A Cocktail Protocol with the Authentication and Key Agreement on the UMTS *

Hsia-Hung Ou[‡] 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.

August 19, 2009

*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: Prof. Min-Shiang Hwang.

A Cocktail Protocol with the Authentication and Key Agreement on the UMTS

Abstract

At present, the Universal Mobile Telecommunications System (UMTS) is very popular in most parts of the world. It is a third-generation mobile communication technique known for its ability to conduct user authentication and for its security of communication with the use of Authentication and Key Agreement (AKA) protocol. A mobile station (MS), a service network (SN) and a home environment (HE) use the protocol to authenticate each other and make an agreement with a session key. With the UMTS-AKA protocol standard, all authentication vectors (AV) produced by the HE are transferred to the SN for mutual authentication with the MS. In this scenario, authentication is exposed to two kinds of defects. One defect is computational overhead concentrating on the HE and the other is the communication overhead for delivering the AVs. To overcome these congenital defects, this study proposes a unique UMTS-AKA protocol called the cocktail-AKA protocol. The goal of this protocol is to allow the SN to share some medicated authentication vectors (MAV) that are calculated in advance and combined with a prescription at the authentication stage. So, the HE only needs to produce a prescription authentication vector (PAV). Once the authentication stage is initiated, the SN distributes MAV and PAV and produces an effective AV for mutual authentication with the MS. The cocktail-AKA protocol can overcome both the aforesaid defects.

Key Words: AKA, authentication, Key Agreement, UMTS

1 Introduction

Cell phones are a significant innovations of the present-day world. The ease of carriage has made cell phones an essential part of the personal outfit in modern lifestyle.

Currently, the Global System for Mobile communication (GSM) [12] and the Universal Mobile Telecommunications System (UMTS) [1] are the technologies most widely used world over. Presently, a third-generation (3G) cell phone technology, based on UMTS, which originated from GSM, is becoming popular to gradually replace GSM. The difference between cell phones and telephones is in their convenience. Cellular phones allow constant communication from virtually anyplace covered by the network of a base station (BS). A BS or service network (SN) is a stationary facility that receives and sends them telephonic signals between users. It acts as a medium between cellular phones and the home SN. Each BS covers an exclusive service region with a specific range of signals. However, it is possible for a mobile subscriber to move out of one's home BS and enter into the service region of another BS. In such circumstances, communication is possible by two ways: the subscriber has moved into the service region of the same SN, or the subscriber has moved into the service region of a third-party SN who is in partnership with the home SN. When a subscriber moves from one SN to another, he must prove his identity to both service providers and establish himself as a privileged user.

The UMTS's Authentication and Key Agreement (AKA) protocol [2] was proposed by the 3GPP (3rd Generation Partnership Project) [1] for this purpose. The authentication foundation of the AKA protocol comes into action when the subscriber's mobile station (MS) and the home environment (HE) enter into a secure key (K) agreement. In a practical situation, the MS is usually unable to establish HE authentication directly and must resort to another SN. The MS moves the authentication call with the SN, which ultimately obtains HE authentication. For this to happen, the SN must belong to the same network as the MS or to other networks with a partnership. This means that the SN and the HE must have prior security association. In the UMTS-AKA protocol, the SN should obtain authentication vectors (AV) from the HE of the MS when the MS moves into the service region of the SN and requests service. The HE produces all AVs and relinquishes them

to the SN for mutual authentication with the MS. In the step, the HE calculates n-sets of AVs and then relinquishes the results to the SN. The SN uses this data to authenticate n-times with the MS. Generally, the SN uses one set of AV a time to facilitate the authentication of the MS. The following challenge/response method is being adopted in the UMTS-AKA protocol. First, the SN sends some data as a challenge to the MS, which than verifies the correctness of the data and calculates a reply response. Next, the SN compares the response of the MS with the parameters in the AV to verify the identity of the MS. Lastly, the MS calculates the relational encryption key from the challenge, and the SN deciphers the key from the AV. In this scenario, authentication is exposed to two kinds of congenital defects. One is computational overhead concentrating on the HE. The other is communication overhead for delivering the AVs.

This study proposes an innovative UMTS-AKA protocol, called the cocktail protocol, to solve the aforesaid defects. In this protocol, the SN shares some AVs (as the medicated authentication vectors, MAV) calculated in advance and combines them with the instructions at the authentication stage. The HE simply needs to produce an instruction authentication vector (as the prescription authentication vectors, PAV). When the authentication stage is initiated, the SN dispenses these two AV's and produces an effective authentication vector for mutual authentication with the MS. The cocktail protocol can reduce both computational overhead concentrating on the HE and communication overhead for delivering the AVs.

The remainder of this article is organized as follows. Section 2 introduces the concepts of UMTS-AKA protocol, reviews the literature, and points out the scope for its improvements. Section 3 presents the cocktail protocol as an improvement over the UMTS-AKA protocol. Section 4 provides relevant discussion and analysis. Finally, Section 5 presents the conclusion.

2 Related works

Many of research efforts have been directed to investigate the UMTS AKA protocol, with a keen focus on security enhancements. A great deal of improvements has been studied to the cryptographic technology [10, 11, 15, 16, 18, 19, 20, 21]. These documents used symmetric cryptography, asymmetric cryptography or other methods to improve security. Given the outstanding characteristics of cryptography, it can be employed to reinforce the security of the UMTS-AKA protocol; however, the main disadvantage with it is higher computation overhead. Generally, the efficiency of mobile phones is limited by the complexity of computation. Although some high-priced phones can fulfill this demand, one should assimilate the fact that excessive equipment requirements can limit the popularity of the UMTS technology.

In 2003, Harn and Hsin used the concept of hash chain and MAC (message authentication code) to design an ER-AKA protocol [13], which is expected to enhance the security of the original UMTS-AKA protocol. However, the protocol has greatly increased space overhead and communication overhead in the storage and transmission of hash chain. In 2005, Zhang and Fang cited some failures in the UMTS-AKA protocol, such as redirection attack and active attack in corrupted networks. In view of this, AP-AKA [22] was introduced to meet higher security requirements. The newer protocol offered a solution of combining the service network's identity within the authentication token to prevent redirection attack and active attack in corrupted networks. However, it adopts a way similar as the original UMTS-AKA protocol to inherit the original shortcoming, that is, computation overhead and communication overhead in calculating and transmitting AVs. In 2005, Huang and Li introduced the X-AKA protocol [14] to overcome the program of low bandwidth consumption. In this protocol, the MS's HE distributes a TK (temporary key) to SN. The SN uses a TK to carry out mutual authentication between the SN and the MS. This is different from the original method in that the HE calculates n-sets of AV and passes them to the SN. A TK is much smaller than n-sets of AV, and therefore

it can save bandwidth consumption. In fact, the approach of this protocol is to transfer the HE's computation overhead to the SN, so as to reduce communication overhead. However, a closer look at the design of UMTS-AKA protocol reveals that the SN almost makes no computation expect that it compares the messages. It can speculate that the SN may not be able to afford complex computations with the current equipment. Therefore, this proposal may not be suitable for the existing environment. By virtue, the designs of AP-AKA and X-AKA were expected to adhere to the UMTS standard. Unfortunately, both use a number of custom encryption functions. These may interfere with the practicability in these protocols combining with the current mechanism. Recently, in 2006, AI-Saraireh and Yousef suggested some remedies to avoid the bottlenecks of the UMTS-AKA protocol, which include reducing the number of messages transmitted between mobile phones and authentication center and minimizing authentication time delay, call setup time and signaling traffic. Their proposal [7] has reversed the original authentication process, that is, AVs are not generated by the HE but instead the MS sends the authentication token via SN to the HE. In accordance with the authors' simulation, signaling messages between the mobile network entities are reduced and moreover, authentication time delay, call setup time and signaling traffic are minimized as compared to the original UMTS-AKA protocol. However, their protocol showed two deadly mistakes. One is that the MS must save n-sets of AVs on a limited space in mobile devices. The other mistake is that each authentication SN must pass the authentication token back to the HE; this will cause delay in authentication.

It is not the aim of this study to design the safest or the best AKA protocol. Indeed, the purpose is to design a most practical AKA protocol. It is very important to adopt the same technology as the current UMTS-AKA protocol. It ensures painless transfer to a new environment without extra cost. In addition, the attractive benefits of other literatures with are also been proposed on this study.

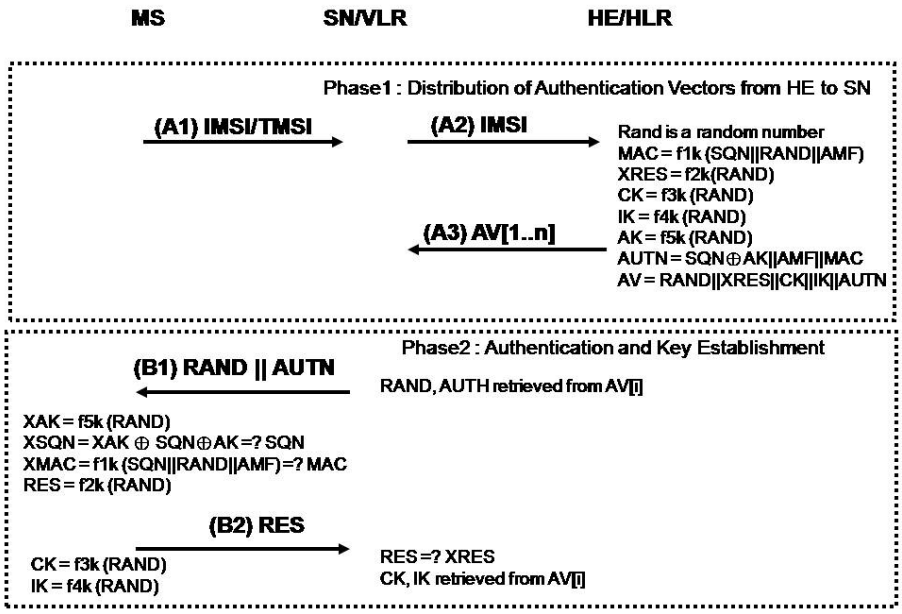

Figure 1: The UMTS-AKA Protocol.

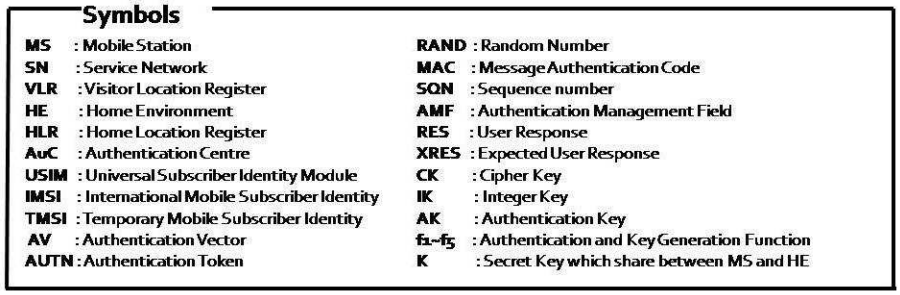

Figure 2: Abbreviations and Notations Used in the UMTS-AKA Protocol.

3 Review of the UMTS-AKA Protocol

Figure 1 illustrates UMTS-AKA protocol, and Figure 2 lists the abbreviations and notations used in the UMTS-AKA protocol. The UMTS-AKA protocol has two phases. Phase 1 involves the distribution of AVs, and Phase 2 involves authentication and key establishment. In phase 1, the SN receives AVs by delivery of the MS's IMSI (International Mobile Subscriber Identity) to the HE. In Phase 2, there are n-sets of AV's for n-times of AKA between the MS and the SN.

In the UMTS-AKA protocol, the MS should have mutual authentication with

the HE. However, the HE actually has mutual authentication with the SN.Since MS has the characteristic of mobile, it will be remote from the HE and will connects to the local SN. With the intention of the mutual identity, the SN must obtain authorization from the HE of the MS. Therefore, the MS makes authentication with the SN to obtain HE authorization. To meet this requirement, the HE should generate and delivere all AVs to the SN when the MS reaches the service region of the SN. In addition, since the SN holds n-sets of AVs, the last SN will relinquish unused AVs to the current SN that arrived at the MS. This can occur if the last SN has an unused AV when the MS moves into the service region of another SN. In this scenario, two defects are expose. One is computational overhead concentrating on the HE. The other is communication overhead for delivering the AVs. To solve these congenital defects, this study proposes an innovative UMTS-AKA protocol called the cocktail protocol. The details of the cocktail protocol are specified in the next section.

4 The Cocktail AKA Protocol

The basic principle of the proposed cocktail-AKA protocol is similar as an eminent cocktail therapy used in the treatment of acquired immune deficiency syndrome (AIDS). This therapy is called highly active anti-retroviral therapy (HAART). According to specific medical conditions, HARRT combines several kinds of drugs to make the treatment effective. The proposed cocktail-AKA protocol follows the essence of the aforesaid medical therapy, that is, using two varieties of AVs in the protocol.

In the UMTS-AKA protocol, all AVs are computed by the HE and transferred to the SN. The protocol suffers two congenital defects as mentioned earlier. Usually, it can works in generalized cases, since the HE has a higher computing capability and the HE is joined to the SN by a wire cable. However, this may become a bottleneck of the UMTS when simultaneous connections are required in a deal of

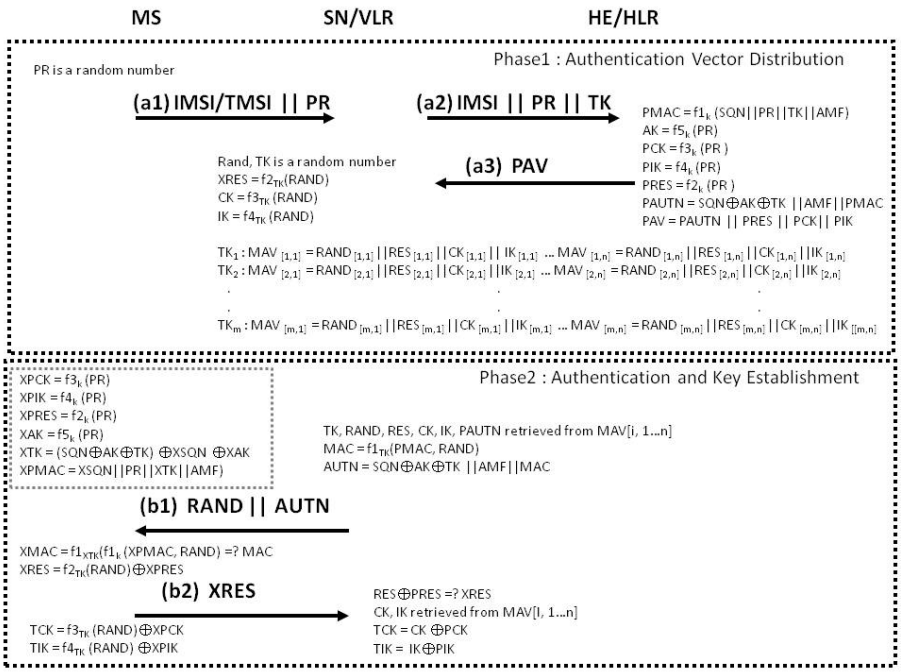

Figure 3: The Cocktail AKA Protocol.

MSs. In reality, one HE needs to serve a large number SNs, and one SN joins many MS. If all computation work were to concentrate on HE, this load is quite heavy. Therefore, the basic principle in cocktail proposal is to permit the SN to share some computation work of the HEs and give the HE only some responsibility for simple key computations. Using medical terms to study this protocol, all SNs produce their own AVs in advance, which are called medicated authentication vectors (MAV). These MAVs can be reused; therefore, they need to be produced only once. When the authentication stage is initiated, the HE calculates a private AV for the MS and transfers this to the SN. This is called the prescription authentication vector (PAV). The SN dispenses the PAV with the MAV, which can produce many effective AVs that can be used later for mutual authentication with the MS.

Figure 3 illustrates the cocktail-AKA protocol on the UMTS. It is composed of two phases: (1) the authentication vector distribution phase and (2) the authentication and key establishment phase. When an MS enters into the service region of SN for the first time, it executes the phase 1 to complete a registration procedure.

If the MS is already registered in the SN, then the MS requests the use of service in the SN. Due to this, the MS can execute Phase 2 to authenticate each other and establish relevant encryption key. Thus, the legitimacy of each is confirmed.

Phase 1: Authentication Vector Distribution Phase

This phase is executed when an MS enters into the service region of another SN for the first time. The main purpose of registration is to issue PAV from the MS's HE to the local SN. Therefore, the SN and the MS interact with and authenticate each other. A detailed procedure on Phase 1 of the cocktail-AKA protocol is given below.

(**a1**) **MS** $\rightarrow$ **SN** : **IMSI/TMSI** $\|$ **PR**: Where IMSI is the International Mobile Subscriber Identity, TMSI is the Temporary Mobile Subscriber Identity and PR is a 128-bit random number. The MS transmits its IMSI/TMSI along with the PR to the SN. The IMSI/TMSI is used for recognizing the MS, and the PR is used for producing the PAV. The MS calculates the relevant parameters in advance and stores them temporarily for use in Phase 2. The relevant parameters include the sets of $XPCK = f3_k(PR), XPIK = f4_k(PR)$ and $XPRES = f2_k(PR)$. In these, $f3$, $f4$, and $f2$ are key generation functions that qualify the 3GPP specifications [4, 5], and K is the secret share key, which is shared between the MS and the HE when the MS has previously joined the HE.

(**a2**) **SN** $\rightarrow$ **HE** : **IMSI** $\|$ **PR** $\|$ **TK**: Before the SN offers the connecting service to the MS, the SN calculates numerous MAVs in advance. These MAVs are delineated to $(TK[i], MAV[i,j])$ and $MAV[i,j] = (RAND[i,j] \parallel RES[i,j] \parallel CK[i,j] \parallel IK[i,j])$ where (i = 1 to m) and (j = 1 to n). $TK[i]$ is the temporary key used as a secret key, K, for producing the relevant Key in Phase 2; $RAND[i,j]$ is a random number(note: $RAND[i,k]$ must be bigger than $RAND[i,k]$ for $k = 0ton-1$); and $RES[i,j] = f2_{TK[i]}(RAND[i,j]), CK[i,j] = f3_{TK[i]}(RAND[i,j])$ and $IK[i] = f4_{TK[i]}(RAND[i,j])$. One $TK[i]$ is used to collocate n-sets of $MAV[i,j](j = 1\ to\ n)$. First, the SN will randomly generate m-sets of $TK[i](i = 1\ to\ m)$ and then each

$TK[i]$ will generate n-sets of $MAV[i,j]$. The computations are done well in advance and can be reused in future. It is important to consider that $MAV[i,j]$, under the same $TK[i]$, must be arranged from small to large according to the value of $RAND[i,j]$. This can prevent replay-attack in Phase 2.

After receiving the IMSI/TMSI and PR from the MS, the SN randomly selects one $TK[i]$ of the $MAV[i,j]$ as TK and delivers the $IMSI$, PR and TK to the MS's HE.

(**a3**) $\mathbf{HE \rightarrow SN : PAV}$: The HE calculates the $PMAC = f1_K(SQN,\ PR,\ TK,\ AMF)$, $AK = f5_K(PR)$, $PCK = f3_K(PR)$, $PIK = f4_K(PR)$, $PRES = f2_K(PR)$, $PAUTN = (SQN \bigoplus AK \bigoplus TK,\ AMF,\ PMAC)$, and $PAV = (PAUTN, PRES, PCK, PIK)$. In these, SQN is a sequence number which maintains consistency between the MS and the HE; the AMF is the authentication and key management file used to indicate the algorithm and key used to generate a particular AV; $f*$ is a key generating function that qualifies the 3GPP specifications [4, 5], AK is the anonymity key; PCK is the prescription cipher key; PIK is the prescription integrity key; $PRES$ is the prescription response; $PAUTN$ is the authentication token; and $PMAC$ is the message authentication code. Upon generating these, the HE sends the PAV back to the SN to combine with the MAV and to authenticate the MS in Phase 2.

Phase 2: Authentication and Key Establishment Phase

If an MS wants to acquire the service of a SN and the MS is already registered at Phase 1, then Phase 2 is deployed. In Phase 2, the MS and the SN will mutually authenticate each other and establish several relational encrypting keys in the session. If the MS continuously resides in the signal range of the same SN, then the SN must first proceed with Phase 1 to obtain the mandate of the HE. Then the MS can proceed with Phase 2 n-times for connectivity.

A detailed procedure for Phase 2 of the cocktail-AKA protocol is listed below.

(**b1**) **SN** $\rightarrow$ **MS** : **RAND** $\|$ **AUTN**: The SN chooses an unused set of parameters from $MAV[i,j]$ according to the (a2) $RAND$ sequence case for the TK_i. The SN retrieves $RAND[i,j]$ from $MAV[i,j]$ and in coordination with $AUTN = (SQN \bigoplus AK \bigoplus TK, AMF, MAC)$ in which $MAC = (f1_{TK}(PMAC, RAND))$ and $(SQN \bigoplus AK \bigoplus TK, AMF, PMAC)$ is retrieved from PAV, uses $(RAND, AUTN)$ as a challenge and sends this to the MS.

(**b2**) **MS** $\rightarrow$ **SN** : **XRES**: When the MS proceeds with Phase 2 for the first time after processing Phase 1, the MS must calculate some values in advance. These are $XPCK = f3_K(PR), XPIK = f4_K(PR), XPRES = f2_K(PR), XAK = f5_K(PR)$ and the retrieval of $(SQN \bigoplus AK \bigoplus TK)$ from $AUTN$. When calculating $XTK = (SQN \bigoplus AK \bigoplus TK) \bigoplus XSQN \bigoplus XAK$, in which $XSQN$ is a sequence number that maintains consistency between the MS and the HE, the MS then calculates whether $XMAC = (f1_{XTK}(f1_k(XSQN \parallel PR \parallel XTK \parallel AMF), RAND))$ equals MAC in AUTN. If they are equal, the MS regards the SN to have legal authorization with the HE. Afterwards, $XRES = f2_{XTK}(RAND) \bigoplus XPRES$ is calculated and returned to the SN. The SN compares $XRES$ with $RES \bigoplus PRES$, from which RES is retrieved from the MAV, and $PRES$ comes from the PAV. If they are equal, the SN understands that the MS is a legal user. What difference the UMTS-AKA protocol with the cocktail-AKA protocol has in the way of counting the SQN. In the UMTS-AKA protocol, a connection counts once. But in the cocktail-AKA protocol, the one-time register counts once. The cocktail-AKA protocol has a restriction in the number of connection times within one time register. There is an AMF in the $PAUTH$ within the PAV that the HE transfers to the SN, which can record the number of connections that is being allowed. Therefore, the MS will record the connection times of the same SQN after the registered stage of Phase 1 to confirm that the MS does not exceed the authorization times. In addition, the MS will also compare $RAND$ that must be greater than the previous RAND with the purpose of preventing a replay-attack.

Based on the above authentication steps are sustained, both the template cipher key (TCK) and the template integer key (TIK) can be calculated. The MS calculates $TCK = f3_{TK}(RAND) \bigoplus XPCK$ and $TIK = f4_{TK}(RAND) \bigoplus XPIK$. The SN calculates $TCK = CK \bigoplus PCK$ and $TIK = IK \bigoplus PIK$, in which CK, IK retrieved from both MAV and PCK, and PIK are retrieved from PAV.

5 Discussion and Analysis

The purpose of this proposed protocol is to solve the inherent problems of the UMTS-AKA protocol, including computational overhead concentrating on the HE and communication overhead for delivery of the AVs from the HE to the SN. In the UMTS-AKA protocol, all the AVs are generated by the HE and then relinquished to the SN, which causes the problems. In the cocktail-AKA protocol, the HE needs to produce only one set of AVs instead of n-sets of AVs. The result will be better than that of the UMTS-AKA protocol, and can solve the first inherent problem of computational overhead concentrating on the HE. This is due to the smaller number of AVs being delegated to the SN from the HE in the cocktail-AKA protocol. The protocol can reduce computational overhead and communication overhead by around $1/n$ in the HE, which is calculated in a simple way. The cocktail AKA protocol does increase the computation load of the SN; however, these computations are anticipated and can be reused. Therefore, it will not increase the burden on the time of authentication. In addition, in the UMTS-AKA protocol, the SN will request n-sets of AV from the HE when the MS is registered with the SN. To clarify, when the MS is registered once, it can make n-times of connection requests. If the system is very busy and the MS requires a connection, the HE will not be able to deal with all registrations. This is similar to the denial of service (DoS) attack [6, 8]. In the cocktail-AKA protocol, when one registered SN receives authorization by the HE, the time for this authorized connection can be decided by the HE. The HE can authorize the SN to extend connection times, when the system is busy. The

aforesaid situation will not arise in the cocktail-AKA protocol.

The AKA protocol has two phases; one is the registration phase in that MS sends a registration request to the HE and SN obtains the authorization to authenticate MS at another phase; the other is the AKA phase to authenticate each other and to establish a new pair of cipher and integrity keys between the MS and the SN when the MS stays in the same region of SN. Most of the relevant AKA protocols, such as UMTS-AKA, ER-AKA, and AP-AKA; HE sends the n-sets of AV to SN in the registration phase, and the SN can process n-time AKA phase with the MS. This has an advantage that the SN does not need to burden the calculation, so that the AKA phase will speed up, but it causes computation overhead and communication overhead on HE. Cocktail AKA and X-AKA have other consideration that HE's load be shared with the SN. The principle of X-AKA is that the HE authorizes SN a template key, TK. The SN can use TK instead of the HE to calculate AV. So, HE does not need to calculate and transmit n-sets of AVs; therefore, computation overhead and communication overhead can be resolved. In fact, the approach is to transfer the HE's computation overhead to SN so that SN also needs to calculate AVs. In view of this, the cocktail-AKA protocol proposed a novel concept; that is, the HE only needs to calculate and transmit one-set of PAV; the SN combines PAV with MAV calculated in advance and produces an effective AV for mutual authentication with the MS. This approach not only solves computation overhead and communication overhead but also transfers the HE's load to the SN on X-AKA.

In the following sections, we will compare two main contributions of the cocktail-AKA protocol, namely which is the communication overhead and computation overhead. Then, we will prove that the cocktail-AKA protocol can meet the necessary security issues that are questioned by other literatures.

5.1 Communication Overhead

It is an alarming situation when many MS authentication requests are received at the same time. So, in order to prevent network congestion, message sizes must be minimized. In this section, we will calculate messages sizes to understand their communication overhead with respect to the UMTS-AKA and cocktail-AKA protocols. Other relevant protocols are not included in the comparison because they use custom functions therefore unable to calculate the length of the messages. The bit numbers for the messages on the 3GPP standard are as follows [2].

- IMSI/TMSI, RAND, K, CK, IK = 128 bits;
- AMF, SQN, AK = 48 bits;
- RES, MAC = 64 bits.

According to the above values, bit numbers can be calculated. The total bit numbers of messages for the UMTS-AKA and cocktail-AKA protocols are listed as follows.

The UMTS-AKA protocol:

Phase 1: $(A1) + (A2) + (A3) = 128 + 128 + 608 \times n = 256 + 608 \times n$ bits. (n indicates the set number of AV, which HE each generated to SN.)

Phase 2: $(B1) + (B2) = 288 + 64 = 352$ bits.

Total: All the message sizes on the UMTS-AKA protocol for n-times connected are $Phase\ 1 + Phase\ 2 \times n = 256 + 608 \times n + 352 \times n = 256 + 960 \times n$ bits.

The cocktail-AKA protocol:

Phase 1: $(a1) + (a2) + (a3) = 256 + 384 + 560 = 1200$ bits;

Phase 2: $(b1) + (b2) = 368 + 64 = 432$ bits;

Total: All the message sizes on the cocktail-AKA protocol for n-times connected are $Phase\ 1 + Phase\ 2 \times n = 1200 + 432 \times n$ bits.

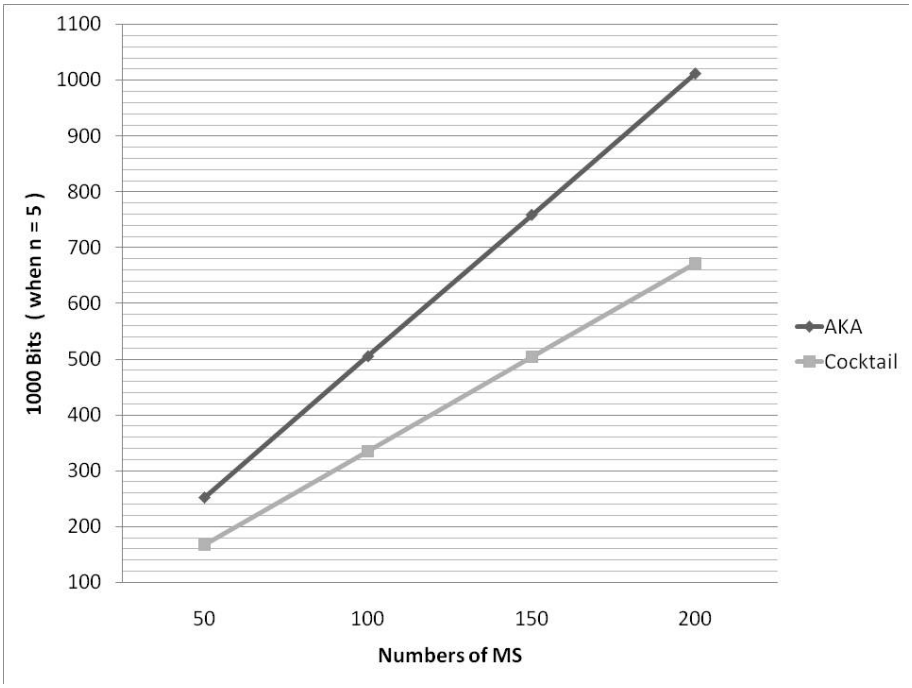

Figure 4: The Comparison of the Communication Overhead.

Figure 4 compares communication overhead of the UMTS-AKA protocol and the cocktail-AKA protocol, when the numbers of the MS is not fixed. The cocktail-AKA protocol has smaller communication overhead in all comparisons, and that more numbers of MS, which demonstrates the advantages of cocktail AKA protocol.

5.2 Computation Overhead

Figure 5 calculates the number of security functions ($f*$) to compare computation overhead. Although different security functions have different computational overheads, to calculate conveniently without exception, all computational overhead of the security function will be designated to 1. Other operations will not be performed due to their computation overhead being relative small.

Below are some statistics, that is, the numbers of function on the UMTS-AKA, AP-AKA, X-AKA and cocktail-AKA. Note that all the protocols must be calculated under the same standards for fairness. Under this principle, the function G will be multiplied by 2 while calculating AP-AKA. UMTS should have both CK and IK. The two encrypt keys at communication in the planning of 3GPP. However, AP-

AKA is designed with only one encryption key, SK. So, it will need to be multiplied by 2.

The UMTS-AKA protocol:

Phase 1: $(f1 + f2 + f3 + f4 + f5) \times n = 5 \times n$. ($n$ indicates the set number of AV, which HE each generated to SN.)

Phase 2: $f5 + f1 + f2 + f3 + f4 = 5$;

Total: All of the computation numbers on the UMTS-AKA protocol for n-times connected are $Phase\ 1 + Phase\ 2 \times n = (5 \times n) + (5 \times n) = 10 \times n$.

The AP-AKA protocol:

Phase 1: $Fk + Fk + (Fk + Gk \times 2 + Hk + Fk) \times n = 2 + 5 \times n$. ($n$ indicates the set number of AV, which HE each generated to SN.)

Phase 2: $Fk + Fk + Gk \times 2 = 4$;

Total: All of the computation numbers on the AP-AKA protocol for n-times connected are $Phase\ 1 + Phase\ 2 \times n = (2 + 5 \times n) + (4 \times n) = 2 + 9 \times n$.

The X-AKA protocol:

Phase 1: $(f1 + fx + f1 + fx + f1) = 5$;

Phase 2: $f1 + f2 + f3 + f4 + f1 + f2 + f3 + f4 = 8$;

Total: All of the computation numbers on the X-AKA protocol for n-time connected are $Phase\ 1 + Phase\ 2 \times n = 5 + 8 \times n$.

The cocktail-AKA protocol:

Phase 1: $f3 + f4 + f2 + f5 + f1 + f1 + f5 + f3 + f4 + f2 = 10$;

Phase 2: $f1 + f1 + f2 + f3 + f4 = 5$;

Total: All of the computation numbers on the cocktail-AKA protocol for n-times connected are $Phase\ 1 + Phase\ 2 \times n = 10 + 5 \times n$.

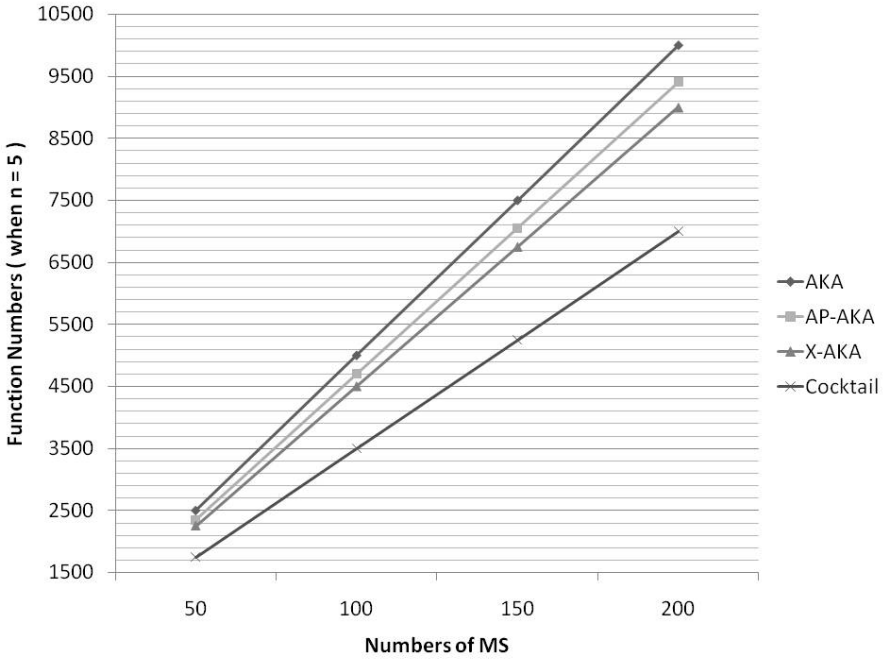

Figure 5: The Comparison of the Computation Overhead.

Figure 5 compares the computation overheads of the UMTS-AKA, AP-AKA, X-AKA and Cocktail-AKA protocols, when the numbers of the MS are different. This comparison is similar as communication overhead comparison in that the cocktail AKA protocol has a smaller computation overhead in all comparisons, and that more numbers of MS, which demonstrates the advantages of cocktail-AKA.

5.3 Security Analysis

The cocktail-AKA protocol adopts the same secured architecture as the UMTS-AKA protocol. Therefore, it has the same security threshold in most situations. This section examines the cocktail-AKA protocol to see if the same security requirements are reached and perform better than the UMTS-AKA protocol and other AKA protocols.

In this section, we state how the cocktail-AKA protocol is security. We first introduced the security requirements of UMTS-AKA protocol and explain how the cocktail-AKA protocol to meet them. Then we demonstrated that cocktail-AKA protocol is the tough and difficult to injury. Finally, we review the defects of the

UMTS-AKA protocol that have exposing in the relevant literatures and that we show cocktail-AKA protocol can overcome the defects.

5.3.1 UMTS Security Requirements

In the 3GPP [1] standard, five security architectures are defined [2]. They are (1) network access security, (2) network domain security, (3) user domain security, (4) application domain security, and (5) visibility and configurability of security. The UMTS-AKA protocol belongs to the network access security [2]. In view of the fact that the cocktail-AKA protocol adopts the same secured architecture as the UMTS-AKA protocol, it can reach these demands with the same technology. In this section, how the UMTS-AKA protocol and the cocktail-AKA protocol reach these demands is explained.

The 3GPP lists some possible security threats to the 3G system in their standards [18] and clause threats in the following categories.

1. **Threats associated with attacks on the radio interface**, such as: unauthorized access to data, threats to integrity, denial of service attacks, and unauthorized access to services.

2. **Threats associated with attacks on other parts of the system**, such as unauthorized access to data, threats to integrity, denial of service attacks, repudiation, and unauthorized access to services.

Benefiting from the experience of operating mobile systems, most of the above-mentioned significant threats can be categorized into the three groups [3], their are

1. **Masquerading:** Intruders may gain unauthorized access to services.

2. **Eavesdropping:** Subscribers may compromise user data traffic confidentiality or call-related information, such as dialed numbers and location data.

3. **Subscription fraud:** Subscribers may exploit the services with heavy usage without any intention of paying.

To solve the above threats, the 3GPP also defines some secure requirements according to the design of the related protocol [3]. These requirements are listed as follows, and how the cocktail-AKA protocol meets them is explained.

1. **Requirements on security of 3GPP services** (secure service access, secure service provision): Based on this demand, both the MS and the SN are the best solutions for mutual authentication. The cocktail-AKA protocol is an authentication and key agreement protocol, in which mutual authentication is the principal design concept. In the UMTS-AKA protocol, the foundation of authentication is a secret key that is shared in advance. According to the key and a random number, the SN authenticates the MS by the challenge/response method and the MS authenticates the SN by the MAC. To overcome the congenital defects of the UMTS-AKA protocol, the cocktail-AKA improves the production way of AVs. Based on the cocktail-AKA protocol, the foundation of authentication is a random-number, original secret key and an additional template key. The MS must have the secret key that is shared with the HE to produce the correct response of the SN's challenge for passing authentication to the SN. Correspondingly, the SN must provide the authentication token that the MS's HE grants to prove the identity. This authentication token is produced by the secret key and the random number. The secret key is shared between the MS and the HE in advance, and the random number is offered by the MS. Given that the mechanism of mutual authentication in the cocktail-AKA protocol has a certain implementation, the requirements on secured service access and secured service provision can be achieved.

2. **Requirements on system integrity**: Based on this demand, the cocktail-AKA protocol will protect against unauthorized modification of user-related data, certain signaling data and control data, particularly radio interfaces. This is due to the addition of MAC after the authentication token and the use of IK to guarantee the dependability of the control data and signal.

3. **Requirements on protection of personal data** (security of user-related transmitted data, user-related stored data): Based on the cocktail-AKA protocol, CK is designed to protect the confidentiality of user-related transmitted data; the $TMSI$ is used to protect the confidentiality of user identity data and location data.

In order to achieve the above request to maintain the security of UMTS, the following are all of these security conditions must be considered [2].

1. **User identity confidentiality** (user identity confidentiality, user location confidentiality, user untraceability): To achieve this demand, users have the $IMSI$ to verify the identity and use the $TMSI$ to hide the location.

2. **Entity authentication** (user authentication, network authentication): Both the UMTS-AKA protocol and the cocktail-AKA protocol are designed to achieve this demand. The Phase 1 of these protocols can satisfy the feature of user authentication, due to the serving network corroborating the user identity of the user. The feature of network authentication can be achieved in the Phase 2 of these protocols. This phase corroborates that the user is connected to a serving network that is authorized by the HE.

3. **Confidentiality** (cipher algorithm agreement, cipher key agreement, confidentiality of user data, confidentiality of signaling data): The cocktail-AKA protocol follows the mechanism of the UMTS-AKA protocol and is successful with these demands. The HE carries the field of the AMF in the $PAUTH$ token to meet the feature of cipher algorithm agreement. A random number collocates with a secure key to make the feature of cipher key agreement, and all the user data will be encrypted with the template cipher key that the MS and the SN agree on in each time session.

4. **Data integrity** (integrity algorithm agreement, integrity key agreement, data integrity, origin authentication of signaling data): Similar to the demand of

confidentiality, the HE carries the field of the AMF in the $PAUTH$ token to meet the feature of integrity algorithm agreement. A random number collocates with a secure key to make the feature of integrity key agreement, and all the user data will be verified with the template integrity key that the MS and the SN agree in each time session. In addition, the original authentication of signaling data will be protected with the message authentication code (MAC).

5.3.2 The security of cocktail-AKA-protocol

As described above, the cocktail-AKA-protocol's design has conforms the needs and requirements of UMTS. Then we continue to explain that cocktail-AKA-protocol is designed to allow it to resist the threat of attack. In order to clear statement of our analysis, we use the BAN-Logic [9] symbols to help it. These symbols as follows [9]:

- $\mathbf{P}|\equiv\mathbf{X}$: P believes X, or P would be entitled to believe X.
- $\mathbf{p}\triangleleft\mathbf{X}$: P sees X. Someone has sent a message containing X to P, who can read and repeat X.
- $\mathbf{P}|\sim\mathbf{X}$: P once said X. P at some time sent a message including the statement X.
- $\mathbf{P}|\Rightarrow\mathbf{X}$: 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.
- $\mathbf{P}\overset{K}{\leftrightarrow}\mathbf{Q}$: P and Q may use the shared key K to communicate.
- $\mathbf{P}\overset{X}{\rightleftharpoons}\mathbf{Q}$: 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.

1. The formally messages

a. (Message a1) $MS \rightarrow SN : Na$

b. (Message a2) $SN \rightarrow HE : Na, SN \overset{TK}{\leftrightarrow} HE$

c. (Message a3) $HE \rightarrow SN : (TK)_{SQN,AK}, f_1(SQN, Na, TK)_k, f_2(Na)_k, f_3(Na)_k, f_4(Na)_k$

d. (Message b1) $SN \rightarrow MS : Nb, (TK)_{SQN,f_5(Na)_k}, f_1(f_1(SQN, Na, TK)_k, Nb)_{TK}$

e. (Message b2) $MS \rightarrow SN : f_2(f_2(Na)_k, Nb)_{TK}$

f. (Cipher Key) $CK : f_3(f_3(Na)_k, Nb)_{TK}$

g. (Integer Key) $IK : f_4(f_4(Na)_k, Nb)_{TK}$

2. Security Assumptions

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

(1) MS has the secure key K and MS$|\equiv$ MS$\overset{K}{\rightleftarrows}$ HE

(2) HE has the secure key K and HE$|\equiv$ MS$\overset{K}{\rightleftarrows}$ HE

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

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

(2) $\dfrac{SN \mid\sim P, HE \lhd P}{HE \mid\equiv SN \mid\equiv P}$

(3) $\dfrac{HE \mid\sim P, SN \lhd P}{SN \mid\equiv HE \mid\equiv P}$

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

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

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

d. It is assumed that a sequence number SQN is maintain with MS and HE.

3. 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.

e. Resistance replay attack between MS and SN.

4. Statements and analysis

a. (Goal 3.a) Mutual authentication between MS and SN.

(1) (1.a),(1.b),(1.c)$\rightarrow MS \mid\equiv \sharp(Na) \wedge SN \mid\equiv \sharp(Na) \wedge HE \mid\equiv \sharp(Na)$

(2) (1.b),(2.b)$\rightarrow HE \mid\equiv \forall(SN \overset{TK}{\leftrightarrow} MS)$

(3) (1.c),(2.a),(2.b)$\rightarrow$

$SN \mid\equiv \forall((TK)_{SQN,AK}, f_1(SQN, Na, TK)_k, f_2(Na)_k, f_3(Na)_k, f_4(Na)_k) \cdot$

$(HE \mid\Rightarrow ((TK)_{SQN,AK}, f_1(SQN, Na, TK)_k, f_2(Na)_k, f_3(Na)_k, f_4(Na)_k))$

(4) $\because$ (1.a),(1.d),(2.a), and (2.b), MS can retrieve TK from $(TK)_{SQN,f_5(Na)_k}$, and checking the correct and $\sharp(Nb)$ by $f_1(f_1(SQN, Na, TK)_k, Nb)_{TK}$

(5) For message-meaning rule and (1.d),(2.a)

$$\frac{MS \mid\equiv (MS \overset{K}{\leftrightarrow} HE) \wedge (SN \overset{TK}{\leftrightarrow} MS), MS \triangleleft f_1(f_1(SQN, Na, TK)_k, Nb)_{TK}}{MS \mid\equiv HE \mid\sim f_1(f_1(SQN, Na, TK)_k, Nb)_{TK}}$$

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

$$\frac{MS \mid\equiv \sharp(Na) \wedge \sharp(Nb), MS \mid\equiv HE \mid\sim f_1(f_1(SQN, Na, TK)_k, Nb)_{TK}}{MS \mid\equiv HE \mid\equiv f_1(f_1(SQN, Na, TK)_k, Nb)_{TK}}$$

(7) For jurisdiction rule and (4.a.b),(1,d)

$$\frac{MS \mid\equiv HE \Rightarrow f_1(f_1(SQN,Na,TK)_k,Nb)_{TK}, MS \triangleleft SN \mid\sim f_1(f_1(SQN,Na,TK)_k,Nb)_{TK}}{MS \mid\equiv HE \mid\equiv SN}$$

(8) For message-meaning rule and (1.c),(1.e),(2.a)

$$\frac{SN \mid\equiv (f_2(Na)_k \wedge (SN \overset{TK}{\leftrightarrow} MS), SN \triangleleft f_2(f_2(SQN, Na, TK)_k, Nb)_{TK}}{SN \mid\equiv HE \mid\sim f_2(f_2(SQN, Na, TK)_k, Nb)_{TK}}$$

(9) For nonce-verification rule and (4.a.1),(4,a,4),(4,a,8)

$$\frac{SN \mid\equiv \sharp(Na) \wedge \sharp(Nb), SN \mid\equiv HE \mid\sim f_2(f_2(SQN, Na, TK)_k, Nb)_{TK}}{MS \mid\equiv HE \mid\equiv f_2(f_2(SQN, Na, TK)_k, Nb)_{TK}}$$

(10) For jurisdiction rule and (4.a.b),(1,d)

$$\frac{SN \mid\equiv HE \Rightarrow f_2(f_2(SQN,Na,TK)_k,Nb)_{TK}, SN \triangleleft SN \mid\sim f_2(f_2(SQN,Na,TK)_k,Nb)_{TK}}{SN \mid\equiv HE \mid\equiv MS}$$

(11) By (4.a.7), (4.a.10) $\rightarrow MS \mid\equiv HE \mid\equiv SN \wedge SN \mid\equiv HE \mid\equiv MS \rightarrow MS \mid\equiv SN \wedge SN \mid\equiv MS$, So, the goal of mutual authentication between MS and SN is holds.

b. (Goal 3.b) Key agreement between MS and SN.

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

(2) $\because (4.a.1) \rightarrow MS \mid\equiv \sharp(Na), (4.a.4) MS \mid\equiv TK \wedge \sharp(Nb)$

$Since (2.a) \rightarrow MS \mid\equiv k, \therefore MS \mid\equiv CK = f_3(f_3(Na)_k, Nb)_{TK} \wedge IK = f_4(f_4(Na)_k, Nb)_{TK}$

(3) $\because (4.a.1) \rightarrow SN \mid\equiv \sharp(Na), (1.a), (1.d) SN \mid\equiv TK \wedge \sharp(Nb)$

$Since (4.a.3) \rightarrow SN \mid\equiv f_3(Na)_k \wedge f_4(Na)_k, \therefore SN \mid\equiv CK = f_3(f_3(Na)_k, Nb)_{TK} \wedge IK = f_4(f_4(Na)_k, Nb)_{TK}$

(4) By (4.b.2),(4.b.3)→ The goal of key agreement between MS and SN is holds.

c. (Goal 3.c) Key freshness between MS and SN.

(1) Since (1.f),(1.g)→ $CK = f_3(f_3(Na)_k, Nb)_{TK}, and IK = f_4(f_4(Na)_k, Nb)_{TK}$

(2) $\because (4.a.1) \rightarrow MS \mid\equiv \sharp(Na) \wedge SN \mid\equiv \sharp(Na), and (1.a), (4.a.4) \rightarrow SN \mid\equiv \sharp(Nb) \wedge MS \mid\equiv \sharp(Nb)$

(3) $\therefore$ The key freshness between MS and SN is holds.

d. (Goal 3.d) Confidentiality between MS and SN.

(1) Since (4.a),(4.b),(4.c) for message-meaning rule $\frac{MS \mid\equiv (MS \overset{CK \wedge IK}{\leftrightarrow} SN), MS \triangleleft (Messages)_{CK \wedge IK}}{MS \mid\equiv SN \mid\sim Messages} \wedge \frac{SN \mid\equiv (SN \overset{CK \wedge IK}{\leftrightarrow} MS), SN \triangleleft (Messages)_{CK \wedge IK}}{SN \mid\equiv MS \mid\sim Messages}$

(2) By (4.d.1) the goal of confidentiality between MS and SN is holds.

e. (Goal 3.e) Resistance replay attack between MS and SN.

(1) $\because (2.c) \therefore$ (1.b),(1.c) is safe.

(2) If the attacker gets Na from (1.a); he is unable forge (1.d) or (1.e) since he did not know k.

(3) If the attacker gets the (1.d); he also unable use it at the next time because Na will change.

(4) If the attacker gets the (1.e); he can not do any matter because Na and Nb will be change at the next time.

(5) By (4.e.1),(4.e.2),(4.e.3),(4.e.4) the goal of resistance replay attack between MS and SN is hold.

5.3.3 Other literature proposed security requirements

In recent years, numerous studies and literature [13, 14, 17, 22] have been written on the defects of the UMTS-AKA protocol. They point out the following defects, and in this study the defects are explained with resolutions provided by the cocktail-AKA protocol.

1. **Bandwidth consumption between the SN and the HE** [14, 17]: By the standards of the UMTS-AKA protocol, the HE produces n-sets of AVs and relinquishes these to the SN, which lends to the program of bandwidth consumption between the SN and the HE. The cocktail-AKA protocol is designed to solve this program. It allows the SN to share some $MAVs$ that are calculated in advance. The HE only needs to produce a PAV and transfer it to the SN. When the authentication stage is initiated, the SN dispenses these two AVs and produces an effective AV for mutual authentication with the MS. The cocktail-AKA protocol reduces both bandwidth consumption between the SN and the HE and computational overhead concentrating on the HE.

2. **Storage space of the SN [14]**: The role that the SN performs in the cocktail-AKA protocol is similar to a druggist. The SN dispenses the MAV it holds and combines this with the PAV produced by the HE. The produced effective AV can then be used for mutual authentication with the MS. In principal, the SN must possess some MAV for allocating. However, the size of these vectors is regular and can be reused, which will not increase the burden on the SN. The SN needs to store only one PAV instead of n-sets of AVs whenever there is a new MS to reach the service area. In the cocktail-AKA protocol, as well as the original disposing space that stores the MAVs, the storage space of the SN will be smaller.

3. **The SQN has synchronization program and operational difficulty** [14, 22]: In the original UMTS-AKA protocol, the HE maintains a counter in the SQN for each MS, and the MS also maintains a same counter in SQN. This counter is dynamic and changes and upgrades in the connection each time. If a crash happens, the SQN will be asynchronous. Although resynchronization can solve this program, it will bring about communication overhead on both sides. In the cocktail-AKA protocol, the SQN has only been maintained in the Phase 1 and not in phase 2. It can prevent a crash that does not need to maintain SQN on the connection each time.

4. **Redirection attack** [22]: This scenario presumes that an adversary has a device that can simulate the functionality of a base station (BS) and a MS. The adversary replays messages that come from a legitimate MS to a genuine BS with the forging device. Thus, the adversary can redirect messages to an unintended network. To solve this problem and not damage the principle of the original protocol, the cocktail-AKA protocol uses a simplified method that records the relevant position message in the AMF. The AMF is an AKA management field that is included in the authentication token of each AV. This field is used to support multiple authentication algorithms and keys. It may be used to indicate the algorithm and key used to generate a particular AV. It can also record the relevant position message to identity the SN, $IMSI/TMSI$ of the MS and the service area. In addition, the AMF on the authentication token, plus the inside MAC guarantees the exactness of the messages. Therefore, the cocktail-AKA protocol guards against redirection attack.

5. **Active attack in corrupted networks** [22]: In the original UMTS-AKA protocol, the AVs may be transferred between different SNs when the MS roams through the service area of other SN's. If a network is corrupted, an adversary can acquire the AVs of the MS from the corrupted network and

Table 1: Comparisons among the authentication and key agreement schemes

Item	BSR	FTS	RBC	RSV	SPS	SRA	SAC	RAD	MCM	MCP
UMTS-AKA	yes	yes	no	no	no	no	no	yes	no	no
AP-AKA	no	no	no	no	yes	yes	yes	yes	no	no
X-AKA	yes	no	yes	yes	yes	yes	yes	no	no	no
Cocktail	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes

BSR: basic security requirements; FTS: follow the standard;
RBC: reduction of bandwidth consumption between SN and HE;
RSV: reduction of SN storage; SPS: solve the program of synchronization; SRA: solve the redirection attack;
SAC: solve the active attack in corrupted networks; RAD: reduction of authentication delay;
MCM: minimize communication overhead; MCP: minimize computation overhead

forge a legal SN to connect with the MS. The cocktail-AKA protocol does not have this mechanism, so this situation will not take place. In addition, the relevant identity message that has been recorded on the AMF is unable to forge-attack.

In addition to the above-mentioned issues, the key generation function that the cocktail-AKA protocol utilizes is the same as the original UMTS system. Therefore, it may not have compatibility problems in real-life applications. For this reason, the cocktail AKA protocol relinquishes other cryptography mechanisms and chooses to follow the standards of the UMTS.

5.4 Summary

From our analysis and comparison, we derive the properties of the relative AKA protocols as follows, and show the results in Table 1.

- Basic security requirements (BSR): The cocktail protocol conforms to the basic security requirements as defined by 3GPP and can resist security threats on outlined in Section 5.3. Here, we state some interesting findings of this key approach.

    1. Mutual authentication: The AKA protocol is using challenge/response method for authentication. The key point is all in the secret key K that the MS shared with the HE in advance. The SN sends $(RAND, AUTN)$

to the MS as the challenge. The MS checks whether they are fresh and accurate. Calculating the corresponding $XRES$ by $RAND$, it returns the response to confirm its identity. Only the authorized SN's AUTN has a correct MAC to confirm that TK is credible, and the exactness of MAC relies on the secret key K and PR that MS produces. So, their authentication is all linked with one another. Therefore, their mutual authentication is established.

2. Confidentiality and integrity: There are cipher key (CK) and integrity key (IK) to establish on the authentication. The CK is designed to protect the confidentiality of user-related transmitted data, and the IK is designed to guarantee the dependability of the control data and sign. Therefore the AP-AKA only produces a key SK to service this, it does not need to accord with this demand. In addition, AP-AKA cannot reach the cipher algorithm agreement and integrity algorithm agreement because the protocol does not include the AMF. Moreover, TMSI and MAC are crucial to hold confidentiality and integrity. The cocktail protocol can reach this demand while having these mechanisms.

3. Masquerading and eavesdropping: Masquerading and eavesdropping are invalid in the cocktail protocol. The cocktail protocol uses fresh RAND and relate key each time. Any interception and replay attack are unable to pass authentication and obtain any interests.

- Follow the standard (FTS): Only the cocktail protocol and the UMTS-AKA follow the standard established by 3GPP. AP-AKA and X-AKA have introduced new encryption function or key generation function in their protocol, such as fx, F, G, and H.

- Reduction of bandwidth consumption between SN and HE (RBC): UMTS-AKA and AP-AKA must deliver n-sets of AVs from the HE to the SN; therefore, bandwidth consumption problem happens. X-AKA and cocktail protocol

overcome this program by delivering a template key and *one*-set of PAV. So, X-AKA and cocktail protocol can reduce bandwidth consumption between SN and HE, whereas the other protocols cannot.

- Reduction of SN storage (RSV): The UMTS-AKA and AP-AKA protocols must store n-sets of AVs; therefore, SN needs larger space to store n-sets of AVs. X-AKA and cocktail protocols do not needs; one TK and one PAV are replaced with n-sets of AVs.

- Solve the program of synchronization (SPS): AP-AKA, X-AKA, and cocktail protocols use different methods to solve this problem, except UMTS-AKA.

- Solve the redirection attack (SRA): AP-AKA puts the identity of the SN into the authentication token to verify the SN and prevent redirection attack. The cocktail protocol uses a simplified method to record the relevant position message in the AMF. Rely on the protection of MAC to solve the redirection attack. It is unknown if X-AKA has this function.

- Solve the active attack in corrupted networks (SAC): AP-AKA and cocktail protocols can solve this program, whereas UMTS-AKA and X-AKA cannot.

- Reduction of authentication delay (RAD): The reason for authentication delay is computation overhead between MS and SN. In UMTS-AKA and AP-AKA, the SN does not need to do any calculation, but just compares. Therefore, authentication delay will not happen there. In the cocktail protocol, the SN only needs to calculate a security function. Because of the application of a template key, X-AKA needs to calculate all authentication token and related keys. This makes computation overhead to cause authentication delay.

- Minimize communication overhead (MCO): According to the conclusion in Section 5.1, the cocktail protocol owns minimum communication overhead.

- Minimize computation overhead (MCP): According to the conclusion in Section 5.2, the cocktail protocol owns minimum computation overhead.

According to the above discussion and Table 1, we confirm that the cocktail protocol is superior to other relevant protocols.

6 Conclusions

The cocktail-AKA protocol adopts a unique schema to solve the congenital defects of the UMTS-AKA protocol. It is similar to a well-known cocktail therapy in medicine, where two different types of AVs are combined to make an effective AV. The advantage is that the HE can share some computation processes with the SN, and the SN can compute MAVs that are allocated in advance. Another advantage is that MAVs can be reused.

Based on the discussion and analysis section of this article, it can be confirmed that the cocktail-AKA protocol can solve the congenital defects of the UMTS-AKA protocol. It has been shown that the security demands of the 3GPP standard and other literature protocols are met. Furthermore, the cocktail-AKA protocol is more practical, since it follows the UMTS standard.

References

[1] 3GPP *http://www.3gpp.org.*

[2] 3rd Generation Partnership Project. "Technical specification group services and systems aspects, security architecture,". tech. rep., 3GPP TS 33.102.

[3] 3rd Generation Partnership Project. "Technical specification group services and systems aspects, security threats and requirements,". tech. rep., 3GPP TS 21.133.

[4] 3rd Generation Partnership Project. "Technical specification group services and systems aspects, specification of the MILENAGE algorithm set, document 1: General,". tech. rep., 3GPP TS 35.205.

[5] 3rd Generation Partnership Project. "Technical specification group services and systems aspects, specification of the MILENAGE algorithm set, document 5: Summary and results of design and evaluation,". tech. rep., 3GPP TS 35.909.

[6] A. Agah and S. K. Das, "Preventing DoS attacks in wireless sensor networks: A repeated game theory approach," *International Journal of Network Security*, vol. 5, pp. 145–153, 2007.

[7] Jaafer Al-Saraireh and Sufian Yousef, "A new authentication protocol for umtsmobile networks," *EURASIP Journal onWireless Communications and Networking*, vol. 2006, pp. 1–10, 2006.

[8] V. Bocan, "Threshold puzzles: The evolution of DoS-resistant authentication," *Periodica Politechnica, Transactions on Automatic Control and Computer Science*, vol. 49, 2004.

[9] M. Burrows, M. Abadi, and R. Needham, "A logic of authentication," *ACM Tran. Computer Systems*, vol. 8, no. 1, pp. 18–36, 1990.

[10] Atul Chaturvedi and Sunder Lal, "An authenticated key agreement protocol using conjugacy problem in braid groups," *International Journal of Network Security*, vol. 6, pp. 181–184, Mar. 2008.

[11] Christos K. Dimitriadis and Siraj A. Shaikh, "A biometric authentication protocol for 3g mobile systems: Modelled and validated using csp and rank functions," *International Journal of Network Security*, vol. 5, pp. 99–111, July. 2007.

[12] European Telecommunication Standards Institute (ETSI). "Recommendation GSM 03.20, security related network functions,". tech. rep., June 1993.

[13] L. Harn and W.-J. Hsin, "On the security of wireless network access with enhancements," in *Proceedings of the 2003 ACM workshop on Wireless security*, pp. 88–95, San Diego, CA, USA, 2003.

[14] C.-M. Huang and J.-W. Li, "Authentication and key agreement protocol for UMTS with low bandwidth consumption," in *Proceedings of 19th International Conference on Advanced Information Networking and Applications (AINA 2005)*, vol. 1, pp. 392–397, March 2005.

[15] Wen-Shenq Juang and Jing-Lin Wu, "Efficient user authentication and key agreement with user privacy protection," *International Journal of Network Security*, vol. 7, pp. 120–129, July 2008.

[16] Wei-Bin Lee and Chang-Kuo Yeh, "A self-concealing mechanism for authentication of portable communication systems," *International Journal of Network Security*, vol. 6, pp. 285–290, May 2008.

[17] Men Long, Chwan-Hwa John Wu, and J. David Irwin, "Reducing communication overhead for wireless roaming authentication: Methods and performance evaluation," *International Journal of Network Security*, vol. 6, pp. 331–341, May. 2008.

[18] Kumar Mangipudi, Rajendra Katti, and Huirong Fu2, "Authentication and key agreement protocols preserving anonymity," *International Journal of Network Security*, vol. 3, pp. 259–270, NOV. 2006.

[19] Shengbao Wang, Zhenfu Cao, and Haiyong Bao, "Efficient certificateless authentication and key agreement (CL-AK) for grid computing," *International Journal of Network Security*, vol. 7, pp. 342–347, Nov. 2008.

[20] Shengbao Wang, Zhenfu Cao, and Feng Cao, "Efficient identity-based authenticated key agreement protocol with pkg forward secrecy," *International Journal of Network Security*, vol. 7, pp. 181–186, Sept. 2008.

[21] Chou-Chen Yang, Kuan-Hao Chu, and Ya-Wen Yang, "3G and WLAN interworking security: Current status and key issues," *International Journal of Network Security*, vol. 2, pp. 1–13, JAN. 2006.

[22] M. Zhang and Y. Fang, "Security analysis and enhancements of 3GPP authentication and key agreement protocol," *IEEE Transactions on Wireless Communications*, vol. 4, pp. 734–742, March 2005.