6-2005

Efficient anonymous roaming and its security analysis

Guomin YANG
Singapore Management University, gmyang@smu.edu.sg

Duncan S. WONG
City University of Hong Kong

Xiaotie DENG
City University of Hong Kong

Citation

YANG, Guomin; WONG, Duncan S.; and DENG, Xiaotie. Efficient anonymous roaming and its security analysis. (2005). *Applied Cryptography and Network Security: 3rd International Conference, ACNS 2005, New York, June 7-10: Proceedings*. 3531, 334-349.
Available at: https://ink.library.smu.edu.sg/sis_research/7439

Efficient Anonymous Roaming and Its Security Analysis

Guomin Yang, Duncan S. Wong*, and Xiaotie Deng

Department of Computer Science
City University of Hong Kong
Hong Kong, China
{csyanggm,duncan,deng}@cs.cityu.edu.hk

Abstract. The Canetti-Krawczyk (CK) model uses resuable modular components to construct indistinguishability-based key exchange protocols. The reusability of modular protocol components makes it easier to construct and prove new protocols when compared with other provably secure approaches. In this paper, we build an efficient anonymous and authenticated key exchange protocol for roaming by using the modular approach under the CK-model. Our protocol requires only four message flows and uses only standard cryptographic primitives. We also propose a one-pass counter based MT-authenticator and show its security under the assumption that there exists a MAC which is secure against chosen message attack.

Keywords: Authenticated Key Exchange, Anonymous Roaming

1 Introduction

Secure key-exchange (KE) protocols provide the basis to build secure communications using symmetric cryptography. But numerous claimed secure protocols have later been found insecure. In 1993, Bellare and Rogaway [3] proposed the first security model for provably secure KE protocols under the symmetric setting. In 1998, Bellare, Canetti and Krawczyk [2] proposed a different model which treats authentication and key exchange separately. The major advantage of this approach is that the proved building blocks can be reused to construct new provably secure protocols. Later, Canetti and Krawczyk [5] extended this work and changed the definition of secure key exchange from simulation-based to indistinguishability-based.

A traditional key exchange protocol involves two parties connected by wired networks, or three parties in the trusted third party setting. With the rapid development of mobile technology, wireless networks become widely available. People can travel around with their mobile devices without being limited by the geographical area of their home networks. This capability is called *roaming*. A

* The work was supported by a grant from the Research Grants Council of the Hong Kong Special Administrative Region, China (Project No. 9040904 (RGC Ref. No. CityU 1161/04E)).

J. Ioannidis, A. Keromytis, and M.Yung (Eds.): ACNS 2005, LNCS 3531, pp. 334–349, 2005.

typical roaming scenario involves three parties: *a roaming user*, *a foreign server*, and *a home server*. The roaming user subscribed to a home server is now in a foreign network and wants to get services from the foreign server. In order to ensure confidentiality of the communications, an authenticated key exchange protocol is carried out among the three parties, e.g. the 2G and 3G cellular network key exchange protocols.

Another important issue regarding the roaming scenario is user privacy. It concerns about hiding the roaming user's identity and movements from eavesdroppers and even the foreign servers, namely *user anonymity* and *user untraceability*. In cellular networks, the latest generation, 3GPP[1], is urging roaming services to be provided with some assurance on the privacy of mobile users. There are many other roaming networks that want user privacy. One is the inter-bank ATM networks or the credit card payment systems [1]. Ideally, a user should not have to reveal anything to the serving network (i.e. the foreign server) other than the confirmation of his good standing with respect to his ATM card or credit card issued by his home server. However, current systems are having users given out their personal information inevitably. Some other scenarios which require anonymous roaming include hopping across meshed WLANs (Wireless Local Area Networks) administered by different individuals, joining and leaving various wireless ad hoc networks operated by different foreign operators, etc.

There have been a number of work on anonymous and authenticated key exchange for roaming [6, 9]. In [9], there is a session key established in each protocol execution between a user and a foreign server. However, the key is also known to the user's home server. This is undesirable because when a roaming user is visiting a foreign server, services are actually provided by the foreign server to the user but not the home server. The home server is called in only as a guarantor for giving a promise that the user is indeed a legitimate subscriber of the home server. For example, in the WLAN Roaming, when a user accesses the Internet through a foreign server, the user may not want his home server to know which network sites he is visiting. In [6], a protocol was designed for protecting a roaming user's identity from all entities other than his home server and the serving foreign server. However, according to results from [11], a malicious server which is not communicating with the roaming user can launch an active attack to reveal the user's identity. In addition, it is recently found that both [9] and [6] cannot provide *Subscription Validation* as described below as both of them are vulnerable to the Deposit-Case Attack [12].

In this paper, we consider a key exchange protocol involving three parties: a user and two servers, namely a home server and a foreign server. In each protocol execution, the following five properties will be attained.

1. **(Server Authentication)** The user is sure about the identity of the foreign server.
2. **(Subscription Validation)** The foreign server is sure about the identity of the home server of the user.

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

3. **(Key Establishment)** The user and the foreign server establish a random session key which is known only to them and is derived from contributions of both of them. In particular, the home server should not obtain the key.
4. **(User Anonymity)** Besides the user himself and his home server, no one including the foreign server can tell the identity of the user.
5. **(User Untraceability)** Besides the user himself and his home server, no one including the foreign server is able to identify any previous protocol runs which have the same user involved.

Since this notion is very useful in the roaming scenario for users to travel from their home servers to foreign servers anonymously while at the same time establishing secure session keys with the foreign servers, we call such a scheme as an **Anonymous and Authenticated Key Exchange for Roaming Networks (AAKE-R)**. To our best knowledge, there is no protocol proposed previously which satisfies all the five properties above.

We adopt the modular approach of Canetti and Krawczyk [5] to construct an AAKE-R protocol from reusable building blocks. The protocol is not only provably secure but also network friendly. There are only four message flows and only standard cryptographic primitives are used. Hence efficient implementation of the protocol is possible by choosing appropriate primitives to use. Besides constructing an authenticated key exchange protocol for anonymous roaming, we also build one for conventional roaming (i.e. not anonymous). In fact, we start with some reusable building blocks based on the modular approach of Canetti and Krawczyk and build one for conventional roaming. Then, we extend it to a version which supports user anonymity and user untraceability. As a side-product of our construction, we propose a one-pass counter based MT-authenticator which is used as one of the building blocks in our protocols. Like other MT-authenticators, this new authenticator can be reused to build new protocols in the future.

Organization. In Sec. 2, we briefly introduce the Canetti-Krawczyk model and describe a new MT-authenticator. An authenticated key exchange protocol for roaming is then built in Sec. 3 and the anonymous version of the protocol is derived in Sec. 4. We conclude the paper in Sec. 5.

2 The CK Model and Some Reusable Building Blocks

In the Canetti-Krawczyk (CK) model [5], there is a system of n parties denoted by P_1,...,P_n, which may carry out multiple concurrent executions of a message-driven protocol over an adversary controlled network.

There are two models with respect to the adversary.

1. Authenticated-links adversarial model (AM). An AM adversary can corrupt any parties and sessions particular parties. However, the adversary is not allowed to inject or modify messages (except that the sender is corrupted or if the message belongs to an exposed session). The adversary is restricted to deliver messages faithfully, and the message is only to be delivered once

(i.e. all the messages the adversary received are in a set M of undelivered messages with format (m,P_i,P_j) where P_i and P_j are the sender and the receiver, m is some message, and once m is delivered, it is removed from M). However, the adversary can choose not to deliver it.

2. **Unauthenticated-links adversarial model (UM).** A UM adversary is essentially the same as an AM adversary but without those restrictions above on delivering messages.

For the rest of the paper, we denote an AM adversary by $\mathcal{A}$, and a UM adversary by $\mathcal{U}$. Let $AUTH_{\pi,\mathcal{A}}$ be the global output (i.e. the output of the adversary and all the parties in the system) of running π in AM and $UNAUTH_{\pi,\mathcal{U}}$ be that in UM.

Definition 1. *Let π and π' be n-party message-driven protocols in AM and UM, respectively. We say that π'* **emulates** *π in UM if for any UM-adversary $\mathcal{U}$ there exists an AM-adversary $\mathcal{A}$ such that $AUTH_{\pi,\mathcal{A}}$ and $UNAUTH_{\pi',\mathcal{U}}$ are computationally indistinguishable.*

Since the authentication in AM is explicitly ensured, if π' emulates π in UM, the authentication in UM is also ensured.

Definition 2 (Authenticator). *An authenticator $\mathcal{C}$ is an algorithm that for any protocol π in AM, the protocol $\mathcal{C}(\pi)$ emulates π in UM.*

The way to construct an authenticator is given in [2], where a layered approach is used. An authenticator $\mathcal{C}_\lambda$ can be constructed from an **MT-authenticator** λ which emulates the basic message transmission protocol. The basic idea is that whenever a party P_i wants to send or receive a message, we emulate it using λ. Readers can refer to [2] for details.

According to [5, Theorem 6], it states that if π is a SK-secure key-exchange protocol in AM and λ is a MT-authenticator, $\pi' = \mathcal{C}_\lambda(\pi)$ is a SK-secure key-exchange protocol in UM.

In the subsection below, we review some MT-authenticators and also introduce a new one which will be used to construct our protocols.

2.1 MT-Authenticators

MT-authenticators can be built from various cryptographic primitives. In case public key cryptosystems are used, it is assumed that each party has its private key and also knows the authentic public key of other parties. We also assume that each message m sent by a sender is different. This can be realized by adding a message-ID.

A Signature Based MT-Authenticator [2]

$$
\begin{aligned}
&P_i \rightarrow P_j : m \\
&P_i \leftarrow P_j : m,\ N_j \\
&P_i \rightarrow P_j : m,\ SIG_{P_i}(m, N_j, P_j)
\end{aligned}
$$

The figure above illustrates a signature based MT-authenticator for party P_i to send a message m to party P_j in an authenticated way. Let k be a security parameter, $N_j \in_R \{0,1\}^k$ be a random challenge and SIG_{P_i} be the signature generation function of P_i. The signature scheme is assumed to be secure against chosen message attack [7].

An Encryption Based MT-Authenticator [2]

$$P_i \rightarrow P_j : m$$
$$P_i \leftarrow P_j : m, ENC_{P_i}(N_j)$$
$$P_i \rightarrow P_j : m,\ MAC_{N_j}(m, P_j)$$

This figure shows an encryption based MT-authenticator. As above, $N_j \in_R \{0,1\}^k$ is a random challenge. ENC_{P_i} denotes the public key encryption function of P_i and MAC_{N_j} denotes a MAC (Message Authentication) scheme under the key N_j. It is assumed that the encryption scheme is semantically secure against chosen ciphertext attack (CCA) [8] and the MAC scheme is secure against chosen message attack.

In this MT-authenticator, after P_i obtains N_j by decrypting the ciphertext from the message sent by P_j, P_i uses N_j as an authentication key to generate the MAC value in the third message. To thwart some interleaving attack which make use of information obtained from session-state reveal queries, we require that N_j is not part of the state information of the involving session of P_i. Instead, N_j should be handled in some secure part of P_i and destroyed once it is no longer needed, that is, after generating the MAC value in the third message.

A One-Pass Counter Based MT-Authenticator

We construct a new one-pass counter based MT-authenticator and propose to use it in authenticated key exchange protocols for simplifying the authentication procedures and improving the efficiency of the underlying mobile applications. The first attempt of constructing a counter based MT-authenticator is found in [2]. However, it has been pointed out in [5] that the security proof of the authenticator in [2] cannot be true due to several shortcomings in the definition of secure KE protocols. We now build a new one and show its security. The idea follows that proposed in [2], but we use a different composition.

Let π be a key exchange protocol in UM. Suppose a party P_i shares a session key κ with another party P_j by running one copy of π. Each of P_i and P_j initiates a counter starting with 0. Our one-pass counter based MT-authenticator λ_{COUNT} proceeds as follows.

- Whenever P_i wants to send a message m to P_j, P_i increases its local counter $COUNT_{P_i}$ by one, sends $m, COUNT_{P_i}, MAC_\kappa(m, COUNT_{P_i}, P_j)$ to P_j, and adds a message "P_i sent m to P_j" to P_i's local output.
- Upon receiving $m, COUNT_{P_i}, MAC_\kappa(m, COUNT_{P_i}, P_j)$, P_j verifies that the MAC is correct and $COUNT_{P_i} > COUNT_{P_j}$ where $COUNT_{P_j}$ is the local counter of P_j. If all of the verifications succeed, P_j outputs "P_j received m from P_i" and sets $COUNT_{P_j} = COUNT_{P_i}$.

Note that the receiver ID is included in the MAC in our scheme, which is not included in [2]. Here we argue that in case the receiver ID is not included and the same key is used for communications in both directions with two counters (one for send and one for receive) at each side, reflection attack will work.

Theorem 1 (One-Pass Counter Based MT-Authenticator). *If π is a SK-Secure key exchange protocol in UM, and MAC is secure against chosen message attack, the MT protocol λ_{COUNT} described above is an MT-authenticator.*

The proof is given in Appendix A.

Remark. Note that a delayed previous message will be rejected by the receiver after a later sent message is accepted in UM, while the delayed message will still be accepted in AM, but it will not affect $\mathcal{A}$ on emulating $\mathcal{U}$.

2.2 SK-Secure Key Exchange Protocols in *AM*

The Diffie-Hellman key-exchange protocol is SK-secure under the Decisional Diffie-Hellman (DDH) assumption [4]. Let G be a subgroup of prime order q of a multiplicative group $\mathbb{Z}_p^*$. Let g be a generator of G. Below is the review of the protocol which is proven SK-secure in AM [5]. P_i and P_j are two parties and s is the session ID.

Diffie-Hellman (DH) Key Exchange in *AM*

1. On input (P_i, P_j, s), P_i chooses $x \in_R \mathbb{Z}_q$ and sends $(P_i, s, \alpha = g^x)$ to P_j.
2. Upon receipt of (P_i, s, α), P_j chooses $y \in_R \mathbb{Z}_q$ and sends $(P_j, s, \beta = g^y)$ to P_i, then computes $\kappa = \alpha^y$, erases y, and outputs the session key κ under session ID s.
3. Upon receipt of (P_j, s, β), P_i computes $\kappa' = \beta^x$, erases x, and outputs the session key κ' under session ID s.

Theorem 2 (Theorem 8 of [5]). *Under the Decisional Diffie-Hellman (DDH) assumption, Diffie-Hellman (DH) Key Exchange above is SK-secure in AM.*

3 Authenticated Key Exchange for Roaming (AKE-R)

In the following, we design a key exchange protocol in the roaming scenario which satisfies the first three properties listed in Sec. 1: Server Authentication, Subscription Validation, and Key Establishment. Thanks for the modular approach, we will see that the separation of key exchange and authentication using the modular approach makes the design work and security analysis easier.

Let k be a system-wide security parameter. Let $\mathsf{C}(k) = \{C_1, \cdots, C_{Q_1(k)}\}$ be the set of roaming users (clients) in the system and $\mathsf{S}(k) = \{S_1, \cdots, S_{Q_2(k)}\}$ be the set of servers in the system, where Q_1 and Q_2 are some polynomials and $C_i, S_j \in \{0,1\}^k$ are the corresponding identities of the parties, for $1 \le i \le Q_1(k)$ and $1 \le j \le Q_2(k)$.

Subscription. The term 'subscribe' is commonly used to describe some special relationship between a user and a server without clear definition. Based on the widely-used concept of subscription in mobile communications, we give the following definition for subscription.

Definition 3 (Subscribe). *Given a security parameter k, 'subscribe' is a computable function Subscribe from $\mathsf{C}(k)$ into $\mathsf{S}(k)$. We say that C_A is 'subscribed' to S_H if $Subscribe(C_A) = S_H$ where $C_A \in \mathsf{C}(k)$ and $S_H \in \mathsf{S}(k)$.*

We assume that each user has subscribed to one and only one server, and the subscription is persistent. Hence scenarios related to changing subscriptions of users are excluded.

Based on the terminologies of mobile communications, S_H is said to be the home server of C_A and S_V is said to be a foreign server of C_A if $S_V \neq S_H$. We also assume that the inverse $Subscribe^{-1}$ is computable. Hence for any $S_H \in \mathsf{S}(k)$, $Subscribe^{-1}(S_H)$ is the set of all $C_A \in \mathsf{C}(k)$ such that $Subscribe(C_A) = S_H$.

3.1 The Security Definition of AKE-R

An AKE-R (Authenticated Key Exchange for Roaming) protocol is a message-driven protocol. In the CK-model, each session is modelled by running a subprocess within a party with input $(P_i, P_j, P_\ell, s, role)$. We extend the CK-model so that the parties are categorized as roaming users and servers. For a user C_A, the input of his session will be in the form $(C_A, S_V, S_H, s, \mathsf{initiator})$ where the *role* must be initiator. For a server, the *role* can either be responder or credential. We say that three sessions of a user and two servers, C_A, S_V and S_H, respectively, are *3-party matching*, if in an execution of the AKE-R protocol, user C_A has a session with input $(C_A, S_V, S_H, s, \mathsf{initiator})$, server S_V has a session with input $(S_V, C_A, S_H, s', \mathsf{responder})$, server S_H has a session with input $(S_H, C_A, S_V, s'', \mathsf{credential})$, and $s = s' = s''$ and $Subscribe(C_A) = S_H$.

Definition 4 (SK-Secure AKE-R Protocol). *An AKE-R protocol π runs among C_A, S_V, and S_H is called **SK-secure** if the following properties hold.*

1. *If uncorrupted C_A, S_V and S_H complete 3-party matching sessions, then upon the completion of the protocol, C_A and S_V output the same key.*
2. *The probability that anyone except C_A and S_V guesses correctly the bit b' (i.e., $b' = b$) is no more than $1/2$ plus a negligible fraction in the security parameter.*

Having an AKE-R protocol SK-secure is not enough in practice. In particular, the definition does not capture Subscription Validation. For example, suppose we extend the two-party Diffie-Hellman key exchange protocol which is proven SK-secure in *AM* (reviewed in Sec. 2.2) to a three-party version in such a way that C_A and S_V conduct the key exchange. After completed, S_V sends the session ID to S_H. Then S_H accepts and the protocol is completed. This three-party version can be shown to be SK-secure with respect to Def. 4 but obviously not satisfying the requirement of Subscription Validation.

For Subscription Validation, S_V has to make sure that a credential issued by a server has been received claiming that C_A is subscribed to the server and is connecting to S_V. In addition, C_A has to make sure that S_V has received a credential issued by S_H.

Definition 5 (Secure AKE-R Protocol). *An AKE-R protocol π run among C_A, S_V, and S_H is secure if the following properties hold.*

1. *π is SK-secure.*
2. *Upon the completion of the 3-party matching sessions,*
   (a) *the matching session of S_H has sent the message below to S_V;*
   "C_A is subscribed to S_H and is talking to S_V"
   (b) *the matching session of S_V has received the message below from S_H;*
   "C_A is subscribed to S_H and is talking to S_V"
   (c) *the matching session of S_V has sent the message below to C_A;*
   "S_H claimed that C_A is its subscriber and is talking to S_V"
   (d) *the matching session of C_A has received the message below from S_V.*
   "S_H claimed that C_A is its subscriber and is talking to S_V"

Remark: Based on Def. 5 above, if a server S_H outputs the message no matter the actual home server of C_A is S_H or not, then it relies on C_A to check (item (d) above) if S_H is cheating. This helps detect the Deposit-Case Attack [12].

In the following, we state an important theorem which allows us to reuse all the proven MT-authenticators given in Sec. 2 for constructing secure AKE-R protocols.

Theorem 3. *Let $\mathcal{C}_\lambda$ be an authenticator (Def. 2) constructed from an MT-authenticator λ exemplified in Sec. 2. If π is a secure AKE-R protocol in AM, then $\pi' = \mathcal{C}_\lambda(\pi)$ is a secure AKE-R protocol in UM.*

Proof is given in Appendix B.

We now start describing our AKE-R protocol. Our protocol consists of two steps. In the first step, a user carries out a *Pre-authentication* protocol with his home server when he is in the network operated by his home server. In this step, a 'long-term' *user authentication key* will be established. This key will be used in the second step for authenticating the user.

In the second step, the user is roaming and communicating with a foreign server. The roaming key exchange protocol will be carried out by the user, the foreign server and the home server. The purpose of the protocol is to let the roaming user and the foreign server establish a fresh session key so that a secure channel can be built.

3.2 Pre-authentication

The purpose of pre-authentication is to have C_A and S_H establish a *user authentication key*, $authK_A$. One way to achieve the task is to run a SK-secure key exchange protocol. Alternatively, like in the cellular networks, the key has

already been embedded in the SIM card. No matter in which of these two cases, we assume that $authK_A$ is randomly chosen and only shared by C_A and S_H.

After running the pre-authentication protocol, C_A stores $authK_A$ and a *counter* initialized to 0 in some secure and non-volatile memory location. S_H creates an entry for C_A in its own database and sets the attribute of $authK_A$ as the primary key of the database. In the entry for C_A, other attributes such as the identity of C_A and a counter value are included. The counter is also initialized to 0, and will be increased in each run of the AKE-R Main protocol below.

3.3 The AKE-R Main Protocol

We first describe our AKE-R Main Protocol in AM. Then we *compile* it to a secure AKE-R protocol in UM using those authenticators described in Sec. 2.1.

A Secure AKE-R Protocol in *AM*

We extend the two-party DH key exchange protocol which is SK-secure in AM (Sec. 2.2) to a three-party variant. Since the network is controlled by the adversary, here we use a virtual link between A and H although they are not physically linked in our target applications. In the following, we describe the AKE-R protocol in AM in an informal way. We think that the presentation below can give readers a better picture of the protocol. But it can always be converted to the form according to CK-model.

Extended DH Protocol in *AM*: (Fig. 1)

1. A roaming user C_A initiates the protocol execution by choosing $x \in_R \mathbb{Z}_q$ and sends $(C_A, S_V, s, \alpha = g^x)$ to S_H.
2. Upon receipt of (C_A, S_V, s, α), S_H checks if C_A is its subscriber, if not, it rejects and halts. Otherwise, S_H sends (S_H, C_A, s, α) to S_V.
3. Upon receipt of (S_H, C_A, s, α), S_V checks if S_H is a legitimate server in its server list, if not, it rejects and halts. Otherwise, S_V chooses $y \in_R \mathbb{Z}_q$, sends $(S_V, S_H, s, \beta = g^y)$ to C_A, then computes $\kappa = \alpha^y$, erases y, and outputs the session key κ under session ID s.
4. Upon receipt of (S_V, S_H, s, β), C_A checks if S_H is indeed its home server. If not, it rejects and halts. Otherwise, it computes $\kappa' = \beta^x$, erases x, and outputs the session key κ' under session ID s.

Corollary 1. *Under the DDH assumption, the Extended DH protocol is SK-secure in AM if S_H is uncorrupted and honest.*

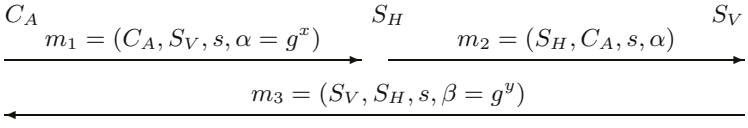

Fig. 1. Extended DH protocol in AM.

Proof. For the first condition in Def. 4, it is easy to see that if the protocol completes, and S_H is uncorrupted and behaves honestly, C_A and S_V share the same key. It is also straight forward that the second condition of Def. 4 is satisfied by following Theorem 2. □

Corollary 2. *Under the DDH assumption, the Extended DH protocol is secure in AM.*

Proof. Trivial. □

A Secure AKE-R Protocol in *UM*

An AKE-R protocol in UM can be derived by applying MT-authenticators to the Extended DH protocol in AM. We apply the one-pass counter based MT-authenticator to m_1, the signature based MT-authenticator to m_2, and the encryption based MT-authenticator to m_3. The resulting protocol is in Fig. 2.

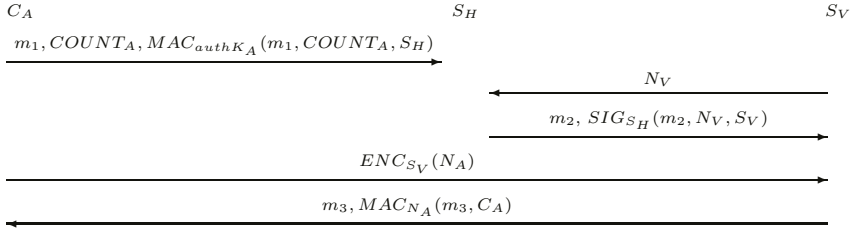

Fig. 2. Extended DH protocol in UM.

After deriving the AKE-R protocol in UM, an optimization [10] of message flows can be applied. And the final AKE-R protocol in UM is illustrated in Fig. 3.

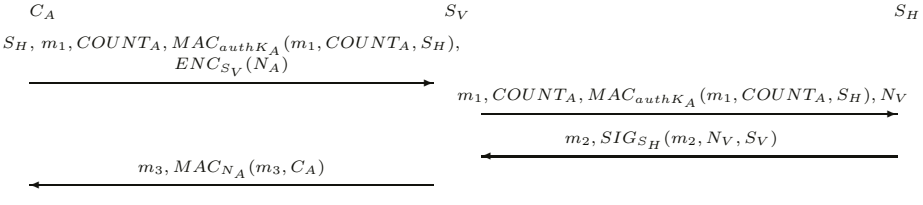

Fig. 3. Optimized Extended DH protocol in UM.

Remark: Note that the identity of the home server S_H is added in the first message from C_A to S_V (Fig. 3). This is required as the one-pass counter based MT-authenticator to m_1 from C_A is not sent directly to S_H. Instead, it is now relayed by S_V to S_H. This is because in our target applications, we assume that C_A is roaming in a foreign network and does not have a direct link with S_H. Therefore, C_A has to tell S_V the identity of his home server, which is S_H.

4 Anonymous and Authenticated Key Exchange for Roaming (AAKE-R)

We now start specifying the security definition of the *anonymous* version of an AKE-R protocol. This version of AKE-R protocol will satisfy all the five properties listed in Sec. 1. In particular, the protocol should provide User Anonymity and User Untraceability. We call such a protocol an Anonymous and Authenticated Key Exchange for Roaming (AAKE-R) protocol.

The adversarial model is based on the adversarial model UM defined in CK-model [5, Sec. 3.2]. The adversary controls the communications of the system and is free to choose any scheduling of activations and action requests of the AAKE-R protocol. The adversary can also adaptively corrupts parties in the system using corrupt a party query, session-state reveal query, and session-key reveal query. By following the notations of [5], we denote the adversary by $\mathcal{U}$.

4.1 The Security Definition of User Anonymity and Untraceability

Game A: "The game is carried out by a simulator $\mathcal{S}$ which runs an adversary $\mathcal{U}$. It is based on the adversarial model UM.

1. $\mathcal{S}$ sets up a system with users in $\mathsf{C}(k)$ and servers in $\mathsf{S}(k)$.
2. $\mathcal{S}$ then runs $\mathcal{U}$ and answers $\mathcal{U}$'s queries.
3. $\mathcal{U}$ can execute the AAKE-R protocol on any parties in the system by activating these parties and making queries.
4. Among all the parties in the system, $\mathcal{U}$ picks two users $C_i, C_j \in \mathsf{C}(k)$ and two servers $S_V, S_H \in \mathsf{S}(k)$ such that $Subscribe(C_i) = Subscribe(C_j) = S_H$.
5. $\mathcal{U}$ sends a test query by providing C_i, C_j, S_V and S_H.
6. The simulator $\mathcal{S}$ tosses a coin b, $b \stackrel{R}{\leftarrow} \{0,1\}$. If $b = 0$, the simulator emulates one AAKE-R protocol run among C_i, S_V and S_H. Otherwise, a protocol run among C_j, S_V and S_H are emulated. The simulator then returns to $\mathcal{U}$, all messages generated in the emulated protocol execution.
7. After receiving the response of the test query, $\mathcal{U}$ can still launch all the allowable attacks through queries and also activate parties for protocol executions as before.
8. At the end of $\mathcal{U}$'s run, it outputs a bit b' (as its guess for b)."

$\mathcal{U}$ wins the game if (1) S_H is uncorrupted. Note that C_i, C_j and S_V can be corrupted (so they are not in the restriction). (2) $\mathcal{U}$ guesses correctly the bit b (i.e. outputs $b' = b$.). Define $\mathbf{Adv}_{\pi,\mathcal{U}}(k)$ to be the probability that $\mathcal{U}$ wins the game.

Definition 6 (User Anonymity and Untraceability). *For sufficiently large security parameter k, $\boldsymbol{Adv}_{\pi,\mathcal{U}}(k)$ is negligible.*

The formulation of Def. 6 is very powerful and can be shown to ensure both user anonymity and user untraceability required by a good AAKE-R protocol.

4.2 A Secure AAKE-R Protocol

To provide user anonymity, the identity of the user should not be sent in clear. In addition, it should not be known to the foreign server according to the anonymity definition above. To do so, we first change the identity of C_a in m_1 and m_2 in Fig. 3 to a *temporary ID* or an *alias* which is a fixed-length binary string chosen randomly from $\{0,1\}^k$. Then, put $authK_A$ into m_1 so that S_H is able to use $authK_A$ as the search key to get C_A's information from S_H's subscriber database. Also, encrypt the one-pass counter based MT-authenticator using S_H's public key encryption function E_{S_H}. Finally, since S_V does not know the real identity of C_A, the identity of the receiver in the MAC of the encryption based MT-authenticator (i.e. the last message in Fig. 3) has to be changed to *alias* which essentially identifies the initiator of the protocol execution.

In addition, for anonymity, all the counters used in the system should have the same length. We define a counter $COUNT \in \{0,1\}^{Q_3(k)}$ for some polynomial Q_3 and assume that the value of $COUNT$ would not reach $2^{Q_3(k)} - 1$ in the lifetime of the system.

The complete AAKE-R main protocol is illustrated in Fig. 4.

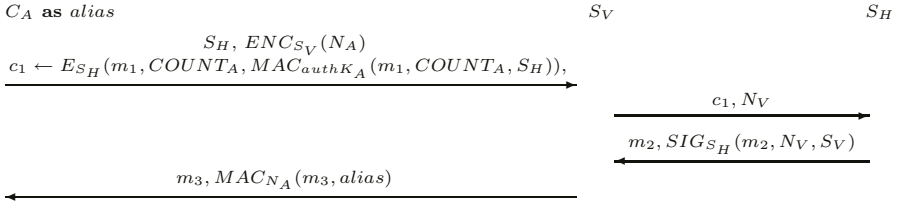

Fig. 4. The AAKE-R Main Protocol.

Theorem 4. *If E_{S_H} is CCA-secure, $\boldsymbol{Adv}_{\pi,\mathcal{U}}(k)$ is negligible.*

Proof is given in Appendix C.

5 Conclusions

Based on the modular approach of the CK-model, we build an anonymous and authenticated key exchange protocol for roaming which is provably secure and efficient. Our scheme requires only four message flows and is the first provably secure key exchange protocol for anonymous roaming. As a side-product from our modular construction, we propose a one-pass counter based MT-authenticator and show its security under the assumption that there exists a secure MAC against chosen message attack. Like other proven secure MT-authenticators, this new authenticator will also be reused to construct new protocols in the future.

References

1. G. Ateniese, A. Herzberg, H. Krawczyk, and G. Tsudik. On traveling incognito. In *Proc. of the IEEE Workshop on Mobile Systems and Applications*, Dec 1994.

2. M. Bellare, R. Canetti, and H. Krawczyk. A modular approach to the design and analysis of authentication and key exchange protocols. In *Proc. 30th ACM Symp. on Theory of Computing*, pages 419–428. ACM, May 1998.
3. Mihir Bellare and Phillip Rogaway. Entity authentication and key distribution. In *Proc. CRYPTO 93*, pages 232–249. Springer-Verlag, 1994. LNCS 773.
4. D. Boneh. The decision Diffie-Hellman problem. In *Proc. of the Third Algorithmic Number Theory Symposium*, pages 48–63. Springer-Verlag, 1998. LNCS 1423.
5. R. Canetti and H. Krawczyk. Analysis of key-exchange protocols and their use for building secure channels. In *Proc. EUROCRYPT 2001*, pages 453–474. Springer-Verlag, 2001. LNCS 2045. `http://eprint.iacr.org/2001/040/`.
6. J. Go and K. Kim. Wireless authentication protocol preserving user anonymity. In *Proc. of the 2001 Symposium on Cryptography and Information Security (SCIS 2001)*, pages 159–164, January 2001.
7. S. Goldwasser, S. Micali, and R. Rivest. A digital signature scheme secure against adaptive chosen-message attack. *SIAM J. Computing*, 17(2):281–308, April 1988.
8. C. Rackoff and D. R. Simon. Non-interactive zero-knowledge proof of knowledge and chosen ciphertext attack. In *Proc. CRYPTO 91*, pages 433–444. Springer, 1992. LNCS 576.
9. D. Samfat, R. Molva, and N. Asokan. Untraceability in mobile networks. In *Proc. of MobiCom '95*, pages 26–36, 1995.
10. Y. Tin, C. Boyd, and J. Gonzalez-Nieto. Provably secure key exchange: An engineering approach. In *Australasian Information Security Workshop (AISW2003)*, 2003.
11. D. Wong. Security analysis of two anonymous authentication protocols for distributed wireless networks. In *Proc. of the 3rd IEEE Intl. Conf. on Pervasive Computing and Communications Workshops (PerCom 2005 Workshops)*, pages 284–288. IEEE Computer Society, March 2005.
12. G. Yang, D. Wong, and X. Deng. Deposit-case attack against secure roaming. In *Information Security and Privacy, 10th Australasian Conference, ACISP 2005*. Springer, 2005. LNCS.

A Proof of Theorem 1

We start with the following lemma.

Lemma 1. *For sufficiently large security parameter k, if κ is replaced by an unknown random key from $\{0,1\}^k$ and MAC is secure against chosen message attack, λ_{COUNT} is an MT-authenticator.*

Proof. Here we assume that for the two communicating parties, one of them always has the role of initiator and the other always has the role of responder. But the proof also applies to the case when the communication is in bidirectional and both directions are using the same shared key but with two different counters (one for send and one for receive) at each side.

Let $\mathcal{U}$ be a UM adversary interacts with λ_{COUNT}. We define an AM adversary $\mathcal{A}$ which simulates $\mathcal{U}$ as follows.

$\mathcal{A}$ runs $\mathcal{U}$ on a simulated interaction with a set of parties running λ_{COUNT}. First, $\mathcal{A}$ chooses and distributes keys for the imitated parties, note that in this

case, the shared secret keys are chosen and distributed. Then $\mathcal{A}$ proceeds its simulation as follows.

1. When $\mathcal{U}$ activates an imitated party $\tilde{P}_i$ for sending a message m to imitated party $\tilde{P}_j$, $\mathcal{A}$ activates P_i in AM for sending m to P_j.
2. When some imitated party $\tilde{P}_j$ outputs "$\tilde{P}_j$ received m from $\tilde{P}_i$", $\mathcal{A}$ activates party P_j in AM with incoming message m from P_i.
3. When $\mathcal{U}$ corrupts a party, $\mathcal{A}$ corrupts the same party in AM and hands the corresponding information (from the simulated run) to $\mathcal{U}$.
4. $\mathcal{A}$ outputs whatever $\mathcal{U}$ outputs.

Let $\mathbf{E}$ denote the event that imitated party $\tilde{P}_j$ outputs "$\tilde{P}_j$ received m from $\tilde{P}_i$" where $\tilde{P}_i$ is uncorrupted and the message (m, P_i, P_j) is not currently in the set M of undelivered messages. In other words, either P_i is not activated for sending m to P_j, or P_j has already had the same output before. Since neither $\tilde{P}_i$ nor $\tilde{P}_j$ is corrupted, $\mathbf{E}$ is also the event that either $\tilde{P}_i$ is not activated for sending m to $\tilde{P}_j$, or $\tilde{P}_j$ has already had the same output before.

If $\mathbf{E}$ never occurs, then $AUTH_{MT,\mathcal{A}}$ and $UNAUTH_{\lambda,\mathcal{U}}$ are equally distributed. It remains to show that $\mathbf{E}$ occurs only with negligible probability.

We prove it by contradiction. Assume $\mathbf{E}$ occurs with non-negligible probability, then we construct a forger $\mathcal{F}$ that breaks the MAC with non-negligible probability.

The forger $\mathcal{F}$ has a MAC oracle O_{MAC} that uses an unknown random key, $\mathcal{F}$ can request O_{MAC} on any message or any verification pair (m, σ). The task of $\mathcal{F}$ is to produce a valid (m, σ) pair but m has not been queried to the oracle before.

$\mathcal{F}$ starts by running $\mathcal{U}$ on a set of parties running λ_{COUNT}. $\mathcal{F}$ chooses and distribute shared keys between parties with one exception, $\mathcal{F}$ randomly chooses one pair of users $\tilde{P}_i$ and $\tilde{P}_j$, and whenever one party is required to produce or verify an MAC for some value, $\mathcal{F}$ queries the oracle and hands the result to that party. If $\mathcal{U}$ chooses to corrupt either $\tilde{P}_i$ or $\tilde{P}_j$, $\mathcal{F}$ fails and aborts.

Note that running λ_{COUNT} in this case is equivalent to a regular run. Assume $\mathbf{E}$ occurs with probability $\upsilon(k)$, the probability it occurs between $\tilde{P}_i$ and $\tilde{P}_j$ is then $\frac{2\upsilon(k)}{n(n-1)}$ since $\tilde{P}_i$ and $\tilde{P}_j$ are randomly chosen. Also note that when $\mathbf{E}$ occurs between $\tilde{P}_i$ and $\tilde{P}_j$, neither $\tilde{P}_i$ nor $\tilde{P}_j$ is corrupted.

In the case $\tilde{P}_i$ is not activated for sending m to $\tilde{P}_j$, $\mathcal{F}$ outputs the MAC value $\mathcal{U}$ delivered to $\tilde{P}_j$ in the last message as its forgery. On the other hand, if $\tilde{P}_j$ has already had the same output before, then $\tilde{P}_j$ has received m from $\tilde{P}_i$ with a counter, say $COUNT^{old}_{\tilde{P}_i}$ before. Now when $\tilde{P}_j$ accepts m the second time, then $\tilde{P}_j$ must have accepted another incoming message from $\tilde{P}_i$ with the same m but with a more updated counter value, say $COUNT^{new}_{\tilde{P}_i}$ such that $COUNT^{new}_{\tilde{P}_i} > COUNT^{old}_{\tilde{P}_i}$. However, $\tilde{P}_i$ ($\mathcal{F}$) has never queried with $(m, COUNT^{new}_{\tilde{P}_i}, \tilde{P}_j)$ because each message from $\tilde{P}_i$ to $\tilde{P}_j$ is assumed to be different. Hence $\mathcal{F}$ outputs the MAC value that $\mathcal{U}$ has delivered to $\tilde{P}_j$ in the last message as its forgery as well. Thus $\mathcal{F}$ successfully produces a forgery with probability $\frac{2\upsilon(k)}{n(n-1)}$. □

Since the SK-security of the key-exchange protocol guarantees that the session key is indistinguishable from a random key, Theorem 1 follows directly from Lemma 1 above. □

B Proof of Theorem 3

Proof. We prove it by showing that the following requirements are satisfied.

1. If π is SK-Secure, then π' is also SK-Secure:
   The proof follows that of Theorem 6 in [5] directly. Since that proof only requires π' emulates π, which is done due to the definition of $\mathcal{C}_\lambda$, and also it does not depend on the parties involved in the protocol. Therefore, if π is SK-Secure, $\pi' = \mathcal{C}_\lambda(\pi)$ "inherits" this property from π.
2. If π satisfies requirement 2 of Def. 5 in AM, π' satisfies the requirement in UM as well:
   Suppose π' does not satisfy this requirement. Then there exists an adversary $\mathcal{U}$ in UM such that the global output of running π' with $\mathcal{U}$ does not follow the requirement, but there has no adversary in AM can do this since π satisfies the requirement. Thus the global outputs are distinguishable, in contradiction to $\mathcal{C}_\lambda$ being an authenticator. □

C Proof of Theorem 4

Proof. We prove it by contradiction. Namely, if the protocol is not anonymous, that is, if $\mathcal{U}$ wins the game with non-negligible advantage, $\upsilon(k)$, over random guess (which is half chance), we construct a distinguisher $\mathcal{D}$ to break E_{S_H}.

We start by describing a game for the distinguisher $\mathcal{D}$. First, $\mathcal{D}$ adaptively queries a decryption oracle with any ciphertext. Then $\mathcal{D}$ chooses two messages msg_1 and msg_2 and asks the game simulator for a ciphertext. The simulator randomly picks $b \stackrel{R}{\leftarrow} \{0,1\}$ and gives $\mathcal{D}$ the ciphertext c such that $c = E_{S_H}(msg_b)$. After receiving c, $\mathcal{D}$ adaptively queries the decryption oracle with any ciphertext except c. $\mathcal{D}$ is to output a value $b' \in \{0,1\}$ as its guess for b.

Now we construct $\mathcal{D}$ which simulates Game A. First, $\mathcal{D}$ sets up the system appropriately by creating a set $\mathsf{C}(k)$ of users and another set $\mathsf{S}(k)$ of servers. It then initializes all the users in $\mathsf{C}(k)$ with randomly chosen authentication keys from $\{0,1\}^k$ and randomly chosen counter values from $\{0,1\}^{Q_3(k)}$, and initializes all the servers in $\mathsf{S}(k)$ with randomly chosen public key pairs for encryption and another set of public key pairs for signature. Afterwards, $\mathcal{D}$ randomly picks a server, S_H, and replace its encryption public key corresponding to E_{S_H}. Let $COUNT_{min}$ be the smallest counter value initialized for the users in $\mathsf{C}(k)$.

$\mathcal{D}$ runs $\mathcal{U}$ and answers all its queries and emulates all the responses of party activation due to protocol execution. If $\mathcal{U}$ picks S_H as the home server, two users C_i, C_j such that $Subscribe(C_i) = Subscribe(C_j) = S_H$, and some server S_V as the foreign server during the test query, $\mathcal{D}$ answers the query by providing the transcript of a protocol run constructed as follows.

First, $\mathcal{D}$ randomly chooses a in $\mathbb{Z}_q$, $alias$ in $\{0,1\}^k$, a session ID $s \in_R \{0,1\}^k$, and constructs two messages msg_1 and msg_2 as follows.

$$msg_1 = alias \mid\mid S_V \mid\mid s \mid\mid authK_{C_i} \mid\mid COUNT_{min} - t \mid\mid g^x$$
$$msg_2 = alias \mid\mid S_V \mid\mid s \mid\mid authK_{C_j} \mid\mid COUNT_{min} - t \mid\mid g^x$$

where $t \in_R \{0,1\}^{Q_3(k)}$ such that $COUNT_{min} - t \geq 0$. Note that the counter value is always encoded into a $Q_3(k)$-bit binary string. $\mathcal{D}$ queries the CCA-security simulator with msg_1 and msg_2. Suppose the CCA-security simulator returns a ciphertext c. Then, $\mathcal{D}$ constructs

$$message_1 = \langle\ S_H, ENC_{S_V}(N_A), c\ \rangle$$
$$message_2 = \langle\ c, N_V\ \rangle$$
$$message_3 = \langle\ S_H, alias, s, g^x, SIG_{S_H}(S_H, alias, s, g^x, N_V, S_V)\ \rangle$$
$$message_4 = \langle\ S_V, S_H, s, g^y, MAC_{N_A}(S_V, S_H, s, g^y, alias)\ \rangle.$$

where $N_A, N_V \in_R \{0,1\}^k$, ENC_{S_V} is S_V's public key encryption function, and Sig_{S_H} is the signature generation function of S_H.

The transcript returned by $\mathcal{D}$ to $\mathcal{U}$, as the response for $\mathcal{U}$'s test query, is $(message_1, message_2, message_3, message_4)$. $\mathcal{D}$ continues the game by answering all the queries made by $\mathcal{U}$ and emulating all the responses of party activation due to protocol execution. When $\mathcal{U}$ outputs a bit value b' as its guess, $\mathcal{D}$ outputs b' and halts.

If $\mathcal{U}$ does not pick S_H as the home server in his test query, $\mathcal{D}$ just randomly picks a value $b' \stackrel{R}{\leftarrow} \{0,1\}$, outputs it and halts.

Analysis: Let $\mathbf{E}$ be the event that $\mathcal{U}$ picks S_H as the home server in its test query. Since $\mathcal{D}$ chooses S_H from $\mathsf{S}(k)$ in the game uniformly, $\Pr[\mathbf{E}] = \frac{1}{Q_2(k)}$. Hence we have

$$\begin{aligned}\Pr[\mathcal{D} \text{ guesses } b \text{ correctly}] &= (\frac{1}{2} + \upsilon(k))\Pr[\mathbf{E}] + \frac{1}{2}(1 - \Pr[\mathbf{E}]) \\ &= \frac{1}{2} + \frac{\upsilon(k)}{Q_2(k)}\end{aligned}$$

which is non-negligible. □