One-Pass Key Establishment Model and Protocols for Wireless Roaming with User Anonymity*

Yuan Wang[1], Duncan S. Wong[2], and Liusheng Huang[1]
(Corresponding author: Yuan Wang)

School of Computer Science and Technology, University of Science and Technology of China, No.96, Jinzhai Road, Hefei 230026, P.R.China[1]
Department of Computer Science, City University of Hong Kong, Kowloon, Hong Kong[2]
(Email:wya@mail.ustc.edu.cn, duncan@cityu.edu.hk, lshuang@ustc.edu.cn)

(Received Sep 23, 2011; revised and accepted Feb. 1, 2012)

Abstract

Key Establishment for Wireless Roaming (KE-WR) is expected to ensure a mobile user to establish a fresh session key with a foreign WSP and also roam from one foreign network domain to another while enjoying the roaming services. However, so far there is no ID-based KE-WR protocol proposed in the literatures with a formal security proof in an appropriate model. The main work of this article address the current gap by first proposing a variation of classic CK and eCK model to support the wireless roaming scenario, which is called rCK model. This article extend classic security model by introducing the simulation of broadcast query and multiple Key Generation Center scenario and also giving the re-defined session definitions and additional adversary capability. Second, this article proposes a novel suite of One-Pass Key Establishment Protocols for Wireless Roaming.

Keywords: One-Pass key establishment, user anonymity, security model, wireless roaming

1 Introduction

High-speed, low-cost, ubiquitous Internet access has been dreamed by human beings for decades and attracts extensive attention from both academia and industry. Nowadays, with the advancement of wireless mobile technology, broadband wireless Internet access has been readily available across a limited local area through IEEE 802.11 (or commonly called Wi-Fi) based wireless hotspots, also known as public Wireless LANs, especially in densely populated areas such as airport, subway, cafes and libraries. However, comparing with the traditional CDMA cellular networks, a typical Wi-Fi hotspot has a much smaller coverage and more restricted scalability thus may limit the moving range of mobile users in Wireless LANs [10]. To achieve interoperable and cost-effective large-scale city-wide wireless access without impairing the performance of throughput, the deployment of metropolitan-area Wireless Mesh Networks (WMNs) seems to be a great strategy. The metropolitan-area WMNs are expected to accommodate thousands of self-managed network domains operated by numerous different Wireless Service Providers (WSPs), which are mainly composed of a number of publicly or privately owned Mesh Access Points (MAPs; supporting either existing Wi-Fi hotspot or emerging technologies such as WiMAX) and a Mesh Gateway (MGW) as the interface to connect the MAPs with the public Internet. As a result, portable end users can access the network via freely roaming among these network domains to enjoy much broader network coverage than WLANs.

Generally, A number of popular roaming solutions, such as Global System for Mobile Communications (GSM), Universal Mobile Telecommunication System (UMTS) and Mobile IP Networks, all depend on an authentication architecture called home-foreign-domain architecture [10]. A wireless user U first subscribes to a WSP, says *home WSP* H, and signs up an account in order to have access to the wireless Internet services via the MAPs managed by H [15]. However, U has no roaming access to the wireless service when moves into the coverage of any other WSP called *foreign WSP* F unless H has a default bilateral agreement with F that specify each other's users can have access to the MAPs managed by itself. This contractual roaming model works well in cellular networks, however, it has two main drawbacks mak-

*This paper is an extended version of our paper which has been published in the proceedings of IEEE International Conference on Communications (ICC), with a detailed security analysis included.

ing it less suitable for practically realizing roaming service in metropolitan-area WMNs. First, it often involved potentially time-consuming and expensive execution of an authentication protocol among a user, its home WSP H and the foreign WSP F. As the user base grows large, the overall network signalling overhead would be significant. Second, a bilateral service level agreement (SLA) has to be established between each pair of WSPs to permit user roaming among them, which can be very difficult to set up between large number of WSPs[6]. To avoid the establishment of numerous pairs of SLAs, an alternative model is proposed [8, 18] to having each WSP established an agreement with a trusted third party (i.e. a Centralized Authorization (CA)). The tradeoff of this model is that it may consequentially run into the risk of making the CA as the performance and security bottleneck.

Recently, *Localized Roaming architectures* have been reported in some studies (i.e. [12, 17]) for inter-network roaming service. In contrast to the conventional solutions above, it localizes the authentication process between U and F so that it eliminates the need for establishing bilateral SLAs and the deployment of centralized CAs. In [17], each WSP plays as a CA and each subscriber authenticates itself to F using a certificate issued by H. And also in both protocols proposed in [12], an authenticated key establishment can be performed between U and F without contacting any third party such as home WSP H, which can significantly reduce the computation and signalling overload and eliminate centralized CAs. In these two protocols, the number of message flows is reduced to three and therefore has the communication efficiency improved. In addition, they support Perfect Forward Secrecy (PFS), namely, all previous established session keys will remain secure even after the long-term keys of U and F are compromised. Furthermore, they are secure against Key-Compromise Impersonation (KCI), namely, no attacker is able to complete a protocol run successfully with U (resp. F) after compromising U's (resp. F's) long-term key.

As a notable security issue especially in wireless communication, privacy protection for a roaming user has become an increasingly demanding requirement. Since eavesdropping is much easier to launch but more difficult to be detected when given the open nature of radio media, it is desirable to keep mobile users' identities and whereabouts anonymous. We consider two levels [12] on user privacy protection: (1) **Normal User Anonymity** concerns about keeping U's identity from being known by eavesdroppers or anyone else except F and H in the system *after the first protocol run between U, F and possibly H as well*; (2) **Strong User Anonymity** concerns about further keeping U anonymous from F *even in the first protocol run between U, F and possibly H as well.* Existing GSM and 3GPP roaming protocols provide Normal User Anonymity. In the first protocol run, the real ID of U is sent in clear, while for all subsequent protocol runs with the same F, a random temporary identity called TMSI (Temporary Mobile Subscriber Identity) is used by U for hiding its real identity from eavesdroppers and anyone else except F and H. In [11], Yang et al. proposed a roaming protocol achieving Strong User Anonymity based on home-foreign-domain architecture, and yet the protocol efficiency is low, requiring at least 8 message flows to complete one protocol run.

As a basic research topic, one of the most essential problems associated with WMNs is how to provide a protocol in letting U gain roaming access from F authentically, and especially along with the considerations on the limitations about the wireless environments, such as the limited computation power and battery capacity of portable devices. In this paper, we suppose three main properties should be satisfied for a roaming protocol, in terms of *security assurance*, *privacy protection*, and *cost-effectiveness*. 1) For security assurance, after two communicating parties U and F execute the scheme and when both parties terminate and accept, each of them should have certain assurance that it shares a fresh session key only with its intended partner. In addition, this session key should be secure against both of passive and active attacks, such as eavesdropping, replay attack, and man-in-the-middle attack. Besides, this session should also be unaccessible for home WSP H; 2) For user privacy protection, no eavesdropper or any entity in the system can find out the identity of U from a protocol run except F and U's home server H. For cost-effectiveness, considering the *imbalanced* network architecture which consists of the limited end devices of users and the powerful servers of WSPs, the protocol should be as lightweight as possible at U side with both light computation load and small number of message flows in order to reduce latency and save energy, while the heavy computation load at WSP side may be acceptable. We call such a protocol as a ***Key Establishment Protocol for Anonymous Wireless Roaming*** **(KE-AWR)**. Below we reviews some previous works on KE-AWR.

1.1 Related Work

The traditional solution of KE-AWR is by use of an *Authentication, Authorization, and Accounting* (AAA) server at WSP side which coordinates the login requests from the subscriber of itself. When a roaming event occurs, U is required to present an *evidence* according to the information distributed by AAA server of H, which appears unintelligible to eavesdroppers. F then forwards the *evidence* to the AAA server of H which then verifies the validity and freshness of the *evidence*. Some previous works on secure anonymous wireless roaming follow this approach, for instance, [11, 13, 16]. Unfortunately, these protocols based on AAA architecture require the involvement of all the three parties and at least four message flows for completing one protocol run thus subject to a significant amount of signalling overhead. On the other hand, some academic works, such as [8, 18], has been proposed based on an alternative model that each WSP only needs to have an agreement with a same trusted

Certificate Authority (CA) instead of a pairwise bilateral trust relationship for each pair of WSPs. However, it consequentially runs the risk of making the CA to become the bottleneck when numerous authentication requests are processing.

In [11], Yang et al. proposed a KE-AWR achieving Strong User Anonymity, and yet the protocol efficiency is low, requiring at least 8 message flows to complete one protocol run. In [12], Yang et al. proposed two *two-party* KE-AWR protocols achieving Normal User Anonymity and Strong User Anonymity, respectively. Both protocols are based on the *localized roaming authentication* architecture and thus reduce the authentication latency and eliminate centralized CAs. These protocols still require three message flows and incur heavy computation.

1.2 Overview of Our Contribution

To achieve secure and efficient roaming service, this paper proposes a novel suite of *One-Pass Key Establishment Protocols for Wireless Roaming* (OP-KE-WR for short) in WMNs. Our OP-KE-WRs are performed between the roaming user U and the foreign WSP F without intervention of any other third party (i.e. the home WSP H), and only one message flow from U to F is needed (so called One-Pass) while by making use of the broadcast wireless communication, F broadcasts some commonly agreed public parameters (i.e. the identity and the public key of F in most application scenarios) which are shared by all users communicating with F. We propose two distinct OP-KE-WRs: First we propose Protocol I so that a fresh and secure session key only known by U and F is established in each run of protocol and no other entity in WMNs (including H) is able to compute the session key. It also supports pre-computation at U side. Actually, most computational operations of U can be precomputed if U knows the public key of F and it leaves almost no operation to be performed online for U. The protocol achieves secure key establishment and high efficiency. However, it does not support the desirable properties that multi-round key establishment protocols usually do [11, 12, 13], namely, it only supports *partial* forward secrecy and *partial* KCI. Consequently, in Protocol II, we solve the problems by supporting all the desirable properties mentioned. Namely, the protocol can support both *Perfect Forward Secrecy* (PFS) and *Perfect Key Compromise Impersonation* (Perfect KCI) and so called OP-KE-WR with PFS. The total computational complexity of Protocol II is comparable to that of previous protocol. However, as a tradeoff between efficiency and security, it needs an additional online Bilinear Pairing operation for U during the runtime of the protocol.

In addition, We find that both CK and eCK models do not provide the security consideration in wireless roaming environment: 1) there is no available broadcast query in CK or eCK models. 2) they do not consider the involvement of multiple KGC servers. Therefore, refer to [1, 5], we propose a variation of CK model to support our ID-based one-pass protocols for wireless roaming. And We call this new model as rCK model. We also give the formal security proof in our rCK model for each proposed protocol.

Furthermore, we extend our OP-KE-WRs to support *User Anonymity* (i.e. One-Pass KE-AWR). The level of user anonymity achieved in our protocol is comparable to that of current mobile systems such as 3GPP and GSM, namely, no other entity except home and foreign WSPs can associate the IDs to any particular user, or telling if two roaming sessions are corresponding to the same user or not. Besides, our protocols are very suitable for the *imbalanced* architectures in wireless network environments, namely, the WSP is considered as a much more powerful server than the mobile device held by user. The protocols focus on lightening the computational load on user side, while the heavy load on WSP side is acceptable. Table 1 compares the main features of all three OP-KE-WRs along with some existing multi-round KEPs for wireless roaming.

2 Preliminaries and Security Requirements

2.1 Preliminaries and Assumptions

A bilinear pairing is a map $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_T$ where $\mathbb{G}_1$ and $\mathbb{G}_T$ are cyclic groups of large prime order q. The map $\hat{e}$ satisfies the following properties: (1) Bilinear: $\hat{e}(aP, bQ) = \hat{e}(P, Q)^{ab}$ for all $P, Q \in \mathbb{G}_1$ and $a, b \in \mathbb{Z}_q$; (2) Non-degenerate: $\hat{e}(P, P) \neq 1$ (the identity element of $\mathbb{G}_T$) for a generator P of $\mathbb{G}_1$ and (3) Computable: $\hat{e}(P, Q)$ can be computed in polynomial time for all $P, Q \in \mathbb{G}_1$.

The (Computational) Bilinear Diffie-Hellman (BDH) problem is to compute $\hat{e}(P, P)^{xyz}$, when the adversary is given a random instance $\langle P, xP, yP, zP\rangle$, where P is an arbitrary generator of $\mathbb{G}_1$ and $x, y, z \in_R \mathbb{Z}_q^*$.
(Computational) BDH Assumption: The advantage of a probabilistic polynomial time (PPT) adversary $\mathcal{A}$ in solving the Computational BDH problem defined below is negligible.

$$Adv_{C-BDH}^{\mathcal{A}} = \Pr[\mathcal{A}(P, xP, yP, zP) = \hat{e}(P, P)^{xyz}]$$

2.2 Desirable Security Properties

Reference to [11, 13, 14], a Key Establishment Protocol for Anonymous Roaming should satisfy the following security properties after successfully being carried out:

1) Secure Key Establishment: a fresh and random session key should be established between roaming user and the roaming target server. The session key should also maintain the following security attributes:

- Known-Key Security: The knowledge of a session key should not enable an adversary to compromise other

Table 1: Comparison among different KE-WR protocols

	#P	#F	PFS	KCI	ANON	Online. OPs
Zhang05 [16]	3	6	¬	¬	N	-
Yang05 [13]	3	4	√	¬	S	3 ECSM
Yang07 [11]	3	8	√	¬	S	$\geq$ 4 ECSM
YHW 1 [12]	2	3	√	√	N	$\geq$ 2 BP+ 1.25 ECSM
YHW 2 [12]	2	3	√	√	S	$\geq$ 3 BP + 8 ECSM
Protocol I	2	1	P	P	N	2 ECSM
Protocol II	2	1	√	√	N	1 BP

#P : Parties Involved
KE : Key Establishment
KCI : KCI Security
ECSM : Elliptic Curve Scalar Multiplication
P : Partially Satisfied
N : Normal User Anonymity
¬ : Not Considered
#F : Number of Flows
PFS : Perfect Forward Secrecy
ANON : User Anonymity
BP : Bilinear Pairing Operation
√ : Fully Satisfied
S : Strong User Anonymity

session keys.

- Unknown-key Share: An entity A should not be leading to the situation that A believes that she is sharing the key with entity B while actually she is sharing the key with another entity C.
- Forward secrecy: If long-term private keys of one or more of the communication participants are compromised, the secrecy of previously established session keys should not be affected. We say that a system has *partial* Forward Secrecy if the compromise of one (or more but not all) of the entities' long-term keys can be corrupted without compromising previously established session keys, and we say that a system has *Perfect* Forward Secrecy (PFS) if the long-term keys of all the entities involved may be corrupted without compromising any session key previously established by these entities.
- Key Compromise Impersonation: The compromise of an entity As long-term private key will allow an adversary to impersonate A, but it should not enable the adversary to impersonate other entities in presence of A.

2) User Anonymity and Untraceability: no other entity except home and the visiting foreign server can associate the transferred IDs to any particular user, or telling if two roaming sessions are corresponding to the same user or not.

2.3 Scope of The Paper

In this paper, we do not have the ambition to solve all related security problems in WMNs. In particular, we just consider the security attacks aims at authentication and key establishment in wireless roaming scenario. Namely, how to deal with the communication and key establishment between users, though important, is not addressed in this paper. Moreover, we will not investigate the DoS type attacks such as physical-layer jamming or routing disruption.

3 One-Pass Secure Roaming

In this section, A One-Pass Key Establishment Protocol is proposed for inter-network secure roaming in WMNs based on Localized Roaming Architecture and aim to achieve secure session key agreement with only one message flow from the mobile user U to the foreign WSP F. It also supports pre-computation at U side so that most computational operations of U can be pre-computed, that is, it leaves only two Scalar Multiplication operations to be performed online for U. Below we first give the detailed description of the protocol and analysis its security properties. A novel formal model which is suitable for id-based roaming scenario is proposed in Section 4 as well as the formal security proof.

3.1 Protocol Description

SYSTEM SETUP. On the network architecture, we consider that there are multiple Wireless Service Providers (WSPs) in a collection of interconnected WMNs. Each WSP is considered as a server and may manage a number of Wi-Fi hotspots and each hotspot can be considered as the MAP of a wireless network domain. For simplicity, we assume that U can directly communicate with the server of F via these MAPs. There is also a PKI (Public Key Infrastructure) where the CA issues certificates $Cert_{WSP}$ with respect to the public keys of WSPs. Each wireless user in the system is subscribed to one and only one WSP, and the subscription is persistent. Hence the scenarios related to changing subscription of users are not considered in this paper. Suppose that a wireless user is denoted as U while the server that U is subscribed to is called the home server H of U. For all other WSPs, they are foreign servers. Without loss of generality, we denote the foreign server that U is communicating with as F. Due to the nature of wireless communication, we assume that a broadcast channel is available in each wireless network domain. In practice, a server always needs to notify users in its coverage by periodically broadcasting the server's public parameters via MAPs, possibly including its public key and certificate, in a wireless network domain.

Notation Definitions. Let $ID_U, ID_H, ID_F \in \{0,1\}^*$ be the identities of roaming user U, home server H and foreign server F, respectively. As shown in Table 2, $\mathbb{G}_1$ is denoted as an additive group of prime order q, while $\mathbb{G}_T$ as a multiplicative group. P is a commonly agreed generator of $\mathbb{G}_1$. Let $H : \{0,1\}^* \rightarrow \mathbb{G}_1$ be a cryptographic hash function. All of these are system-wide public parameters. Here we assume that these public parameters $(\mathbb{G}_1, \mathbb{G}_T, q, P, H)$ have been shared by all the entities involved in WMNs including roaming users and servers.

Long-term Key Generation. 1) For server H (resp. F), a random number $s_H \in_R \mathbb{Z}_q$ (resp. $s_F \in_R \mathbb{Z}_q$) is selected and the public/private key pair of H (resp. F) is set as $(PK_H = s_H P, s_H)$ (resp. $(PK_F = s_F P, s_F)$) where P is a commonly agreed generator of $\mathbb{G}_1$. A certificate on (ID_H, PK_H) (resp. (ID_F, PK_F)) is then obtained from the CA. 2) For each user U with identity ID_U, say a subscriber of a server H, after proving its identity, U obtains a user secret key $S_U = s_H H(ID_U)$ from H during subscription. Please refer to Table 2 for the notations used.

Table 2: Notations used in Protocol I

$\mathbb{G}_1$	an additive group
$\mathbb{G}_T$	a multiplicative group
$\hat{e}$	a bilinear map $\mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_T$
q	the prime order of $\mathbb{G}_1$
P	a generator of $\mathbb{G}_1$
H	hash function $\{0,1\}^* \rightarrow \mathbb{G}_1$
ID_H	identity of home server
ID_F	identity of foreign server
ID_U	identity of roaming user
(PK_H, s_H)	the public/private key pair of home server
(PK_F, s_F)	the public/private key pair of foreign server
S_U	user private key of roaming user
T	a timestamp

Protocol I. Suppose that a user U of H roams and is about to connect to the network domain via the MAP operated by a foreign server F. As a requirement of time synchronization, we assume that U and F have their clocks loosely synchronized by having each device synchronize its clock to time information received from the GPS, or some atomic clock services available over the Internet. Below is the description of the One-Pass Key Establishment Protocol which is illustrated in Figure 1.

- Step 0: As mentioned above, a server always notify users in its coverage by periodically broadcasting beacon signal via MAPs, along its public parameters such as the public key and certificates. Therefore it is reasonable to assume that U has received F's public parameters $(PK_F, Cert_F)$ once U moves into the radius coverage of F's hotspots. In this case, U can easily ensure the security of the public key of F after validating the certificate on CA's public key. U will accept the broadcast massage if and only if the validation passes. Otherwise, U will abandon the parameters and wait for new valid ones. Note that when doing this, U might be standing on the overlap range in the coverage of two servers and still get connected with previous server. Consequently, We consider this step as a part of system setup and denote it as Step 0.

- Step 1: To establish a fresh session key with F, U randomly chooses $r \in \mathbb{Z}_q^*$, and computes $X = rP$. After that U computes a shared secret as $K_{U,F} = \hat{e}(rH(ID_U) + S_U, PK_F)$, then erases r from its memory. U sends a login request $\langle ID_U, ID_H, X, T\rangle$ where T is a current timestamp . Subsequently, U generates the session key SK by using the following: $SK \leftarrow KDF(K_{U,F}, ID_U||ID_H||ID_F||T||X)$ where KDF is a key derivation function.

- Step 2: On receiving login request $\langle ID_U, ID_H, X, T\rangle$ from U at time T', F first checks the validity of the time interval between T' and T_{now}. F will rejects this message if $(T' - T_{now}) > \delta T$, where δT denotes the expected valid time interval for transmission delay. if valid, F will continue the session key computation. SK, as the session key, are calculated by using key derivation function: $SK \leftarrow KDF(K_{F,U}, ID_U||ID_H||ID_F||T||X)$, where $K_{F,U}$ is obtained by the following computation: $K_{F,U} = \hat{e}(s_F H(ID_U), X + PK_H)$.

3.2 Security Properties Analysis

We proposed a new One-Pass KE-WR which requires fewer message flows and achieves higher online efficiency when compared with existing protocols. We now analysis the security properties of Protocol I in detail as below:

Secure Key Establishment. A One-Pass KE-WR is said to securely establish a session key between U and F if U (resp. F) is assured that only F (resp. U) is able to obtain a fresh session key. We give the detailed analysis in terms of "Key Security against H" and "Key Security against Adversary".

1) *Key Security against H.* Suppose H is an honest-but-curious home server and tries to compute $K_{U,F}$ or $K_{F,U}$ from the message flow between U and F. Though H knows s_H, H cannot compute $K_{U,F}$ as H cannot compute the terms $\hat{e}(H(ID_U), P)^{rs_F}$ without knowing $\langle r, s_F\rangle$ since the discrete logarithm problem that deriving r from $X = rP$ is intractable.

2) *Key Security against Adversary.* In the following, we consider that an active adversary $\mathcal{A}$ which has full control of the communication between U and F but has neither S_U (i.e. s_H) nor s_F is trying to establish a session key with F in presence of U. To do so, $\mathcal{A}$

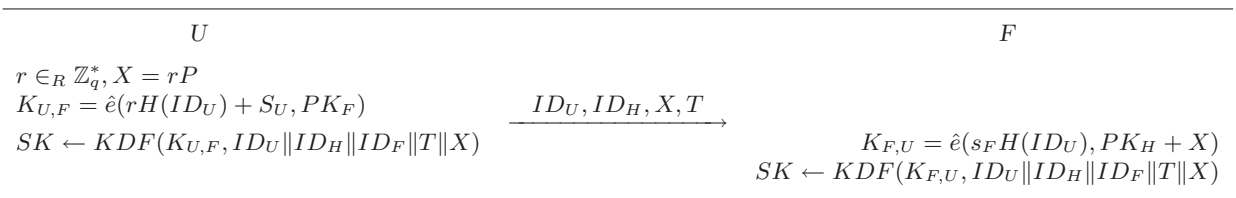

Figure 1: Protocol I: One-Pass KE-WR using IBS

needs to find $K = \hat{e}(H(ID_U), P)^{s_F(r+s_H)}$. Similar with H, $\mathcal{A}$ cannot compute both components $K = \hat{e}(H(ID_U), P)^{rs_F}$ and $K = \hat{e}(H(ID_U), P)^{s_F s_H)}$. Note that in an impersonation attack, $\mathcal{A}$ may randomly choose r' and compute $X' = r'P$ through impersonating U. However, in this case, $\mathcal{A}$ still cannot compute $K = \hat{e}(H(ID_U), P)^{s_F s_H)}$ as $\mathcal{A}$ does not know s_F, s_H or the discrete logarithm of $H(ID_U)$ under the assumption that Bilinear Diffie-Hellman (BDH) Problem is intractable. Therefore, the protocol can also prevent impersonation attack.

Known-Key Security. For any PPT adversary $\mathcal{A}$ which is holding a compromised session key and the transcript of the corresponding protocol run, $\mathcal{A}$ is not able to impersonate U and compute a new session key. The reason is that for each new session, a new timestamp value, say T_{new} is generated by F and used in computing the new session key. Hence, even if $\mathcal{A}$ replays X_{old} and σ_{old} from the compromised session at timestamp T_{new}, the new session key $SK_{new} \leftarrow KDF(K_{old}, ID_U \| ID_H \| ID_F \| T_{new} \| X)$ has to be computed by $\mathcal{A}$ which is different from the old session key $SK_{old} \leftarrow KDF(K_{old}, ID_U \| ID_H \| ID_F \| T_{old} \| X)$ established at timestamp T_{old} with overwhelming probability under the random oracle assumption of KDF [3].

Unknown Key-share. In the OP-KE-WR, we adopt a suitable key derivation function KDF (e.g. [3]) to derive a session key SK from a shared secret K instead of using K directly. Otherwise, an adversary $\mathcal{A}$ who has captured the message flow of a session and also compromised the session key can launch replay attack easily share a session key with F in presence of U even without compromising U's private key.

Partial Key Compromise Impersonation. This protocol satisfies partial KCI that it prevents adversary $\mathcal{A}$ from impersonating F after compromising U's long-term private key S_U. In order to find K when communicating with U in presence of F, $\mathcal{A}$ needs to know either random number r or F's private key s_F. As analyzed above, both of them cannot be accessible for $\mathcal{A}$. Therefore, $\mathcal{A}$ which has already compromised S_U, is still not able to find K through impersonating F. However, revealing s_F will allow $\mathcal{A}$ to launch an impersonation attack to F in presence of any user.

Partial Forward Secrecy. As shown above, even if $\mathcal{A}$ has compromised S_U, it cannot compromise any session key by launching the user-key-compromise impersonation. Therefore, our OP-KE-WR protocol also supports user forward secrecy, namely the old session keys will still be secure even if $\mathcal{A}$ has compromised U's long-term private key [4]. However, the leakage of s_F will allow $\mathcal{A}$ to compromise all previously established session key. This means that the protocol achieves partial Forward Secrecy but not Perfect Forward Secrecy (PFS)[1].

4 Formal Security Proof

This section presents a verification of CK model,called rCK model, to support wireless roaming scenario. A formal security proof of Protocol I in the rCK security model is also proposed.

4.1 Security Model

The model proposed by Canetti-Krawczyk (CK model for short) [1] is now widely regarded as one of the acceptable formal model under which the security of a key establishment protocol can be analyzed. It provides a security consideration on the session-state leakage. By specifying the session-state to be the ephemeral private key, an extended version (eCK model) was defined by LaMacchia et al. [5]. The eCK model captures several properties such as resistance to the key compromise impersonation attacks, partial forward secrecy. Recently, Cremers [2] drew a separation result among some variants of CK model (including CK and eCK models) which stated none of them was stronger than the others. However, both CK and eCK models do not provide the security consideration in wireless roaming environment: 1) As in most wireless application, a server may always periodically broadcast its public parameters to notify users in its coverage. However, there is no available broadcast query in CK or eCK models. 2) In roaming scenario, It becomes more common that a user which is a subscriber of server KGC_1 needs to authenticated itself to server KGC_2. In CK or eCK models, they do not consider the involvement of multiple KGC servers. Therefore, refer to [1, 5], we now propose a verification of CK model to support our ID-based one-pass protocols for wireless roaming. And We call this new model as rCK model.

Parties. In our rCK model, there are a collection of $n(\theta)$ $User$ and $m(\theta)$ $Server$ where $\theta \in \mathbb{N}$ is a security parameter, and $n(\cdot)$ and $m(\cdot)$ are polynomial. We consider each

[1] *Perfect forward secrecy* in the identity-based cryptographic setting requires that even if both of two communication parties' long-term private keys are corrupted, previously established session keys should remain secure.

instance of $Server$ as a KGC server. And every $User$ is a subscriber of one of $Server$. Each of these parties has a unique identity. For simplicity, we denote $\hat{U}, \hat{H} \in \{0,1\}^*$ as an individual instance in $User$ and $Server$ respectively, where $\hat{U}$ is a subscriber of $\hat{H}$. And we denote $\hat{F} \in \{0,1\}^*$ as the expecting communication peer with $\hat{U}$. Then a protocol π is modeled as running the proposed OP-KE-WR between $\hat{U}$ and $\hat{F}$. As specified in Section 2.3, in roaming scenario, we consider the communication is restricted only between $User$ and $Server$.

Sessions. We call a particular instantiation of the protocol executed by either $\hat{U}$ or $\hat{F}$ an OP-KE-WR session. A party can be activated to execute the protocol by an incoming message with the following form:i)A broadcast message $(\hat{F}, \perp)$ or ii)A send message $(\hat{U}, \hat{F}, msg)$. Different from CK or eCK model, $\hat{U}$ can only be activated by $(\hat{F}, \perp)$ and be the *session initiator* while $\hat{F}$ can only be activated by $(\hat{U}, \hat{F}, msg)$ and be the *session responder*. Each of initiator sessions and responder sessions is identified via a session identifier *sid*.

Definition 1 (Session Identifier). *A session identifier is define to consist of the identities of two participants and the information they send. Specifically, A session is identified by a session identifier* $sid = (role, ID_1, ID_2, msg)$*, where* $role \in \{I, R\}$ *(initiator or responder),* $ID_1 \in \{0,1\}^*$ *is the identities of the participant executing the session,* $ID_2 \in \{0,1\}^*$ *is the identities of the intended communication peer, and* msg *is the message transferred in the protocol.*

Besides, we assume that each participants has a $SecVal$ which preserves the status of session. When activated, the party $\hat{U}$ or $\hat{F}$ initialize its own $SecVal$ to 0. Once successfully finishing the calculation of session key, it sets $SecVal$ to a non-zero value.

Definition 2 (Complete Sessions). *A session is said to be a completed session if and only if* $SecVal$ *is a non-zero value.*[2]

Definition 3 (Matching Sessions). *For a legitimate protocol run, two* ***completed*** *sessions sid and* sid^* *are said to be match if and only if the participant identity* $\hat{U}$ *and* $\hat{F}$*, and communication message* msg*, such that the session identifier of sid is* $(I, \hat{U}, \hat{F}, msg)$ *and the session identifier of* sid^* *is* $(R, \hat{F}, \hat{U}, msg)$.

Adversary capability. We consider a probabilistic polynomial-time adversary $\mathcal{A}$ is modeled as a probabilistic Turing machine and can make oracle queries to above honest parties. That is, the adversary $\mathcal{A}$ has full control of the communications links between the protocol participants,i.e., it can listen to all the transmitted information, decide what message will reach their destination and when, change these message at will or even inject its own generated messages. $\mathcal{A}$ is given the ability to reveal the session key for any specified session, and the private keys for both $User$ parties and $Server$ parties. In addition, $\mathcal{A}$ is also allowed to register fictitious users and servers. Here, to formalize the adversary capability, we allow $\mathcal{A}$ to make the following queries during the protocol run:

- $Broadcast(\hat{F}, \perp)$: This query allows $\mathcal{A}$ to broadcast the public parameters such as identity and pubic key in presence of $\hat{F}$.
- $Send\,(ID_1, ID_2, msg)$: This query allows $\mathcal{A}$ to send a message msg to ID_1 in presence of ID_2 and the response of ID_1 is return to $\mathcal{A}$. If msg is an empty message λ and ID_1 is $\hat{U}$, It actives $\hat{U}$ as the session initiator. Otherwise, if msg is a non-empty message and ID_1 is $\hat{F}$, it makes the role of $\hat{F}$ as the session responder.
- $Establish\,(ID)$: This query allows $\mathcal{A}$ to register a fictitious user or server party with arbitrary selected identity ID, and obtain the corresponding static private key. Parties against whom $\mathcal{A}$ did not issue this query are called *honest*.
- $Reveal\,(sid)$: This query allows $\mathcal{A}$ to obtain the value of session key for the complete session *sid*.
- $EpKeyReveal\,(sid)$: This query allows $\mathcal{A}$ to ephemeral private key possessed by a session *sid*.
- $LtKeyReveal\,(ID)$: This query allows $\mathcal{A}$ to obtain the long-term private key for selected party with identity ID.

In addition, $\mathcal{A}$ is allowed, at any time during its entire execution, to selects a completed session *sid*, and makes one and only once $Test\,(sid)$ query. In response to this query, a challenge value is given to $\mathcal{A}$.

- $Test\,(sid)$: This query tries to capture the advantage of $\mathcal{A}$ to tell apart a real session key from a random bit string. On this query, the challenge (the owner of session sid) picks a random $b \in \{0,1\}$. if b = 0 it returns the session key in the session *sid* to $\mathcal{A}$. Otherwise, a random l_{sk}-bit (l_{sk} is the length of the session key) string is returned if $b = 1$.

$\mathcal{A}$ is allowed to continue its execution after the Test query. The entire execution terminates as soon as $\mathcal{A}$ outputs its guess b'. $\mathcal{A}$ wins the game if the selected test session is *clean* and $b' = b$.

Definition 4 (Clean Sessions). *Let sid be the session identifier of a completed session, owned by* $\hat{U}$ *with communication peer* $\hat{F}$*, and denote by* sid^* *the matching session (if exists). We say that an OP-KE-WR session is not clean if* $\mathcal{A}$ *can trivially compute the corresponding session key. Namely, a session sid is said* ***not to be clean*** *if any of following conditions holds:*

1) $\hat{U}$ *or* $\hat{F}$ *is an adversary controlled party (issued by Extract query before session execution);*

[2] In case of one-pass scenario, there is only one message msg transmitted during the protocol run which means that the initiator will not receive any respondence after sending msg. Therefore, a session of a one-pass protocol is considered to be completed as long as the session key SK is computed by the session executor.

2) $\mathcal{A}$ issues a Reveal(sid) or a Reveal(sid^*) query;

3) $\mathcal{A}$ issues either of the following queries:
- Both $LtKeyReveal(\hat{U})$ and $EpKeyReveal(sid)$, or
- $LtKeyReveal(\hat{F})$

Definitions 5 (rCK Security). *The advantage of the adversary $\mathcal{A}$ in the experiment of rCK model with protocol π is defined as*

$$Adv_{\pi}^{rCK}(\mathcal{A}) = \Pr[Suc_{\mathcal{A}}] - 1/2.$$

We say that an OP-KE-WR is secure under the rCK model if following conditions hold:

1. If two honest and uncorrupted parties complete matching sessions, then they could calculate the same session key.

2. The PPT adversary $\mathcal{A}$ has no more than a negligible advantage to distinguish a real session key from a random bit string. Namely, $Adv_{\pi}^{rCK}(\mathcal{A})$ is negligible.

4.2 Security Proof

This section proposes a formal security proof for our one-pass protocol in the rCk security model.

Theorem 1 (rCK-security of OP-KE-WR). If H is a random oracles, KDF is a random, non-collision key derivation function and $\mathbb{G}_1$ is a finite group where the BDH assumption holds, Then One-Pass KE-WR is secure in the rCK model. More precisely, if there is a PPT bounded adversary $\mathcal{A}$ can break the rCK-security of OP-KE-WR with non-negligible advantage $Adv_{\pi}^{rCK}(\mathcal{A})$, which involves at most $n(\theta)$ users and $m(\theta)$ servers, and activates at most $k(\theta)$ sessions, where $\theta \in \mathbb{N}$ is a security parameter and $n(\cdot)$, $m(\cdot)$, $k(\cdot)$ are polynomial, Then we can construct a PPT attacker $\mathcal{S}$ who can solve the BDH challenge with probability

$$Adv^{BDH}(\mathcal{S}) \geq max\{\frac{1}{k(\theta)n(\theta)m(\theta)^2}, \frac{1}{k(\theta)^2n(\theta)m(\theta)}\}Adv_{\pi}^{rCK}(\mathcal{A}) \quad (1)$$

Proof: To prove the above theorem, we start by assuming that the adversary $\mathcal{A}$ can distinguish the session key from a random key and break the rCK-security of our protocol with non-negligible advantage. Observing that the adversary $\mathcal{A}$ has only two ways to distinguishing SK from a random string: forging attack or key-replication attack. However, due to the randomness and non-collisions of KDF query, by a key-replication attack, $\mathcal{A}$ can win the game no better than tossing a coin. Therefore, at some point $\mathcal{A}$ must issue KDF query for a specific input and perform a forging attack. Next, we show that if $\mathcal{A}$ can perform a successful forging attack, then we can use $\mathcal{A}$ as a subroutine to constructed an efficient BDH solver $\mathcal{S}$. Let $\langle P, xP, yP, zP\rangle$ be the BDH challenge given to $\mathcal{S}$. Then the BDH solver $\mathcal{S}$ whose goal is finding $\hat{e}(P,P)^{abc}$, is simulated as below:

$\mathcal{S}$ randomly picks up two sets of identities: $\{U_i|1 \leq i \leq n(\theta)\}$ for $n(\theta)$ users and $\{S_j|1 \leq j \leq m(\theta)\}$ for $m(\theta)$ servers. Also let $k(\theta)$ be the maximum number of sessions activated by $\mathcal{A}$. $\mathcal{S}$ chooses $\hat{U} \in \{U_1, U_2...U_{n(\theta)}\}$, $\hat{H} \in \{S_1, S_2...S_{m(\theta)}\}$, $\hat{F} \in \{S_1, S_2...S_{m(\theta)}\}$. Let tid denote the test session chosen by $\mathcal{A}$ and owned by the honest party $\hat{U}$ with honest communication peer $\hat{F}$, and let tid^* denote the matching session (if exists) of tid. In addition, the queries of $\mathcal{A}$ are answered by $\mathcal{S}$ as follows:

H(U_i) Query: $\mathcal{S}$ starts with an empty list $\mathbb{L}_{\mathcal{H}}$ and on the input U_i it first checks to see if there is entry for it in $\mathbb{L}_{\mathcal{H}}$. If so, it returns the stored value to $\mathcal{A}$. Otherwise, $\mathcal{S}$ chooses $h_i \in_R \mathbb{Z}_q^*$ and return h_iP to $\mathcal{A}$. Then the entry (U_i, h_i, h_iP) is stored in $\mathbb{L}_{\mathcal{H}}$. If $U_i = \hat{U}$, $\mathcal{S}$ returns zP and adds $(\hat{U}, \perp, zP)$ to the list.

KDF Query: In this query, a list $\mathbb{L}_{\mathcal{K}}$ is maintained to ensure previously asked input would receive the same answer. On a fresh query, $\mathcal{S}$ randomly picks a value and returns it to $\mathcal{A}$.

Extract/LtKeyReveal($ID, role$) Query: This query is handled as below based on the input of $role$:

1. $role = R(responder/server)$. $\mathcal{S}$ maintains a list $\mathbb{L}_{\mathcal{R}}$ to ensure the consistency of its response. for any new input, say S_j, $\mathcal{A}$ selects $s_j \in_R \mathbb{Z}_q^*$, returns (s_j, s_jP) to $\mathcal{A}$[3]. The input and output are stored in $\mathbb{L}_{\mathcal{R}}$.

2. $role = I(initiator/user)$. $\mathcal{S}$ checks if there is an entry of ID in $\mathbb{L}_{\mathcal{H}}$. On no match, say U_i, it first makes a **H(U_i)** query and retrieves h_iP from $\mathbb{L}_{\mathcal{H}}$. Then it randomly select s_j from $\mathbb{L}_{\mathcal{R}}$ and returns the tuples $(s_j, s_j(h_iP)$ to $\mathcal{A}$. The tuples $(s_j, s_jP, U_i, s_j(h_iP)$ is stored in a new list $\mathbb{L}_{\mathcal{I}}$ as an entry. For $ID = \hat{U}$, $\mathcal{S}$ randomly chooses s_j from $\mathbb{L}_{\mathcal{R}}$ and returns $(s_j, s_j(zP))$.

Broadcast(S_j) Query: On the input of S_j, $\mathcal{S}$ first look over $\mathbb{L}_{\mathcal{E}}$ for matching record. if S_j is fresh, $\mathcal{S}$ makes a **Extract** query and return the public key s_jP to $\mathcal{A}$. Also note that though an **Extract** query on $\hat{F}$ will leads to abortion of execution, a Broadcast($\hat{F}$) query is available.

Send(ID_1, ID_2, msg) Query: This query is handled as below:

1. if $ID_1 = \hat{U}, ID_2 = \hat{F}$ and the session identifier $sid = tid$. In this case, $\mathcal{S}$ will response in different ways based on the complementary events. We will discuss this case in detail later.

2. For all other sessions (including all other sessions between $\hat{U}$ and $\hat{F}$), say between U_i and S_j, if $(msg = \lambda)$, $\mathcal{S}$ chooses a random number $r_{i,j} \in \mathbb{Z}_q^*$ and returns $X = r_{i,j}s_jP$. The session is then marked as completed after issuing this query; Otherwise $(msg \neq \lambda)$, $\mathcal{S}$ accepts the session and marks it as completed.

Reveal(sid) Query: If it is the session between $\hat{U}$ and $\hat{F}$, and $sid = tid/tid^*$, $\mathcal{S}$ aborts this simulation and fails. In other cases, $\mathcal{S}$ returns the session key with the knowledge of the static and ephemeral private keys.

EpKeyReveal(sid) Query: If it is the session between $\hat{U}$ and $\hat{F}$, and $sid = tid/tid^*$, $\mathcal{S}$ aborts this simulation

[3]Note that here an **Extract/LtKeyReveal** query on input $\hat{F}$ is considered to be against the Definition 4. On such a query, $\mathcal{S}$ will abort the execution.

and fails. In other cases, $\mathcal{S}$ returns the ephemeral private keys to $\mathcal{A}$.

Test(sid) Query: If $sid \neq tid$, or the session sid is not *clean*, $\mathcal{S}$ aborts the simulation. Otherwise, $\mathcal{S}$ has to return the session held in the session sid or a random string after flipping a coin. However, $\mathcal{S}$ cannot compute the real session key, which requires solving the BDH problem,. Hence it returns a random string.

For events that $\mathcal{A}$ perform a successful forging attack to and correctly guess the private bit b involved in Test Query, we consider the following complementary conditions according to the clean definition:

- The adversary $\mathcal{A}$ does not issue neither $LtKeyReveal(\hat{F})$ and either of following happens:
  - $\mathcal{A}$ does not issue $LtKeyReveal(\hat{U})$ - Event E_1;
  - $\mathcal{A}$ issues $LtKeyReveal(\hat{U})$ but does not issue $EpKeyReveal(sid)$ - Event E_2

Event E_1: In this case, $\mathcal{S}$ sets PK_H to xP and PK_F to yP and simulates the protocol executions and oracles queries by following the description above. It is perfect, since $\mathcal{A}$ can not detect the simulation without issuing $LtKeyReveal$ queries for two participants.

With probability at least $1/n(\theta)m(\theta)^2$ $\mathcal{A}$ chooses honest user $\hat{U}$ as the initiator of tid with peer $\hat{F}$ and server $\hat{H}$ as $\hat{U}$'s home server. If $\mathcal{A}$ wins the game, $\mathcal{S}$ can solve the BDH as follows: $\mathcal{S}$ randomly picks an item $(K, \hat{U}||\hat{H}||\hat{F}||T||X)$ in list $\mathbb{L}_{\mathcal{K}}$ (at most $k(\theta)$ items), and computes $\rho = K/\hat{e}(zP, PK_F)^r$, then answers the BDH challenge with ρ. Thus $\mathcal{S}$ can solve the challenge with advantage:

$$Adv^{BDH}(\mathcal{S}) \geq \frac{1}{k(\theta)n(\theta)m(\theta)^2} Adv_{\pi}^{rCK}(\mathcal{A}) \quad (2)$$

Event E_2: In this case, $\mathcal{S}$ sets PK_F to yP. $\mathcal{S}$ simulates the protocol executions and oracles queries by following the description above except for sessions tid. For simulations of tid, by modifying the Send query, $\mathcal{S}$ embeds $X = xP$. $\mathcal{A}$ can not detect the simulation due to the definition of events E_2.

With probability at least $1/k(\theta)$, $\mathcal{A}$ picks tid as the test session, and with probability at least $1/n(\theta)m(\theta)$ $\mathcal{A}$ chooses $\hat{U}$ as the session owner and server $\hat{F}$ as the communication peer. If $\mathcal{A}$ wins the game, $\mathcal{S}$ can solve the BDH as follows: $\mathcal{S}$ randomly picks an item $(K, \hat{U}||\hat{H}||\hat{F}||T||X)$ in list $\mathbb{L}_{\mathcal{K}}$ (at most $k(\theta)$ items), and computes $\rho = K/\hat{e}(zP, PK_F)^{s_H}$ ($\mathcal{S}$ can retrieve the value of s_H from $\mathbb{L}_{\mathcal{R}}$), then answers the BDH challenge with ρ. Thus $\mathcal{S}$ can solve the challenge with advantage:

$$Adv^{BDH}(\mathcal{S}) \geq \frac{1}{k(\theta)^2 n(\theta)m(\theta)} Adv_{\pi}^{rCK}(\mathcal{A}) \quad (3)$$

Overall analysis. Combining the result of equation 2 to 1, the advantage of $\mathcal{S}$ to solve the BDH challenge is

$$Adv^{BDH}(\mathcal{S}) \geq max\{\frac{1}{k(\theta)n(\theta)m(\theta)^2}, \frac{1}{k(\theta)^2 n(\theta)m(\theta)}\} Adv_{\pi}^{rCK}(\mathcal{A}) \quad (4)$$

At the beginning of proof, we assume that $Adv_{\pi}^{rCK}(\mathcal{A})$ is non-negligible. Therefore, $\mathcal{S}$ can solve the BDH challenge with an advantage which is also non-negligible. It leads to a contradiction to the BDH assumption. Hence, there exists no polynomially bounded adversary that succeeds in breaking rCK-security of our one-pass protocol with non-negligible advantage. □

5 One-Pass Secure Roaming with PFS

As analyzed in Section 3.2, previous protocol only support *partial* forward secrecy and *partial* KCI. Consequently, in this section, we solve the security problems by proposing a novel protocol that supporting both *Perfect Forward Secrecy* (PFS) and *Perfect Key Compromise Impersonation* (Perfect KCI).

5.1 Protocol Description

The system setup and long-term key generation are the same as described in Protocol I. Please recall Section 3.1 for detail. In addition, an extra Broadcast phase is introduced in this protocol.

Broadcast Phase. Suppose that the whole service time is divided into time periods. For each time period i, there is an index, TP_i. For example, a time period can be set to 30 seconds and i is counted from a particular starting time, say, 00:00 on January 1st, 2011. At the beginning of a time period TP_i, each server, say F, erases the old ephemeral key a_{i-1} used in TP_{i-1} from its memory and selects a new random number $a_i \in_R \mathbb{Z}_q$ as the ephemeral key for TP_i. Then F computes $A_i = a_i P$ and broadcast (TP_i, A_i, PK_F) together with the certificate of (ID_F, PK_F) during the time period TP_i.

Protocol II. Let the current time period be TP_τ. Suppose that user U, as a subscriber of H, is roaming to a network domain operated by a foreign server F. Figure 2 shows the protocol.

1) Step 0: Given the open nature of wireless communication, it is reasonable to assume that F has already broadcast (TP_τ, A_τ, PK_F) and its certificate to all users who are in the radius coverage of its hotspots. In this case, U can easily ensure the security of the public key of F after validating the certificate.

2) Step 1: U checks if the current system time is in time period TP_τ and the certificate is valid with respect to (ID_F, PK_F). If not, U rejects this broadcast message and waits for a new one. Otherwise, U randomly picks $b \in_R \mathbb{Z}_q$, computes $K_{U,F} = \hat{e}(bH(ID_U) + S_U, PK_F + A_\tau)$ and sends $\langle ID_U, ID_H, B = bP, TP_\tau \rangle$ to F and erases b from its memory. Subsequently, U generates the session key $SK \leftarrow KDF(K_{U,F}, TP_\tau \| ID_U \| ID_H \| ID_F \| A_\tau \| B)$ where KDF is a key derivation function [7].

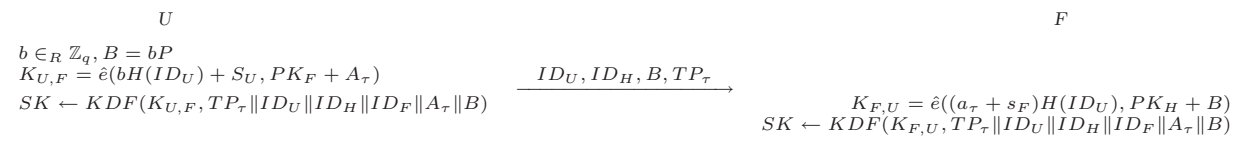

Figure 2: Protocol II: One-Pass KE-WR with PFS

3) Step 2: Upon receiving the message, F checks if the current time period is TP_τ. If not, F rejects it. Otherwise, F computes $K_{F,U} = \hat{e}((a_\tau + s_F)H(ID_U), PK_H + B)$ and $SK \leftarrow KDF(K_{F,U}, TP_\tau \| ID_U \| ID_H \| ID_F \| A_\tau \| B)$. F stores SK as the session key shared with U.

5.2 Security Analysis

We now analyze the proposed Protocol II in terms of Perfect Forward Secrecy (PFS), the security of Perfect Key-Compromise Impersonation (Perfect KCI).

Perfect Forward Secrecy (PFS). Suppose that adversary $\mathcal{A}$ has compromised all the long-term secret keys, i.e. S_U, s_F and s_H, but no ephemeral keys. For any previous session key, say established in TP_i, $\mathcal{A}$ is to find

$$\begin{aligned} K_{U,F} &= \hat{e}(H_1(ID_U), P)^{(b+s_H)(s_F+a_i)} \\ &= \hat{e}(H_1(ID_U), P)^{bs_F + ba_i + s_H s_F + s_H a_i} \end{aligned} \quad (5)$$

However, $\mathcal{A}$ is not able to find $K_{U,F}$ since at least one term $\hat{e}(H_1(ID), P)^{ba_i}$ cannot be computed by $\mathcal{A}$ under the BDH assumption as both b and a_i as well as the discrete logarithm of $H_1(ID_U)$ are unknown to $\mathcal{A}$.

In particular, we should note that an active corruption that reveals both of the current ephemeral private key and the master secret key of F may be launched by adversary $\mathcal{A}$ during the service time. That is to say, at time period TP_i, both a_i and s_F are disclosed to $\mathcal{A}$. Then A is able to compromise the previous session keys established in current time period TP_i. However, our protocol ensure that the sessions established in all other time periods remain secure. The reason is that: with the ephemeral key a_i in current time period TP_i and s_F, $\mathcal{A}$ gets all necessary information to calculate the session keys established in time period TP_i. Nevertheless, she cannot obtain any session keys established in all of the time periods before TP_i since the old ephemeral keys $a_0, a_1, ..., a_{i-1}$ had been erased from memory. Moreover, the randomness of ephemeral private keys prohibits the deduction of future ephemeral keys so that adversary $\mathcal{A}$ could not derive the ephemeral private keys of future time periods (e.g. a_{i+1} of TP_{i+1}) from a_i.

Perfect KCI. To achieve the security of perfect KCI (Key-Compromise Impersonation), the OP-KE-WR should i) prevent adversary $\mathcal{A}$ from impersonating F after compromising U's long-term secret key S_U and ii) prevent $\mathcal{A}$ from impersonating U after compromising F's long-term secret key s_F. For the first case, $\mathcal{A}$ has the control of a_i for each time period TP_i. However, $\mathcal{A}$ cannot obtain $K_{U,F}$ as the term $\hat{e}(H_1(ID_U), P)^{bs_F}$ in Eq. (5) remains secret to $\mathcal{A}$ due to the BDH assumption as $\mathcal{A}$ does not know b, s_F or the discrete logarithm of $H_1(ID_U)$. For the second case, $\mathcal{A}$ has the control of b. However, $\mathcal{A}$ cannot obtain $K_{U,F}$ either as the term $\hat{e}(H_1(ID_U), P)^{s_H a_i}$ is not known to $\mathcal{A}$ due to the secrecy of s_H, $S_U = s_H H_1(ID_U)$, a_i, and the discrete logarithm of $H_1(ID_U)$ under the BDH assumption.

5.3 Security Proof

Theorem 2 (rCK-security of OP-KE-WR with PFS). If H is a random oracles, KDF is a random, non-collision key derivation function and $\mathbb{G}_1$ is a finite group where the BDH assumption holds, Then Protocol II is secure in the rCK model.

Proof: The model for Protocol II is the same as previous protocol, except for the session identifier definition and clean definition should be modified. We redefine them as below:

Definition 6 (Session identifier). *A session of Protocol II is identified by a session identifier* $sid = (role, ID_1, ID_2, msg, T_i)$, *where* $role \in \{I, R\}$ *(initiator or responder),* $ID_1 \in \{0,1\}^*$ *is the identities of the participant executing the session,* $ID_2 \in \{0,1\}^*$ *is the identities of the intended communication peer, msg is the message transferred in the protocol and* T_i *is the index of the time period in which the session is executed.*

And before we redefine the clean sessions, the Broadcast queries are needed to be modified for simulating the capability of adversary $\mathcal{A}$ more precisely and formally.

Broadcast $(\hat{F}, \perp, counter)$: This query now allows $\mathcal{A}$ to select a number $a_{\hat{F}} \in_R \mathbb{Z}_q^*$ and broadcast $a_{\hat{F}}$ along with the public parameters such as identity and pubic key in presence of $\hat{F}$. The value of counter will add by 1 when $\mathcal{A}$ request a Broadcast query and never decreases.

Definition 7 (Clean Sessions). *Let sid be the session identifier of a completed session, owned by* $\hat{U}$ *with communication peer* $\hat{F}$, *and denote by* sid^* *the matching session (if exists). We say that an session is not clean if* $\mathcal{A}$ *can trivially compute the corresponding session key. Namely, a session sid is said* ***not to be clean*** *if any of following conditions holds:*

1) $\hat{U}$ *or* $\hat{F}$ *is an adversary controlled party (issued by Extract query before session execution);*

2) $\mathcal{A}$ *issues a Reveal(sid) or a Reveal(*sid^**) query;*

3) if sid^* exists, and $\mathcal{A}$ makes either of following queries:
- both LtKeyReveal $(\hat{U})$ and EpKeyReveal (sid/sid^*), or
- both LtKeyReveal $(\hat{F})$ and Broadcast $(\hat{F})$;

4) if sid^* does not exist, and $\mathcal{A}$ makes either of following queries:
- both LtKeyReveal $(\hat{U})$ and EpKeyReveal (sid/sid^*),or
- LtKeyReveal $(\hat{F})$;

The security proof is similar to that of our previous protocol in Section 4. An efficient BDH solver $\mathcal{S}$ given the BDH challenge $\langle P, xP, yP, zP\rangle$ wishes to find $\hat{e}(P,P)^{xyz}$ by using $\mathcal{A}$ as a subroutine. Let *sid* be the session identifier of a completed session, owned by $\hat{U}$ with communication peer $\hat{F}$, and denote by sid^* the matching session (if exists). Due to the clean session in Definition 7, for events that $\mathcal{A}$ perform a successful forging attack and correctly guess the private bit b involved in Test Query, we consider the following complementary conditions according to the clean definition:

- if sid^* exists and $\mathcal{A}$ does not issue Broadcast$(\hat{F})$; and either of the following:
  - $\mathcal{A}$ does not issue $LtKeyReveal\,(\hat{U})$, or - Event E_1
  - $\mathcal{A}$ does not issue $EpKeyReveal\,(sid)$ - Event E_2
- if sid^* exists and $\mathcal{A}$ issues $Broadcast\,(\hat{F})$, or sid^* does not exist, but $\mathcal{A}$ does not issue $LtKeyReveal\,(\hat{F})$; and either of following:
  - $\mathcal{A}$ does not issue $LtKeyReveal\,(\hat{U})$, or - Event E_3
  - $\mathcal{A}$ does not issue $EpKeyReveal\,(sid)$ - Event E_4

Event E_1: *In this case, $\mathcal{S}$ sets PK_H to xP and selects random long-term key pairs for the remaining servers. $\mathcal{S}$ simulates the protocol executions and queries by following the description in Section 4.1 as usual. For simulation of tid where $T_i = T_\tau$, $\mathcal{S}$ embeds yP into the ephemeral public key of T_τ as $A_\tau = yP$. The reminder of session tid is simulated as the protocol definition.*

With probability at least $1/k(\theta)$ $\mathcal{A}$ picks tid as the test session and with probability at least $1/n(\theta)m(\theta)$ $\mathcal{A}$ chooses honest user $\hat{U}$ as the initiator of tid and server $\hat{H}$ as the $\hat{U}$'s home server. If $\mathcal{A}$ wins the game, $\mathcal{S}$ can solve the BDH as follows: $\mathcal{S}$ randomly picks an item $(K, TP_\tau||\hat{U}||\hat{H}||\hat{F}||B)$ in list $\mathbb{L}_\mathcal{K}$ (at most $k(\theta)$ items), and computes $\rho = K \cdot \hat{e}(zP, PK_F)^{-b} \cdot \hat{e}(zP, A_\tau)^{-b} \cdot \hat{e}(zP, PK_H)^{-s_{\hat{F}}}$, then answers the BDH challenge with ρ. Thus $\mathcal{S}$ can solve the challenge with advantage:

$$Adv^{BDH}(\mathcal{S}) \geq \frac{1}{k(\theta)^2 n(\theta) m(\theta)} Adv_\pi^{rCK}(\mathcal{A}) \tag{6}$$

Event E_2: *In this case, $\mathcal{S}$ simulates the protocol executions and queries as usual except for the session tid. For simulation of tid, For simulation of tid where $T_i = T_\tau$, $\mathcal{S}$ embeds yP into the ephemeral public key of T_τ as $A_\tau = yP$. And by modifying the Send query, $\mathcal{S}$ embeds xP into the ephemeral key as $B = xP$. $\mathcal{A}$ can not detect the simulation due to the definition of Event E_2.*

With probability at least $1/k(\theta)$, $\mathcal{A}$ picks tid as the test session, and with probability at least $1/n(\theta)$ $\mathcal{A}$ chooses $\hat{U}$ as the session owner. If $\mathcal{A}$ wins the game, $\mathcal{S}$ can solve the BDH as follows: $\mathcal{S}$ randomly picks an item $(K, TP_\tau||\hat{U}||\hat{H}||\hat{F}||B)$ in list $\mathbb{L}_\mathcal{K}$, and computes $\rho = K \cdot \hat{e}(zP, B)^{-s_{\hat{H}}} \cdot \hat{e}(zP, A_\tau)^{-s_{\hat{F}}} \cdot \hat{e}(zP, PK_{\hat{H}})^{-s_{\hat{F}}}$, then answers the BDH challenge with ρ. Thus $\mathcal{S}$ can solve the challenge with advantage:

$$Adv^{BDH}(\mathcal{S}) \geq \frac{1}{k(\theta)^2 n(\theta) m(\theta)} Adv_\pi^{rCK}(\mathcal{A}) \tag{7}$$

Event E_3: *In this case, $\mathcal{S}$ sets $PK_{\hat{H}}$ to xP and $PK_{\hat{F}}$ to yP and simulates the protocol executions and oracles queries as usual. It is perfect, since $\mathcal{A}$ can not detect the simulation without issuing LtKeyReveal queries for two participants.*

With probability at least $1/n(\theta)m(\theta)^2$ $\mathcal{A}$ chooses $\hat{U}$ as the session owner with peer $\hat{F}$ and server $\hat{H}$ as the $\hat{U}$'s home server. If $\mathcal{A}$ wins the game, $\mathcal{S}$ can solve the BDH as follows: $\mathcal{S}$ randomly picks an item $(K, TP_\tau||\hat{U}||\hat{H}||\hat{F}||B)$ in list $\mathbb{L}_\mathcal{K}$, and computes $\rho = K \cdot \hat{e}(zP, PK_{\hat{H}})^{-a_\tau} \cdot \hat{e}(zP, A_\tau)^{-b} \cdot \hat{e}(zP, PK_{\hat{f}})^{-b}$, then answers the BDH challenge with ρ. Thus $\mathcal{S}$ can solve the challenge with advantage:

$$Adv^{BDH}(\mathcal{S}) \geq \frac{1}{k(\theta) n(\theta) m(\theta)^2} Adv_\pi^{rCK}(\mathcal{A}) \tag{8}$$

Event E_4: *In this case, $\mathcal{S}$ sets PK_F to yP simulates the protocol executions and queries as usual except that, for simulation of tid, by modifying the Send query, $\mathcal{S}$ embeds xP into the ephemeral public key of $\hat{U}$ as $B = xP$. $\mathcal{A}$ can not detect the simulation due to the definition of Event E_4.*

With probability at least $1/k(\theta)$, $\mathcal{A}$ picks tid as the test session, and with probability at least $1/n(\theta)m(\theta)$ $\mathcal{A}$ chooses $\hat{U}$ as the session owner with peer $\hat{F}$. If $\mathcal{A}$ wins the game, $\mathcal{S}$ can solve the BDH as follows: $\mathcal{S}$ randomly picks an item $(K, TP_\tau||\hat{U}||\hat{H}||\hat{F}||B)$ in list $\mathbb{L}_\mathcal{K}$, and computes $\rho = K \cdot \hat{e}(zP, B)^{-a_\tau} \cdot \hat{e}(zP, PK_H)^{-a_\tau} \cdot \hat{e}(zP, PK_{\hat{F}})^{-s_{\hat{H}}}$, then answers the BDH challenge with ρ. Thus $\mathcal{S}$ can solve the challenge with advantage:

$$Adv^{BDH}(\mathcal{S}) \geq \frac{1}{k(\theta)^2 n(\theta) m(\theta)} Adv_\pi^{rCK}(\mathcal{A}) \tag{9}$$

Overall analysis. *Combining the result of equation 6 to 9, the advantage of $\mathcal{S}$ to solve the BDH challenge is*

$$Adv^{BDH}(\mathcal{S}) \geq max\{\frac{1}{k(\theta) n(\theta) m(\theta)^2}, \frac{1}{k(\theta)^2 n(\theta) m(\theta)}\} Adv_\pi^{rCK}(\mathcal{A}) \tag{10}$$

At the beginning of proof, we assume that $Adv_\pi^{rCK}(\mathcal{A})$ is non-negligible. Therefore, $\mathcal{S}$ can solve the BDH challenge with an advantage which is also non-negligible. It leads to a contradiction to the BDH assumption. Hence, there exists no polynomially bounded adversary that succeeds in breaking rCK-security of our one-pass protocol with non-negligible advantage. □

Table 3: Comparison

	#P	#F	KE	PFS	KCI	Anon	Computational Complexity (total)	Computation Cost of User (online)
Zhang05 [16]	3	6	E	¬	¬	N	10 SK	3 SK
Yang05 [13]	3	4	EH	√	¬	S	4 EXP + 6 PK	1 EXP + 2 PK
Yang07 [11]	3	8	EH	√	¬	S	≥ 10 PK	≥ 4 PK
YHW 1 [12]	2	3	EH	√	√	N	≥ 4 BP + 4 ECSM + 2 SK	≥ 2 BP + 1 ECSM + 2 SK
YHW 2 [12]	2	3	EH	√	√	S	≥ 6 BP + 20 ECSM	≥ 3 BP + 8 ECSM
Protocol I	2	1	EH	P	P	N	2 BP + 4 ECSM	0
Protocol II	2	1	EH	√	√	N	2 BP + 4 ECSM	1 BP

#P : Parties Involved
#F : Number of Flows
KE : Key Establishment
PFS : Perfect Forward Secrecy
KCI : Perfect KCI Security
Anon : User Anonymity
E : Key Establishment
H : Home server cannot calculate the session key
P : Partially Satisfied
√ : Fully Satisfied
N : Normal User Anonymity
S : Strong User Anonymity
¬ : Not Considered
SK : Symmetric Key Encryption/Decryption
PK : Public Key Encryption/Decryption
EXP : Modular Exponentiation
ECSM : Elliptic Curve Scalar Multiplication
BP : Bilinear Pairing Operation

6 Achieving User Anonymity

As user privacy preservation has become an increasingly demanding requirement for wireless communications, a secure KE-WR should not only establish a secure session key between U and F, but also provide privacy protection so to keep user identity from being exposed and user movement from being tracked. Refer to Section 1, there are two security levels [12] on user privacy protection, *Normal User Anonymity* and *Strong User Anonymity*. Both two levels concerns about keeping U's identity and whereabouts from being known by any eavesdropper. But *Strong User Anonymity* concerns further about keeping U anonymous from F. In this section, we extend our protocol to achieve *normal user anonymity and untraceability*. Following shows how to extend our KE-WR to support user anonymity, using protocol I as the example.

Protocol I': The session key generation phase of U in Protocol I could be changed to

$$TID\|SK \leftarrow KDF(K_{U,F}, ID_U\|ID_H\|ID_F\|T\|X)$$

and U sets TID as a temporary ID of U, which will be used in the next protocol run with F, and SK as the session key. F also generates the same TID and SK according to

$$TID\|SK \leftarrow KDF(K_{F,U}, ID_U\|ID_H\|ID_F\|T\|X)$$

F then stores TID as the temporary ID of U corresponding to ID_U, and sets the session key to SK.

After the first protocol run, if U wants to establish a new session with F, the Protocol I' will be carried out again but have ID_U replaced by the temporary ID TID in the message from U to F. All the rest of the protocol will remain unchanged. After the protocol run of this new session, another new temporary ID TID' will be generated. It will be used as the new temporary ID of U for the next protocol run with F.

Achieving User Anonymity and Untraceability. In this case, since the established TID is protected by the secure key establishment in the first protocol run, an eavesdropper cannot associate U's real ID ID_U to the temporary ID TID to be used in the second protocol run. Furthermore, TID will be renewed and replaced by another temporary ID TID'. Hence, the temporary ID is updated after every session so that eavesdroppers cannot tell if any two roaming sessions are corresponding to the same U or trace U across multiple sessions.

7 Performance and Comparison

As of existing KE-AWR protocols, U is usually considered as a power-constrained mobile device. Its computational complexity requirement is therefore usually more stringent than that of F. In our scheme, the two most expensive operations at U are Bilinear Pairing (BP) and Elliptic Curve Scalar Multiplication (ECSM). Other operations such as elliptic curve point addition and KDF (which can be instantiated using an efficient MAC) [7] are in the order of at least a hundred times faster than BP and ECSM. Besides, H is a map-to-point hash function. As shown in [9], the evaluation of H is slightly faster than ECSM. Here we assume that the speed of doing one H_1 evaluation is comparable to that of ECSM. For U, $H(ID_U)$ can be stored as a constant point at U. As a result, the computational complexity of both protocols at U is mainly incurred by one BP and two ECSMs.

Protocol I: Online Computations. We notice that Protocol I could be further optimized in computational complexity by pre-computing, since we assume that in step 0, F has already delivered (PK_F, T) and its certificate to all users through the broadcast channel. As shown in Figure 1, once the random number r is selected, rP, $rH(ID_U)$ as well as $K_{U,F} = \hat{e}(rH(ID_U) + S_U, PK_F)$ can be computed by U. This means, one BP and two ECSMs can be extremely pre-computed by U as long as U moved into the radio coverage of F's hotspots, and leave almost no computation to be performed online in the run of protocol execution.

Protocol II: Online Computations. The computational complexity of Protocol II can also be optimized by pre-computation. Once the random number b is selected, $B = bP$ and $bH(ID_U)$ can be pre-computed by U. However, as A_τ is only valid in a limited time, U must respond quickly and perform a bilinear paring operation online to compute $K_{U,F} = \hat{e}(rH(ID_U) + S_U, PK_F + A_\tau)$. Therefore, for each run of protocol, U has to perform an online BP operation.

Table 3 shows a comparison in terms of both complexity and security features between our novel OP-KE-WRs and existing ones. And our protocols are the only two achieving One-Pass. Furthermore, Protocol II supports PFS and perfect KCI security. Though Yang et al.'s protocols [11, 12, 13] can provide a stronger level of user anonymity and untraceability to U and also eliminate the involvement of HS in their protocols, our protocol requires much lower bandwidth with comparable computational performance as bilinear pairing implementations are getting significant improvements recently [9].

8 Conclusion

Two protocols are proposed: Protocol I ensures that a fresh session key secreted from all other entities except user and foreign WSP is established in each run of protocol, by just sending one message from the user to the foreign WSP and eliminate any intervention of a third party. However, it does not support the desirable properties that multi-round key establishment protocols usually do, namely, it only supports partial forward secrecy and partial KCI. Consequently, Protocol II solves the problems by supporting all the desirable properties mentioned. Namely, the protocol can support both Perfect Forward Secrecy (PFS) and Perfect Key Compromise Impersonation (Perfect KCI). The total computational complexity of Protocol II is comparable to that of previous protocol. Third, both protocols are proved to be security when presenting formal security proofs in the rCK security model. Furthermore, both protocols are extended to support User Anonymity and Untraceability.

References

[1] R. Canetti and H. Krawczyk, "Analysis of key-exchange protocols and their use for building secure channels," in *Lecture Notes in Computer Science*, vol. 2045, pp. 453–474, 2001.

[2] C. Cremers, "Formally and practically relating the CK, CK-HMQV, and eCK security models for authenticated key exchange," in *Cryptology ePrint Archive*, 2009.

[3] Y. Dodis, R. Gennaro, J. Hastad, H. Krawczyk, and T. Rabin, "Randomness extraction and key derivation using the cbc, cascade and hmac modes," in *Crypto 2004*, vol. 3152, pp. 494–510, 2004.

[4] M. Gorantla, C. Boyd, and J. Gonzalez Nieto, "ID-based one-pass authenticated key establishment," in *Proceedings of the 6th Australasian Conference on Information Security*, pp. 39–46, 2008.

[5] B. LaMacchia, K. Lauter, and A. Mityagin, "Stronger security of authenticated key exchange," in *Proceedings of the 1st international conference on Provable security*, vol. 4784, pp. 1–16, 2007.

[6] M. Long, C. H. Wu, and J. D. Irwin, "Reducing communication overhead for wireless roaming authentication: Methods and performance evaluation," *International Journal of Network Security*, vol. 6, no. 3, pp. 331–341, 2008.

[7] A. Menezes, P. Van Oorschot, and S. Vanstone, *Handbook of Applied Cryptography*. 1997.

[8] N. Thompson, Z. Yin, H. Luo, P. Zerfos, and J. P. Singh, "Authentication on the edge: Distributed authentication for a global open wi-fi network," in *Proceedings of of the 13th Annual ACM International Conference on Mobile Computing and Networking*, pp. 334–337, 2007.

[9] X. Xiong, D. Wong, and X. Deng, "TinyPairing: A fast and lightweight pairing-based cryptographic library for wireless sensor networks," in *Proceedings of 2010 IEEE Wireless Communications and Networking Conference (WCNC 2010)*, pp. 1–6, 2010.

[10] C. C. Yang, K. H. Chu, and Y. W. Yang, "3G and WLAN interworking security: current status and key issues," *International Journal of Network Security*, vol. 2, no. 1, pp. 1–13, 2006.

[11] G. Yang, D.Wong, and X. Deng, "Anonymous and authenticated key exchange for roaming networks," *IEEE Transactions on Wireless Communications*, vol. 6, no. 9, pp. 3491–3472, 2007.

[12] G. Yang, Q. Huang, D.Wong, and X. Deng, "Universal authentication protocols for anonymouswireless communications," *IEEE Transactions on Wireless Communications*, vol. 9, no. 1, pp. 168–174, 2010.

[13] G. Yang, D. Wong, and X. Deng, "Efficient anonymous roaming and its security analysis," in *Applied Cryptography and Network Security (ACNS 2005)*, pp. 334–349, 2005.

[14] G. Yang, D. Wong, and X. Deng, "Formal security definition and efficient construction for roaming with a privacy-preserving extension," *Journal of Universal Computer Science*, vol. 14, no. 3, pp. 441–462, 2008.

[15] Y. Zeng, J. Ma, and M. Sangjae, "An improvement on a three-party password-based key exchange protocol using weil pairing," *International Journal of Network Security*, vol. 11, no. 1, pp. 17–22, 2010.

[16] M. Zhang and Y. Fang, "Security analysis and enhancements of 3 GPP authentication and key agreement protocol," *IEEE Transactions on Wireless Communications*, vol. 4, no. 2, pp. 734–742, 2005.

[17] H. Zhu, X. Lin, R. Lu, P. Ho, and X. Shen, "SLAB: A secure localized authentication and billing scheme for wireless mesh networks," *IEEE Transactions on*

Wireless Communications, vol. 7, no. 10, pp. 3858–3868, 2008.

[18] H. Zhu, X. Lin, M. Shi, P. Ho, and X. Shen, "PPAB: A privacy-preserving authentication and billing architecture for metropolitan area sharing networks," *IEEE Transactions on Vehicular Technology*, vol. 58, no. 5, pp. 2529–2543, 2009.

Yuan Wang is currently a Ph.D. student in the School of Computer Science and Technology at University of Science and Technology of China, under the supervision of Prof. Liusheng Huang. He also joins in the collaborated Ph.D. education scheme of the City University of Hong Kong in 2008, under the supervision of Dr. Duncan S. Wong. He is now studying at USTC-CityU Joint Advanced Research Center of Suzhou Institute of USTC. Prior to that, he received his B.S. degree from University of Science and Technology of China in 2006. His primary research interest is applied cryptography; in particular, cryptographic protocols, and anonymous systems.

Duncan S. Wong received his B.Eng. degree in Electrical & Electronic Engineering with first class honors from the University of Hong Kong in 1994, M.Phil. degree in Information Engineering from the Chinese University of Hong Kong in 1998 and Ph.D. degree in Computer Science from Northeastern University, Boston, MA, USA in 2002. After graduation, he has been a visiting assistant professor at the Chinese University of Hong Kong for one year before joining City University of Hong Kong in September 2003. He is now an associate professor in the Department of Computer Science. His primary research interest is applied cryptography; in particular, cryptographic protocols, encryption and signature schemes, and anonymous systems. His publications have continuously appeared in reputable conferences and journals in the fields of cryptography and information security. With extensive experience in designing cryptographic algorithms and protocols, and developing computer security systems, he also provides security consultancy services to the industry and governmental institutions.

LiuSheng Huang is currently a professor and doctoral supervisor of the School of Computer Science of USTC, receiving the state council special government allowances. He used to be Head of the department of Computer Science and Technology of USTC, and the vice president of the School of Information Science and Technology of USTC. Now, he is Executive Director of Suzhou Institute for Advanced Study of USTC, and the deputy director of Suzhou sub-director of the National High Performance Computing Center at Hefei. In recent years, as the leader or key member, LiuSheng Huang has taken part in more than 20 projects, including the national 973 Project, 863 key projects, National Science and Technology Major Project, and pre-assembly Eleventh Five-year research project, etc. He has published more than 200 papers in major international conferences, such as ACM Computing Surveys, IEEE Transaction on Services Computing, Journal of Parallel and Distributed Computing and other well-known journals, among which more than 160 are indexed by SCI or EI.