TK-AKA: Using Temporary Key on Authentication and Key Agreement Protocol on UMTS *

Hsia-Hung Ou[†] Iuon-Chang Lin[‡] Min-Shiang Hwang[‡§] Jinn-Ke Jan[†]

Department of Management Information Systems [‡]
National Chung Hsing University
250, Kuo Kuong Road,
Taichung, Taiwan 402, R.O.C.
Email: mshwang@nchu.edu.tw

Department of Computer Science and Engineering [†]
National Chung Hsing University
Taichung, Taiwan 402, R.O.C.

September 8, 2012

*This work was supported in part by Taiwan Information Security Center (TWISC), National Science Council under the Grants NSC 96-2219-E-001-001, and NSC 96-2219-E-009-013.

§Corresponding author.

TK-AKA: Using Temporary Key on Authentication and Key Agreement Protocol on UMTS

Abstract

Mobile communication is definitely one of the major high-technology applications that offer present-day people a modern life of appropriate convenience. In recent years, the third generation cell-phone system has approached maturity. The Universal Mobile Telecommunication System (UMTS) is currently the most widely used system around the world. The 3rd Generation Partnership Project is equipped with the Authentication and Key Agreement (AKA) protocol to maintain secret and secure both during the authentication process and during the telecommunication session through UMTS. However, Hung and Li have pointed out that the UMTS-AKA protocol has three problems involving the bandwidth consumption, space overhead, and synchronization of the sequence number that are yet to be resolved. In addition, they have proposed an extension of the UMTS-AKA protocol, named the X-AKA protocol, to overcome these obstacles. Nevertheless, the X-AKA protocol too appears to have problems of its own. In this article, the weaknesses of X-AKA are enumerated. In addition, a more practical AKA protocol for UMTS is presented. The new protocol, based on the same framework as its predecessor, proves to be more efficient and practical, satisfying the requirements of modern living.

Key Words: AKA, UMTS, Authentication, Mobile Communication

1 Introduction

A new generation of modern mobile communication has been recently developed. The Universal Mobile Telecommunication System (UMTS)[?] is the backbone of the next generation of mobile communication devices and is the mainstream that mobile-related technologies have to match in pace. Because the UMTS is based on the same framework that the currently widespread Global System for Mobile Communication (GSM)[**?**] system relies on, the individual specifications of both systems are compatible with each other. In other words, the original GSM consumers can still enjoy the services of the existing telecom companies when they upgrade their GSM units to the UMTS system.

1.1 3^{rd} Generation Partnership Project UMTS Authentication and Key Agreement Protocol

To achieve mutual authentication between the user and the network system, and to come another step closer to the goal of key coordination, the 3^{rd} Generation Partnership Project (3GPP) [**?**] has been included as an essential part of the UMTS-Authentication and Key Agreement (AKA) protocol [**?**]. The UMTS-AKA protocol is envisaged as a secret key shared between the Universal Subscriber Identity Module (USIM) and the Authentication Center (AuC) in the user's Home Environment (HE). In addition, the protocol combines a challenge/response method with a sequence number provided by ISO/IEC 9798-4 [**?**]. Figure 1 shows the method by which the 3GPP UMTS-AKA protocol operates, and Figure 2 defines the symbols that are used.

The user's Mobile Station (MS) sends his/her International Mobile Subscriber Identity (IMSI) to the Service Network (SN) when the user enters a new part of the service network. The SN then delivers this IMSI to the user's HE. After receipt of the IMSI, the HE can find the secret key 'K' shared between the IMSI's USIM and its AuC and further use it to calculate 'n' sets of Authentication Vectors (AVs) for the authentication and key establishment between the MS and the SN, to be carried out at Phase 2. If the user wishes to obtain some service provided by the SN, then both the parties have to follow the challenge/response approach and pass each other's check, wherein the SN chooses a set of parameters from the AVs on the basis of the sequence number (SQN) as the challenge and sends it to the corresponding MS. By using the secure key 'K' and the 'SQN', the MS first checks the accuracy

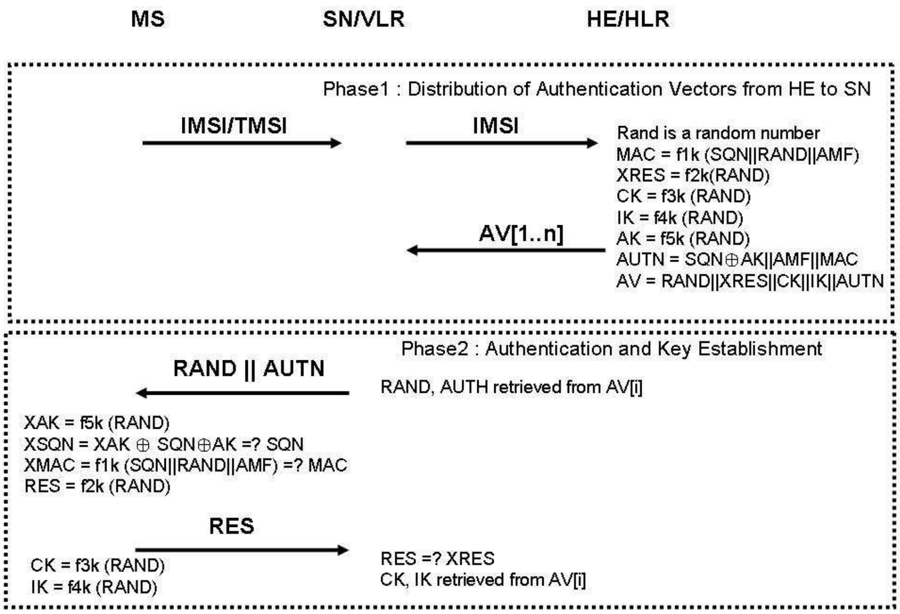

Figure 1: 3GPP UMTS AKA Protocol

Symbols

MS	: Mobile Station	**RAND**	: Random Number
SN	: Service Network	**MAC**	: Message Authentication Code
VLR	: Visitor Location Register	**SQN**	: Sequence number
HE	: Home Environment	**AMF**	: Authentication Management Field
HLR	: Home Location Register	**RES**	: User Response
AuC	: Authentication Centre	**XRES**	: Expected User Response
USIM	: Universal Subscriber Identity Module	**CK**	: Cipher Key
IMSI	: International Mobile Subscriber Identity	**IK**	: Integer Key
TMSI	: Temporary Mobile Subscriber Identity	**AK**	: Authentication Key
AV	: Authentication Vector	**f1~f5**	: Authentication and Key Generation Function
AUTN	: Authentication Token	**K**	: Secret Key which share between USIM and AuC

Figure 2: Symbols used in 3GPP UMTS AKA Protocol

of this parameter set and subsequently, after the confirmation, it calculates an apt response and sends it back to the SN. If the challenge is cleared without any error, the mutual authentication between the MS and the SN is said to be completed. However, the 3GPP-UMTS-AKA protocol has the following three problems that are yet to be solved[?]:

1. Bandwidth consumption: The HE must deliver the 'n' sets of AVs to the SN in the authentication-vector distribution phase. When many MSs ask for registration simultaneously, a great deal of bandwidth is necessarily consumed.

2. Space overhead: The SN has to store the 'n' sets of AVs for the HE when the MS enters its domain in the service network. If many MSs are located in the same part of the SN, a large amount of memory space is needed.

3. Sequence number synchronization: In the 3GPP-UMTS-AKA protocol, the MS, SN, and HE have their respective sequence numbers, and consistency has to be suitably maintained, which can be quite a problem because, in reality, network errors and equipment breakdowns can occur at any instant, thereby breaking the synchrony.

In an attempt to solve the above problems, Hung and Li have proposed an UMTS-X-AKA protocol [?] as follows.

1.2 UMTS X-AKA Protocol

To circumvent the problems of bandwidth consumption, space overhead, and sequence-number synchronization that the 3GPP-UMTS -AKA protocol possesses, Hung and Li have substituted a temporary key mechanism for the sequence-number determination. Thus, the SN has the ability to authorize the MS and the SN no longer needs to store the authentication vectors on its own part of the network. As a result, this X-AKA protocol solves the three problems that are associated with the 3GPP-AKA protocol. Figure 3 illustrates the design of the X-AKA protocol.

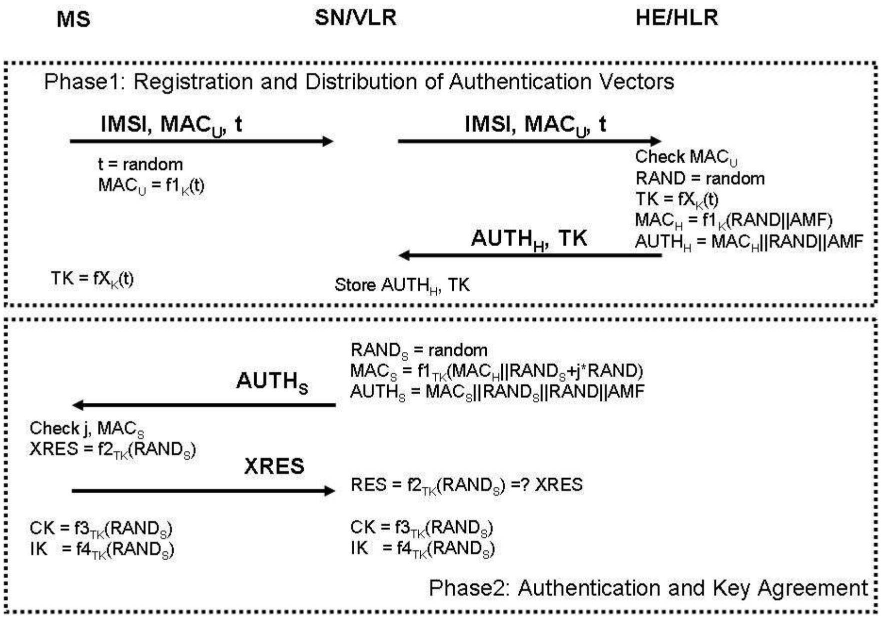

Figure 3: The X-AKA Protocol

Similar to the 3GPP-UMTS-AKA protocol, the X-AKA protocol is composed of two phases. Phase 1 registers and distributes the authentication vectors when the MS arrives at a new Visitor Location Registry (VLR) coverage. Phase 2 carries out the AKA when the MS remains within the same region. In Phase 1, the SN can get a temporary key, TK, from its HE. The TK subsequently carries out the AKA with the MS in Phase 2.

To produce the TK, the MS in Phase 1 sends its IMSI, a random number t and its Message Authentication Code (MAC_U)[?, ?] MAC_u through the SN to its HE. The HE checks both the t and MAC_U to verify the identity of the MS using the secret key K shared with the MS. If the confirmation is positive, then the HE computes the value for $TK = fX_k(t)$ and sends it back to the SN along with other necessary parameters. When the SN receives this TK and the initial parameters, it becomes ready to execute the AKA with the MS later in the second phase.

In Phase 2, the SN generates $AUTH_S = MAC_S||RAND_S||RAND||AMF$ as the challenge to the MS, where MAC_S can be derived as $MAC_S = f1_{TK}(MAC_H||RAND_S+$

$j \times RAND$) where $RAND_S$ is a random number and j is the time of authentication. $RAND$ and MAC_H can be retrieved from Phase 1. Upon receiving the $AUTH_S$ from the SN, the MS verifies MAC_S and authenticates the SN. If everything proceeds smoothly, the MS computes the response as $XRES = f2_{TK}(RAND_S)$ and then sends it back to the SN. The SN verifies $XRES$ and authenticates the MS. After the mutual authentication between the MS and the SN is completed, the cipher key (CK) and integer key (IK) can be computed as $CK = f3_{TK}(RAND_S)$ and $IK = f4_{TK}(RAND_S)$.

The X-AKA protocol thus mends the defects of the original AKA protocol. However, it is nonetheless a few steps away from perfection.

The following list describes the points which demand improvement in the X-AKA protocol.

1. The temporary key generation function, fx, is a creation of Hung and Li; that is, the function cannot be found among the 3GPP standard functions. In contrast to the current UMTS system, this temporary key generation function can cause some compatibility problems for the X-AKA protocol in real-life applications.

2. The use of the message authentication code function $f1$ is tricky. The $f1$ function is a component of the 3GPP standards [?, ?, ?]; it takes in four input parameters $K, RAND, SQN$, and AMF, whose numbers of bits are $128, 128, 48$, and 16 respectively, and gives an output. However, in the X-AKA protocol, the f1 function is involved in the derivations of MAC_U, MAC_H, and MAC_S. There are numerous constituents whereas not enough input parameters are available. Therefore, a compatibility problem is presented herein.

3. In some parts of the X-AKA protocol the use of the MAC is not necessary. For example, in Phase 1, the MS uses MAC_U to prove that the random number t is its own creation; however, this is a redundant procedure because a false random number t cannot possibly pass the identification in Phase 2.

4. The capabilities of the protocols are not effective. Comparing the X-AKA protocol with the UMTS-AKA protocol with reference to the computational, storage, and communication overheads, the capabilities are observed to be not good enough. These are explained in detail in section 3. In addition, a more efficient and practical protocol has been proposed in the section 2.

To mend the flaws in both the X-AKA protocol and the original AKA protocol, an improved version is proposed in this article, called the TK-AKA protocol (Temporary Key-AKA Protocol). Similar to its predecessors, this new protocol is based on the temporary key concept, but it is capable of providing better efficiency with increased simplicity. In the next section, the details of the TK-AKA protocol are presented along with some comparisons and advantages that the new protocol provides.

2 The Solution: TK-AKA Protocol

When the design of the TK-AKA protocol was contemplated, the first idea that was considered was the shifting of the computing overhead from the HE's side onto the SN's side. The use of the temporary key is retained in the X-AKA protocol, but the steps have been simplified, with unnecessary steps being omitted, and an encryption function freshly included in the 3GPP specifications is used to encrypt everything. The purpose of the TK-AKA protocol is not only to revise the programs of UMTS-AKA but also, more importantly, to make practical improvements so that the new protocol is perfectly compatible with the current UMTS system.

Similar to its predecessors, the TK-AKA protocol also has two phases: Phase 1 encompasses the registration and distribution of authentication vectors, and Phase 2 deals with the Authentication and Key Agreement. In addition, key generation functions $f2$, $f3$, and $f4$, all of which are covered by the 3GPP specifications [**?**, **?**, **?**], are efficiently used. The challenge/response authentication method uses the user-authentication function, $f2$ Putting in a random number as a challenge, the user gets a response produced using the $f2$ function, along with a security key (or temporary key). $f3$ and $f4$ are the cipher -key derivation function and the integrity-key derivation function, respectively. With a random number keyed in, the cipher key or integrity key can be obtained as a product of the $f3$ or $f4$ function, along with a security key (or temporary key). Moreover, the proposed TK-AKA protocol also uses the $f4$ function to develop its temporary key and the secret process of sharing keys.

The rationale behind the mechanism of TK-AKA is to enable the HE to authenticate and issue a specific SN a temporary key when an MS gets into the service coverage of that SN. As long as the MS stays within the service coverage of the same SN, the SN and the MS can use this temporary key to authenticate each other. It should be also noted that, in the TK-AKA protocol, to prevent any damage caused by the replay attack, the random number used during the previous connection is considered while producing the new challenge the next round of connection. The benefit of this approach is that an attacker intercepting a message during the user authentication cannot be used again in the next authentication. Because the attacker does not have the TK that the user shares with the MS and SN, it is unable to intercept the $challenge_i$ of the previous usage to calculate the $challenge_{i+1}$ used for authentication the next time. Moreover, the relation and order of $challenge_i$ and $Challenge_{i+1}$ can help the MS to distinguish the freshness of the challenge. Figure 4 illustrates the details of TK-AKA protocol.

2.1 The Proposed Protocol

The details of the phases in TK-AKA protocol are as follows.

Phase 1: Registration and Distribution of Authentication Vectors

1. MS randomly generates a 128-bit number as a *seed* and transmits it along with the $IMSI$ through the SN to the HE. The MS can conveniently calculate $TK = f4_K(Seed)$ and then store it along with the seed.

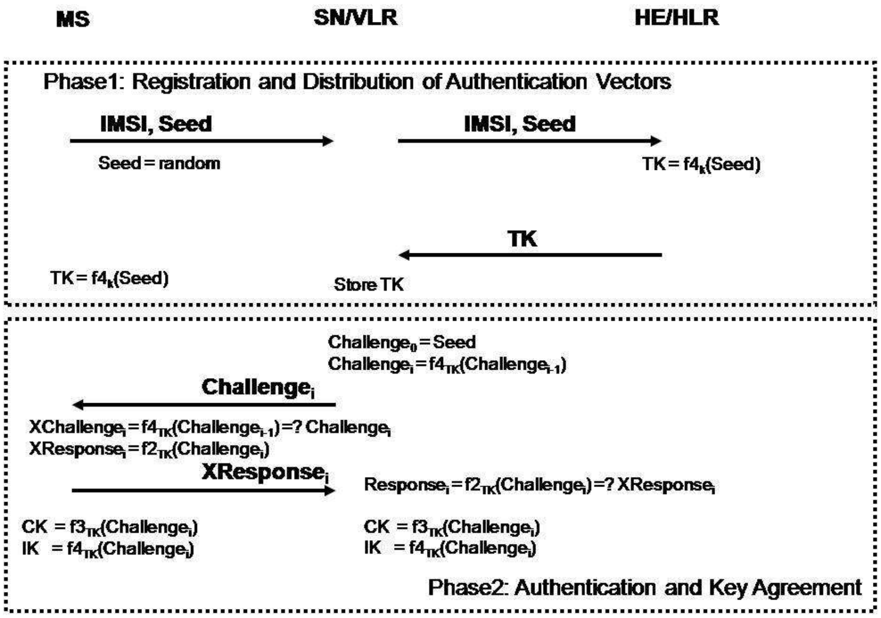

Figure 4: The TK-AKA Protocol

2. The HE calculates the temporary key $TK = f4_K(Seed)$, where K is a secret key previously shared between the MS and HE. Subsequently, TK is sent back to SN. Note that HE does not need to store it.

3. SN receives the *seed* as the first *Challenge* and then stores it along with TK for the authentication in Phase 2.

In this phase, similar to the UMTS-AKA protocol, $IMIS$, *Seed*, and TK send in the clear text between SN and HE. In This step is secure because SN and HE are connected through a special wired network such as SS7[?]. SS7 (Signaling System 7) is the architecture for conducting out-of-band signaling in support of the call-establishment, billing, routing, and information-exchange functions of the public-switched telephone network (PSTN). It identifies functions to be carried out by a signaling-system network and a protocol that enables their efficient performance and enhanced security[?]. Moreover, unlike the MS and SN within a wireless network, a wired network is positioned between the MS and SN and is therefore not vulnerable to eavesdropping and interception.

Phase2: Authentication and Key Agreement procedure

1. SN calculates $Challenge_i = f4_{TK}(Challenge_{i-1})$ as a new challenge and sends it to the MS during a connection request.

2. MS calculates $XChallenge_i = f4_{TK}(Challenge_{i-1})$ and $XResponse_i = f2_{TK}(Challenge_i)$. MS compares the $XChallenge_i$ with $Challenge_i$ and ensure that $Challenge_i$ is a fresh challenge. If the result is positive, MS sends $XResponse$ as a response to SN; otherwise, this connection is brought to a stop.

3. SN calculates $Responsei = f2_{TK}(Challenge_i)$ and compares $Response_i$ with $XResponse_i$. If the result is positive, then the MS is considered legal; otherwise, the connection is cut off.

4. If both parties have confirmed the identity of each other without an error, both calculate the cipher key $CK = f3_{TK}(Challenge_i)$ and the integrity key $IK = f4_{TK}(Challenge_i)$ for this connection. Subsequently, they store $Challenge_i$ to replace $Challenge_{i-1}$ for the next connection.

Moreover, mutual authentication can be achieved during this phase. Because SN sends a $Challenge_i$ to the MS, it is proved to be legal, and MS returns a $XResponse_i$ that proves itself. $Challenge_i$ is derived from $f4_{TK}(Challenge_{i-1})$, where $Challenge_{i-1}$ is the challenge used in the previous authentication to ensure the freshness of this session, and TK is the temporary key. The temporary key is made up from the secure key that the MS and SN have shared previously, which can confirm whether the SN has the authorization in relation to the Home Network (HN).

2.2 Discussions and Analyses

As mentioned earlier, bandwidth consumption, space overhead, and sequence- number synchronization are the three problems that the UMTS-AKA protocol presents.

In this section, the circumvention of these problems by the proposed TK-AKA protocol is elaborated.

1. Bandwidth consumption: In the original UMTS-AKA protocol, the HE must compute and deliver 'n' sets of AVs . Even in the X-AKA protocol, the delivery of a temporary key, TK, and an authentication challenge, $AUTH_H$, is indispensable. In the TK-AKA protocol, to reduce the bandwidth consumption, only a temporary key TK needs to be computed and delivered. Furthermore, the TK-AKA protocol reduces the computation overhead on the HE's side. Compared with UMTS-AKA and X-AKA, TK-AKA has lower communication overhead in every component of the whole process.

2. Space overhead: As noted in point 1, the SN must store the 'n' sets of AVs in the UMTS-AKA protocol, and the X-AKA protocol also requires the SN to store the details corresponding to $(AUTH_H, TK)$. By contrast, in the TK-AKA protocol, the only thing that has to be stored is the TK. Obviously, the TK-AKA protocol has the smallest space overhead.

3. Sequence-number synchronization: In the 3GPP-UMTS-AKA protocol, each of the components-MS, SN, and HE- has to keep a SQN to ensure that the authentication vector is fresh. Such a design easily leads to consistency problems in real-life applications because network errors and equipment breakdowns can occur at any instant and break the synchrony. To avoid such a situation, the counter j is used between the MS and SN in the X-AKA protocol. Therefore, only the synchrony between two parties has to be considered, instead of considering three components. Moreover, the authentication challenge '$AUTH_S$' of the UMTS-AKA protocol is composed of $AUTH_H$, a random number, and j, which is too complicated. In the current TK-AKA protocol, the random number used during the previous connection $RAND_{i-1}$ and the temporary key TK are used to generate the random number for this connection, $RAND_i$, by

using the key-generation function. In this method, the SQN-related synchrony problem generated by the X-AKA protocol can be reduced drastically without any loss of security.

In addition, the three problems associated with the UMTS -AKA protocol are neatly solved as the discussions above indicate. There are nevertheless three weaknesses in the X-AKA protocol: the compatibility problem caused by the use of the alien-to-the system temporary key-generation function fx, the misuse of the message authentication-code function $f1$, and the redundant use of MAC. However, the TK-AKA protocol can mend these weaknesses of the X-AKA protocol as follows:

1. Unlike the X-AKA protocol that uses a temporary key-generation functionfx alien to the 3GPP specifications, all the encryption functions included in the TK-AKA protocol are ingredients of the 3GPP specifications. This feature implies that the TK-AKA protocol can be readily applied in the current UMTS environment.

2. To avoid the misuse of the message authentication-code function $f1$, the TK-AKA protocol is very carefully designed so that its every detail is compatible with the 3GPP specifications.

3. Unnecessary use of MAC is avoided. The TK-AKA protocol follows the simplest route to the same purpose, with no redundancies in its application.

4. A more efficient and practical protocol, TK-AKA, has been proposed in this article. Some related comparisons are presented in the next section.

From the discussions, it can be inferred that the TK-AKA protocol can solve the problems encountered in the current UMTS-AKA protocol, and that it is capable of performing better as it is free from the defects of the X-AKA protocol.

	UMTS-AKA	X-AKA	TK-AKA
Comparison of Computational Overhead on Phase 1			
MS	None	G+f1+fk	G+f4
SN	None	None	None
HE	(G+f1+f2+f3+f4+f5)×n	G+f1×2+fk	f4
Comparison of Computational Overhead on Phase 2			
MS	f1+f2+f3+f4+f5	f1×2+f2+f3+f4	f2+f3+f4×2
SN	None	G+f1+f2+f3+f4	f2+f3+f4×2
HE	None	None	None
Comparison of Storage			
MS	SQN=48	t+TK+j=258+j	TK+RAND$_{i-1}$=256
SN	AV×n=608×n	AUTH$_H$+TK+j=368+j	TK+RAND$_{i-1}$=256
HE	SQN=48	None	None
Comparison of Communication Overhead on Phase 1			
MS→SN	128	320	256
SN→HE	128	320	256
HE→SN	560×n	368	128
Subtotal	256+560×n	1008	640
Comparison of Communication Overhead on Phase 2			
SN→MS	240	368	128
MS→SN	64	64	64
Subtotal	304	432	192

Figure 5: Capability comparisons

3 Comparisons

The X-AKA scheme has already been compared with a number of other related protocols, and its performance has been proven better (Harn&Hsia [?], AP-AKA [?]). The X-AKA scheme is superior because of the following features: (i) it supports mutual authentication between the MS and the SN; (ii) it provides user-traffic confidentiality; (iii) it offers signal integrity; (iv) it reduces the bandwidth consumption between the SN and the HN; and (v) it alleviates the storage space overhead for the SN's database. However, synchronization has to be derived between the MS and the HN. To prove that the TK-AKA scheme is capable of outperforming the X-AKA method, the performances of UMTS-AKA, X-AKA, and the proposed TK-AKA are compared in terms of the following criteria. Figure 5 shows a summary of these comparisons.

1. Computation Overhead: To fit the actual application environment, all AKA protocols are designed to be divided into two phases. Phase 1 is usually limited to the process of registration and authentication-vector distribution when the mobile user arrives in a new SN's coverage, whereas Phase 2 is usually reserved for the establishment of authentication and key when the mobile user accesses the communication services. In other words, Phase 2 may be executed many times after the completion of the registration in Phase 1. To better understand the messages carried by Figure 5, a closer look at the principles of designing the AKA protocols is attempted. At the outset, the original UMTS-AKA protocol holds Phase 1 responsible for the computation of the authentication vectors, which can thus be maintained ready for future use when Phase 2 is reached. Such a design can have adverse results, as the computation overheads are concentrated on the HE's side in Phase 1, and, in Phase 2, the MS has to bear all the overheads. Except for these two parties, others involved in the protocol do not deal with computation. Thus, in this method, the HE has to take an extremely heavy load because it has to serve a lot of MS. If too many registration requirements are requested at the same time, the HE may be over flooded and can crash. The TK-AKA protocol, designed to be more similar to the X-AKA protocol, shares the burden of the HE . Therefore, the computation overhead in Phase 1 in both protocols is reduced but that in Phase 2 increases a little. In addition, there is one parameter in which these AKA protocols differ. In the original UMTS-AKA protocol, the authentication vectors created in Phase 1 have a constant period of validity; in other words, the MS and the HE have to re-run the registration procedure in Phase 1 to obtain new authentication vectors after, say,n times of running the authentication procedure in Phase 2. On the contrary, TK-AKA uses the concept of authorization, similar to X-AKA. After the SN gets the authorization from the HE, the MS can stay authenticated by the SN until the preset threshold is reached, if there

is one. Thus, TK-AKA and X-AKA can reduce computation overhead to some extent. In the UMTS-AKA protocol, the authentication vectors are processed in beforehand, whereas they are computed only on demand in both TK-AKA and X-AKA protocols. According to the results of the comparison in Figure 5, TK-AKA has a lower computation overhead than the other two protocols, and it is lower in both Phase 1 and Phase 2. G, in Figure 5, represents the random number generation function.

2. Storage: The space overhead is also an important factor to be considered while evaluating the AKA protocols. In the UMTS-AKA protocol, the space overhead is concentrated on the SN, whereas it is uniformly shared between the MS and the SN in the other two protocols. The results in Figure 5 show that the TK-AKA protocol consumes the least total-storage space. The bit numbers adopted in Figure 5 are derived from the Figure 6 [**?**].

3. Communication Overhead: Compared with the computational overhead of the other protocols, the proposed TK-AKA protocol has identical characteristics. The UMTS-AKA protocol is responsible for the calculation of all the authentication vectors during the phase 1; therefore, it needs more communication overhead in the stage where HE transmits the authentication vectors to the SN. On the contrary, X-AKA and TK-AKA do not need this procedure; hence, the communication overhead is relatively low. The variable 'n' in $(560 \times n)$ implies the number of authentication vectors involved; therefore, the communication overhead contains larger numbers than the other two protocols. Considering that the other two protocols need more communication overhead to obtain authorization in the stage of communication between the SN and HE, the increased communication overhead of the proposed protocol is cost-effective; it can be easily understood by observing the subtotal of the phase 1 in the Figure 5. Moreover, TK-AKA is superior to X-AKA. In phase 2, TK-AKA access is a comprehensive victory compared to the other two protocols. The

TK-AKA is more outstanding than the X-AKA because the TK-AKA uses a more streamlined process to satisfy the same demands. Obviously, in this aspect, as Figure 5 indicates, the TK-AKA protocol also wins convincingly.

4. Every fragment of the TK-AKA protocol is compatible with the 3GPP specifications. Therefore, TK-AKA is as secure as the UMTS-AKA protocol. Moreover, it avoids the SQN synchronization problem that is found while applying the X-AKA protocol. In the UMTS-AKA protocol, the authentication of the MS is carried out in a method different from that of the SN. The SN uses the authentication vector computed by the HE much in advance to carry out authentication with the MS, but the MS confirms the accuracy of the authentication vector and not the SN. Thus, the authentication vectors created in advance have to be delivered to another fresh SN if the MS roams across the border into its coverage area, which procedure increases the risk of authentication-vector leakages. In contrast, such a security problem does not exist in the TK-AKA protocol; the reason being that the new protocol uses a temporary-key mechanism. Under such a design, the MS has to re-register and the temporary key is re-established whenever the MS roams into another SN. The computation overhead of TK-AKA in Phase 1 (Registration) is therefore lower than that of UMTS-AKA, but the security of the TK-AKA protocol is more robust.

5. The X-AKA protocol uses a key generation function that is alien to the original system and thus may have compatibility problems in real-time applications. Unlike the X-AKA, the TK-AKA protocol is applicable along with the systems currently in use and is therefore far more practical.

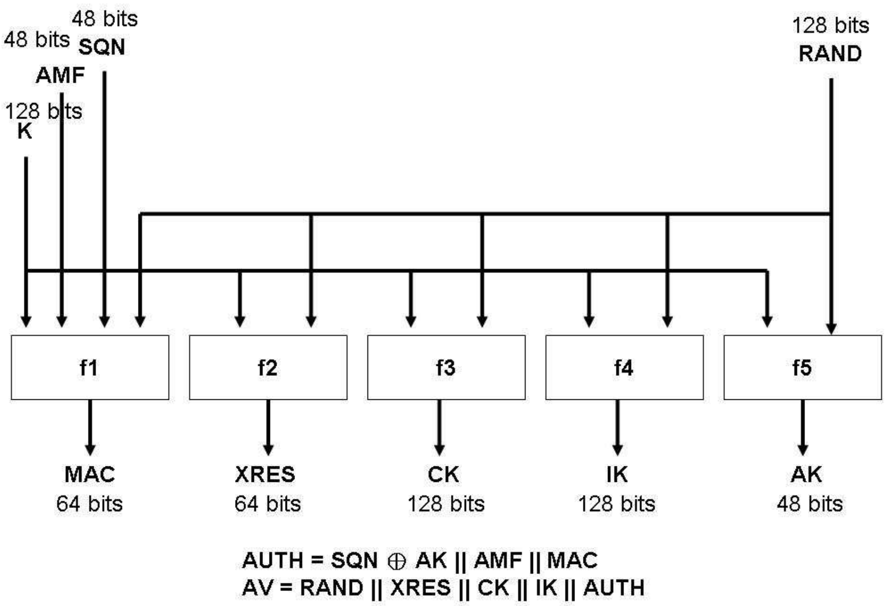

Figure 6: Generation functions and their bit numbers

4 Conclusion

UMTS is now the mainstream communication technology, but just as every new technology, it is subject to alterations, improvements, additions, and even replacement. To improve the original UMTS-AKA protocol, the herein proposed TK-AKA protocol uses a temporary-key mechanism similar to X-AKA, but the design is different. The difference between the two makes the TK-AKA protocol capable of not only amending the major defects of the original UMTS -AKA protocol but also solving the problems associated with the X-AKA protocol. The TK-AKA protocol is thus superior to X-AKA in every aspect-computation, storage and communication overheads, security, and practicability. Better communication services can be expected to be developed soon in the future based on this TK-AKA protocol.

Appendix

The Formal Analysis

This section states why the TK-AKA protocol can reach the goals of authentication and key agreement. The formal method BAN logic[?] is utilized to analyze this protocol. BAN logic is one of the authentication logics to analysis cryptographic protocols, authentication protocol especially. It identifies the messages with statements in the many-sorted modal logic. BAN-Logic proposed connective by a comma conjunction and take for granted properties such as associatively and commutatively. In addition to follow-up statements we simply display the symbol of BAN Logic[?]:

- P$|\equiv$X: P believes X, or P would be entitled to believe X.
- p$\triangleleft$X: P sees X. Someone has sent a message containing X to P, who can read and repeat X.
- P$|\sim$X: P once said X. P at some time sent a message including the statement X.
- P$|\Rightarrow$X: P has jurisdiction over X. P is an authority on X and should be trusted on this matter.
- $\sharp(X)$: The formula X is fresh, that is, X has not been sent in a message at any time before the current run of the protocol.
- P$\overset{K}{\leftrightarrow}$Q: P and Q may use the shared key K to communicate.
- P$\overset{K}{\rightleftharpoons}$Q: The formula X is a secret known only to P and Q.
- $(X)_Y$: This represents X combined with the formula Y that Y be a secret.

A formal analysis is as follows:

1. The proposed protocol

- (Message 1) $MS \rightarrow SN \rightarrow HE : IMSI, Seed$
- (Message 2) $HE \rightarrow SN : IMSI, PK$
- (Message 3) $SN \rightarrow MS : Challenge_i$
- (Message 4) $MS \rightarrow SN : XResponse_i$

The relation parameters as following:

(a) $Seed$: A random number.

(b) PK: $f4_K(Seed)$; K is a secure key which is shared between the MS's USIM and the MS's HE. The $f2, f3$ and $f4$ are the key generating functions.

(c) $Challenge_i$: $f4_{PK}(Challenge_{i-1})$; $Challenge_0 = Seed$.

(d) $XResponse_i$: $f2_{PK}(Challenge_i)$.

(e) CK: $f3_{PK}(Challenge_i)$; A cipher key.

(f) IK: $f4_{PK}(Challenge_i)$; A Integrity Key.

2. Security Assumptions

(a) It is assumed that K is a secure key which is share between the MS and his HE.

(a) MS has the secure key K

(b) HE has the secure key K

(c) MS$|\equiv$ MS$\overset{K}{\rightleftarrows}$ HE

(d) HE$|\equiv$ MS$\overset{K}{\rightleftarrows}$ HE

(b) It is assumed that the trusting relationship between HE and SN.

(a) SN$|\equiv$ HE$|\Rightarrow$ MS$\overset{K}{\rightleftarrows}$ HE

(c) It is assumed that the communication between HE and SN is secure.

(a) SN$|\equiv$ SN$\overset{P}{\rightleftarrows}$ HE, P is the conveyance messages between SN and HE.

(b) HE$|\equiv$ SN$\overset{P}{\rightleftarrows}$ HE, P is the conveyance messages between SN and HE

3. Formally messages

(a) (Message 1) $MS \rightarrow SN \rightarrow HE : IMSI, Seed$

(b) (Message 2) $HE \rightarrow SN : IMSI, f4(K, Seed)$

(c) (Message 3) $SN \rightarrow MS : f4(f4(K, Seed), Challenge_{i-1})$

(d) (Message 4) $MS \rightarrow SN : f2(f4(K, Seed), Challenge_i)$

4. Protocol goals

(a) Mutual authentication between MS and SN.

(b) Key agreement between MS and SN.

(c) Key freshness between MS and SN.

(d) Confidentiality between MS and SN.

5. Statements and analysis

(a) (Goal 4.a) Mutual authentication between MS and SN.

(1) (3.a)(2.c.2)$\rightarrow$ $MS \mid\equiv \sharp(Seed) \wedge SN \mid\equiv \sharp(Seed) \wedge HE \mid\equiv \sharp(Seed)$

(2) Since (2.a.2), (2.a.4), (2.b.1), (2.c.1). By (3.b)$\rightarrow$ $SN \mid\equiv \forall f4(K, Seed).(HE \mid\Rightarrow SN \overset{f4(K,Seed)}{\leftrightarrow} MS)$

(3) Since (2.a.1), (2.a.3) and (5.a.1)$\rightarrow$ $MS \mid\equiv MS \overset{f4(K,Seed)}{\leftrightarrow} HE$

(4) For message-meaning rule and (3.c)

$$\frac{MS \mid\equiv MS \overset{f4(K,Seed)}{\leftrightarrow} HE, MS \triangleleft f4(f4(K, Seed), Challenge_{i-1}}{MS \mid\equiv HE \mid\sim f4(f4(K, Seed), Challenge_{i-1}}$$

(5) For nonce-verification rule, (1.c) and (5.a.4)

$$\frac{MS \mid\equiv \sharp(Challenge_{i-1}), MS \mid\equiv HE \mid\sim f4(f4(K, Seed), Challenge_{i-1}}{MS \mid\equiv HE \mid\equiv f4(f4(K, Seed), Challenge_{i-1})}$$

(6) For jurisdiction rule and (5.a.5)

$$\frac{MS|\equiv HE|\Rightarrow f4(f4(K,Seed),Challenge_{i-1}),MS\triangleleft SN|\sim f4(f4(K,Seed),(Challenge_{i-1})}{MS \mid\equiv HE \mid\equiv SN}$$

(7) For message-meaning rule (5.a.2) and (3.d)
$$\frac{SN \mid\equiv SN \overset{f4(K,Seed)}{\leftrightarrow} MS, SN \lhd f4(f4(K,Seed), Challenge_i}{SN \mid\equiv MS \mid\sim f2(f4(K,Seed), Challenge_i}$$

(8) For nonce-verification rule, (1.b), (1.d), (1.e) and (5.a.7)
$$\frac{SN \mid\equiv \sharp(Challenge_i), SN \mid\equiv MS \mid\sim f2(f4(K,Seed), Challenge_i}{SN \mid\equiv MS \mid\equiv f2(f4(K,Seed), Challenge_i)}$$

(9) For jurisdiction rule and (5.a.8)
$$\frac{SN|\equiv MS|\Rightarrow f2(f4(K,Seed),Challenge_i), SN\lhd MS|\sim f2(f4(K,Seed),(Challenge_i)}{SN \mid\equiv MS}$$

(10) By (5.a.6), (5.a.9) $\rightarrow MS \mid\equiv HE \mid\equiv SN \wedge SN \mid\equiv MS \rightarrow MS \mid\equiv SN \wedge SN \mid\equiv MS$, So, the goal of mutual authentication between MS and SN is holds.

(b) (Goal 4.b) Key agreement between MS and SN.

(1) There are two keys to agreement between MS and SN: CK and IK.

(2) (1.e)$\rightarrow CK = f3(PK, Challenge_i) = f3(f4(K,Seed), Challenge_i)$

(3) Since (5.a.1), (5.a.2), (5.a.3)$\rightarrow SN \mid\equiv (SN \overset{f4(K,Seed)}{\leftrightarrow} MS) \wedge MS \mid\equiv (SN \overset{f4(K,Seed)}{\leftrightarrow} MS)$

(4) (3.c), (1.c)$\rightarrow SN \mid\equiv (SN \mid\sim (Challenge_i) \wedge \sharp(Challenge_i))$

(5) (3.d), (5.a.8)$\rightarrow SN \mid\equiv (MS \mid\equiv \sharp(Challenge_i))$

(6) (3.c), (1.c)$\rightarrow MS \mid\equiv (MS \lhd Challehge_i \wedge SN \mid\sim Challenge_i \wedge \sharp(Challenge_i))$

(7) By (5.b.3), (5.b.5), (5.b.6) $\rightarrow$ CK agreement between MS and SN

(8) (1.f)$\rightarrow IK = f4(PK, Challenge_i) = f4(f4(K,Seed), Challenge_i)$

(9) As (5.b.7)$\rightarrow$ IK agreement between MS and SN.

(10) By (5.b.7), (5.b.9)$\rightarrow$ the goal of key agreement between MS and SN is holds.

(c) (Goal 4.c) Key freshness between MS and SN.

(1) Since (5.a.1), (5.b.3), (5.b.4), (5.b.6)$\rightarrow MS \mid\equiv (\sharp(PK) \wedge \sharp(CK) \wedge \sharp(IK))$. $\rightarrow$ So, the goal of key freshness between MS and SN is hold.

(d) (Goal 4.d) Confidentiality between MS and SN.

(1) Since (5.a), (5.b), (5.c), for message-meaning rule→

$$\frac{MS|\equiv MS(\overset{CK\wedge IK}{\leftrightarrow}SN), MS\lhd(Message)_{(CK\wedge IK)}}{MS|\equiv SN|\sim Message} \wedge \frac{SN|\equiv MS(\overset{CK\wedge IK}{\leftrightarrow}SN), SN\lhd(Message)_{(CK\wedge IK)}}{SN|\equiv MS|\sim Message}$$

(2) By (5.d.1)→ the goal of confidentiality between MS and SN is hold.

Since (5.a.10), (5.b.10), (5.c.1), (5.d.2), all the protocol goals is hold.