: $HMAC_{r1}$(IMSI)⊕k⊕Q = $HMAC_{seaMK}$(SEAid)⊕S*⊕S''
20. Save ($HMAC_{seaMK}$(SEAid)⊕S*⊕S'', SEAid)
21. Compute and save: k_{SEA} = P'($HMAC_{seaMK}$(SEAid)⊕S*)

Fig. 15

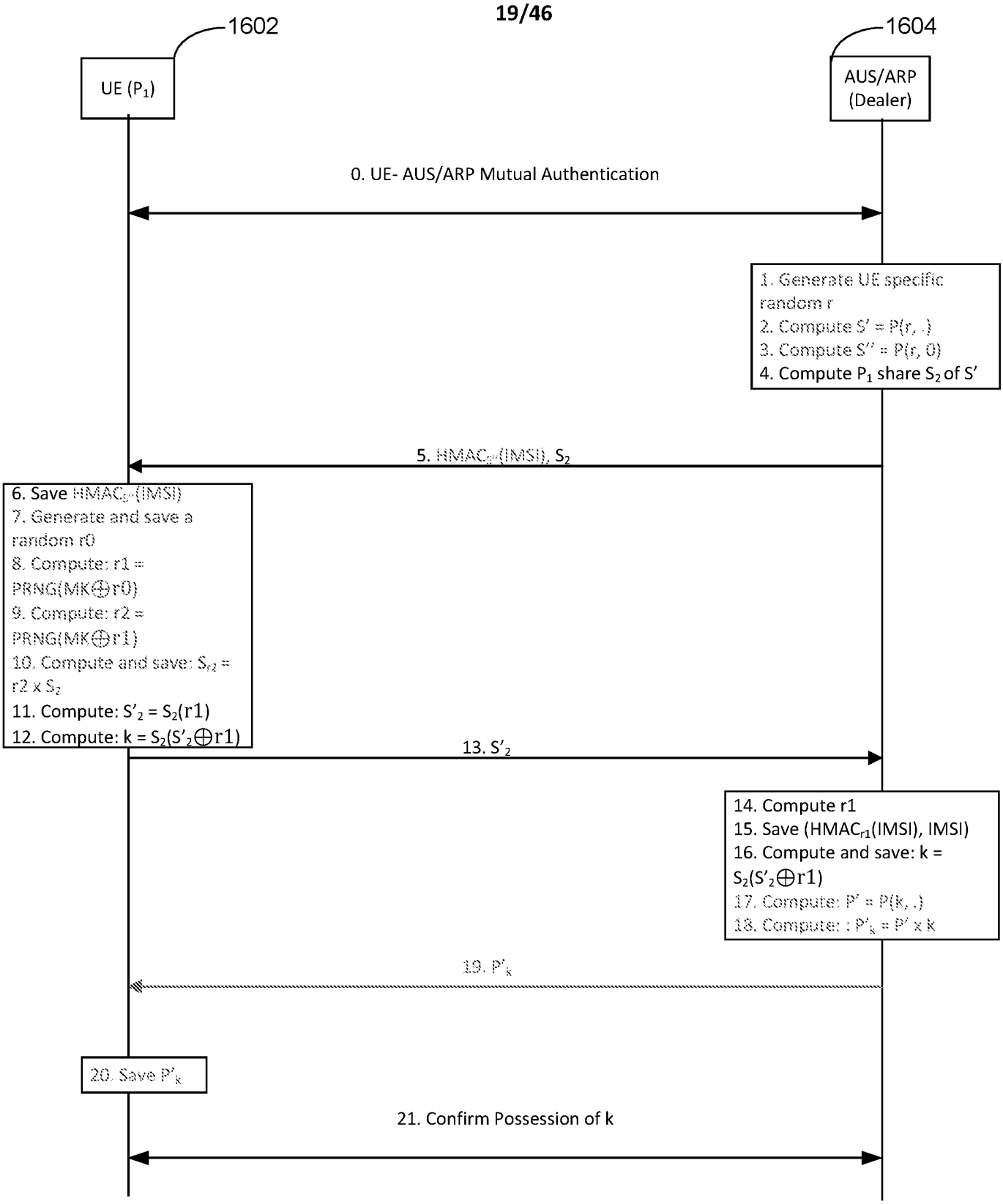

1602
UE (P_1)
1604
AUS/ARP (Dealer)
0. UE- AUS/ARP Mutual Authentication
1. Generate UE specific random r
2. Compute S' = P(r, .)
3. Compute S'' = P(r, 0)
4. Compute P_1 share S_2 of S'
5. $HMAC_{S''}$(IMSI), S_2
6. Save $HMAC_{S''}$(IMSI)
7. Generate and save a random r0
8. Compute: r1 = PRNG(MK⊕r0)
9. Compute: r2 = PRNG(MK⊕r1)
10. Compute and save: S_{r2} = r2 x S_2
11. Compute: $S'_2 = S_2(r1)$
12. Compute: $k = S_2(S'_2 \oplus r1)$
13. S'_2
14. Compute r1
15. Save ($HMAC_{r1}$(IMSI), IMSI)
16. Compute and save: $k = S_2(S'_2 \oplus r1)$
17. Compute: P' = P(k, .)
18. Compute: : $P'_k = P' \times k$
19. P'_k
20. Save P'_k
21. Confirm Possession of k

Fig. 16

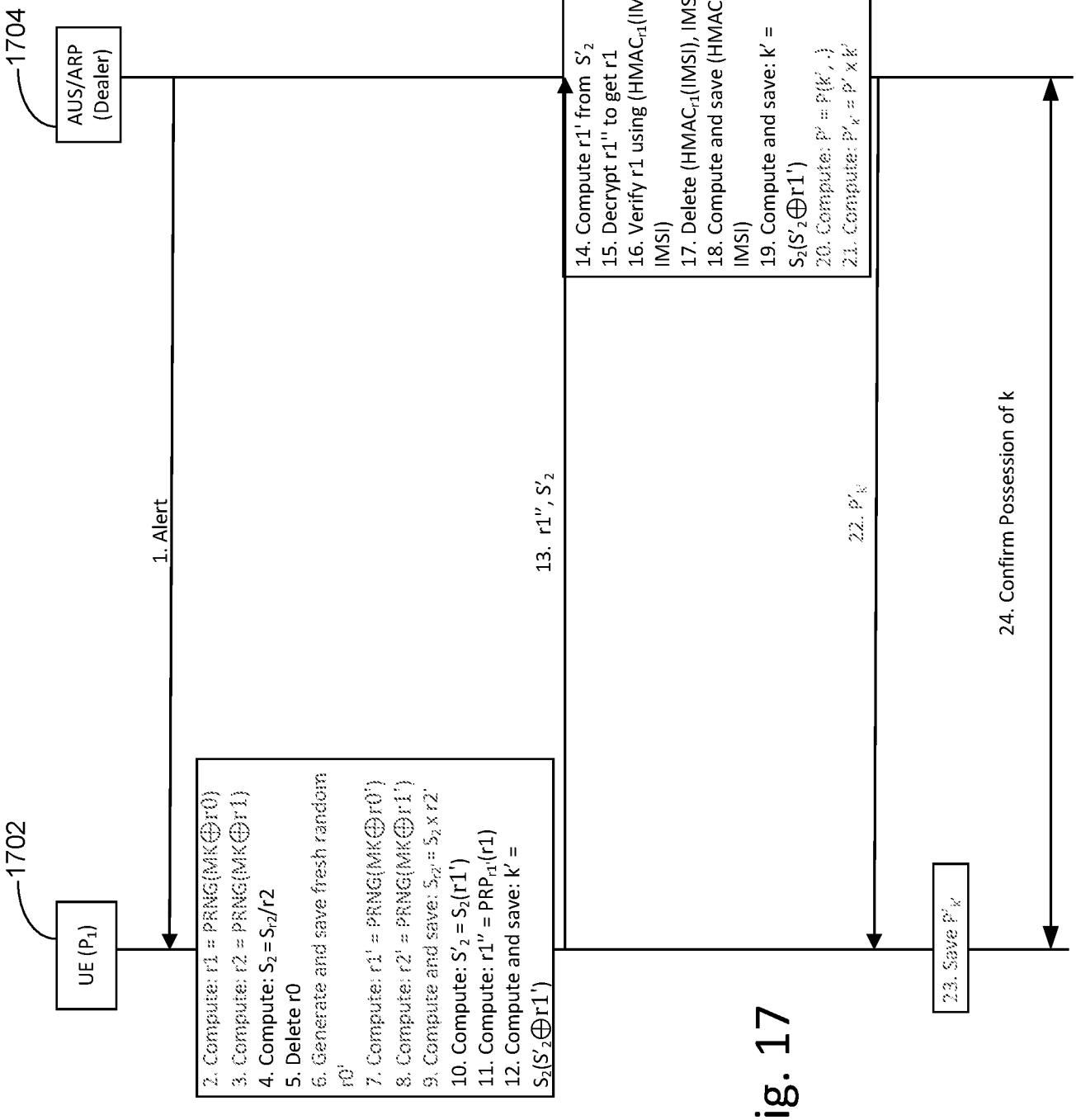

1702
UE (P_1)
1704
AUS/ARP (Dealer)
1. Alert
2. Compute: r1 = PRNG(MK⊕r0)
3. Compute: r2 = PRNG(MK⊕r1)
4. Compute: $S_2 = S_{r2}/r2$
5. Delete r0
6. Generate and save fresh random r0'
7. Compute: r1' = PRNG(MK⊕r0')
8. Compute: r2' = PRNG(MK⊕r1')
9. Compute and save: $S_{r2'} = S_2 \times r2'$
10. Compute: $S'_2 = S_2(r1')$
11. Compute: r1'' = $PRP_{r1'}(r1)$
12. Compute and save: k' = $S_2(S'_2 \oplus r1')$
13. r1'', S'_2
14. Compute r1' from S'_2
15. Decrypt r1'' to get r1
16. Verify r1 using ($HMAC_{r1}$(IMSI), IMSI)
17. Delete ($HMAC_{r1}$(IMSI), IMSI)
18. Compute and save ($HMAC_{r1'}$(IMSI), IMSI)
19. Compute and save: k' = $S_2(S'_2 \oplus r1')$
20. Compute: $P' = P(k',)$
21. Compute: $P'_k = P' \times k'$
22. P'_k
23. Save P'_k
24. Confirm Possession of k

Fig. 17

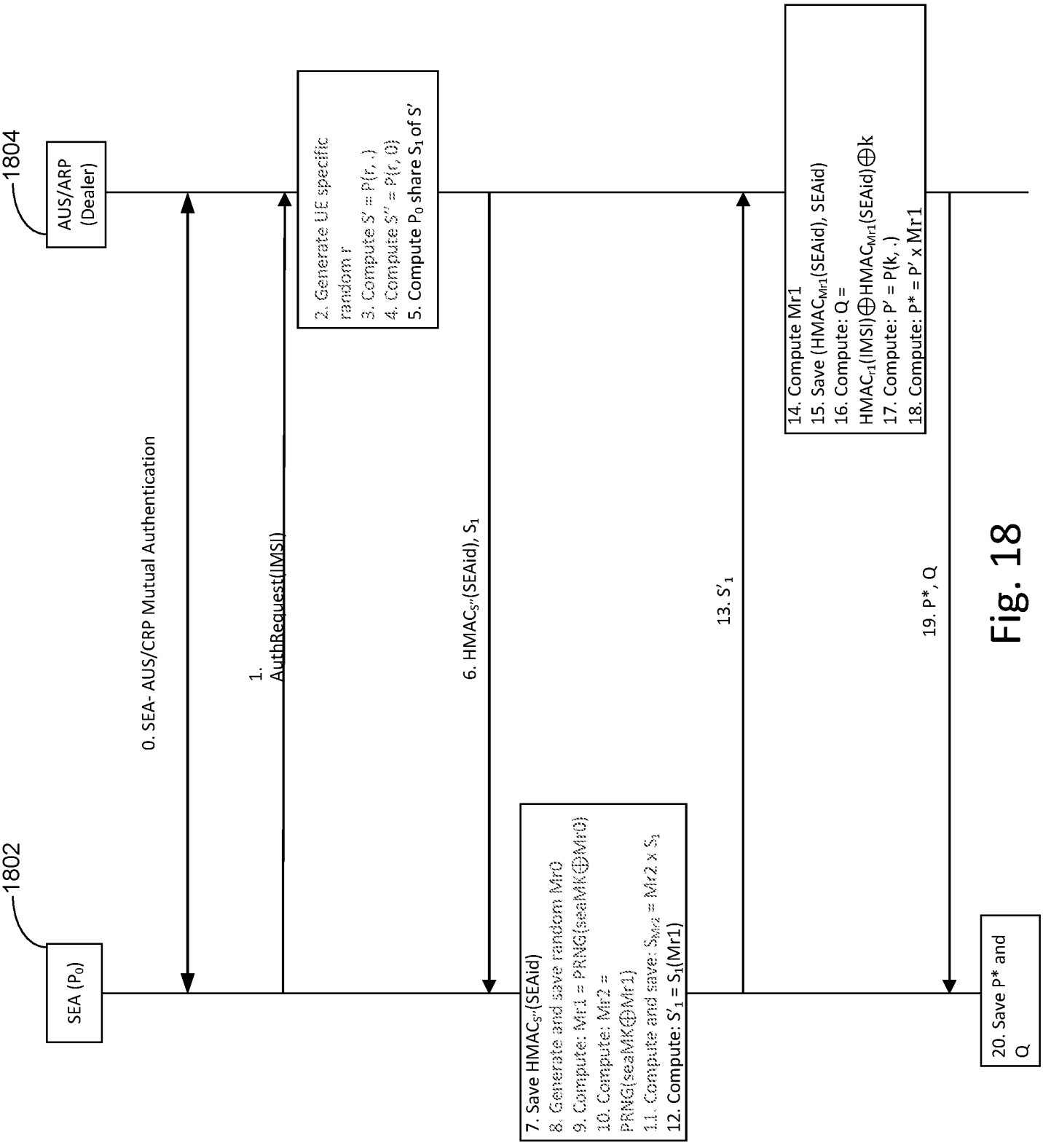

1802
SEA (P_0)
1804
AUS/ARP (Dealer)
0. SEA- AUS/CRP Mutual Authentication
1. AuthRequest(IMSI)
2. Generate UE specific random r
3. Compute S' = P(r, .)
4. Compute S'' = P(r, 0)
5. Compute P_0 share S_1 of S'
6. $HMAC_{S''}$(SEAid), S_1
7. Save $HMAC_{S''}$(SEAid)
8. Generate and save random Mr0
9. Compute: Mr1 = PRNG(seaMK⊕Mr0)
10. Compute: Mr2 = PRNG(seaMK⊕Mr1)
11. Compute and save: S_{Mr2} = Mr2 x S_1
12. Compute: $S'_1 = S_1$(Mr1)
13. S'_1
14. Compute Mr1
15. Save ($HMAC_{Mr1}$(SEAid), SEAid)
16. Compute: Q = $HMAC_{r1}$(IMSI)⊕$HMAC_{Mr1}$(SEAid)⊕k
17. Compute: P' = P(k, .)
18. Compute: P* = P' x Mr1
19. P*, Q
20. Save P* and Q

Fig. 18

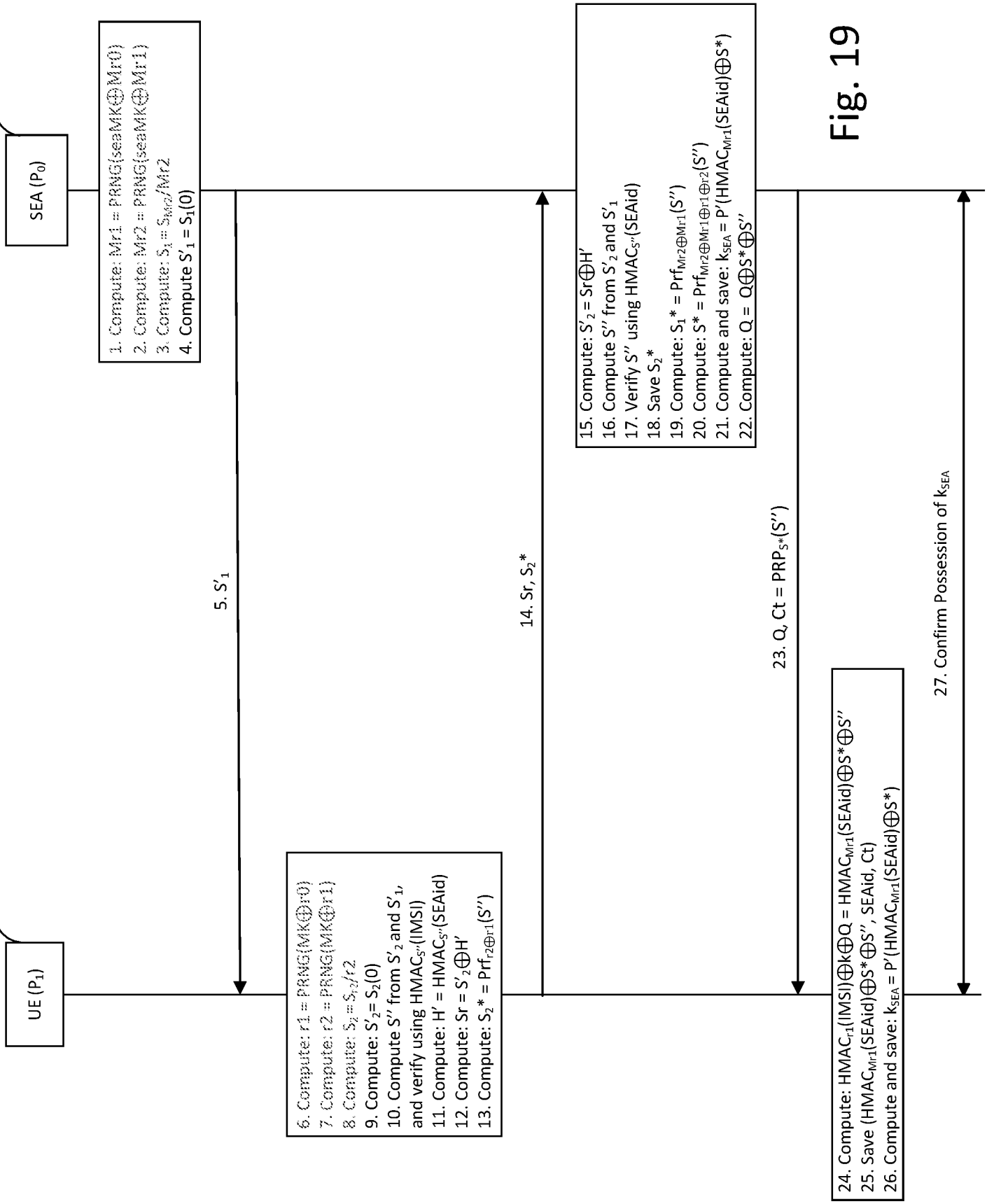

1902
UE (P1)
1904
SEA (P0)
1. Compute: Mr1 = PRNG(seaMK⊕Mr0)
2. Compute: Mr2 = PRNG(seaMK⊕Mr1)
3. Compute: S1 = SKMr2/Mr2
4. Compute S'1 = S1(0)
5. S'1
6. Compute: r1 = PRNG(MK⊕r0)
7. Compute: r2 = PRNG(MK⊕r1)
8. Compute: S2 = S1/r2
9. Compute: S'2 = S2(0)
10. Compute S'' from S'2 and S'1, and verify using HMACS''(IMSI)
11. Compute: H' = HMACS''(SEAid)
12. Compute: Sr = S'2⊕H'
13. Compute: S2* = Prfr2⊕r1(S'')
14. Sr, S2*
15. Compute: S'2 = Sr⊕H'
16. Compute S'' from S'2 and S'1
17. Verify S'' using HMACS''(SEAid)
18. Save S2*
19. Compute: S1* = PrfMr2⊕Mr1(S'')
20. Compute: S* = PrfMr2⊕Mr1⊕r1⊕r2(S'')
21. Compute and save: kSEA = P'(HMACMr1(SEAid)⊕S*)
22. Compute: Q = Q⊕S*⊕S''
23. Q, Ct = PRPS*(S'')
24. Compute: HMACr1(IMSI)⊕k⊕Q = HMACMr1(SEAid)⊕S*⊕S''
25. Save (HMACMr1(SEAid)⊕S*⊕S'', SEAid, Ct)
26. Compute and save: kSEA = P'(HMACMr1(SEAid)⊕S*)
27. Confirm Possession of kSEA

Fig. 19

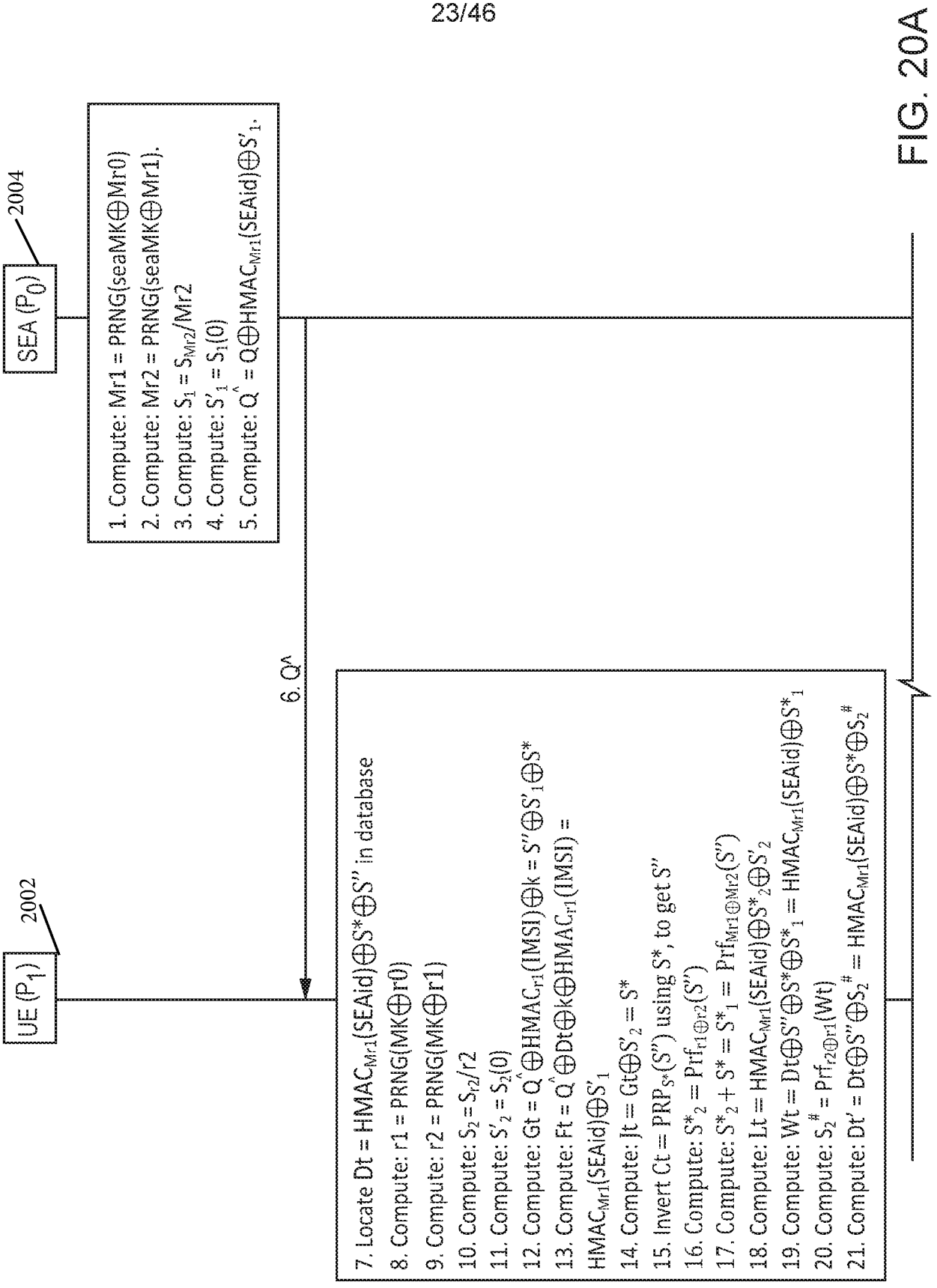

UE (P1)
2002
SEA (P0)
2004
1. Compute: Mr1 = PRNG(seaMK⊕Mr0)
2. Compute: Mr2 = PRNG(seaMK⊕Mr1).
3. Compute: S1 = SMr2/Mr2
4. Compute: S'1 = S1(0)
5. Compute: Q^ = Q⊕HMACMr1(SEAid)⊕S'1.
6. Q^
7. Locate Dt = HMACMr1(SEAid)⊕S*⊕S'' in database
8. Compute: r1 = PRNG(MK⊕r0)
9. Compute: r2 = PRNG(MK⊕r1)
10. Compute: S2 = Sr2/r2
11. Compute: S'2 = S2(0)
12. Compute: Gt = Q^⊕HMACr1(IMSI)⊕k = S''⊕S'1⊕S*
13. Compute: Ft = Q^⊕Dt⊕k⊕HMACr1(IMSI) = HMACMr1(SEAid)⊕S'1
14. Compute: Jt = Gt⊕S'2 = S*
15. Invert Ct = PRPS*(S'') using S*, to get S''
16. Compute: S*2 = Prfr1⊕r2(S'')
17. Compute: S*2 + S* = S*1 = PrfMr1⊕Mr2(S'')
18. Compute: Lt = HMACMr1(SEAid)⊕S*2⊕S'2
19. Compute: Wt = Dt⊕S''⊕S*⊕S*1 = HMACMr1(SEAid)⊕S*1
20. Compute: S2# = Prfr2⊕r1(Wt)
21. Compute: Dt' = Dt⊕S''⊕S2# = HMACMr1(SEAid)⊕S*⊕S2#

FIG. 20A

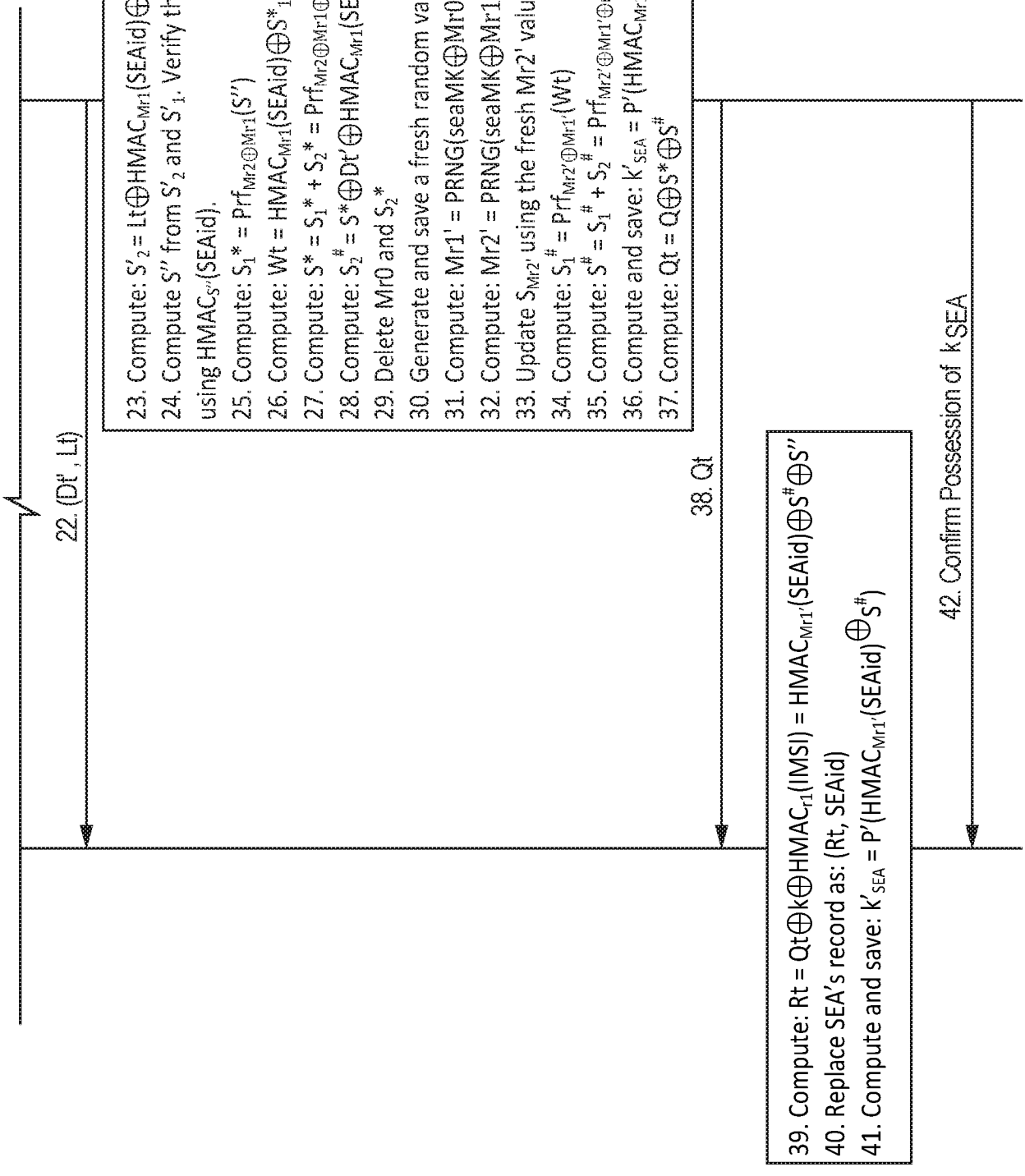

22. (Df', Lt)
23. Compute: S'2 = Lt⊕HMAC_Mr1(SEAid)⊕S*2
24. Compute S'' from S'2 and S'1. Verify the value of S'' by using HMAC_S''(SEAid).
25. Compute: S1* = Prf_Mr2⊕Mr1(S'')
26. Compute: Wt = HMAC_Mr1(SEAid)⊕S*1
27. Compute: S* = S1* + S2* = Prf_Mr2⊕Mr1⊕r2⊕r1(S'')
28. Compute: S2# = S*⊕Dt'⊕HMAC_Mr1(SEAid)
29. Delete Mr0 and S2*
30. Generate and save a fresh random value of Mr0'
31. Compute: Mr1' = PRNG(seaMK⊕Mr0')
32. Compute: Mr2' = PRNG(seaMK⊕Mr1')
33. Update S_Mr2' using the fresh Mr2' value
34. Compute: S1# = Prf_Mr2'⊕Mr1'(Wt)
35. Compute: S# = S1# + S2# = Prf_Mr2'⊕Mr1'⊕r2⊕r1(Wt)
36. Compute and save: K'_SEA = P'(HMAC_Mr1'(SEAid)⊕S#)
37. Compute: Qt = Q⊕S*⊕S#
38. Qt
39. Compute: Rt = Qt⊕k⊕HMAC_r1(IMSI) = HMAC_Mr1'(SEAid)⊕S#⊕S''
40. Replace SEA's record as: (Rt, SEAid)
41. Compute and save: K'_SEA = P'(HMAC_Mr1'(SEAid)⊕S#)
42. Confirm Possession of kSEA

FIG. 20B

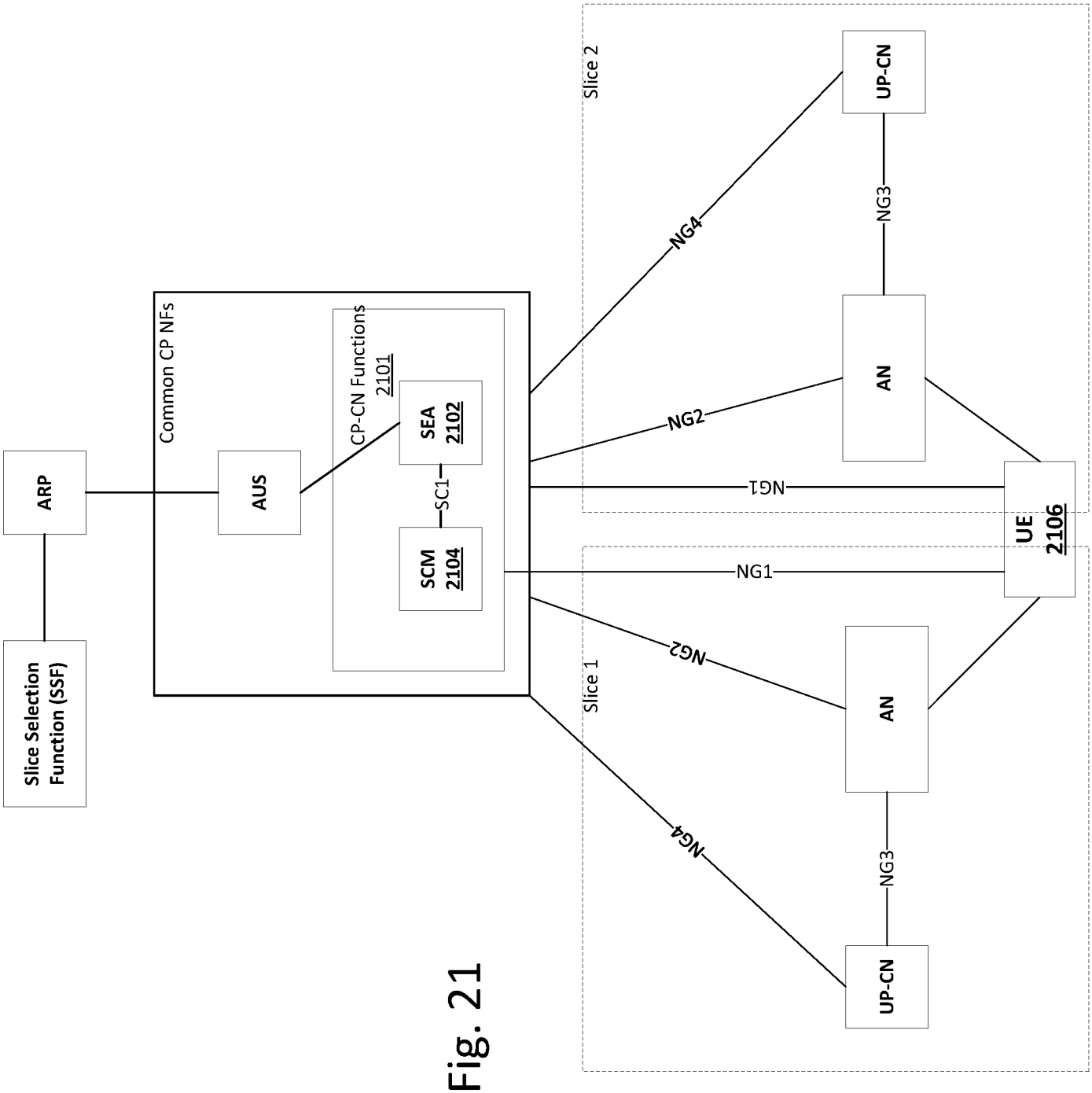

Slice 2
UP-CN
NG3
NG4
AN
NG2
NG1
UE
2106
ARP
Common CP NFs
AUS
CP-CN Functions
2101
SEA
2102
SC1
SCM
2104
NG1
NG2
AN
NG4
NG3
Slice 1
UP-CN
Slice Selection
Function (SSF)

Fig. 21

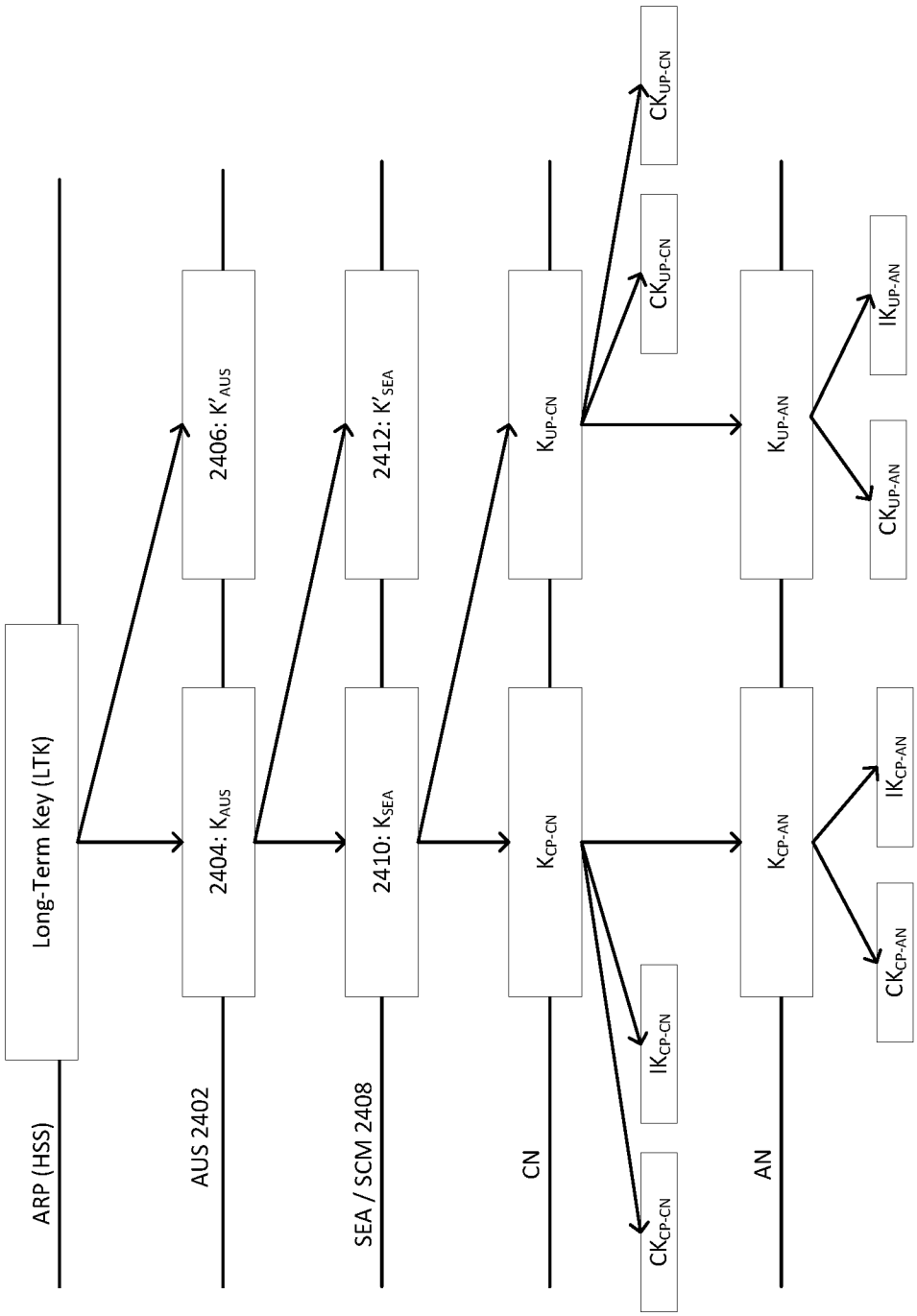

ARP (HSS)
Long-Term Key (LTK)
AUS 2402
2404: K_{AUS}
2406: K'_{AUS}
SEA / SCM 2408
2410: K_{SEA}
2412: K'_{SEA}
CN
K_{CP-CN}
K_{UP-CN}
CK_{CP-CN}
IK_{CP-CN}
CK_{UP-CN}
CK_{UP-CN}
AN
K_{CP-AN}
K_{UP-AN}
CK_{CP-AN}
IK_{CP-AN}
IK_{UP-AN}
CK_{UP-AN}

Fig. 22

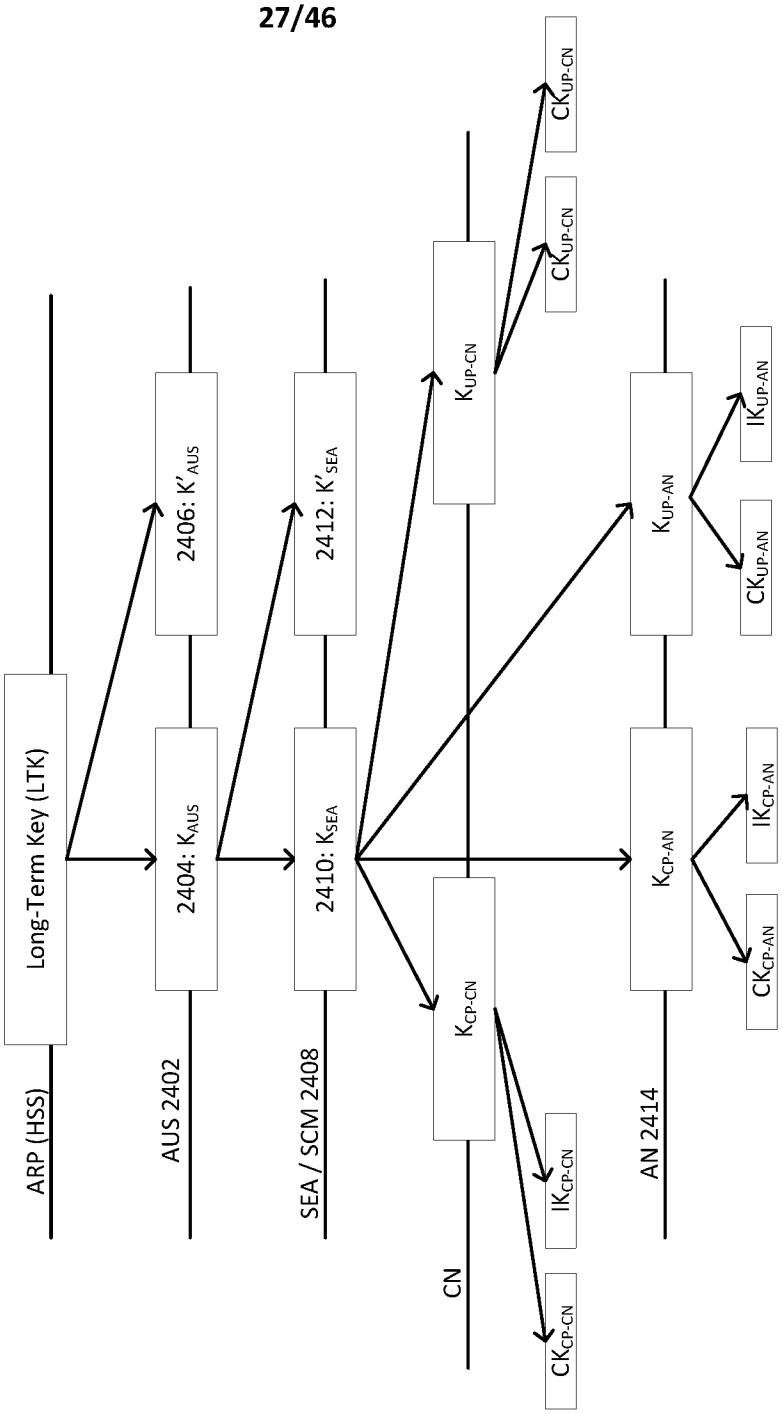

Long-Term Key (LTK)
2404: K_{AUS}
2406: K'_{AUS}
2410: K_{SEA}
2412: K'_{SEA}
K_{UP-CN}
K_{CP-CN}
K_{UP-AN}
K_{CP-AN}
CK_{UP-CN}
CK_{UP-CN}
IK_{CP-CN}
CK_{CP-CN}
IK_{UP-AN}
CK_{UP-AN}
IK_{CP-AN}
CK_{CP-AN}
ARP (HSS)
AUS 2402
SEA / SCM 2408
CN
AN 2414

Fig. 23

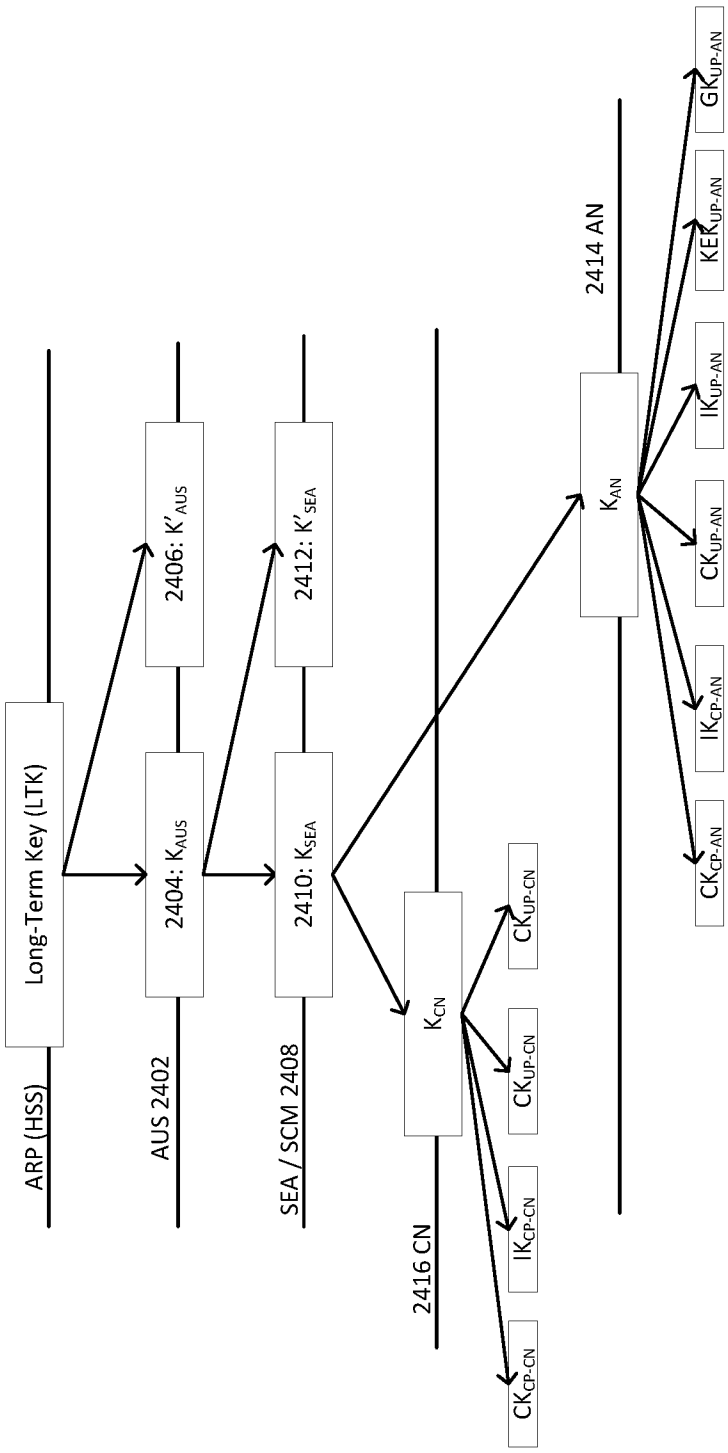

ARP (HSS)
Long-Term Key (LTK)
AUS 2402
2404: K_{AUS}
2406: K'_{AUS}
SEA / SCM 2408
2410: K_{SEA}
2412: K'_{SEA}
2416 CN
K_{CN}
CK_{UP-CN}
CK_{UP-CN}
IK_{CP-CN}
CK_{CP-CN}
2414 AN
K_{AN}
GK_{UP-AN}
KEK_{UP-AN}
IK_{UP-AN}
CK_{UP-AN}
IK_{CP-AN}
CK_{CP-AN}

Fig. 24

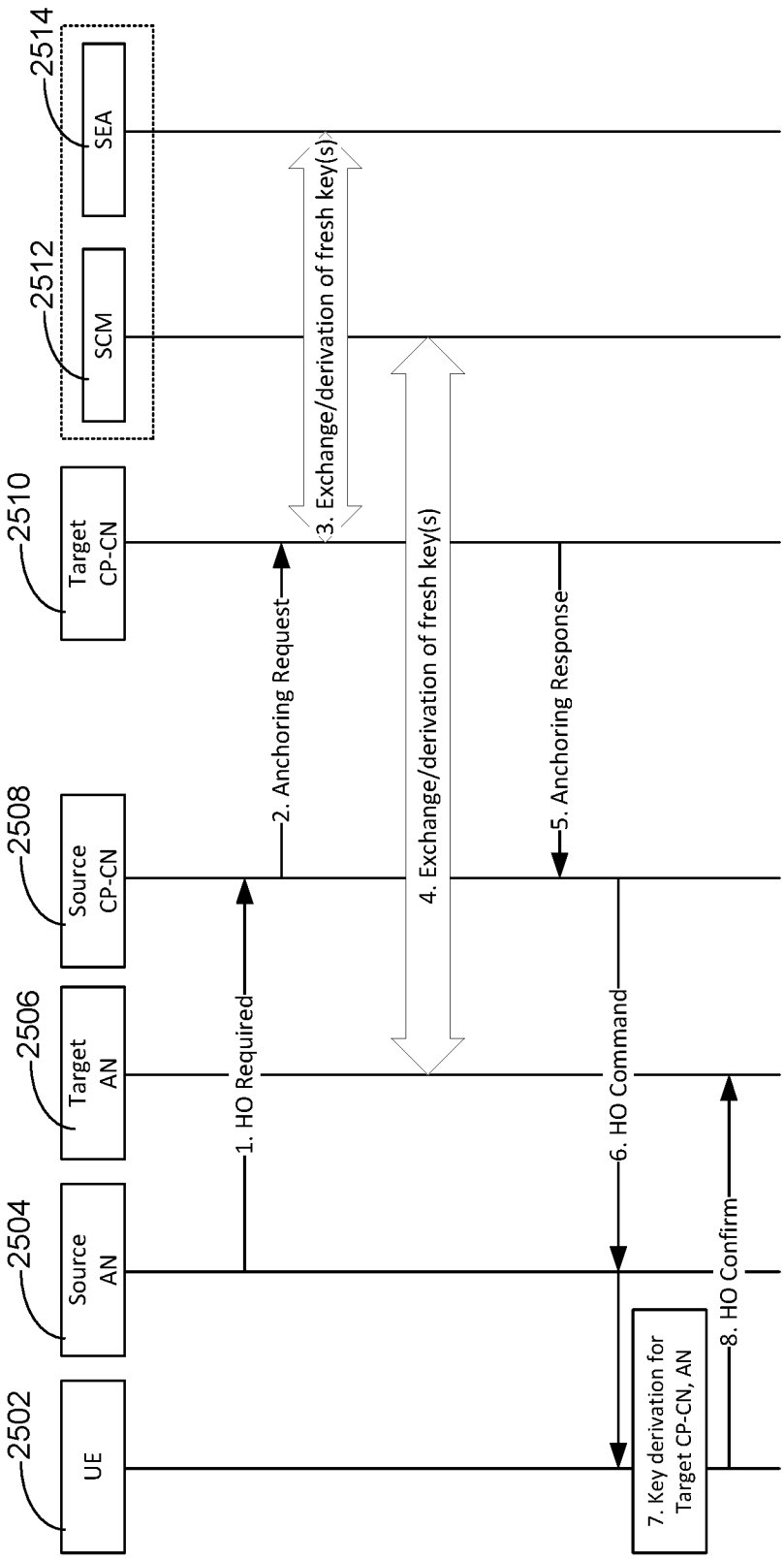

2514
SEA
2512
SCM
2510
Target CP-CN
2508
Source CP-CN
2506
Target AN
2504
Source AN
2502
UE
3. Exchange/derivation of fresh key(s)
2. Anchoring Request
4. Exchange/derivation of fresh key(s)
5. Anchoring Response
1. HO Required
6. HO Command
8. HO Confirm
7. Key derivation for Target CP-CN, AN

Fig. 25

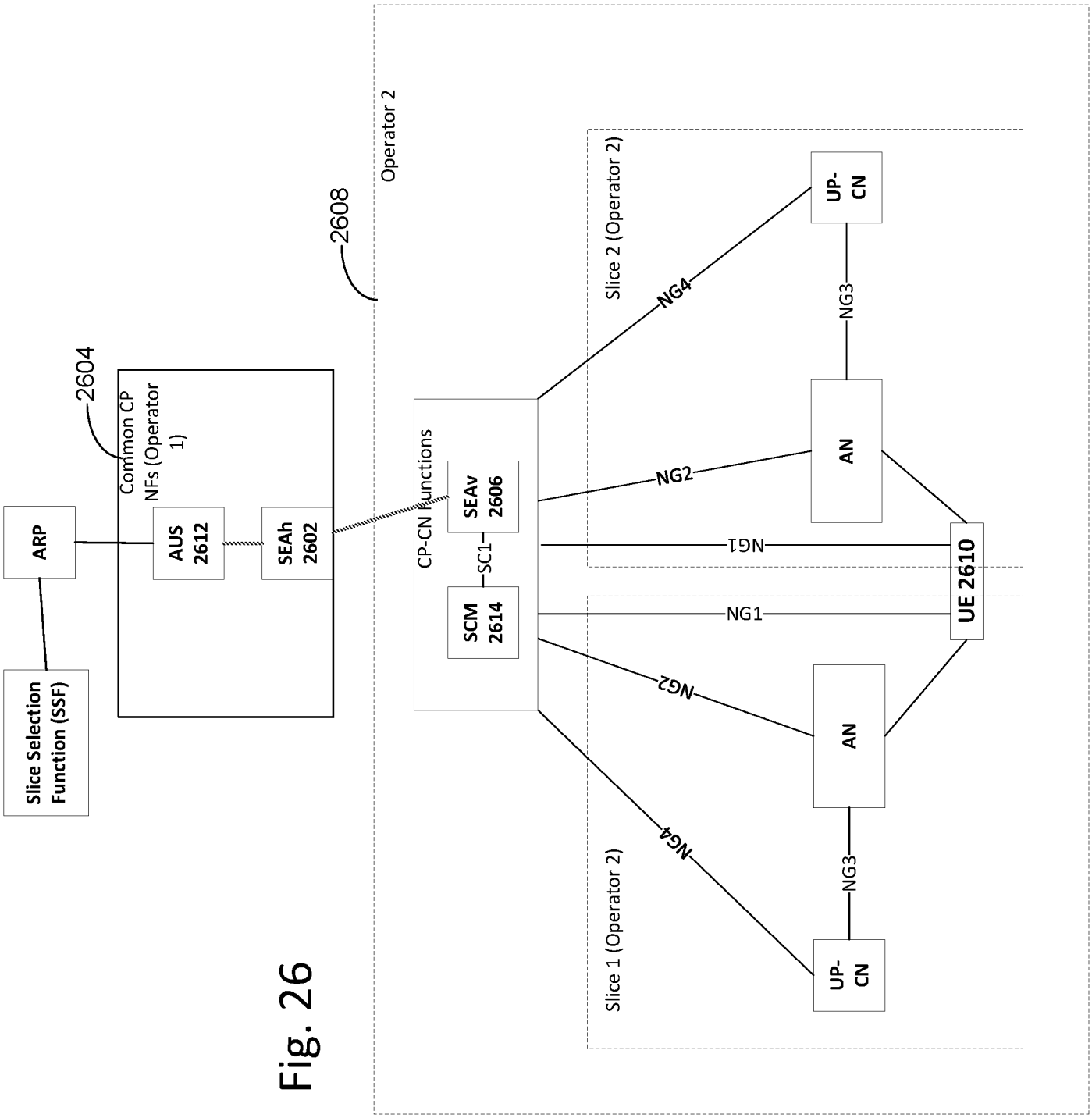

Operator 2
2608
2604
Common CP NFs (Operator 1)
ARP
AUS 2612
SEAh 2602
Slice Selection Function (SSF)
CP-CN functions
SEAv 2606
SC1
SCM 2614
Slice 2 (Operator 2)
Slice 1 (Operator 2)
NG4
NG3
NG2
NG1
UP-CN
AN
UE 2610

Fig. 26

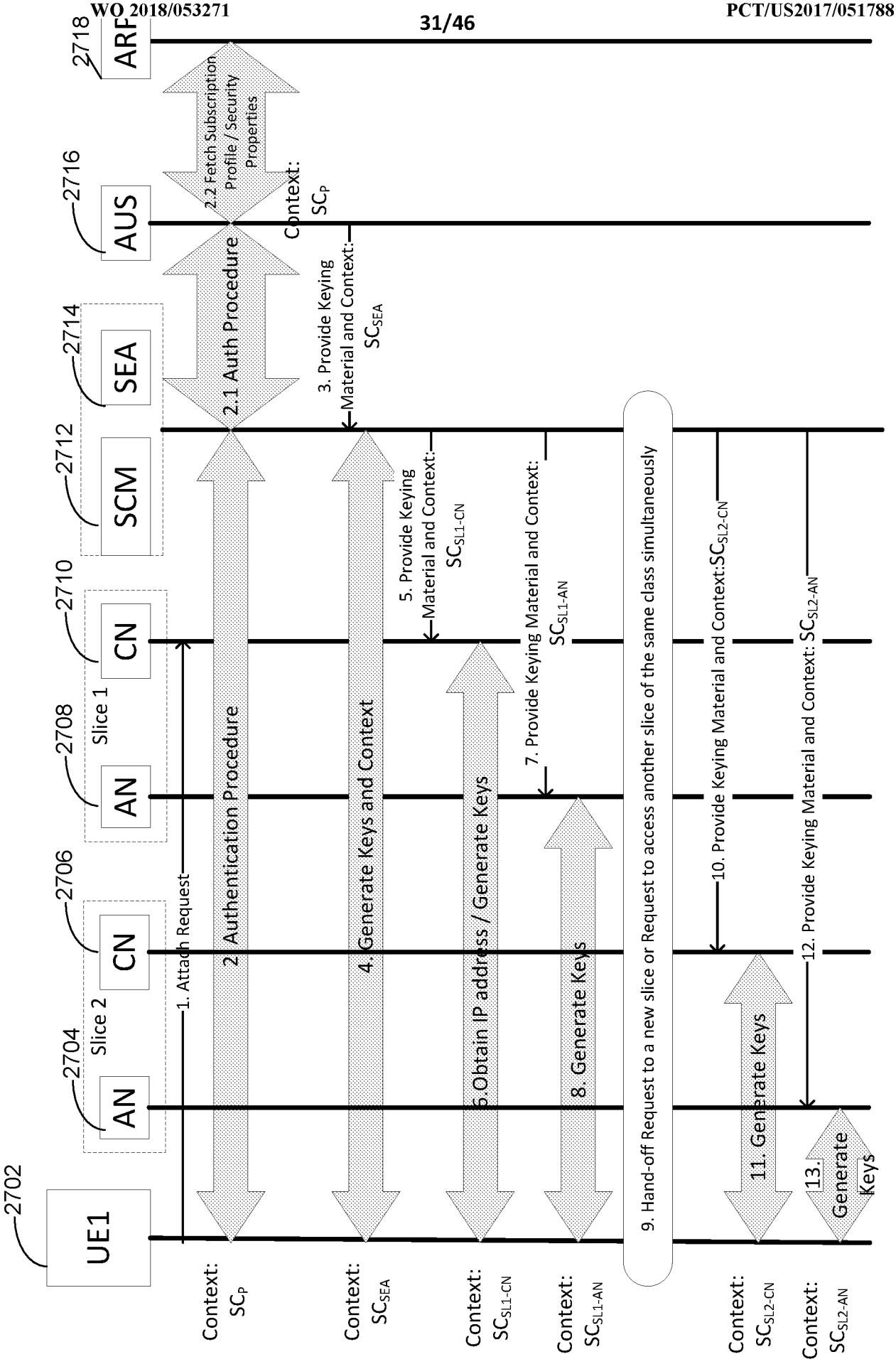

2702
UE1
2704
AN
Slice 2
2706
CN
2708
AN
Slice 1
2710
CN
2712
SCM
2714
SEA
2716
AUS
2718
ARF
1. Attach Request
2. Authentication Procedure
2.1 Auth Procedure
2.2 Fetch Subscription Profile / Security Properties
3. Provide Keying Material and Context: SCSEA
4. Generate Keys and Context
5. Provide Keying Material and Context: SCSL1-CN
6.Obtain IP address / Generate Keys
7. Provide Keying Material and Context: SCSL1-AN
8. Generate Keys
9. Hand-off Request to a new slice or Request to access another slice of the same class simultaneously
10. Provide Keying Material and Context:SCSL2-CN
11. Generate Keys
12. Provide Keying Material and Context: SCSL2-AN
13. Generate Keys
Context: SCP
Context: SCSEA
Context: SCSL1-CN
Context: SCSL1-AN
Context: SCSL2-CN
Context: SCSL2-AN

Fig. 27

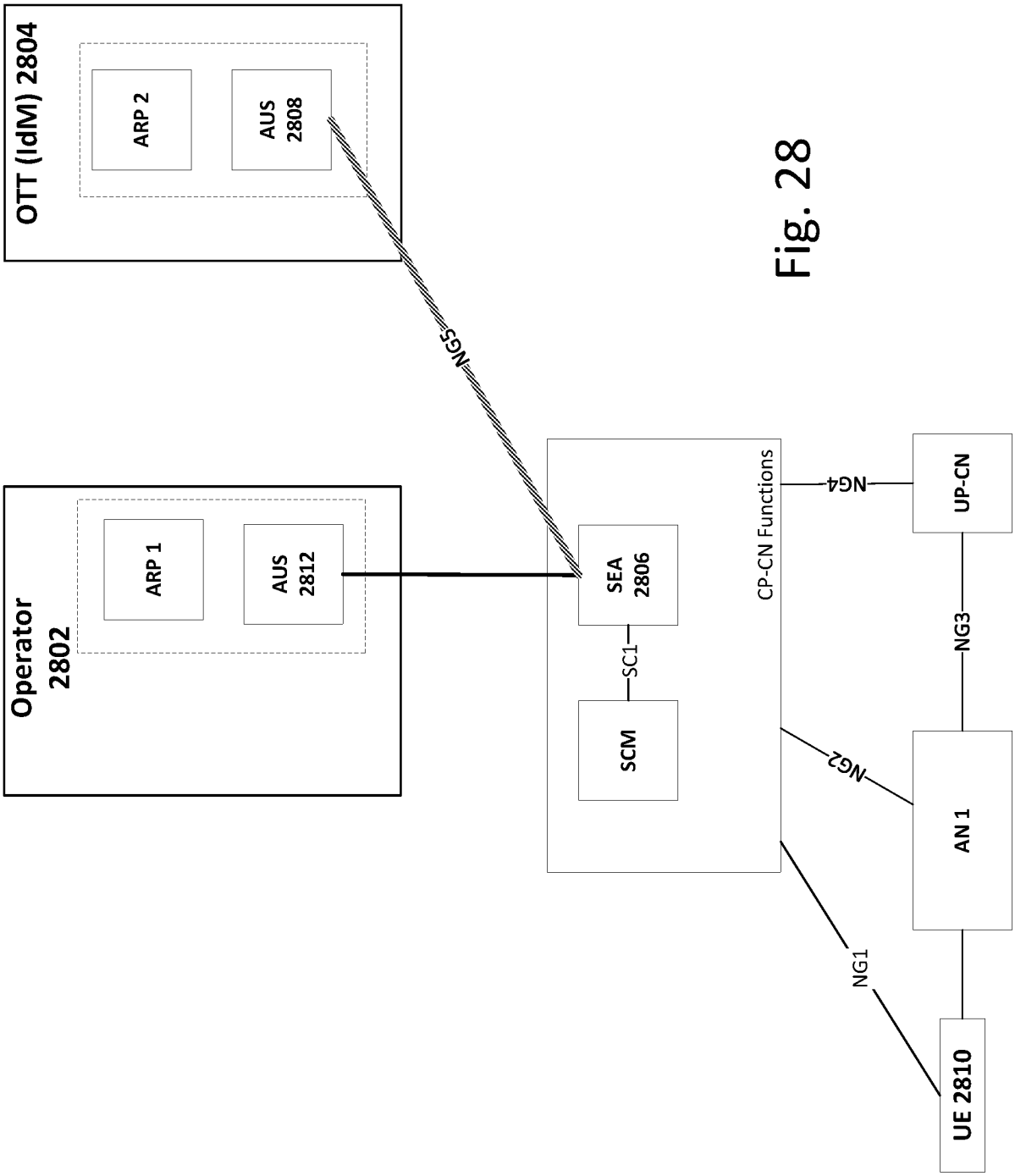

OTT (IdM) 2804
ARP 2
AUS 2808
Operator 2802
ARP 1
AUS 2812
NG5
SEA 2806
SC1
SCM
CP-CN Functions
NG4
UP-CN
NG3
NG2
AN 1
NG1
UE 2810

Fig. 28

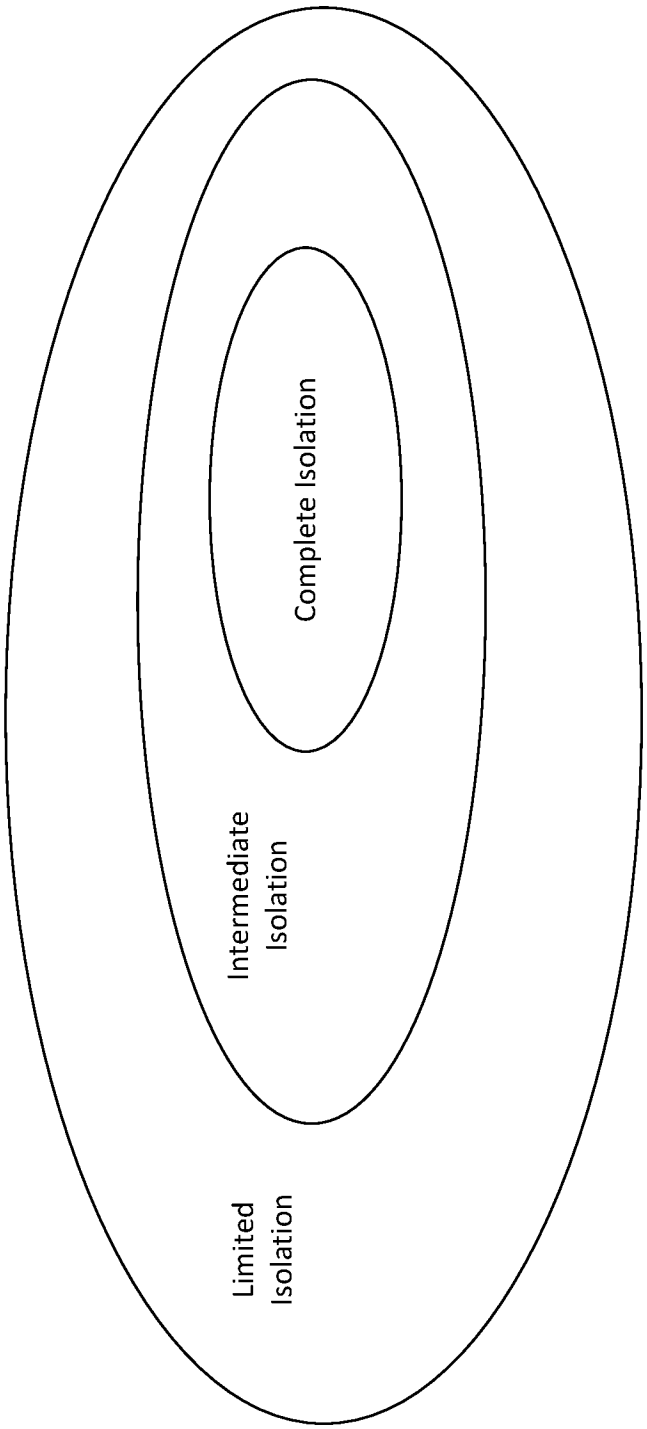

Complete Isolation
Intermediate Isolation
Limited Isolation

Fig. 29

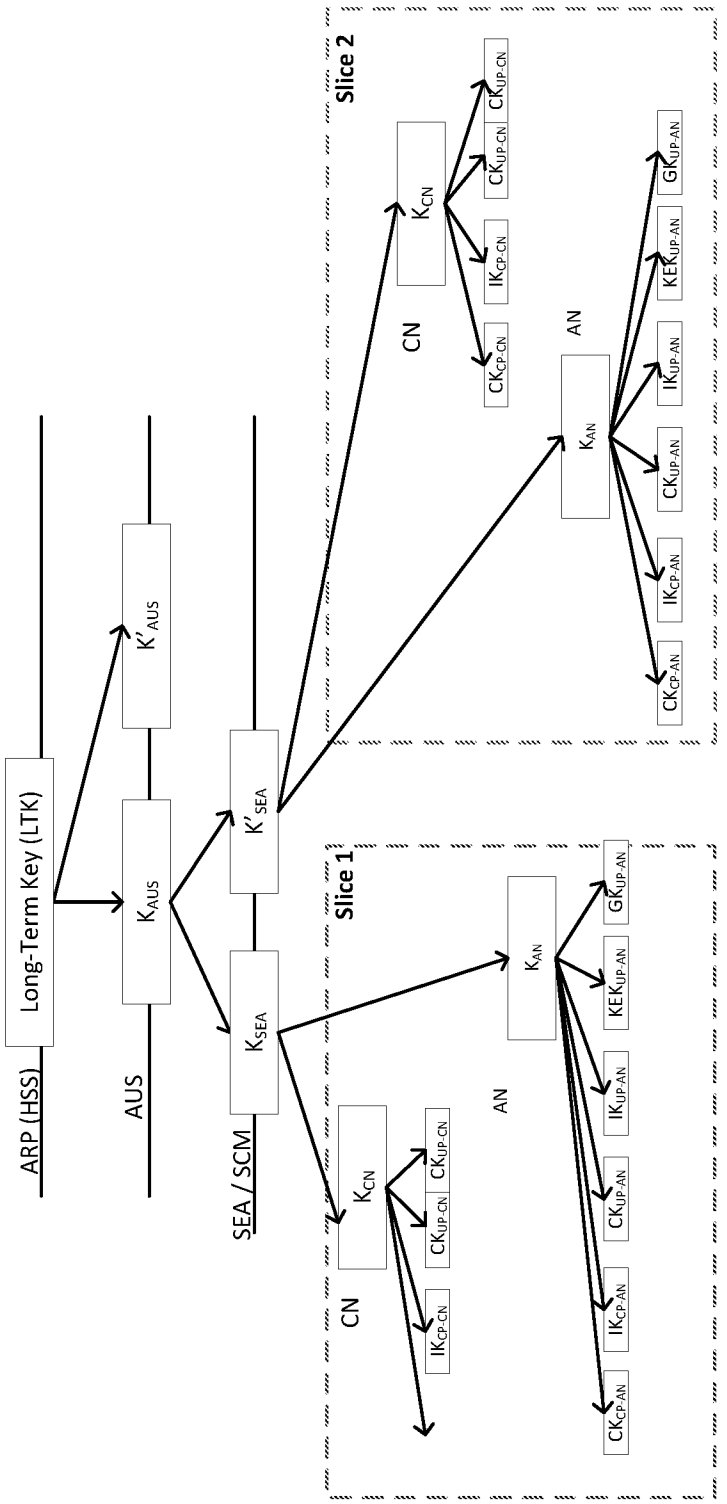

ARP (HSS)
Long-Term Key (LTK)
AUS
SEA / SCM
K_{AUS}
K'_{AUS}
K_{SEA}
K'_{SEA}
Slice 1
Slice 2
CN
AN
K_{CN}
K_{AN}
IK_{CP-CN}
CK_{UP-CN}
CK_{CP-CN}
CK_{CP-AN}
IK_{CP-AN}
CK_{UP-AN}
IK_{UP-AN}
KEK_{UP-AN}
GK_{UP-AN}

Fig. 31

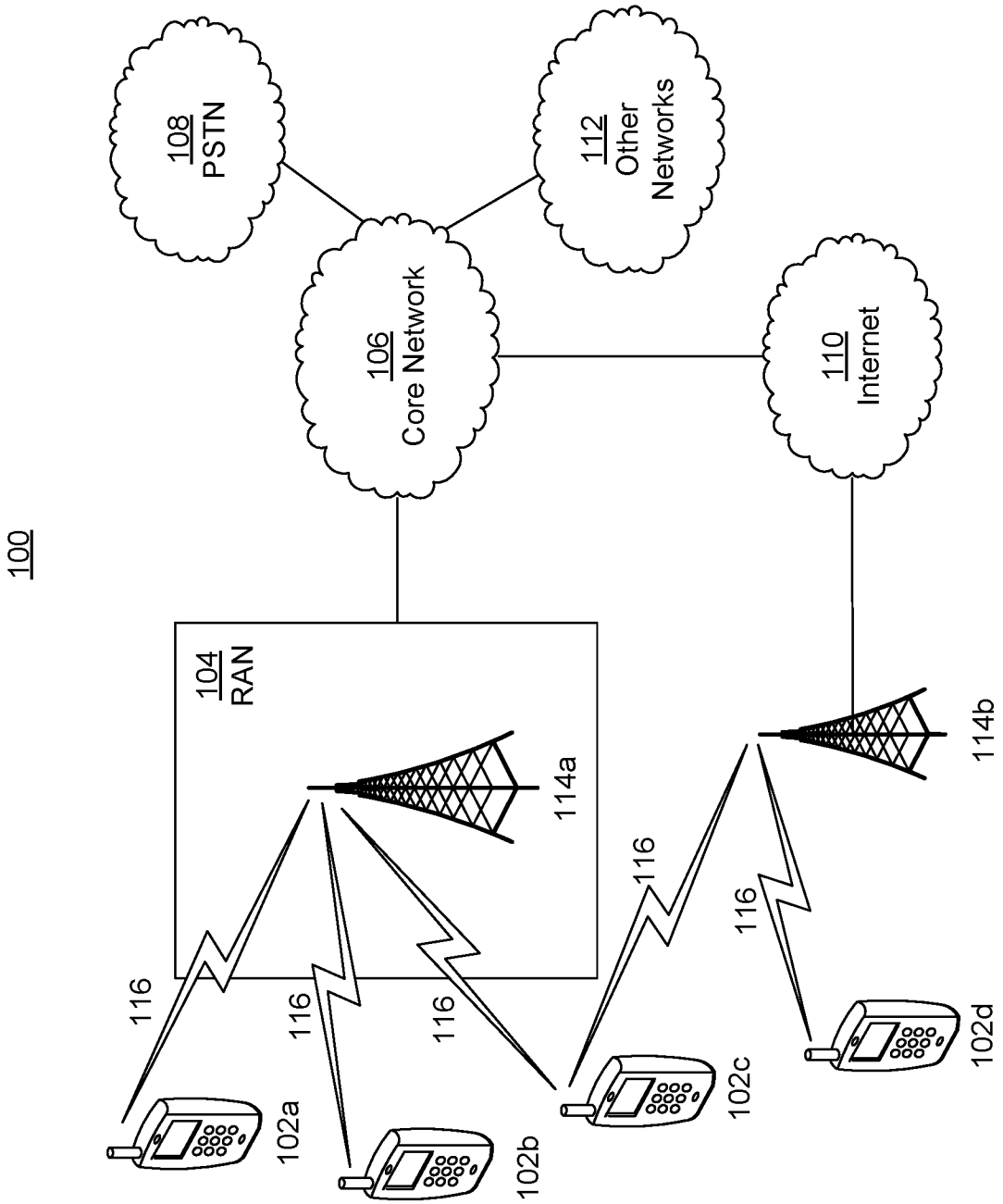

100
108 PSTN
112 Other Networks
106 Core Network
110 Internet
104 RAN
114a
114b
116
102a
102b
102c
102d

Fig. 31A

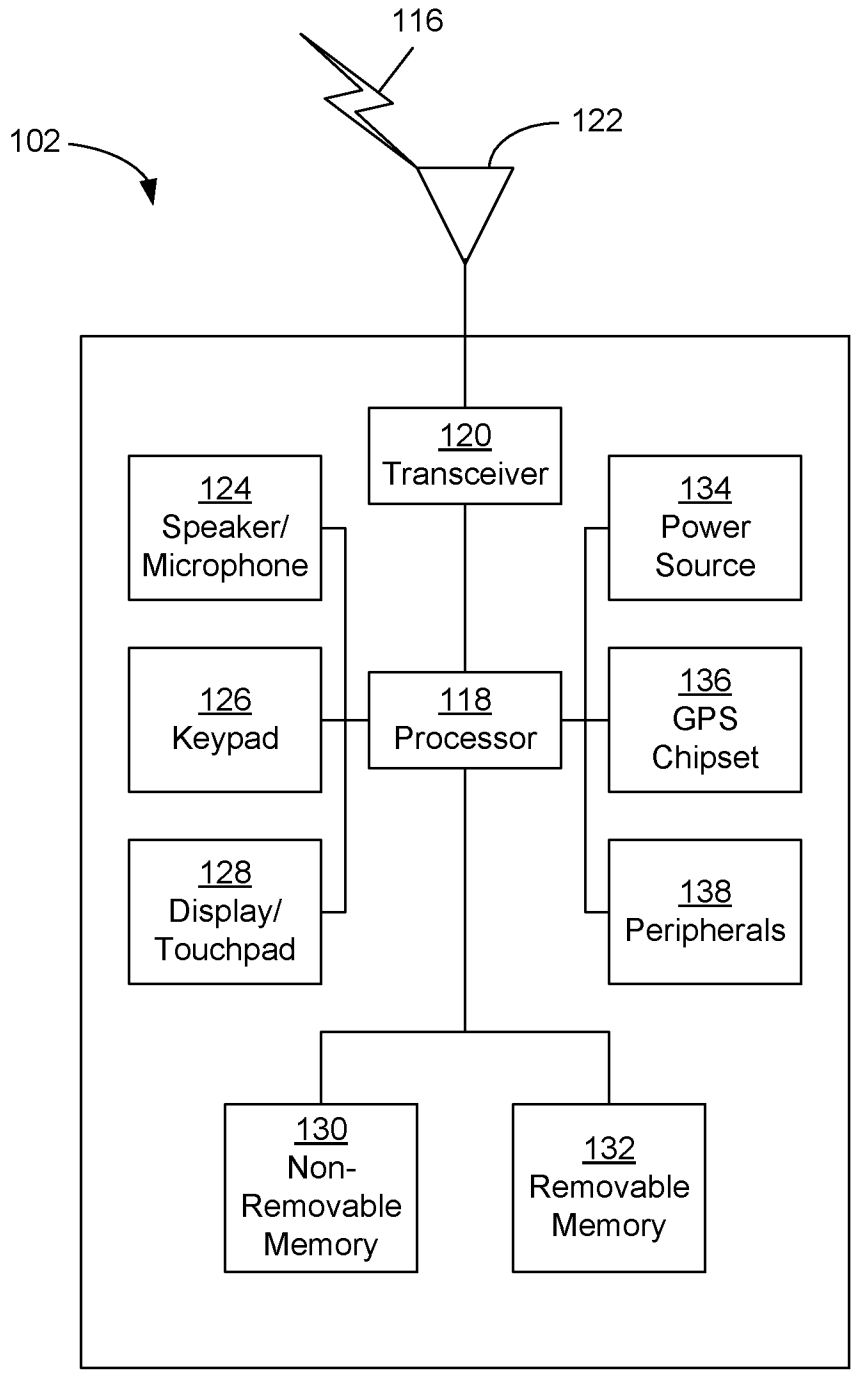

116
122
102
120
Transceiver
124
Speaker/
Microphone
134
Power
Source
126
Keypad
118
Processor
136
GPS
Chipset
128
Display/
Touchpad
138
Peripherals
130
Non-
Removable
Memory
132
Removable
Memory

Fig. 31B

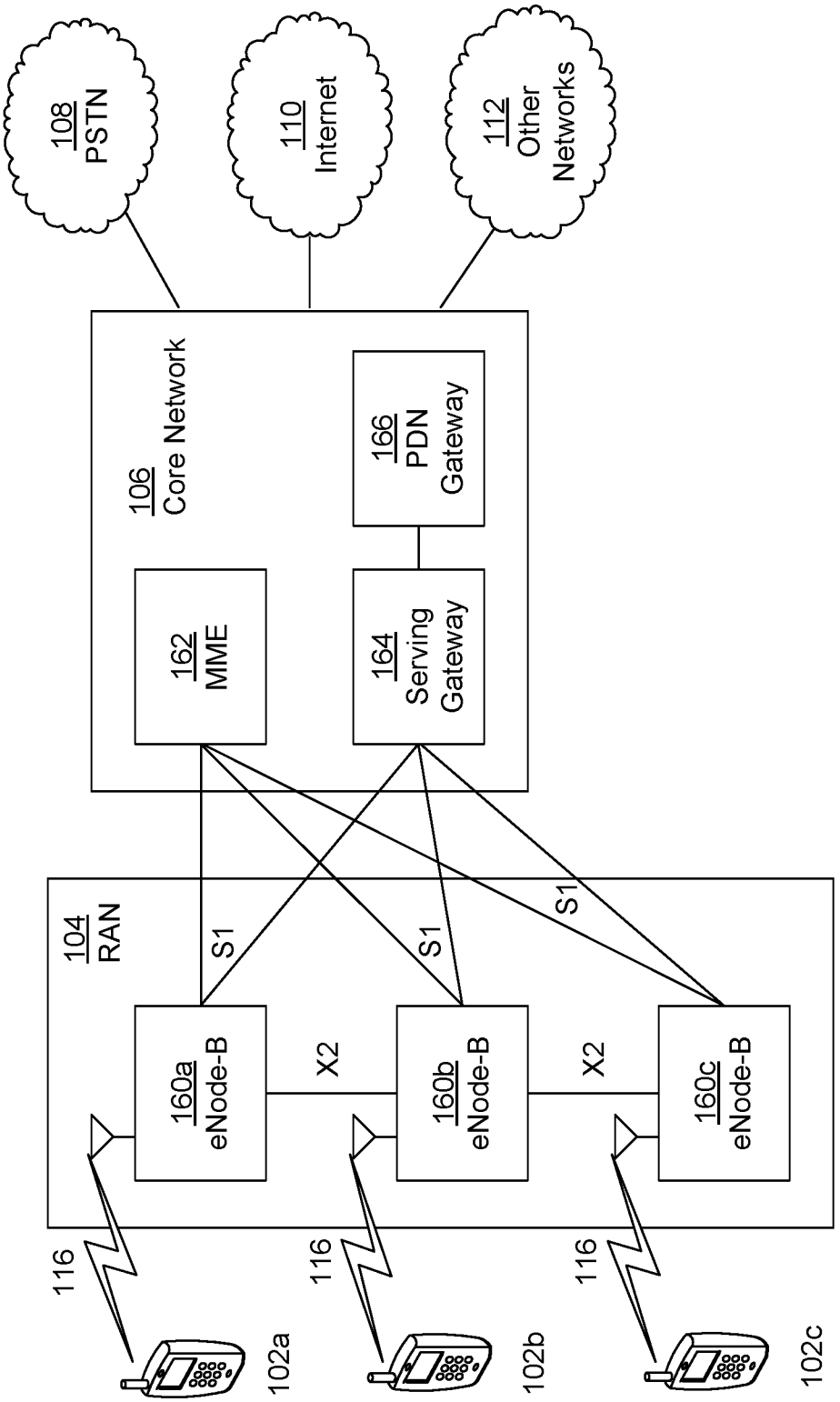

108 PSTN
110 Internet
112 Other Networks
106 Core Network
166 PDN Gateway
162 MME
164 Serving Gateway
104 RAN
S1
S1
S1
160a eNode-B
160b eNode-B
160c eNode-B
X2
X2
116
116
116
102a
102b
102c

Fig. 31C

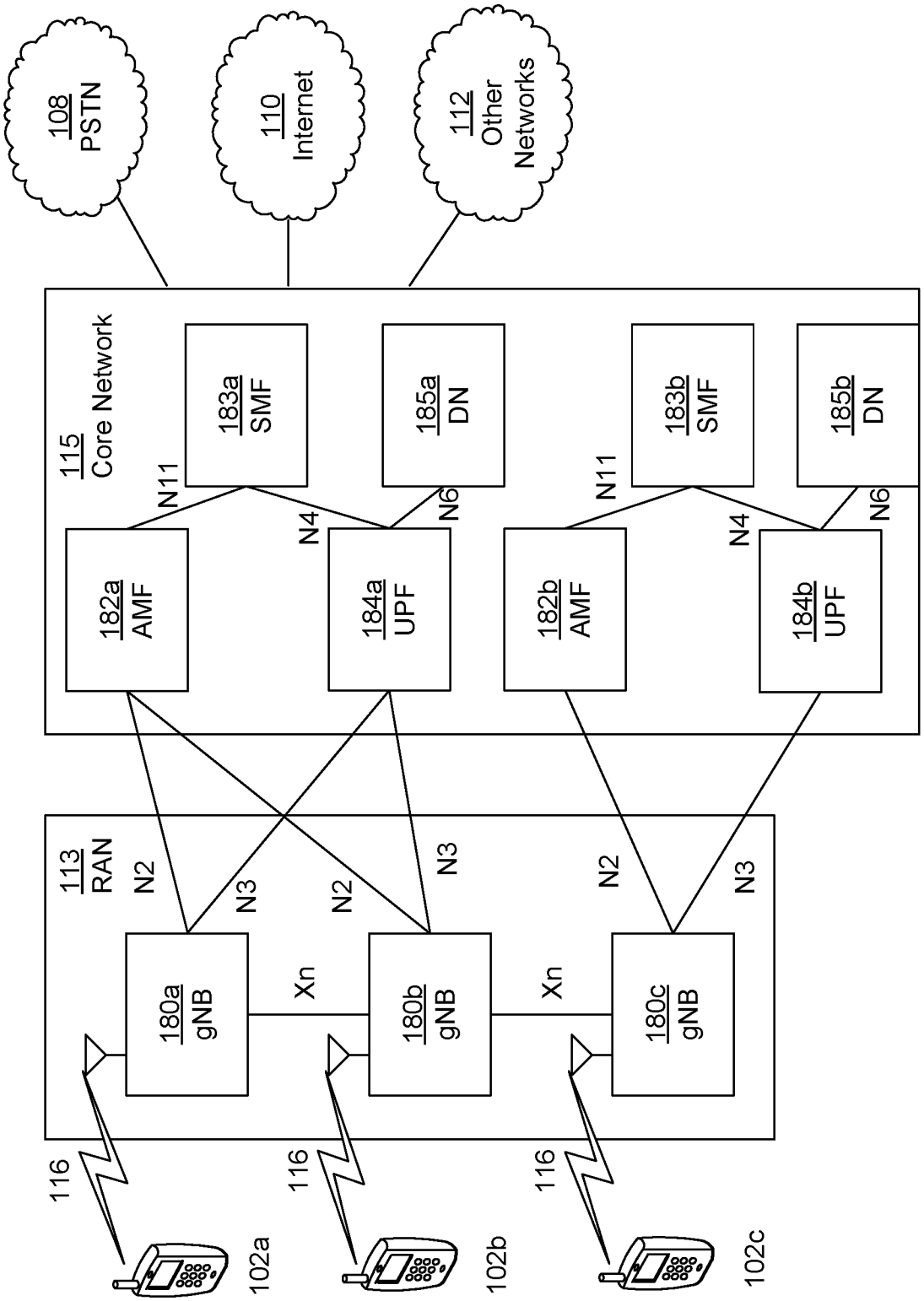

108 PSTN
110 Internet
112 Other Networks
115 Core Network
183a SMF
185a DN
183b SMF
185b DN
N11
N6
N4
182a AMF
184a UPF
182b AMF
184b UPF
113 RAN
N2
N3
180a gNB
180b gNB
180c gNB
Xn
116
102a
102b
102c

Fig. 31D

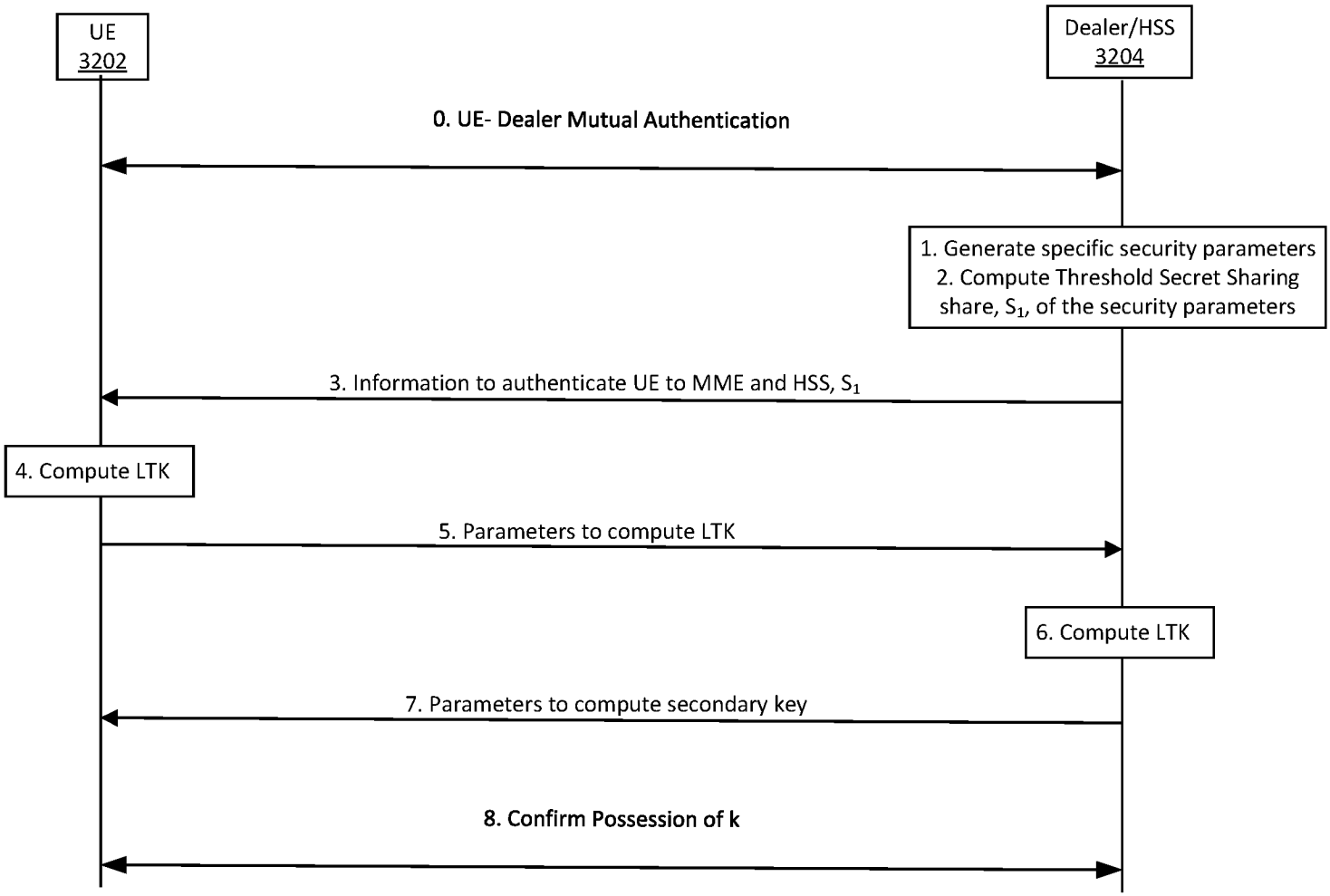

UE
3202
Dealer/HSS
3204
0. UE- Dealer Mutual Authentication
1. Generate specific security parameters
2. Compute Threshold Secret Sharing share, S_1, of the security parameters
3. Information to authenticate UE to MME and HSS, S_1
4. Compute LTK
5. Parameters to compute LTK
6. Compute LTK
7. Parameters to compute secondary key
8. Confirm Possession of k

FIG. 32

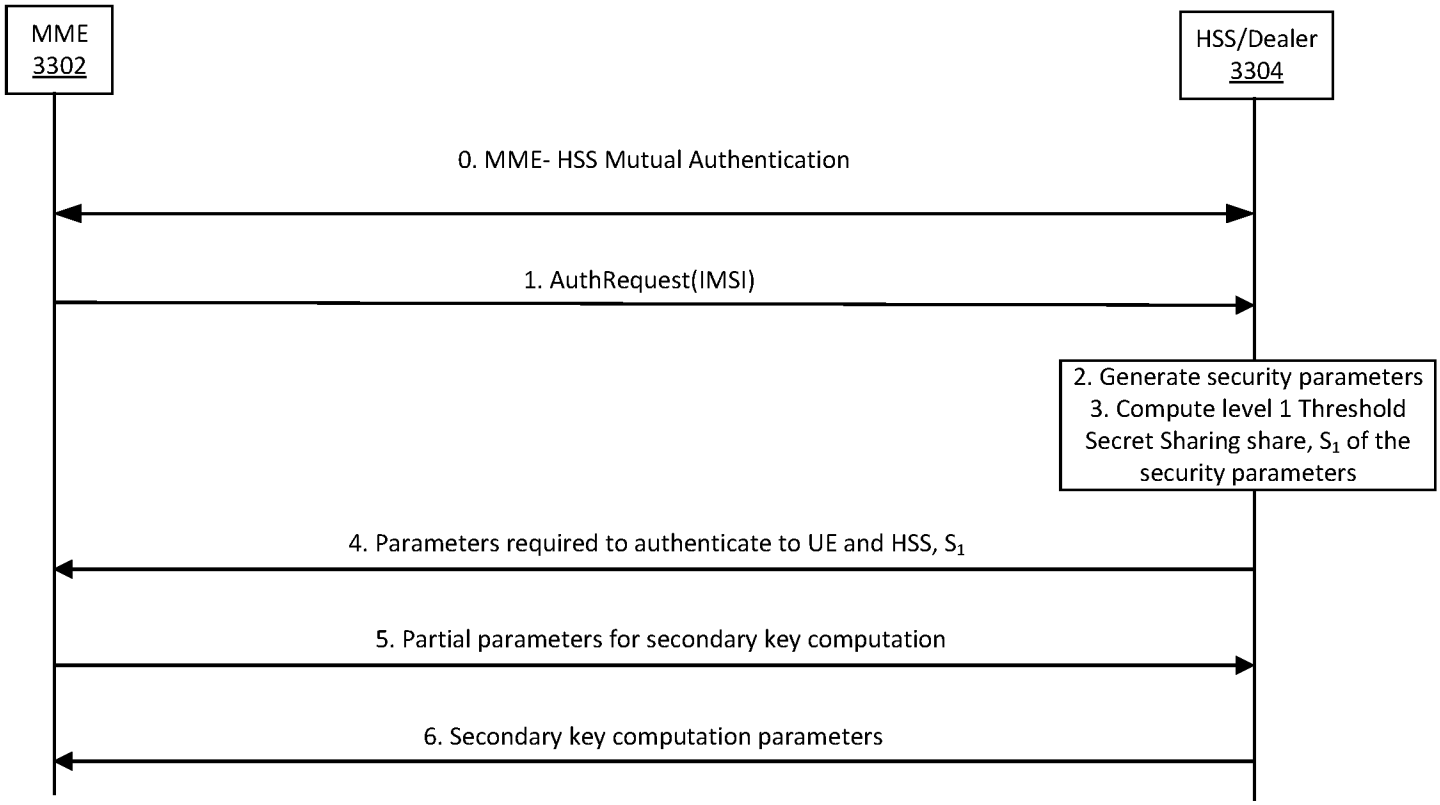

MME
3302
HSS/Dealer
3304
0. MME- HSS Mutual Authentication
1. AuthRequest(IMSI)
2. Generate security parameters
3. Compute level 1 Threshold Secret Sharing share, S_1 of the security parameters
4. Parameters required to authenticate to UE and HSS, S_1
5. Partial parameters for secondary key computation
6. Secondary key computation parameters

FIG. 33

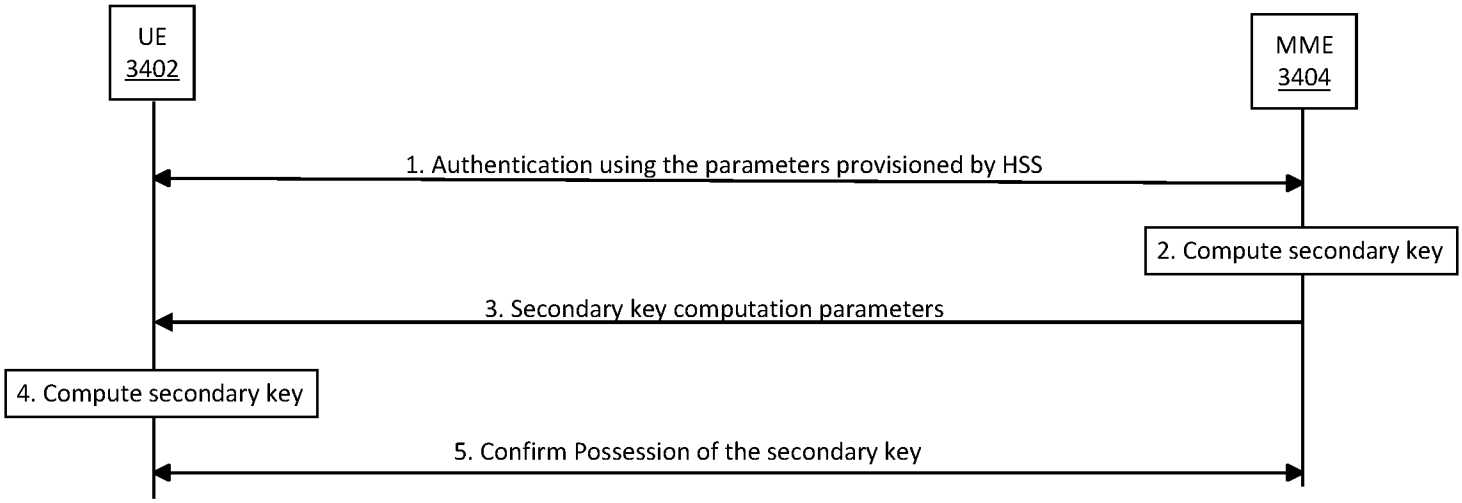

UE
3402
MME
3404
1. Authentication using the parameters provisioned by HSS
2. Compute secondary key
3. Secondary key computation parameters
4. Compute secondary key
5. Confirm Possession of the secondary key

FIG. 34

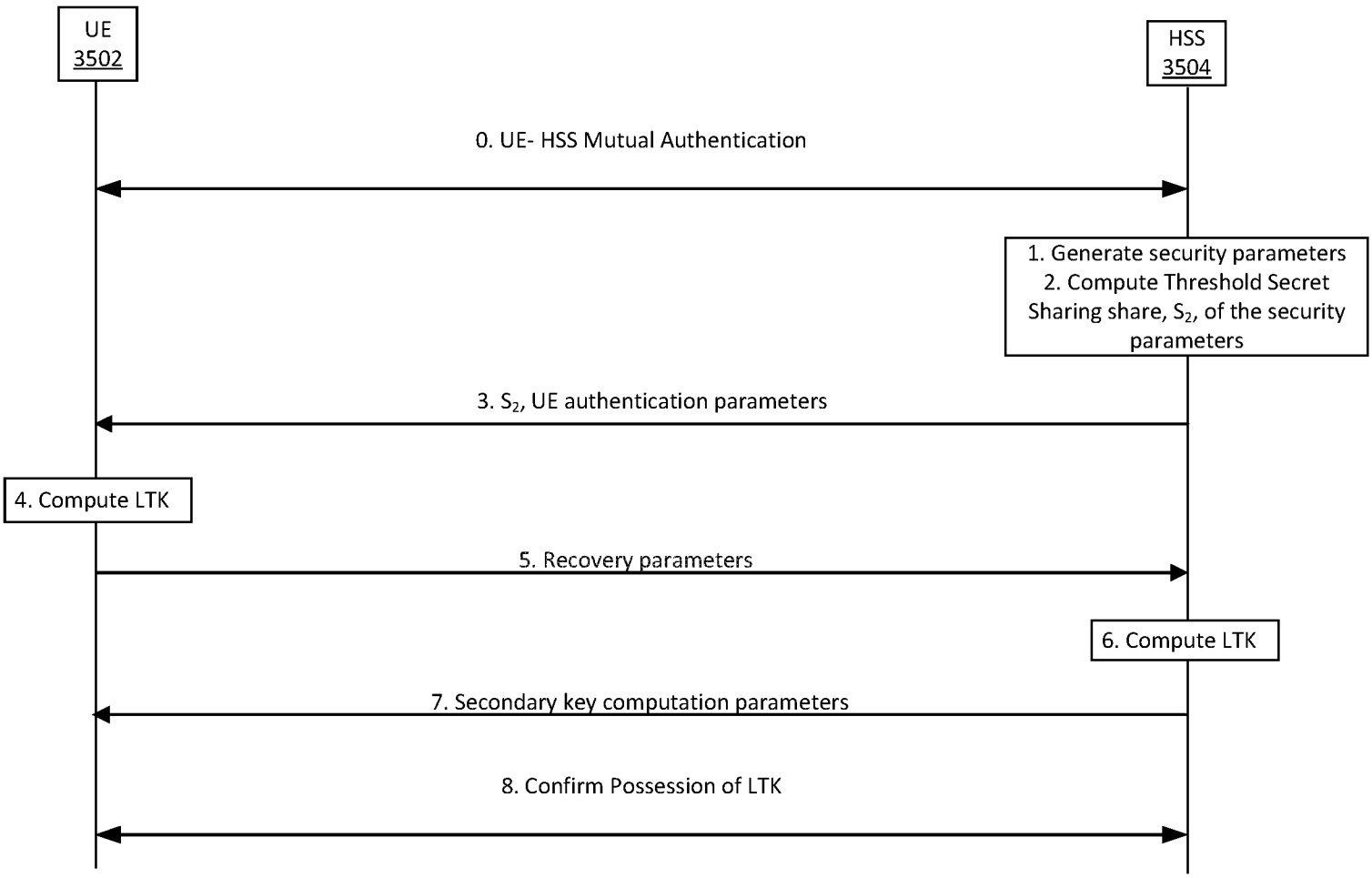

UE
3502
HSS
3504
0. UE- HSS Mutual Authentication
1. Generate security parameters
2. Compute Threshold Secret Sharing share, S_2, of the security parameters
3. S_2, UE authentication parameters
4. Compute LTK
5. Recovery parameters
6. Compute LTK
7. Secondary key computation parameters
8. Confirm Possession of LTK

FIG. 35

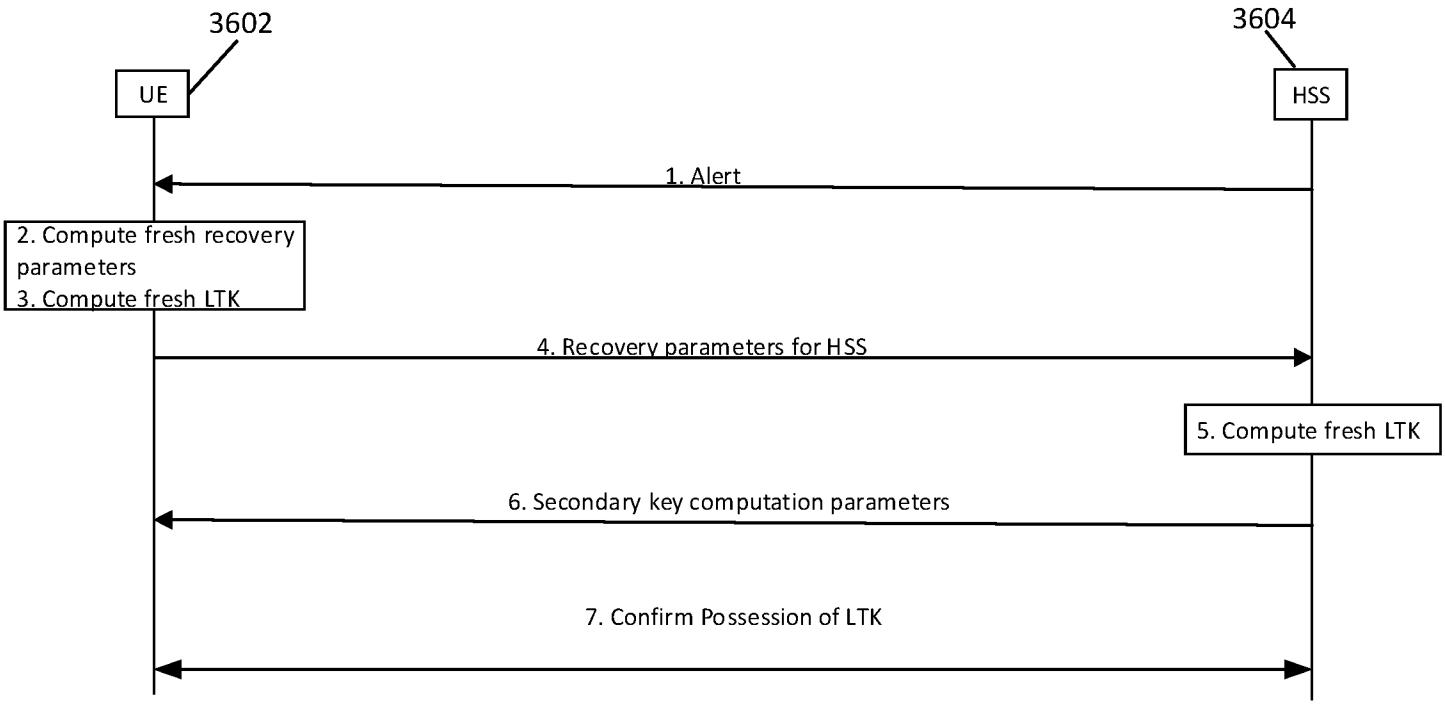

3602
3604
UE
HSS
1. Alert
2. Compute fresh recovery parameters
3. Compute fresh LTK
4. Recovery parameters for HSS
5. Compute fresh LTK
6. Secondary key computation parameters
7. Confirm Possession of LTK

FIG. 36

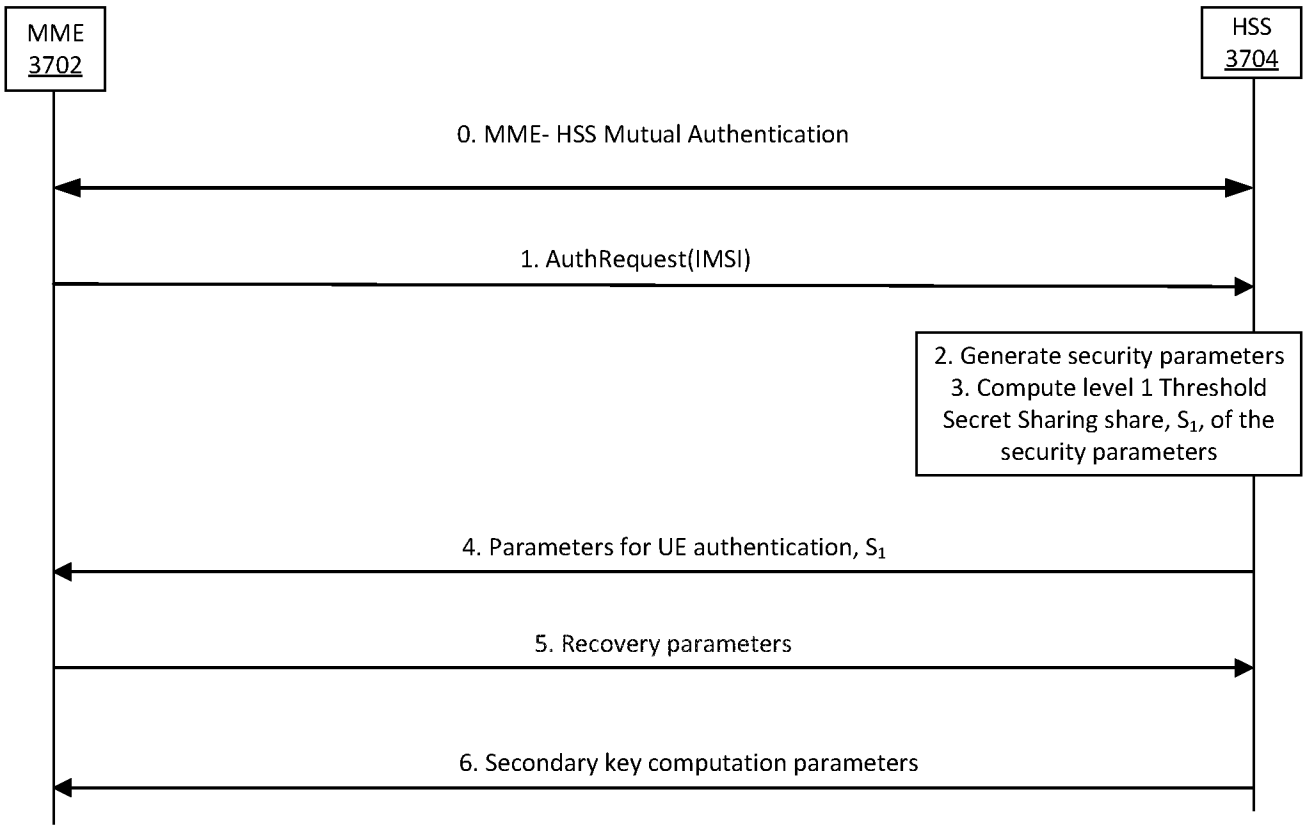

MME
3702
HSS
3704
0. MME- HSS Mutual Authentication
1. AuthRequest(IMSI)
2. Generate security parameters
3. Compute level 1 Threshold Secret Sharing share, S_1, of the security parameters
4. Parameters for UE authentication, S_1
5. Recovery parameters
6. Secondary key computation parameters

FIG. 37

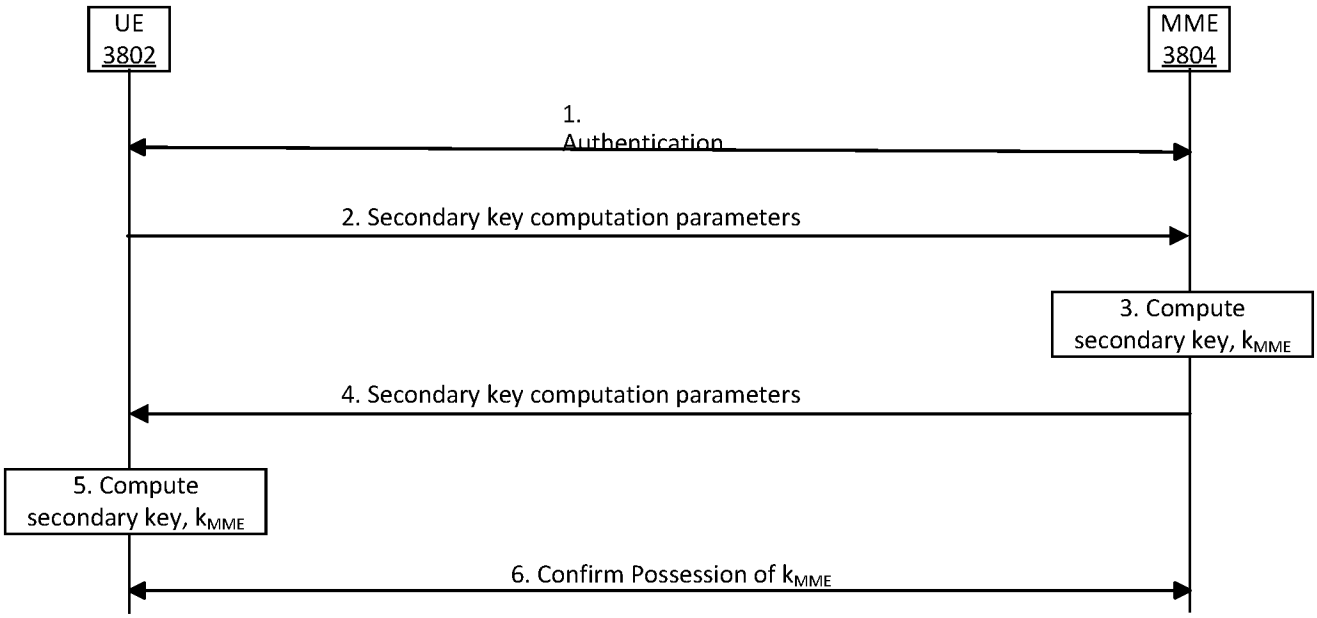

UE
3802
MME
3804
1.
Authentication
2. Secondary key computation parameters
3. Compute
secondary key, k_{MME}
4. Secondary key computation parameters
5. Compute
secondary key, k_{MME}
6. Confirm Possession of k_{MME}

FIG. 38

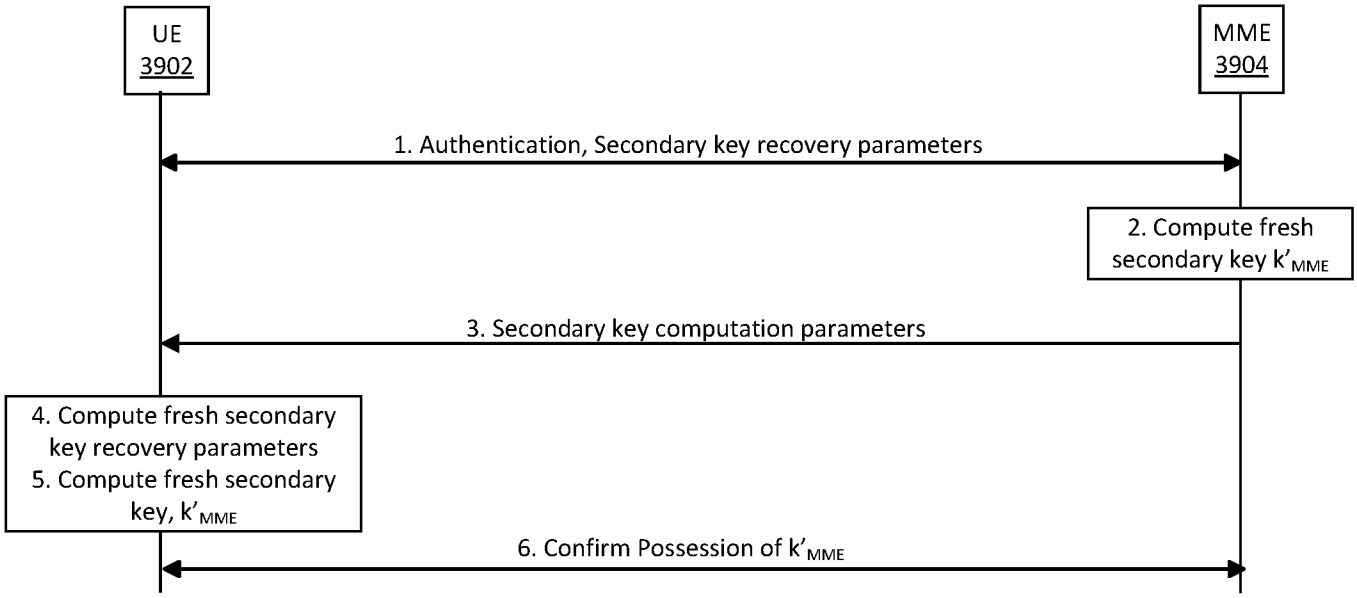

UE
3902
MME
3904
1. Authentication, Secondary key recovery parameters
2. Compute fresh secondary key k'MME
3. Secondary key computation parameters
4. Compute fresh secondary key recovery parameters
5. Compute fresh secondary key, k'MME
6. Confirm Possession of k'MME

FIG. 39

INTERNATIONAL SEARCH REPORT

International application No
PCT/US2017/051788

A. CLASSIFICATION OF SUBJECT MATTER
INV. H04L9/00 H04L9/08 H04L29/06 H04L9/32
ADD.

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

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
H04L

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

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)
EPO-Internal, WPI Data

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	WO 2015/197368 A1 (KONINKL PHILIPS NV [NL]) 30 December 2015 (2015-12-30) page 2, line 16 - page 4, line 19 page 5, line 31 - page 17, line 10 ----- -/--	1-20

[X] Further documents are listed in the continuation of Box C.

[X] See patent family annex.

* Special categories of cited documents :

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

"E" earlier application or patent but published on or after the international filing date

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

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

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

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

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

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

"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
15 November 2017	22/11/2017

Name and mailing address of the ISA/	Authorized officer
European Patent Office, P.B. 5818 Patentlaan 2 NL - 2280 HV Rijswijk Tel. (+31-70) 340-2040, Fax: (+31-70) 340-3016	Spranger, Stephanie

2

Form PCT/ISA/210 (second sheet) (April 2005)

C(Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	FU ANMIN ET AL: "Nframe: A privacy-preserving with non-frameability handover authentication protocol based on (t,n) secret sharing for LTE/LTE-A networks", WIRELESS NETWORKS, ACM, 2 PENN PLAZA, SUITE 701 - NEW YORK USA, vol. 23, no. 7, 29 April 2016 (2016-04-29) , pages 2165-2176, XP036321725, ISSN: 1022-0038, DOI: 10.1007/S11276-016-1277-0 [retrieved on 2016-04-29] abstract page 2176, right-hand column, line 37 - page 2171, right-hand column, line 8; figure 2	1-20
A	TAMIR TASSA ET AL: "Multipartite Secret Sharing by Bivariate Interpolation", JOURNAL OF CRYPTOLOGY, SPRINGER-VERLAG, NE, vol. 22, no. 2, 30 July 2008 (2008-07-30), pages 227-258, XP019699800, ISSN: 1432-1378 page 3, line 32 - page 6, line 31 page 17, line 1 - page 39, line 17	1-20

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No
PCT/US2017/051788

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
WO 2015197368 A1	30-12-2015	CN 106464490 A	22-02-2017
		EP 3161993 A1	03-05-2017
		JP 2017519457 A	13-07-2017
		US 2017155510 A1	01-06-2017
		WO 2015197368 A1	30-12-2015