Noname manuscript No.
(will be inserted by the editor)

Introducing a Novel Authentication Protocol for Secure Services in Heterogeneous Environments using Casper/FDR

Mahdi Aiash · Jonathan Loo

Received: date / Accepted: date

Abstract Next Generation Networks is a convergence of networks such as 2G/3G, WLAN as well as the recently implemented Long Term Evolution (LTE) networks. Future mobile devices will switch between these different networks to maintain the connectivity with end servers. However, to support these heterogeneous environments, there is a need to consider a new design of the network infrastructure, where currently closed systems such as 3G will have to operate in an open environment. Security is a key issue in this open environment; after authenticating the mobile terminal to access the network, there is a requirement for service-level mechanisms to protect the session between the mobile terminal and the remote service provider. Furthermore, since mobile terminals switch between networks of different characteristics in terms of coverage, Quality of Service and security, there is a need for re-assessing the security of the same session over the different networks to comply with the changes at the network level due to the mobility. Therefore, this paper introduces a Service-Level Authentication and Key Agreement protocol to secure the session between the mobile terminal and the end server. The proposed protocol considers user mobilities in an heterogeneous environment and reassesses the session's security level in case of handover. The proposed protocol has been verified using formal methods approach based on the well-established Casper/FDR compilers.

Keywords Authentication And Key Agreement Protocols · Heterogeneous Networks · Casper/FDR

M. Aiash, J. Loo
School of Science and Technology
Middlesex University
London, UK
E-mail: M.Aiash, J.Loo@mdx.ac.uk

1 Introduction

We are currently experiencing huge development and large-scale deployment of several wireless technologies; from next generation cellular networks to personal/home networks such as WLANs and metropolitan ones such as WiMax and LTE. Since the peripheral networks are mainly wireless, they will be of an entirely different nature to the network in the core network and thus will have hugely different characteristics in terms of latency, bandwidth and error rate.

Therefore, it will not be possible to think of the future Internet as a single unified structure; future Internet could be viewed as comprising of a fast core network with slower peripheral networks attached around the core. The core network will consist of a super-fast backbone using optical switches and fast access networks which uses Multiprotocol Label Switching (MPLS), and it will be an open architecture not under the control of a specific mobile operator rather various network operators will coexist in the core network and provide their services. This view emphasises the heterogeneous and open nature of future Next Generation Networks (NGNs), where users will expect to switch between different access networks using handover techniques while maintaining the connectivity to various application Service Providers (SPs) that provides a wide variety of services such as e-Commerce, on-line banking and electronic public services in addition to video/news on-demand, Grid and Cloud resources/services. This situation is shown in Fig 1.

This new open architecture, will bring about new security threats in terms of authenticating and authorizing the mobile terminal to access the network and contact the application Service Provider (SP). Therefore, there is a need to secure the transactions whether at

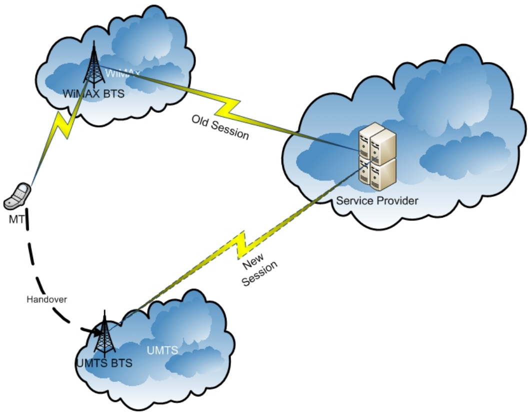


Fig. 1 Ubiquitous Connectivity via Vertical Handover

the network or at the service or application levels. While the first has been addressed by different research efforts such as [1] [2] [3] [4]. Furthermore, in a recent work of our research group two Authentication and Key Agreement (AKA) protocols have been proposed to provide security at the network level [5]. However, few research efforts such as [6] [27] [8] have been dealing with the security at the service level, considering the open nature of the future networks.

This paper introduces a novel Service-Level AKA (SL-AKA) protocol that considers the open architecture of future networks [35] and achieves mutual authentication between the MT and the SP in the initial stage when the MT contacts the SP for the first time as well as in the case of handover, when the MT changes its point of attachment while maintains the connectivity to the SP. The proposed protocol is verified using formal methods such as Communication Sequential Processes (CSP) [11], which is a formal language to describe the interaction and states in concurrent systems, it has been used to model communicating and security protocols as in [12] and [13]. To verify the CSP models, model checkers such as the Failure Divergence Refinement (FDR) [14] is used. Although modelling and verifying security protocols using CSP and FDR have proven to be effective and widely deployed, modelling directly in CSP is a time-consuming and error-prone. Therefore, a new compiler for generating the CSP description of the protocol was designed by Lowe in [15]. The new complier is called Casper and it accepts an abstract description of a system and translates it into CSP. This paper will model the security properties of the proposed protocols using Casper and analyse the CSP output with FDR.

The rest of this paper is organized as follows: Section 2 describes a potential structure of future, heterogeneous networks as introduced by different research frameworks such as Daidalos II and Y-Comm research groups [16] [17]. Section 3 defines the challenges of providing the service-level security in the heterogeneous environment. Related works to address the security at the service-level are discussed in Section 4. While Section 5 describes the approach followed in this paper to analyse the proposed security protocols, Section 6 presents the proposed SL-AKA protocols for the initial authentication as well as authentication in case of the handover. The paper concludes in Section 7.

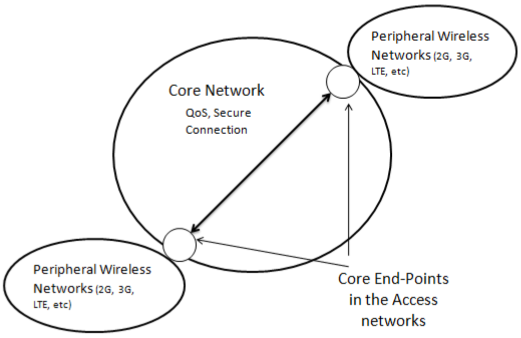

Core Network
QoS, Secure Connection
Peripheral Wireless Networks (2G, 3G, LTE, etc)
Peripheral Wireless Networks (2G, 3G, LTE, etc)
Core End-Points in the Access networks

Fig. 2 The Structure of Future Internet

2 Overview of Future Network Structure

Current systems such as 2nd and 3rd generations are considered as closed ones due to the fact that the core network is owned by a sole operator, who is responsible for managing all aspects of communication including security and QoS provision. However, as explained in section 1, the heterogeneity of future networks leads to a new open architecture of the core network, where the infrastructure is not controlled by a single operator rather multiple operators coexist in the core network. To deal with the interoperability issue between the different operators, the Y-Comm group and the Daidalos II [17] [16] adopted and enhanced the concept of a central management entity, as proposed in [18], to control the different operators. And hence, the concept of the Core End-Point (CEP) as a central administrative domain that controls the operation of different network operators was introduced.

As shown in Fig 2, the future Internet could be viewed as composed of several Core End-Points, interconnected over the super-fast backbone of the Internet. Each CEP is responsible for managing multiple, wireless peripheral networks such as Wimax, WiFi or cellular technologies.

A detailed view of the network along with its components are explained in [17] and shown in Fig 3. It is a hierarchical structure of the network composed of three levels. The top level is the Core End-Point (CEP) which acts as a gateway to the Internet and is responsible for managing multiple, mid-level domains. Each domain is technology-specific and is controlled by a single operator. For instance the CEP might be connected to two domains; each is controlled by different technology operator such as WiMAX and GSM. The bottom level is the peripheral wireless networks, represented by multiple Access Routers (ARs), which make the interface between the network and the mobile terminal (MT). The communication between the CEPs takes place over the backbone of the Internet where architectures like the Intermon [19], which is a research framework to facilitate Inter-domain QoS monitoring and analysis for validation, planning and optimisation of inter-domain QoS, could be used to manage the communication among CEPs. However, the research in this paper is not concerned with discussing inter-CEPs communication framework.

In order to deal with the QoS and security tasks in this architecture, a number of operational entities have been proposed as follows:

- **The Central A3C server (CA3C):** This is the central Authentication, Authorization; Accounting and Cost (A3C) server in the Core End-Point. The CA3C holds the Service Level of Agreements (SLAs) along with the Network Level of Agreements (NLAs), which describe the clients' term of use of the service and access networks, respectively.
- **The Central QoS Broker (CQoSB):** is responsible for negotiating QoS in case of cross-CEP handover.
- **The Domain A3C Server (DA3C):** The DA3C is responsible for handling users' service aspects. Initially, it extracts users' profile information from the CA3C and uses this information for authorizing the users' requests to access services.
- **The Domain QoS Broker (DQoSB):** manages the resources of the attached peripheral networks with respect user preferences and network availability, it also makes a per-flow admission control decision.
- **The Access Router (AR):** This is the link between the domain and the peripheral networks; it enforces the admission control decision, taken by the DQoSB. Since the AR acts as a relay between the Mobile Terminal (MT) in the peripheral network and the DA3C, using security terminology, the AR will be referred to as the Authenticator (Auth).

The proposed network architecture supports two different business models as follows:

1. The Cellular Model: In this model, a single network operator manages both a core network and wireless access, such as the 3G, WLAN and 4G.

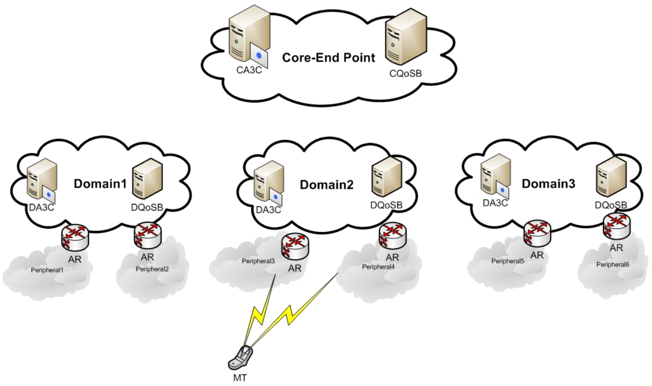


Fig. 3 The Future Network Structure

Table 1 Supporting LTE Networks using the discussed Hierarchical Architecture

The LTE Network Element	The Corresponding Network Entity
The Home Subscriber Server (HSS)	The Central A3C Server (CA3C) and the Central QoS Broker (CQoSB)
The Policy Control and Charging Rules Function (PCRF)	The High-level Access Admission Decision module (HAAD) of the CQoSB
The Policy Control Enforcement Function (PCEF)	The Access Admission Decision (AAD) and the Centralized Network Monitoring Entity (CNME) modules of the DQoSB
The PDN Gateway (P-GW)	The Domain QoS Broker (DQoSB) and the Domain A3C (DA3C) server
The Serving Gateway (S-GW)	The Access Admission Enforcement (AAE) and Network Monitor Entity (NME) modules of the Access Router
The eNB and the MME	The Access Router

2. The Collaboration Model: Here, the core network and each wireless access are managed by different operators.

Furthermore, in a recent work of our group in [9], we discussed how this hierarchical architecture could be used to support new technologies such as LTE and Wimax. By Considering the architecture of LTE networks as detailed in [10], Table shows the mapping between the two architectures.

3 Problem Definition

The aim of ubiquitous computing in heterogeneous environments such as the one described in the previous section is to provide mobile users anytime, anywhere

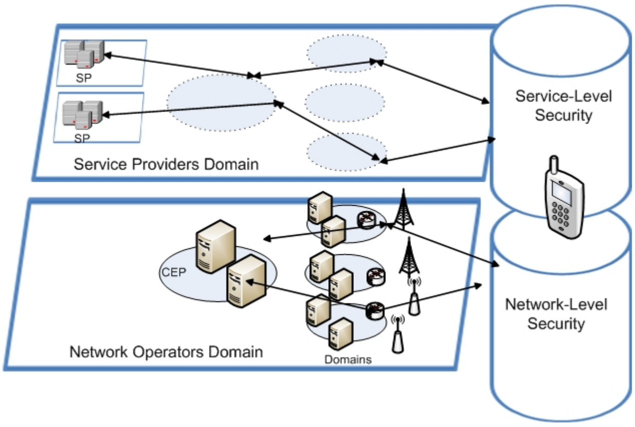


Fig. 4 The Two-Level Security

and any platform access to a wide variety of computing servers. While much research has been performed to provide the infrastructure and mechanisms support for this goal at the network such as Mobile IPv6, Fast Mobile IPv6 and IEEE802.21 [20] [21] [3], few research efforts such as the [6] have considered the need for application supports for connectivity. As shown in Fig 4, from a security perspective, this situation highlights the need for addressing the security at two levels; the Network-Level between the MT and the network operators, and the Service-level between the MT and the application service providers.

Furthermore, the issue of Application Service-Level security has in general been difficult to address in future networks. This is due to many reasons. Firstly, any proposed security protocol has to consider the structure of current mobile devices as well as their limitations in terms of battery and processing power. These conditions put extra restrictions when deciding on security measurements such as encryption algorithms (Symmetric or Asymmetric) as well as keys management.

Secondly, current security mechanisms consider the closed nature of current communication systems such as 2nd and 3rd generation networks. According to this nature, the resources of core network are controlled and managed by a sole operator and thus, the core network will be physically secure. Unlike the closed architecture of current systems, the future network represents an open, heterogeneous environment where multiple network operators and application service providers exist in the core network. These differences highlight the need for enhancing current security mechanism if not introducing new ones that consider the open architecture of the future networks.

Thirdly, when a client subscribes to services, parameters such as the desired QoS and security parameters will be defined as part of the Service Level of Agreement (SLA). However, since the service provider might have different preferences in terms of the security and QoS, the two end-points might need to provide a range of preferences where they could negotiate the required level of security. This highlights the need for a negotiation stage to specify the connection parameters before setting up the connection.

Fourthly, in heterogeneous networks, future mobile devices are expected to switch between various access networks while maintain connected to the service provider. Based on their security characteristics, a server providers might choose to trust some networks more than others and hence apply different security measurements. This highlights the need for the server providers to know about the access network of the mobile terminal in order to re-assess the connection security and to decide on the required security parameters.

However, in the case of handover when the mobile terminal moves into a new network with different characteristics in terms of security and QoS, vertical handover will therefore have an impact on the network service experienced by ongoing applications and services as mobile terminals move around. This implies that in case of handover there is a need for re-negotiating the connection parameters to comply with the characteristics of the new access network. This highlights the need for a lightweight Authentication and Key Agreement (AKA) protocol for handover so the functionality of the this protocol will not disrupt the connection with the server.

4 Related Work

The literature is very rich with AKA protocols that operate at the network-level and provide mutual authentication between the mobile device and the access network such as in [36] [37] [38] [39]. However, fewer protocols have been introduced to address security at the service level in heterogeneous environment. This section describes some potential mechanisms to address the Service-Level security in future networks.

4.1 Transport Layer Security (TLS) and Secure Sockets Layer (SSL)

Transport Layer Security (TLS) and its predecessor, Secure Sockets Layer (SSL), are cryptographic protocols that provide secure communication between two end points over the Internet. SSL is divided into two layers, with each layer using services provided by a lower layer and providing functionality to higher layers. The SSL record layer provides confidentiality, authenticity, and re-play protection over a connection-oriented reliable transport protocol such as TCP. Layered above the record layer is the SSL handshake protocol, a key-exchange protocol which initializes and synchronizes cryptographic state at the two endpoints. After the key-exchange protocol completes, sensitive application data can be sent via the SSL record layer [22]. In this sense, SSL/TLS enables the end-points to negotiate and agree on security parameters such as the encryption and hashing algorithms.

Using public-key encryption techniques, SSL-enabled client and server will authenticate each other and establish an encrypted connection. Although SSL/TLS achieve many desired security properties, and as a result have been widely implemented, there are many issues when it comes to implementing them in future networks; firstly, they are PKI-based protocols and the fact that setting- up a PKI is a complex and costly process that consists of several steps: registration of users, generation of keys, issuance and distribution of certificates. Additionally, PKI involves other complex processes such as certificate retrieval and certification path construction and validation makes it unsuitable for normal mobile terminals [23]. Furthermore, the complexity of the PKI operation will add extra burden on the authentication process in the case of handover. Secondly, The SSL/TLS run above the Transport layer which make them unaware of the characteristics of the underlying access networks. And thus cannot reflect these characteristics in the negotiation stage of the protocol. Thirdly, these protocols do not introduce a lightweight extension for re-authentication in case of handover.

4.2 The Stream Control Transmission Protocol (SCTP)

The Stream Control Transmission Protocol (SCTP) [24] is a connection-oriented transport protocol that operates on the top of the IP protocol. The SCTP has several advantages over the traditional transport protocol such as Transmission Control Protocol (TCP) [25] and User Datagram Protocol (UDP) [26], examples of these advantages are multi-streaming and multi-homing support. Additionally, the Secure SCTP (SSCTP) [6] [27] was designed with security features to set a secure association between the two end-points and thus addresses attacks such replay and SYN flooding. The SSCTP protocol enables the two end-points to negotiate the security parameters and thus agree on the desired algorithms.

However, the security approach proposed of the Secure SCTP is highly dependent on the SCTP protocol as the underlying transport protocol and consequently, it cannot be used with other transport protocols such as the widely implemented TCP and the UDP. Although the SCTP protocol supports client mobility [28], there is neither a clear indication of the impact of this mobility on the security mechanisms nor a lightweight re-authentication protocol in case of handover.

4.3 The Service-Level AKA Protocol of the Mobile Ethernet framework

The Mobile Ethernet framework [8] [29] is an architecture for IP-based, future networks. In order to address the security between the mobile terminal and the service provider,an SL-AKA protocol was introduced in [8].

Although the Mobile Ethernet framework, and thus its security protocols, adopts a network structure that is very similar to our view of future networks in section 2, and despite the fact that the SL-AKA protocol achieves a set of desired security features such as mutual authentication and connection confidentiality, it suffers from some major drawbacks. These are as follows: firstly, the SL-AKA protocol does not have a negotiation stage; thus, it neither considers the variation of QoS and security requirements of the access networks and the service provider nor the client preference. Secondly, it does not consider the case of handover and thus no SL-AKA protocol for handover has been proposed.

5 Verifying Security Protocols

5.1 Verifying Security Protocols Using Formal Methods and Casper/FDR Tool

Analysing security protocol using formal methods goes through two stages. Firstly, modelling the protocol using a theoretical notation or language such as Communication Sequential Processes (CSP) [11]. Secondly, verifying the protocol using a model checker such as Failures-Divergence Refinement (FDR) [14].

However, describing a system or a protocol using CSP is a quite difficult and error-prone task; therefore, Gavin Lowe [15] has developed CASPER tool to model security protocols, it accepts a simple and human-friendly input file that describes the system and compiles it into CSP code which is then checked using the FDR model checker. CASPER's input file consists of eight headers as explained in Table 2. Examples of the assertions that could be checked in the #Specifications header are explained when discussing the formal verification of the proposed protocol in section 6.1.2.

Table 2 Casper's Input File

The Header	Description
#Free Variables	Defines the agents, variables and functions in the protocol
#Processes	Represents each agent as a process
#Protocol Description	Shows all the messages exchanged between the agents
#Specification	Specifies the security properties to be checked
#Actual Variables	Defines the real variables, in the actual system to be checked
#Functions	Defines all the functions used in the protocol
#System	Lists the agents participating in the actual system with their parameters instantiated
#Intruder Information	Specifies the intruder's knowledge and capabilities

Model checking is a fully automated analysis of properties of a finite state system called here the model. In the case of CSP, the property as well as the communicating system are expressed as CSP processes. If a property checks positively for a model this means that the behaviour of the specified system does respect the property. This check is equivalent to a mathematical proof relative to the model since all possible traces of the system are checked by the model checking procedure. Therefore it is superior to testing. Nevertheless, the process depends a lot on the right level of abstraction since the complexity of the checking is exponential in the number of the state components of the model. On the other side, in particular with security protocol these abstraction have to be carefully checked at specification time (by the specifier/modeller) since they implicitly carry all the guarantees about keys, intruder knowledge etc. A great advantage of Model checking as compared to other automated verification techniques like Automated Theorem Proving is the possibility to generate counterexamples. If a model (for example of a protocol) is faulty (or allows an attack) this will show when trying to verify a corresponding property. The model checking process is such that it provides a path through the finite state graph representing the model that leads from an initial state to a state where the property in question is violated. For a protocol, for example, this path represents a possible attack. This is how attacks can be discovered on protocols. In general, it make model checking a tool suitable for an iterated process using a feedback loop of stepwise development as it is common in engineering disciplines.

5.2 Analysing the Security Protocols

To verify SL-AKA protocols, we use a form of formal methods approach based on Casper/FDR tool [15]. The Casper tool accepts an abstract, human-friendly description of the system and compiles it into Communication Sequential Processes (CSP) code, suitable for the Failures-Divergence Refinement (FDR) [14] checker.

Furthermore, as stated in [30], it is desired for AKA protocols to meet certain security properties. Therefore, a list of these properties will be used to analyse the security features of all the proposed AKA protocols. The properties are as follows:

1. *Mutual Entity Authentication*: This is achieved when each party is assured of the identity of the other party.
2. *Mutual Key Authentication*: This is achieved when each party is assured that no other party aside from a specifically identified second party gains access to a particular secret key.
3. *Mutual Key Confirmation*: This requirement means that each party should be ensured that the other has possession of a particular secret key.
4. *Key Freshness*: a key is considered fresh if it can be guaranteed to be new and not reused through actions of either an adversary or authorized party.
5. *Unknown-Key Share resilience*: In this attack the two parties compute the same session key but have different views of their peers in the key exchange. In other words, in this attack an entity A ends up believing she shares a key with B, although this is the case, B mistakenly believes the key is instead shared with an entity $E \neq A$.
6. *Key Compromise Impersonation Resilience*: This property implies that if the Intruder compromised the long-term key of one party, he should not be able to masquerade to the party as a different party.

6 The Proposed SL-AKA Protocol and Key Hierarchy

By considering the network structure in section 2, upon signing the initial contract, the user's profile information including a Unique Key (UK), the Service Level of Agreement (SLA) along with the subscribed services is shared between the Centralized A3C (CA3C) in the administrative domain and the subscriber. Similar to the AKA protocols in GSM, UMTS and LTE [31] [32] [33] where a secret key is burnt onto the SIM card of the mobile device and is shared with the Authentication Server, our proposed protocol presumes the existence of such a secret key which will be referred to as the

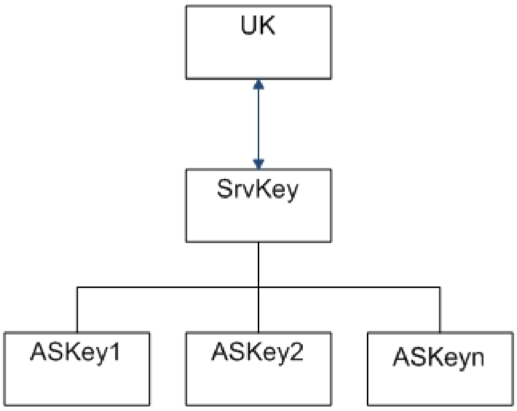


Fig. 5 The Key Hierarchy

Unique Key (UK). This UK will be burnt onto the SIM card of the mobile device and will only be used to derive further keys, in other words it will not be used to encrypt/decrypt any messages, which means that it will not be exposed to intruders. Furthermore, as has initially been highlighted in [32] and then analysed in [34] and [5], it is crucial to secure the communication in the core network to provide security in an open and heterogeneous environment like the one described in section 2. Therefore, secure connections using VPN or IPSec protocols have to be set between the operational network entities in the core network to protect their transactions. Our proposed SL-AKA protocol will presumes the presence of such security mechanisms when considering the communications in the core network.

As shown in Fig 5, a service-specific secret key (Srvkey) is derived by the Mobile Terminal and the CA3C which will securely pass it to the intended application Service Provider (SP). The Srvkey is derived using the UK, service ID (SrvID); the user's subscription ID (SubID) and a lifetime value as follow: Srvkey= F1(UK, SrvID, SubID, lifetime). Using the Srvkey and other information, an Association Key ASKey is derived by the MT and the SP to protect the session between them. When MTs change the access network due to handover, a new ASKey is derived to reflect the security level of the new network as will be explained in section 6.1.1.

6.1 The Initial SL-AKA Protocol

This protocol runs when the Mobile Terminal (MT) initially expresses its intention to contact the Service Provider (SP) to achieve mutual authentication and set up a secure channel between the MT and the SP.

Considering the network structure in section 2 and as explained in [5], the information about the subscribed services and the client's preference of security along with the characteristics of its access network is kept by the Central A3C (CA3C) in the Core-End Point (CEP). Also, as described in section 6, for each subscribed service, the CA3C will derive a service key Srvkey= F1(UK, SrvID, SubID, lifetime) and passes it to the SP. However, sharing the SrvKey between the MT and the SP is not part of the SL-AKA protocol. Therefore, the SL-AKA protocol considers the SrvKey to be pre-shared between the SP and the MT. This key will be used to derive the Association Key (ASKey) to secure the connection between the SP and the MT. By considering the notations in Table 3, the SL-AKA runs as follows:

As shown in Figure 6, the SL-AKA is initiated when the Mobile Terminal (MT) indicates to the CA3C its in-

Table 3 Notation

Abbreviation	Full name and description
MT	Mobile Terminal
SP	Application Service providers residing in the core end point
DesDA3C	The Domain AAAC server of the destination (SP) domain
CA3C	Central AAAC server stores the MT's SLA, which contains the MT's preferred QoS and Security parameters as well as the a list of all the SPs.
Srvkey(SP)	Service key: a pre-shared key between the MT and the SP: Srvkey= F1(UK, SrvID, SubID, lifetime).
r1,r2	Random nonce
HMACList1, HMACList2	Lists of supported hashing algorithms.
EncList1, EncList2	Lists of encryption algorithms.
SrvID	Service ID, which uniquely identifies the service.
SubID	User subscription ID, uniquely identify the subscriber to the SP.
ADname	Access Domain name, defining the domain name of the access network.
SrvCookies	The Cookies, sent by the sever to the MT, these cookies limit replay and DoS attacks.
Vector1	r1,HMACList1,EncList1.
Vector2	r2,HMACList2,EncList2.
ASKey	Association key ASKey = F2 (Srvkey, Vector1, Vector2).
Ackm	Authentication Token Ackm = F (SubID, SrvID, timestamp) used as an acknowledgement messages to indicate the completion of the AKA process.

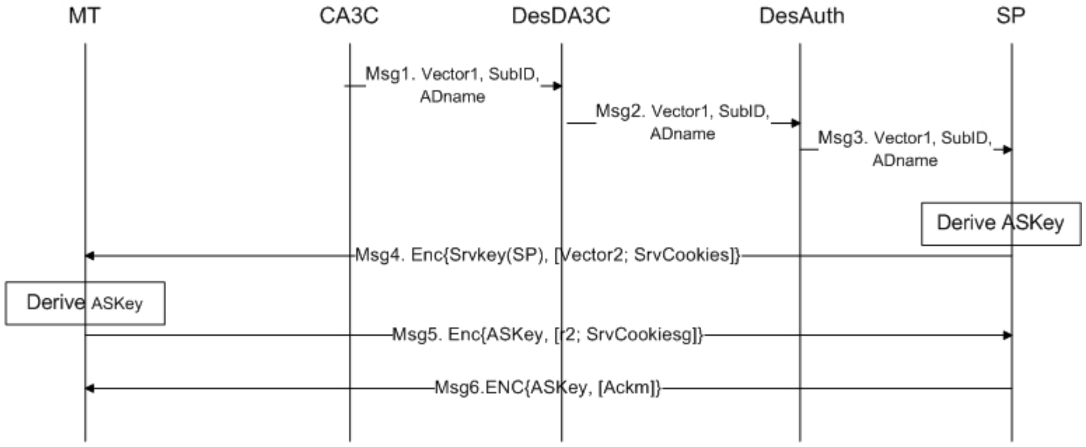

MT
CA3C
DesDA3C
DesAuth
SP
Msg1. Vector1, SubID, ADname
Msg2. Vector1, SubID, ADname
Msg3. Vector1, SubID, ADname
Derive ASKey
Msg4. Enc{Srvkey(SP), [Vector2; SrvCookies]}
Derive ASKey
Msg5. Enc{ASKey, [r2; SrvCookiesg]}
Msg6.ENC{ASKey, [Ackm]}

Fig. 6 The Initial SL-AKA Protocol

tention to access the service provider (SP). The CA3C server knows the services' subscription IDs as well as the corresponding MT's preferences in terms of security and QoS which are part of the Service Level of Agreement (SLA) stored in the CA3C server. The CA3C server passes the MT's preference as a vector of information (Vector1), which contains lists of MT's preferred encryption and hashing algorithms (Enclist) and (HMACList) respectively. Additionally, it contains a fresh random value (r1) to maintain the vector freshness.

This vector along with the domain name of the MT's access network and the MT's SubID are passed to the Service Provider (SP) as messages 1,2 and 3. Once, the SP receives message 3, it derives the Association Key ASKey = F2 (Srvkey(SP), Vector1, Vector2).

$Msg1 : CA3C \rightarrow DesDA3C : Vector1, SubID,$
$ADname$
$Msg2 : DesDA3C \rightarrow DesAuth : Vector1, SubID,$
$ADname$
$Msg3 : DesAuth \rightarrow SP : Vector1, SubID, ADname$
Generate the ASKey= F2 (Srvkey(SP), Vector1, Vector2)

Based on the MT's preference in Vector1, the SP constructs Vector2 which represents the SP preferences in terms of encryption and hashing algorithms (EncList2), (HMACList2) -this negotiation stage will be discussed later in section 6.1.1. In message 4, the SP sends Vector2 and server cookies (SrvCookies) to the MT. These cookies will be used as a challenge and to stop re-play attacks.

$Msg4 : SP \rightarrow MT : \{Vector2, SrvCookies\}_{Srvkey(SP)}$
Generate the ASKey= F2 (Srvkey(SP), Vector1, Vector2)

Message 4 is encrypted using the pre-shared service key SrvKey(SP) between the MT and the SP. Therefore, the MT will decrypt the message to get Vector2 and derive the ASKey. The MT retrieves the nonce number (r2) from the received Vector2, uses the derived ASKey to encrypt message 5, which includes the server's cookies and r2. Upon receiving message 5, the SP verifies the message's contents to ensure that it contains the valid values for the SrvCookies and r2. In case of a successful verification, the SP acknowledges the successful authentication by composing the acknowledgement message (Msg6)

$Msg5 : MT \rightarrow SP : \{r2, SrvCookies\}_{ASKey}$
Verify the message contents
$Msg6 : SP \rightarrow MT : \{Ackm\}_{ASKey}$

6.1.1 The Negotiation Stage

At this stage, we presumed that, each time the user subscribes to a new service, an identity-based authentication token is generated by the MT and the SP. This token is used in the protocol as an acknowledgement Ackm=F(SubID, SrvID, timestamp) to indicate authentication completion and for achieving identity authentication as explained in the SL-AKA Analysis section 6.1.4. In Msg1, the CA3C provides the SP with a list of the supported hashing and encryption algorithms by the MT; it also contains the domain name of the MT's access network. The reason for including the network domain name is to allow the SP to specify its security level with regards to the credibility of the MT's access network.

Three major factors define network's credibility: the network security level in terms of the efficiency of the authentication and encryption mechanisms, geographical location of the MT's access network; some services might choose not to accept access requests from certain countries or domains, which are considered insecure. The third factor is the access network technology (WiFi, 2G, 3G. etc), this factor has to do with making the SP aware of the access network's characteristics in terms of QoS range and coverage which are important to consider in case of handover. Taking these factors into account, the SP specifies three modes of access networks: low, normal and high credibility networks and as a result the SP re-orders its own hashing and encryption lists (HMACList2, EncList2) and sends them to the MT as part of the Vector2.

This way, in addition to its lists, each end has the other end's lists. In the case of HMAC lists for instance, each end takes the first suggested algorithms in the SP's list (HMACList2) and looks it up in the MT's list (HMACList1), if no match found, it takes the second suggested algorithm in list2 and looks it up in list1, then the third and so on. The first match is considered as the adopted hashing algorithm. The same procedure is followed for choosing the session encryption algorithm.

6.1.2 The SL-AKA Formal Verification

The goal of the proposed SL-AKA protocol is to achieve mutual authentication and set a secure connection between the MT and SP. To model the AKA protocol using Casper/FDR tool, we prepared a Casper input file that represents the system. The complete description of the protocol is found in Appendix 8, for conciseness only the # Processes, the # Specification and the # Intruder Information headings are shown here, while the rest are of a less significance in terms of verifying the protocol.

The # Protocol Description section describes the protocol as a sequence of the messages exchanged between the participants. The notation {m}{k} implies that the message (m) is encrypted using the key (k). Also, m%w denotes that the recipient of the message is not supposed to understand the message (m) instead; he should store it in a variable (w) and pass it. In contrast, the notation w%m means that recipient should be able to encrypt the message (m), stored in the variable (w).

The # Processes heading shows the process in the system, where each participant is represented by a single process. Our system comprises five processes: The MT represented by the INITIATOR process, the Destination Authenticator (DesAuthenticator) process corresponds to the DesAuth; the DesAAASERVER process represents the Destination AAA server; the last two processes namely, the CentralSERVER and Responder represent the CA3C in the core end point and the SP. For each process, the parameters- in the brackets- define the agents' initial knowledge before running protocol.

# Processes

```
INITIATOR(MT, Ackm, r1, Vector1, SubID, ADname)
knows Srvkey(SP)
DesAuthenticator(DesAuth, SP, DesDA3C, AuthReq,
Adv, AccRes)
DesAAASERVER(DesDA3C,CA3C,DesAuth)
CentralSERVER( CA3C, DesDA3C, Vector1, SubID,
ADname)
RESPONDER(SP, MT, DesAuth, DesDA3C, r2, Vector2,
SrvCookies, Ackm) knows Srvkey(SP)
```

The security requirements of the system are defined under the # Specification heading. The lines starting with the keyword Secret define the secrecy properties of the protocol. The `Secret(SP,ASKey,[MT])` specifies the ASKey as a secret between the MT and the SP. The lines starting with Agreement define the protocol's authenticity properties; for instance `Agreement(SP, MT, [ASKey])` specifies that, the SP is correctly authenticated to the MT using the key ASKey. The Aliveness assertion checks the availability of the participants, e.g. `WeakAgreement(SP,MT)` assertion could be interpreted as follows: if MT has completed a run of the protocol with SP, then SP has previously been running the protocol, apparently with MT.

# Specification

```
Secret(SP,ASKey,[MT])
Secret(MT,ASKey,[SP])
Secret(SP,SrvCookies,[MT])
Agreement(SP,MT,[ASKey])
Agreement(MT,SP,[ASKey, SrvCookies])
WeakAgreement (SP, MT)
WeakAgreement (MT, SP)
```

The # Intruder Information heading specifies the Intruder identity, knowledge and capability. The first line identifies the Intruder as Mallory, the Intruder Knowledge defines the Intruder's initial knowledge i.e. we assume the intruder knows the identity of the participants.

# Intruder Information

```
Intruder = Mallory
IntruderKnowledge = {mt, desDA3C, ca3c, authID,
F2}
Crackable = ServiceSpecificKeys
```

After generating the CSP description of the systems using Casper and asking FDR to check the security assertions, no attacks were found against any of the assertions.

6.1.3 Security Considerations

Due to the fact that the security of the proposed protocol is based on the secrecy of the derived keys Srvkey and ASKey, this section will discuss all the possible threat models against the secrecy of the keys.

1. **The Secrecy of the Unique Key UK:** The proposed SL-AKA protocol complies with the security design of the AKA protocols of current systems such as 2G and 3G networks, which presume a secret key is brunet onto the SIM card of the mobile terminal and is shared with the network operator. The secret key will exclusively be used to derive further key and thus will not be exposed. The SL-AKA protocol refers to this secret key as the Unique Key (UK) and uses it to derive the Servkey.
2. **The Secrecy of the Service Key (Srvkey):** The Srvkey is derived by the MT and the CA3C from the UK as follows: Srvkey= F1(UK,SrvID, SubID, Lifetime). The secrecy of this key is very crucial for the overall security of the protocol, because exposing this key will lead to exposing the derived key Association Key (ASKey). Therefore, this key should be transferred securely from the CA3C to the SP. This emphasises on the need to secure the communication between the entities in core network, which has been discussed and addressed as part of the Network-Level AKA protocol presented in [5].

It is worth pointing out that unlike the AKA protocol of GSM an UMTS, where the key derivation functions of the derived key were kept secret, our proposed protocol presumes that the derivation function of the ASKey (F2) is known to the Intruder.

Furthermore, the proposed protocol works at the service or the application level, this means that it could operate with any security protcol at the network layer including the IP Security (IPSec) or EAP-based authentication protocols [40] [41].

6.1.4 Security Analysis Based on the Security Requirements List

To give a deep analysis of the security features of the SL-AKA protocol, this section discusses who the initial SL-AKA protocol meets the desired security requirements, explained in section 5.2.

1. Mutual Entity Authentication:
   Similar to the UL-AKA protocol, this security property is achieved, using the Authentication Token Ackm=F(SubID, SrvID, timestamp) which has been generated based on the parties' IDs.
2. Mutual Key Authentication:
   The mutual authentication between the MT and the SP is based on the secrecy of the derived session key Srvkey(SP). We got Casper to check this using the Secret (SP, Srvkey(SP),[MT]) assertion check.
3. Mutual Key Confirmation:
   This property is met by performing the check, using the Decryptable function after Msg9 and 10 in the Protocol Description heading Appendix 8 . By using the Decryptable function each party makes sure that, the valid key is possessed by the other part. If any of the check failed the protocol aborts.
4. Key Freshness: Since Casper does not have any function to check this property, The freshness of the Association key ASKey is guaranteed by including Vector 1 and 2 in its Key Generation Function (KGF) ASKey= F2(Srvkey(SP), Vector1, Vector2). These vectors comprise two fresh random values r1 and r2; thus, a new ASKey is derived for each session. Since Casper does not detect any attack on the secrecy of the ASKey, this implies that key freshness is not violated.
5. Unknown-Key Share resilience:
   This requirement could be met by making a bind between the derived key and the parties' identities. This is considered by including the SrvKey in the deriving function of the ASKey; the SrvKey involves the SubID and SrvID in its derivation function: Srvkey= F1(UK, SrvID, SubID, lifetime). Casper verifies this property by using the WeakAgreement assertion in the Specification heading.
6. Key Compromise Impersonation Resilience: We modelled this requirement by specifying the long-term keys as crackable and using the Agreement assertion to check any breach of the authenticity feature. However, this property will be analysed in more detailed in the following subsection.

6.1.5 Analysing the Key Compromise Impersonation Resilience property:

The key mentioned after the Crackable keyword will be compromised and passed to the intruder when all agents whose runs overlap in time with any agent using that key have finished their runs [15]. Our proposed protocol was not vulnerable to this attack, due to the fact that there was no overlapping among the agents' runs. However, to be very exhaustive, we simulate the case when the Intruder has managed to compromise the Srvkey(SP)- either in a previous run or in the current one-. By adding the Srvkey(SP) to the Intruder Knowledge as shown below:

# Intruder Information

```
Intruder = Mallory
IntruderKnowledge = {mt, desDA3C, ca3c, authID,
F2, Srvkey(SP)}
Crackable = ServiceSpecificKeys
```

The following attack has been discovered:

```
0. ca3c -> I_desDA3C : VECTOR1, SUBID, ADNAME
2. I_desAuth -> sp : VECTOR1, SUBID, ADNAME
3a. sp -> I_mt : {R2, VECTOR2, SRVCookies}
{Srvkey(sp)}
3b. I_sp -> mt : {R2, VECTOR2, SRVCookies}
{Srvkey(sp)}
4. mt -> I_sp : {R2, SRVCookies}{ASKey}
5. I_sp -> mt : {ACKM}{ASKey}
The intruder knows ASKey
```

Where the notations I_mt, I_desDA3C, I_SP represent the Intruder impersonating the MT, the DesDA3C and the SP, respectively. The attack is against the secrecy of the secret key ASKey and it leads to compromising the `Agreement(SP, MT,[ASKey]`. As a result, the MT believes it has completed a run of the protocol, taking role INITIATOR, with the SP, using data items ASKey while in reality it has been running the protocol with the Intruder instead.

6.2 Light Weight SL-AKA Protocol for Handover

When the MT performs handover and changes its point of attachment, the new access network might of a different credibility level. There is a need to consider these changes by re-negotiating the security parameters and deriving a new Association Key (NewASKey) to secure the connection between the MT and the SP.

However, there is a need not to interrupt the ongoing service; therefore, the re-negotiation process in the proposed SL-AKA protocol starts before the MT actually moves to the new network, and hence, the NewASKy is derived by the MT and SP prior to the handover. Furthermore, since the MT and SP have already authenticated each other, the new fast re- authentication will be based on the previous authentication.

The light weight SL-AKA protocol goes as follows: When the MT sends a handover request to a new domain, the CA3C will send the domain name of the new network towards the SP as in messages 1,2 and 3. When the SP receives this information, it re-orders the HMACList2 and the EncList2 to suit the new characteristics of the network, and thus the SP will have a different value of the Vector2. The SP will also use the old Association Key (OldASKey) to derive the new one: $NewASKey = F2(OldASKey, Vector1, Vector2)$.

$Msg1 : CA3C \rightarrow DesDA3C : ADname$
$Msg2 : DesDA3C \rightarrow DesAuth : ADname$
$Msg3 : DesAuth \rightarrow SP : ADname$
Generate the NewASKey= F2 (OldASKey , Vector1, Vector2)

The SP sends the new vector (Vector2) to the MT as message 4, which is encrypted using the OldASKy. Only the MT can decrypt this message to retrieve Vector2, which will be used by the MT to generate the NewASKey. The MT acknowledges the the successful derivation by sending an encrypted acknowledgement using the NewASKey.

$Msg4 : SP \rightarrow MT : \{Vector2\}_{OldASKey}$
Generate the NewASKey= F2 (OldASKey , Vector1, Vector2)
$Msg5 : MT \rightarrow SP : \{Ackm\}_{NewASKey}$

6.2.1 Formal Verification

Since the light weight SL-AKA protocol is based on the initial SL-AKA, it will meet the same desired security features such as the ones in section 5.2. This has been proven by Casper/FDR which found no attacks against the light weight SL-AKA. The full Casper/FDR description of the protocol is in Appendix 9.

Furthermore, in order to analyse the Key Compromise Impersonation Resilience property, we simulate the case when the Intruder knows the previous secret key (OldASKey) and checked for the secrecy of the newly derived one (NewASKey).

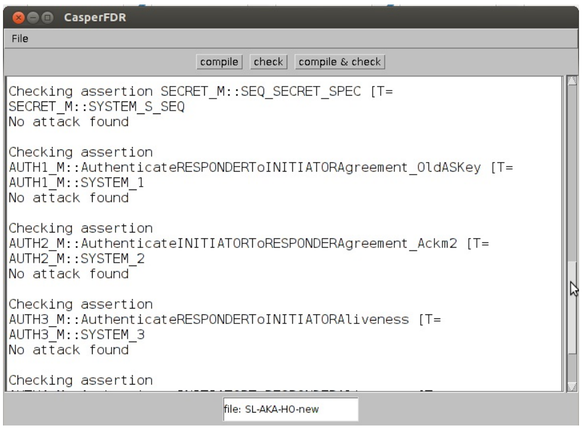
# Intruder Information

```
Intruder = Mallory
IntruderKnowledge = {mt, desDA3C, ca3c, authID,
F, oldASKey}
```

No attack was found as shown in Fig 7.

7 Conclusion

This article discussed several research efforts, which have been trying to address the issue of authenticating the mobile nodes to end servers in heterogeneous environment. The discussion showed that most of the solutions failed to realize the threats resulting from the open nature of future networks and to consider the power and processing restriction of mobile devices. Therefore, a novel Service-Level AKA protocol is introduced in



```
Checking assertion SECRET_M::SEQ_SECRET_SPEC [T=
SECRET_M::SYSTEM_S_SEQ
No attack found

Checking assertion
AUTH1_M::AuthenticateRESPONDERToINITIATORAgreement_OldASKey [T=
AUTH1_M::SYSTEM_1
No attack found

Checking assertion
AUTH2_M::AuthenticateINITIATORToRESPONDERAgreement_Ackm2 [T=
AUTH2_M::SYSTEM_2
No attack found

Checking assertion
AUTH3_M::AuthenticateRESPONDERToINITIATORAliveness [T=
AUTH3_M::SYSTEM_3
No attack found

Checking assertion
```

Fig. 7 Casper/FDR Verification

this paper, the proposed protocol provides mutual authentication and sets up a secure connection between the mobile terminal and the service provider. The protocol considers the initial authentication as well as the case of a handover, and it has been verified using formal method approach.

References

1. Internet Engineering Task Force, Handover keying working group (hokey wg). http://www.ietf.org/html.charters/hokey-charter.html.[Accessed 06 August 2012]
2. 3rd Generation Partnership Project (3GPP). http://www.3gpp.org/. [Accessed 06 August 2012].
3. Institute of Electrical and Electronics Engineers. IEEE 802.21/D8.0, Draft Standard for Local and Metropolitan Area Networks: Media Independent Handover Services, 2007.
4. Ali A.S.2010 *Authentication and Key Management in Heterogeneous Wireless Networks.* Electrical and Computer Engineering, The University of British Columbia.
5. Aiash M, Mapp G, Lasebae A, Phan R, Loo J. 2012 *A Formally Verified AKA Protocol For Vertical Handover in Heterogeneous Environments using Casper/FDR.*EURASIP Journal on Wireless Communications and Networking 2012, 2012:57.
6. Hohendorf C, Unurkhaan E, Dreibholz T. 2005. *Secure SCTP.* draft-hohendorf-secure-sctp-00.
7. Tuexen M, Stewart R, Lei P, Rescorla E. 2007. *Authenticated Chunks for Stream Control Transmission Protocol (SCTP)*, draft-ietf-tsvwg-sctp-auth-08.txt.
8. Masahiro K, Mariko Y, Ryoji O, Shinsaku K, Tanaka T. 2004. *Secure Service and Network Framework for Mobile Ethernet.* Wirel. Pers. Commun
9. Aiash M, Mapp G, Lasebae A, A AL Nemrat: Supporting LTE Networks in Heterogeneous Environment using the Y-Comm Framework. In proceeding of: The Fourth International Conference on Networks Communications (NETCOM-2012).
10. The LTE/LTE Advanced Guidea semi-annual publication on LTE/LTE Advanced, http://lteportal.com/LTE Business Guide
11. Ryan P, Schneider S, Goldsmith M, Lowe G, Roscoe A.W. 2010,*The modelling and analysis of security protocols.* PEARSON Ltd.
12. Roscoe A.W. 1995. *Modelling and verifying key-exchange protocols using CSP and FDR.* 8th IEEE Computer Security Foundations Workshop.
13. Schneider S. 1996. *Security properties and CSP.* IEEE Symposium on Security and Privacy.
14. Formal Systems (Europe) Ltd.: Failures-Divergence Refinement. FDR2 User Manual. http://www.fsel.com/documentation/fdr2/fdr2manual.pdf. [Accessed 19 August 2011]
15. Lowe G, Broadfoot P, Dilloway C, Hui M. 2011 *Casper, A compiler for the Analysis of security protocol.*http://www.comlab.ox.ac.uk/gavin.lowe /Security/Casper/ [Accessed 19 August 2011].
16. Almeida M, Corujo D, Sargento S, Jesus V, and Aguiar R. 2007.*An End-to-End QoS Framework for 4G Mobile Heterogeneous Environments.* OpenNet Workshop (2007).
17. Aiash M, Mapp G, Lasebae A. 2011. *A QoS framework for Heterogeneous Networking.* ICWN2011, London, UK.
18. International Telecommunication Union (ITU-T), Global Information Infrastructure, Internet Protocol Aspects And Next Generation Networks, Y.140.1, 2004.
19. Bartoli M, Baumgartner F, Brandauer C, Braun T, Kardos S, Orl F, Scheidegger M, Seger J. 2004 *The Intermon Simulation Framework.*In Proceedings of the Second Inter-Domain Performance and Simulation Workshop.
20. Johnson D, Perkins C, Arkko J. 2004. *Mobility Support in IPv6.* RFC 3775.
21. McCann P. 2005. *Mobile IPv6 Fast Handovers for 802.11 Networks.* RFC 4260.
22. Dierks T, Allen C. 1999. *The TLS Protocol.* RFC 2246.
23. Jalali-Sohi M, Ebinger P. 2002. *Towards efficient PKIs for restricted mobile devices.* International Conference on Communications and Computer Networks (CCN).
24. Stewart R. 2007. *Stream Control Transmission Protocol.* RFC 4960.
25. Cerf V, Dalal Y. 1974. *SPECIFICATION OF INTERNET TRANSMISSION CONTROL PROGRAM.* RFC 675.
26. Postel J. 1980.*User Datagram Protocol.* RFC 768.
27. Tuexen M, Stewart R, Lei P, Rescorla E. 2007. *Authenticated Chunks for Stream Control Transmission Protocol (SCTP)*, draft-ietf-tsvwg-sctp-auth-08.txt.
28. Riegel M, Tuexen M. 2005. *Mobile SCTP.* draft-riegel-tuexen-mobile-sctp-05.txt
29. Kuroda M, Inoue M, Okubo A, Sakakura T, Shimizu K, Adachi F.2004. *Scalable Mobile Ethernet and Fast Vertical handover.* IEEE Wireless Communications and Networking Conference.
30. Menezes A, van Oorschot P, Vanstone S. 1996. *Handbook of Applied Cryptography.* CRC Press.
31. Chandra P. 2005. *Bulletproof wireless security : GSM, UMTS,802.11 and ad hoc security* Newnes. Oxford, pp. 129-158,2005
32. Aiash M, Mapp G, Lasebae A & Phan R. 2010. *Providing Security in 4G Systems: Unveiling the Challenges.* AICT 2010. Barcelona, Spain, 9-15 May 2010
33. Stig Fr M, Joe-Kai T.2012. *Computational Security Analysis of the UMTS and LTE Authentication and Key Agreement Protocols.* ACM CoRR.
34. Aiash M, Mapp G, Lasebae A & Phan R. 2012. *A SURVEY ON AUTHENTICATION AND KEY AGREEMENT PROTOCOLS IN HETEROGENEOUS NETWORKS.* International Journal of Network Security Its Applications (IJNSA), Vol.4, No.4, July 2012

35. C Han-Chieh., Z Sherali, C Yuh-Shyan., M Gregorio., W Reen-Cheng .: Next Generation Networks (NGNs). Int.J. Commun. Syst.(2010). DOI: 10.1002/dac.1144.
36. He, D., Chen, C., Chan, S., Bu, J.: Strong roaming authentication technique for wireless and mobile networks. Int.J. Commun. Syst.(2012). DOI:10.1002/dac.1387.
37. Chen, C., He, D., Chan, S., Bu, J., Gao, Y., Fan,. R.: Lightweight and provably secure user authentication with anonymity for the global mobility network. Int.J. Commun. Syst.(2010). DOI:10.1002/dac.1158.
38. X Qi.: A new authenticated key agreement for session initiation protocol. Int.J. Commun. Syst.(2011). DOI:10.1002/dac.1286.
39. Y Ilsun., L Jong-Hyouk., K Bonam.: caTBUA: Context-aware ticket-based binding update authentication protocol for trust-enabled mobile networks. Int.J. Commun. Syst.(2010). DOI:10.1002/dac.1113.
40. P Hoffman.: Cryptographic Suites for IPsec. Request for Comments: 4308. December 2005.
41. B Aboba., L Blunk., J Vollbrecht., J Carlson., H Levkowetz.: Extensible Authentication Protocol (EAP). Request for Comments: 3748. June 2004.

8 Code for Formal Analysis of the Proposed Initial SL-AKA Protocol

```
# Free Variables
MT: Agent
SP : Service
DesAuth : DesAccessRouterAuthenticator
DesDA3C : DesDomainA3CServer
CA3C : CentralA3CServer
r1, r2 : Nonce
ADname : AccessDomainname
Srvkey :Service -> ServiceSpecificKeys
F: ServiceSpecificKeys x Vectors x Vectors ->
AssociationKeys
SubID : ServiceSubscribtionID
ASKey : AssociationKeys
Vector1,Vector2:Vectors
SrvCookies: Cookies
Ackm : AcknowledgementMessage
InverseKeys = (Srvkey, Srvkey), (ASKey,ASKey),
(F,F)

# Processes
INITIATOR(MT,Ackm,r1,Vector1, SubID,ADname)
knows Srvkey(SP)
DesAuthenticator(DesAuth,SP,DesDA3C)
DesAAASERVER(DesDA3C,CA3C,DesAuth)
CentralSERVER(CA3C, DesDA3C,Vector1,SubID,
ADname)
RESPONDER(SP, MT, DesAuth, DesDA3C, r2, Vector2,
SrvCookies, Ackm)
knows Srvkey(SP)
# Protocol Description
0. CA3C -> DesDA3C: Vector1, SubID, ADname
1. DesDA3C -> DesAuth: Vector1, SubID, ADname
2. DesAuth -> SP : Vector1, SubID, ADname
$< ASKey := F(Srvkey(SP), Vector1, Vector2) >$
3. SP -> MT : {r2, Vector2, SrvCookies}
{Srvkey(SP)}
$< ASKey := F(Srvkey(SP), Vector1, Vector2) >$
4. MT -> SP : {r2, SrvCookies}{ASKey}%v1
$[decryptable(v1, ASKey) andnth(decrypt$
$(v1, ASKey), 1) == SrvCookies]$
5. SP -> MT : {Ackm}{ASKey}%w3
$[decryptable(w3, ASKey) andnth(decrypt$
$(w3, ASKey), 1) == Ackm]$
# Specification
Secret(SP,ASKey,[MT])
Secret(MT,ASKey,[SP])
Secret(SP,SrvCookies,[MT])
Agreement(SP,MT,[ASKey])
Agreement(MT,SP,[ASKey, SrvCookies])
WeakAgreement (SP, MT)
WeakAgreement (MT, SP)
# Actual Variables
mt, Mallory: Agent
desAuth : DesAccessRouterAuthenticator
desDA3C : DesDomainA3CServer
sp : Service
ca3c : CentralA3CServer
R1, R2 : Nonce
ADNAME : AccessDomainname
VECTOR1,VECTOR2: Vectors
SRVCookies: Cookies
SUBID : ServiceSubscribtionID
ACKM : AcknowledgementMessage
ASKEY : AssociationKeys
InverseKeys = (ASKEY,ASKEY)
# Functions
symbolic Srvkey, F
# System
INITIATOR(mt, ACKM, R1, VECTOR1, SUBID, ADNAME)
DesAuthenticator(desAuth, sp, desDA3C)
DesAAASERVER(desDA3C, ca3c, desAuth)
CentralSERVER(ca3c, desDA3C, VECTOR1, SUBID,
ADNAME)
RESPONDER(sp, mt, desAuth, desDA3C, R2, VECTOR2,
SRVCookies, ACKM)
# Intruder Information
Intruder = Mallory
IntruderKnowledge = {mt, desDA3C, ca3c, authID,
F}
```

9 Code for Formal Analysis of the Light Weight SL-AKA Protocol For Handover

```
# Free Variables
MT: Agent
SP : Service
DesAuth : DesAccessRouterAuthenticator
DesDA3C : DesDomainA3CServer
CA3C : CentralA3CServer
r1, r2 : Nonce
ADname : AccessDomainname
Srvkey :Service -> ServiceSpecificKeys
F: ServiceSpecificKeys x OldAssociationKeys
x Vectors x Vectors -> NewAssociationKeys
SubID : ServiceSubscribtionID
NewASKey : NewAssociationKeys
OldASKey : OldAssociationKeys
Vector1,Vector2: Vectors
SrvCookies: Cookies
Ackm : AcknowledgementMessage
InverseKeys = (Srvkey, Srvkey),
(NewASKey,NewASKey), (OldASKey,OldASKey) ,(F,F)

# Processes
INITIATOR(MT, Ackm,r1,Vector1, SubID,ADname,
OldASKey) knows Srvkey(SP)
DesAuthenticator(DesAuth,SP,DesDA3C)
DesAAASERVER(DesDA3C,CA3C,DesAuth)
CentralSERVER( CA3C, DesDA3C,Vector1,SubID, ADname)
RESPONDER(SP, MT, DesAuth, DesDA3C, r2, Vector2,
SrvCookies, Ackm, OldASKey) knows Srvkey(SP)

# Protocol Description
0. CA3C -> DesDA3C:Vector1, SubID,ADname
1. DesDA3C -> DesAuth:Vector1, SubID,ADname
2. DesAuth -> SP : Vector1, SubID,ADname
< NewASKey := F(Srvkey(SP), OldASKey,
Vector1, Vector2) >
3. SP -> MT : {Vector2}{OldASKey}
< NewASKey := F(Srvkey(SP), OldASKey,
Vector1, Vector2) >
4. MT -> SP : {Ackm}{NewASKey}
# Specification
Secret(SP,NewASKey,[MT])
Secret(MT,NewASKey,[SP])
Secret(SP,SrvCookies,[MT])
Agreement(SP,MT,[OldASKey])
Agreement(MT,SP,[Ackm])
WeakAgreement (SP, MT)
WeakAgreement (MT, SP)
# Actual Variables
mt, Mallory: Agent
desAuth : DesAccessRouterAuthenticator
desDA3C : DesDomainA3CServer
sp : Service
ca3c : CentralA3CServer
R1, R2 : Nonce
ADNAME : AccessDomainname
VECTOR1,VECTOR2: Vectors
SRVCookies: Cookies
SUBID : ServiceSubscribtionID
ACKM : AcknowledgementMessage
InverseKeys = (newASKey,newASKey)
,(oldASKey,oldASKey)
newASKey : NewAssociationKeys
oldASKey : OldAssociationKeys
# Functions
symbolic Srvkey, F
# System
INITIATOR(mt, ACKM, R1, VECTOR1, SUBID, ADNAME,
oldASKey)
DesAuthenticator(desAuth, sp, desDA3C)
DesAAASERVER(desDA3C, ca3c, desAuth)
CentralSERVER( ca3c, desDA3C, VECTOR1, SUBID, ADNAME)
RESPONDER(sp, mt, desAuth, desDA3C, R2, VECTOR2,
SRVCookies, ACKM, oldASKey)
# Intruder Information
Intruder = Mallory
IntruderKnowledge = {mt, desDA3C, ca3c, authID,
F}
```