A Privacy Protecting UMTS AKA Protocol Providing Perfect Forward Secrecy*

Daeyoung Kim[1], Younggang Cui[2], Sangjin Kim[3], and Heekuck Oh[2]

[1] Empas Corporation, Republic of Korea
kdy1029@empascorp.com
[2] Hanyang University, Department of Computer Science and Engineering, Republic of Korea
{ygcui,hkoh}@cse.hanyang.ac.kr
[3] Korea University of Technology and Education, School of Internet Media Engineering, Republic of Korea
sangjin@kut.ac.kr

Abstract. In UMTS (Universal Mobile Telecommunication System), a protocol called UMTS AKA (Authentication and Key Agreement) is used to securely authenticate an MS (Mobile Station). However, the UMTS AKA has several drawbacks such as network bandwidth consumption and synchronization problem. In this paper, we propose a new authentication protocol for UMTS that overcomes these problems. Moreover, our protocol enhances the security of the protocol by providing better privacy and also provides perfect forward secrecy. Furthermore, our protocol also provides mutual authentication between an MS and its HN (Home Network) and between an MS and the SN (Serving Network).

Keywords: UMTS, authentication, privacy, perfect forward secrecy.

1 Introduction

1.1 Overview of UMTS

In recent years, we have observed rapid development of wireless and mobile communication networks, especially in the UMTS. The UMTS is one of the third generation mobile communication standards which is currently being launched throughout the world. It inherits the framework of GSM (Global System for Mobile Communications), its forerunner, which has many disadvantages. The UMTS uses UMTS AKA protocol [1] to authenticate MSs. This protocol is based on the GSM security framework and designed to be secure against the known GSM vulnerabilities.

The network architecture of UMTS is given in Fig 1. In UMTS, an MS is associated with a HLR (Home Location Register) and an AuC (Authentication Center) associated with the HLR maintains authentication information of the MS. The HLR together with AuC is referred to as the HN of the MS. An MS is connected to the UTRAN (UMTS Terrestrial Radio Access Network) via the radio interface. The UTRAN, which

* This work was supported by the Ministry of Information and Communication, Korea, under the HNRC-ITRC program supervised by the IITA.

O. Gervasi and M. Gavrilova (Eds.): ICCSA 2007, LNCS 4706, Part II, pp. 987–995, 2007.

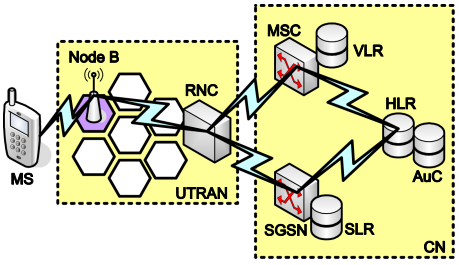


Fig. 1. The UMTS Network Architecture

is composed of Node Bs (base transceiver station) and RNCs (Radio Network Controller), connects an MS with the CN (Core Network). The CN is the long-range network that transports a user's data to its respective destination. Depending on whether circuit-switched or packet-switched is used, an MS is served by the local MSC (Mobile Switching Center), which contains the VLR (Visitor Location Register), or the local SGSN(Serving GPRS (General Packet Radio Service) Support Node), which contains the SLR (Serving Location Register), respectively. The MSC together with VLR or the SGSN together with SLR is referred to as the current SN of the MS.

The main purpose of UMTS AKA protocol is to provide mutual authentication between an MS and a SN. However, both SLR and VLR does not have the necessary information to authenticate a foreign MS. Therefore, the SN contacts the HN of the MS to acquire the subscription and authentication data. An MS is uniquely identified by IMSI (International Mobile Subscriber Identifier). Attackers can use this information to track the user of MS. Therefore, IMSI should be protected as possible to provide better user privacy. To this end, a TMSI (Temporary MSI) is used locally instead of IMSI. But to date, all methods requires an MS to send its IMSI at least once. The UMTS AKA [1] and all subsequent protocols for UMTS [2,3,4] uses a long-term key shared between an MS and its HN to authenticate an MS. However, disclosure of long-term keys result in disclosure of all the previous communications. In other words, to date, all protocols for UMTS do not provide perfect forward secrecy. In this paper, we provide a new protocol that does not require an MS to send its IMSI and provides perfect forward secrecy.

1.2 Notation

In this paper, the following notations are used to describe protocols.

- M, S, H: denotes the MS, the SN, and the HN, respectively.
- K_{XY}: denotes a symmetric key shared between participant X and Y.
- N_X: denotes a nonce generated by the participant X.
- T_X: denotes a timestamp generated by the participant X.
- C_X: denotes a counter maintained by the participant X.
- f_K^i: denotes a MAC using the key K, where i differentiates different MACs.
- g_K^i: denotes a key generation function using the key K.
- h: denotes a collision-resistant hash function.

- $\{M\}.K$: denotes an encryption of a message M with the key K.
- IK_X, CK_X, AK_X, TK_X: denotes a symmetric key used for integrity, confidentiality, anonymity, and ticket key generated by X, respectively.
- AMF: denotes an authentication and key management field.

1.3 Background

In this section, we will give some definitions related to our work.

Definition 1 (forward secrecy). *We say that a protocol provides the forward secrecy, if the advantage of an adversary compromising past session keys is negligible, even though he/she has compromised long-term keys of one or more participants.*

Definition 2 (perfect forward secrecy). *We say that a protocol provides the perfect forward secrecy, if the forward secrecy is preserved, even if an adversary has compromised long-term keys of all the participants.*

For the following, $\mathbb{G}$ is an additive group on an EC (Elliptic Curve) of prime order q, P is a random elements of $\mathbb{G}$, and a and b are random elements of $\mathbb{Z}_q$.

Definition 3 (EC-Discrete Logarithm Problem (EC-DLP)). *Given P and aP, compute a.*

Definition 4 (EC-Computational Diffie-Hellman Problem (EC-CDHP)). *Given P, aP, and bP, compute abP.*

Currently, solving EC-DLP and EC-CDHP is computationally infeasible.

1.4 Our Contribution

In this paper, we propose a new authentication protocol for UMTS. This protocol provides the following characteristics.

- Provides mutual authentication between an MS and its HN.
- Provides mutual authentication between an MS and the current SN.
- Reduces network bandwidth consumption between the SN and the HN by not using multiple authentication vectors.
- Reduces the storage overhead of the SN by not using multiple authentication vectors.
- Provides better privacy protection for the subscribers, since dynamic ID is used instead of IMSI.
- Enhances security by providing perfect forward secrecy.

2 Previous Work

2.1 The UMTS AKA Protocol

In UMTS AKA protocol [1], an MS shares a secret key K_{MH} with its HN. An MS maintains a counter C_M. A HN maintains a counter C_H for each individual subscribers. C_M

and the corresponding C_H must be loosely synchronized. The protocol runs as follows. The SN forwards *IMSI* sent by the MS to the HN to request authentication data. The HN generates m AVs. Each AV consist of N_H, $XRES = f^2_{K_{MH}}(N_H)$, $CK_H = g^2_{K_{MH}}(N_H)$, $IK_H = g^3_{K_{MH}}(N_H)$, and $AUTH = C \oplus AK_H || AMF || MAC_H$, where C is the current value of C_H, $AK_H = g^1_{K_{MH}}(N_H)$, and $MAC_H = f^1_{K_{MH}}(C||N_H||AMF)$. The C_H is incremented each time an AV is constructed. The SN selects one of the AV on FIFO basis and sends N_M and $AUTH$ of the selected AV to the MS. Upon receipt, the MS computes AK_H and obtains C included in the $AUTH$. The C must satisfy $C > C_M$. If C is in the correct range, the MS verifies MAC_H. Finally, it computes $RES = f^2_{K_{MH}}(N_M)$ and sends it back to the SN. The protocol completes by verifying $XRES = RES$.

2.2 The AP-AKA Protocol

Zhang and Fang proposed a protocol called AP-AKA [2] that runs as follows. A SN sends N_S to the MS. The MS generates N_M and calculates $MAC_M = f^1_{K_{MH}}(N_S||N_M||ID_S)$, where ID_S denotes the identity of the SN. The MS sends them with its *IMSI* to the SN. The SN forwards N_S, N_M, MAC_M, and *IMSI* to the HN. The HN verifies MAC_M and generates m AVs. It then sends them to the SN. Each AV consists of N_H, $XRES = f^2_{K_{MH}}(N_H)$, $AK_H = g^1_{K_{MH}}(N_H)$, and $AUTH = i||N_i||MAC_H$, where i is the index number of the newly generated AVs, $N_i = f^3_{K_{MH}}(i||N_M)$, and $MAC_H = f^1_{K_{MH}}(N_H||i||N_i)$. The SN selects unused AV with the lowest index and sends the N_H and $AUTH$ of the AV to the MS. The MS verifies the MAC_H included in the $AUTH$ and replies by sending $RES = f^2_{K_{MH}}(N_H)$. The protocol completes by verifying $XRES = RES$.

2.3 Harn-Hsin Protocol

Harn-Hsin protocol for UMTS do not use AVs and instead employs hash chaining technique [3]. We refer to this protocol as HH-AKA. In this protocol, an MS and a SN generates I and J hash chains, respectively. However, we will simplify our description by assuming that both MS and SN generates a single long hash chain. The simplified version of the protocol runs as follows. An MS generates a hash chain and sends its *IMSI*, $h^m(b_M)$, and a $MAC_M = f^1_{K_{MH}}(IMSI||h^m(b_M))$ to the SN, where b_M is the random seed of the hash chain, and h^m denotes m composition of hash function h. The SN forward this message to the HN. The HN verifies the MAC_M and computes a single AV. The AV consist of *IMSI*, $h^m(b_M)$, N_H, $CK_H = g^2_K(N_H)$, $IK_H = g^3_K(N_H)$, and $AK_H = g^1_K(N_H)$. The SN generates a hash chain and sends N_H, $h^m(b_S)$, and $MAC_S = f^1_{AK_H}(N_H||h^m(b_S))$ to the MS, where b_S is the random seed of SN's hash chain. The MS verifies the MAC_S by constructing AK_H and using it. It also computes IK_H and CK_H. It replies by sending $h^{m-i}(b_M)$. The SN verifies it and computes $CK_S = g^2_{CK_H}(h^{m-i}(b_S)||h^{m-i}(b_M))$ and $IK_S = g^3_{IK_H}(h^{m-i}(b_S)||h^{m-i}(b_M))$. It then sends $h^{m-i}(b_S)$ to the MS. The MS authenticates the SN by verifying the hash chain of SN.

2.4 UMTS X-AKA Protocol

Huang and Li proposed a protocol called X-AKA protocol [4] that runs as follows. An MS first sends *IMSI*, T_M, and $MAC_M = f^1_{K_{MN}}(T_M)$ to the SN. The SN forwards

Table 1. Comparison of Previous Work

	UMTS AKA	AP-AKA	HH-AKA	X-AKA	Ours
Use of AVs	◯	◯	hash-chain	ticket key	ticket key
Synchronization (MS ↔ HN)	counter	×	×	clock	clock
Bandwidth consumption (HN ↔ SN)	◯	◯	×*	×	×
Storage overhead of SN	◯	◯	×*	×	×
Mutual authentication (MS ↔ HN)	MS ← HN	◯	◯	◯	◯
Mutual authentication (MS ↔ SN)	◯	◯	◯	◯	◯
Effect of long-term key disclosure	◯	◯	◯	◯	×
Effect of short-term key disclosure	×	×	×**	◯***	×***

*: assumes a single long hash-chain is used.
**: effect of disclosure of the root of a hash chain.
***: effect of disclosure of TK_H.

these data to the HN. The HN verifies the MAC_M. It then computes ticket key $TK_H = g^1_{K_{MN}}(T_M)$ and $AUTH_H = MAC_H||N_H||AMF$ and sends them to the SN, where $MAC_H = f^1_{K_{MN}}(N_H||AMF)$. The SN generates N_S and computes $MAC_S = f^1_{TK_H}(MAC_H||N_S + j \times N_H)$ and $AUTH_S = MAC_S||N_S||N_M||AMF||j$, where j denotes jth usage of TK_H. It sends $AUTH_S$ to the MS. The MS verifies MAC_S and MAC_H. It then responds by sending $RES = f^2_{TK_H}(N_S)$. Both the MS and the SN computes $IK_S = g^2_{TK_H}(N_S)$ and $CK_S = g^3_{TK_H}(N_S)$.

2.5 Comparison of Previous Work

The UMTS AKA and AP-AKA uses AVs to minimize the number of access to a HN by a SN. However, the use of AVs may cause too much bandwidth consumption between a SN and a HN and storage overhead in a SN. To overcome this, HH-AKA and X-AKA do not use AVs. Instead they use hash chains and Kerberos-like ticket key, respectively. The original HH-AKA uses several hash chains which may results in similar bandwidth consumption and storage overhead compared to UMTS AKA. However, if a single long hash chain is used, it can be classified in the same category as X-AKA.

In UMTS AKA, the HN cannot authenticate an MS. However, in other protocols, the HN can authenticate the requesting MS by verifying the MAC_M. In all four protocols, an MS can authenticate its HN by verifying the MAC_H or MAC_S. In HH-AKA, we must note that an MS cannot verify the timeliness of MAC_S. However, in UMTS AKA and AP-AKA, C and N_M is included in MAC_H, respectively. And in X-AKA T_M is used to compute TK_H. These allows an MS to verify the freshness of MAC_H or MAC_S.

In all four protocols, the disclosure of the long-term key K_{MH} results in disclosure of all previous communications. On the other hand, the disclosure of short-term key used in AVs does not effect other AVs. In HH-AKA, the root of hash chains, CK_H, and IK_H must all be disclosed to affect past or future communications. In X-AKA, the disclosure of TK_H results in disclosure of all previous communications involving TK_H.

In all four protocols, the SN can authenticate an MS by verifying the response of the MS. In UTMS AKA, the MS cannot directly authenticate the current SN. However, in order for the protocol to complete, some legitimate entity must have received the correct

AVs from the HN. A HN is trusted to give the necessary authentication data only after securely authenticating the requesting SN. As a result, if SNs are trusted authorities, the MS can conclude that the local SN was involved. Moreover, in AP-AKA, the MS includes the *ID* of the SN in its request. This inclusion improves the security of the protocol, since only the SN with the given *ID* can receive the AVs.

3 The Proposed Protocol

3.1 Assumption

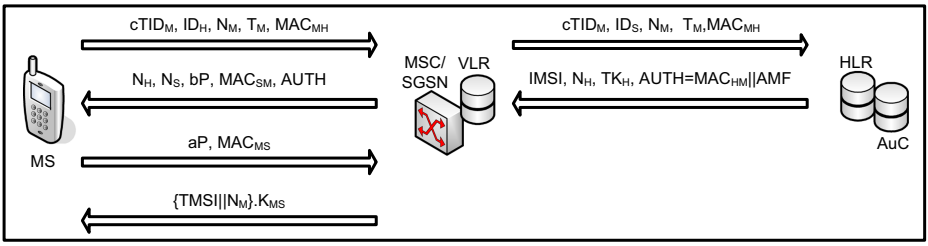
In our protocol, we assume the following.

- A secure and authenticated channel can be established between a HN and a SN.
- An MS can identify the correct ID of the current SN in which it resides.
- HNs and SNs are regarded as a trusted authority and we do not consider security problems that may occur due to an attacker gaining control of a HN or a SN.
- We do not consider protocol weakness resulting from using weak algorithms, nor do we consider security holes resulting from coexistence of GSM and UMTS [5].

3.2 The Protocol

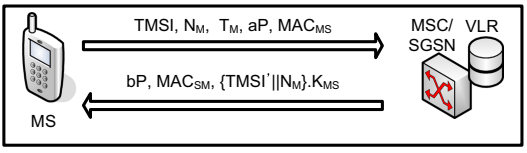
The proposed initial protocol depicted in Fig. 2 runs as follows.

- **Step 1.** An MS generates N_M and computes $MAC_{MH} = f^1_{K_{MH}}(N_M||T_M||ID_S)$. It then sends $cTID_M$, ID_H, N_M, T_M, and MAC_{MH} to the current SN. $cTID_M$ is the current temporary ID of an MS, which is computed as $f^2_{K_{MH}}(IMSI)$ or $f^2_{K_{MH}}(pTID_M)$, where $pTID_M$ is the previous temporary ID of the MS.
- **Step 2.** The SN forwards the received data to the HN using the ID_H to locate the HN of the MS.
- **Step 3.** The HN searches the entire database to identify the requesting MS. For each subscriber, *IMSI* and current and previous TID_M is stored in the database. Previous TID_M is maintained to solve the synchronization problem of TID_M. It then uses K_{MH} to verify the MAC_{MH}. During this process, the HN verifies that the SN is the



$MAC_{MH} = f^1_{K_{MH}}(N_M||T_M||ID_S)$, $MAC_{HM} = f^1_{K_{MH}}(N_M||N_H)$, $MAC_{SM} = f^1_{TK_H}(N_M||N_S||bP)$,
$MAC_{MS} = f^1_{K_{MS}}(N_S||aP)$ $TK_H = g^1_{K_{MH}}(N_M||N_H)$, $K_{MS} = g^2_{TK_H}(abP)$

Fig. 2. The ECC-AKA Initial Protocol



$$MAC_{SM} = f^1_{K_{MS}}(N_M||bP),\ \ MAC_{MS} = f^1_{TK_H}(N_M||aP||T_M),\ \ K_{MS} = g^2_{TK_H}(abP)$$

Fig. 3. The ECC-AKA Subsequent Protocol

same SN identified by ID_S included in the MAC_{MH} and the timestamp T_M is valid. It then generates the next $nTID_M = f^2_{K_{MH}}(cTID_M)$ and N_H. The $nTID_M$ is stored in the database as the current TID_M and $cTID_M$ is stored as the previous TID_M. It also computes $MAC_{HM} = f^1_{K_{MH}}(N_M||N_H)$ and $TK_H = g^1_{K_{MH}}(N_M||N_H)$. Finally, it sends $IMSI$, N_H, $AUTH = MAC_{HM}||AMF$ to the SN. These data are sent through an authenticated and secure channel.

- **Step 4.** The SN selects $b \in_R \mathbb{Z}_q$ and computes bP. It then generates N_S and computes $MAC_{SM} = f^1_{TK_H}(N_M||N_S||bP)$. Finally, it sends N_H, N_S, bP, MAC_{SM} and $AUTH$ to the MS.
- **Step 5.** The MS first verifies MAC_{HM} and then computes TK_H. It then verifies MAC_{SM}. If everything confirm, it computes its next temporary ID and stores it with TK_H. It then selects $a \in_R \mathbb{Z}_q$ and computes aP and $K_{MS} = g^2_{TK_H}(abP)$. Finally, it computes $MAC_{MS} = f^1_{K_{MS}}(N_S||aP)$ and sends aP and MAC_{MS} to the SN. To ease the computation burden on MS, MSs can precompute aP in advance. However, this would require some storage overhead.
- **Step 6.** The SN computes K_{MS} and use it to verify MAC_{MS}. It then generates $TMSI$ for the MS and sends it encrypted to the MS. Although sending $TMSI$ is a necessity, other previous protocols do not explicitly include this step in the protocol description. Both the MS and SN computes CK_{MS} and IK_{MS} using the new K_{MS}.

Subsequent protocols between an MS and a SN uses the protocol depicted in Fig. 3. The protocol runs as follows.

- **Step 1.** The MS first selects $a \in_R \mathbb{Z}_q$ and computes aP. It then generates N_M and computes $MAC_{MS} = f^1_{TK_H}(N_M||aP||T_M)$. Finally, it sends $TMSI$, N_M, T_M, aP, and MAC_{MS} to the SN.
- **Step 2.** The SN uses $TMSI$ to retrieve the TK_H and verifies MAC_{MS}. If it confirms, it selects $b \in_R \mathbb{Z}_q$ and computes bP and abP. It then generates the session key $K_{MS} = g^2_{TK_H}(abP)$. It also computes $MAC_{SM} = f^1_{K_{MS}}(N_M||bP)$. It also generates a new TMSI. The SN must maintain previous $TMSI$ to solve the synchronization problem of $TMSI$. Finally, it sends bP, MAC_{SM} and encrypted $TMSI$ to the MS. The MS computes $K_{MS} = g^2_{TK_H}(abP)$ and verifies MAC_{SM}. Both the MS and SN computes CK_{MS} and IK_{MS} using the new K_{MS}.

4 Analysis

4.1 Security Analysis

In this section, we analyze our protocol's security.

- Mutual authentication between an MS and its HN: The HN and MS can mutually authenticate each other by verifying MAC_{MH} and MAC_{HM}, respectively. These MACs are generated using a secret key K_{MH} shared between each other.
- Mutual authentication between an MS and a SN: Let's first consider the initial protocol. We assume that MSs can identify the ID of SN they are residing. This ID is included in MAC_{MH}. Moreover, the HN securely authenticates the SN before sending authentication data of the MS. Furthermore, the MS can verify that TK_H was generated by its HN and the freshness of it. Therefore, the MS can authenticate the SN by verifying MAC_{SM} which is generated using TK_H. The SN also securely authenticates the HN of the MS before receiving authentication data. Moreover, these data are exchanged through a secure channel. Therefore, the SN can believe that TK_H is a secure key generated by the HN for it and the MS. This key is used to compute K_{MS}. As a result, the SN can authenticate the MS by verifying MAC_{MS} which is generated using K_{MS}. In the subsequent protocol, the above argument still holds, since MAC_{MS} and MAC_{SM} is computed using TK_H and K_{MS}, respectively.
- Privacy protection for the subscribers: In our protocol, the unique identity of MS is never exchanged in a cleartext. The MS always uses a TID_M generated using K_{MH}. As a result, only the MS and the HN can generate the next sequence ID. Therefore, third parties cannot link temporary IDs of an MS. To date, in previous protocols, an MS must send its IMSI at least once each time the MS visits a new foreign network.
- Perfect forward secrecy: In our protocol, we use EC-based Diffie-Hellman key agreement protocol to establish a session key between an MS and a SN. Since EC-CDHP is computationally infeasible, third parties cannot compute the abP even though aP and bP are sent in clear. These values are required to compute the session key K_{MS}. Moreover, to defend against man-in-the-middle-attack, these values are included in MACs exchanged between an MS and a SN.

Now, we will review some possible attacks.

- Replay attacks: A replay attack using any of the message in both protocols can be detected using either the timestamp or the nonce included in the MACs.
- Problem related to synchronization of TID_M: If an attacker suppresses the message from the SN to the MS, the HN will have updated the TID_M, whereas the MS have not. In this case, the MS will use $cTID_M$ again. However, since the HN maintains previous ID, the HN can still identify the requesting MS.

4.2 Efficiency Analysis

- Reduction of network bandwidth consumption between the SN and the HN: Only a single authentication data are exchanged between SNs and HNs. The multiple use of TK_H is limited by setting a validation period. As a result, our approach is similar to that of X-AKA.

- Reduction of storage overhead of the SN: The SN only stores a single authentication data for each MS. Therefore, the storage overhead is similar to that of X-AKA.
- Use of public key operation: In order to provide perfect forward secrecy, we use EC-based Diffie-Hellman key agreement protocol. Every time an MS runs the protocol with a SN, both of them perform two EC operation. Previous protocols do not use any kind of public key operation. The reason for this is because MSs have computational and battery limitation. However, the technology is improving rapidly and currently people are even looking at fully public-key based solution for UMTS [6]. Since our system only use ephemeral keys, we do not require any certificates and two EC operation per authentication will not be too much of a burden on MSs. Moreover, if several aPs are precomputed in advance, only a single EC operation per authentication is required. We must note that this requires some secure storage space to maintain $\langle a, aP \rangle$ pairs.

5 Conclusion

In this paper, we proposed a new authentication protocol for UMTS that overcomes problems of UMTS AKA such as network bandwidth consumption and storage overhead. Moreover, our protocol provides better privacy by using encrypted dynamic IDs and also provides perfect forward secrecy using EC-based Diffie-Hellman key agreement protocol. Our protocol also provides mutual authentication between an MS and its HN and an MS and the SN.

References

1. 3GPP TS 33.102, Security Architecture. V7.0.0 (2005)
2. Zhang, M., Fang, Y.: Security Analysis and Enhancements of 3GPP Authentication and Key Agreement Protocol. IEEE Trans. on Wireless Communications 4(2), 734–742 (2005)
3. Harn, H., Hsin, W.: On the Security of Wireless Network Access with Enhancements. In: Proc. of the ACM Workshop on Wireless Security, pp. 88–95. ACM Press, New York (2003)
4. Huang, C., Li, J.: Authentication and Key Agreement Protocol for UMTS with Low Bandwidth Consumption. In: Proc. of the 19th IEEE Conf. on AINA, pp. 392–397. IEEE Computer Society Press, Los Alamitos (2005)
5. Meyer, U., Wetzel, S.: A Man-in-the-Middle Attack on UMTS. In: Proc. of the ACM Workshop on Wireless Security, pp. 90–97. ACM Press, New York (2004)
6. Kambourakis, G., Roukas, A., Gritzalis, S.: Performance Evaluation of Public key based Authentication in Future Mobile Communication Systems. EURASIP J. on Wireless Communications and Networking 2004(1), 184–197 (2004)