Sensors **2014**, *14*, 11379-11394; doi:10.3390/s140711379

OPEN ACCESS
sensors
ISSN 1424-8220
www.mdpi.com/journal/sensors

Article

A Secure and Efficient Handover Authentication Protocol for Wireless Networks

Weijia Wang [1,*] and Lei Hu [2]

[1] School of Science, Beijing Jiaotong University, Beijing 100044, China
[2] The State Key Laboratory of Information Security, Institute of Information Engineering, Chinese Academy of Sciences, Beijing 100093, China; E-Mail: hu@is.ac.cn

* Author to whom correspondence should be addressed; E-Mail: wangwj@bjtu.edu.cn; Tel.: +86-10-5168-2052 (ext. 216).

Received: 11 May 2014; in revised form: 19 June 2014; / Accepted: 23 June 2014 / Published: 26 June 2014

Abstract: Handover authentication protocol is a promising access control technology in the fields of WLANs and mobile wireless sensor networks. In this paper, we firstly review an efficient handover authentication protocol, named PairHand, and its existing security attacks and improvements. Then, we present an improved key recovery attack by using the linearly combining method and reanalyze its feasibility on the improved PairHand protocol. Finally, we present a new handover authentication protocol, which not only achieves the same desirable efficiency features of PairHand, but enjoys the provable security in the random oracle model.

Keywords: wireless network; security; privacy; handover authentication

1. Introduction

In today's world, wireless communication networks are ubiquitous, and mobile handheld devices, such as PDAs, smart phones and laptop PCs, have a wide influence on various aspects of people's lives. To overcome the restriction of geographical coverage, seamless access services are highly desirable for WLANs and mobile wireless sensor networks (WSNs), but how to ensure the security and efficiency of this process is still challenging. Recently, as a promising seamless access control technology, handover authentication protocols have received much attention [1–12]. A handover authentication scenario is always assumed to involve three kinds of parties: mobile nodes (MNs), access points (APs) and the

authentication server (AS). An MN is a registered user on AS, who accesses its subscribed services by connecting any AP. An AP acts as a guarantor for vouching for an MN as a legitimate subscriber. When an MN leaves the service area of the current AP (e.g., AP1) and tries to connect a new AP (e.g., AP2), the new AP will start its handover authentication process to identify the MN. If the authentication succeeds, a session key will be built between the MN and AP2 to escort the MN's later access. Otherwise, the requirement for accessing will be rejected by AP2. A promising application of this kind of protocol appears in three-tiered mobile WSNs [13], which consist of a base station, access points, mobile agents and sensor nodes. In the highest layer, the base station works as the AS to deploy access points and to register mobile agents by granting the corresponding authentication keys. The access points are the APs with the task of receiving and verifying the message from the medium layer. The medium layer is composed of the mobile agents, which can be mobile phones, vehicles, men and even animal, acting as the MNs and responsible for gathering data from the sensor nodes in the lowest layer and, then, forwarding to the upper layer.

Recently, He *et al.* [14] introduced an interesting handover authentication protocol, named PairHand. For improving the communication efficiency and reducing the burden on AS, PairHand only requires two handshakes between MN and AP for mutual authentication and key establishment, instead of relying on the participation of AS. Furthermore, considering the high cost and the inconvenience of revoking users due to using a group signature in the authentication process, PairHand makes its construction directly based on the pairing-based cryptography and uses a pool of shorter-lived pseudonyms to protect users' privacy. Unfortunately, shortly after, He *et al.* [15] found that there is a serious design weakness in PairHand protocol that enables an adversary to easily obtain the private key from the message transported in the first round of the protocol and presented an improvement by utilizing a composite order bilinear group, claiming that the improved version fixes the security problem without losing any of the desirable feature of PairHand. However, Yeo *et al.* [16] showed that if an attacker obtains multiple authenticated messages generated with the same pseudo-ID, it will be likely to recover the private key of the mobile node. Furthermore, Yeo *et al.* [16] and Tsai *et al.* [17] pointed out another dilemma of the improved version that suggested that the 160-bit composite is insecure, but using a 1,024-bit composite-order group will lead to a great drop in the efficiency. At the same time, Tsai *et al.* [17] presented a provably secure handover authentication protocol, which solves the above security problem, but increases the size of the public key.

In this paper, we provide a linear combination method to reduce the number of captured signatures corresponding to the same pseudo-ID required by the key recovery attack on the improved PairHand [15]. By repeatedly linearly combining arbitrary two-captured signatures from the same pseudo-ID in a random way, the attacker can compute out the private key of MN with a very high probability. To improve the security without losing the desirable features, we present a new handover authentication protocol that overcomes the security weakness of the original PairHand and achieves the same level of high efficiency. Finally, in the random oracle model, we prove that this protocol enjoys both semantic security and authentication security.

2. The Bilinear Maps and Complexity Implications

In this section, we briefly recall bilinear maps and some difficult problems that will be used in the followings.

Let $\mathbb{G}$ be a cyclic additive group of composite order q and $\mathbb{G}_T$ be a cyclic multiplicative group of the same order. Let $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ be a bilinear map that satisfies the following properties.

- Bilinearity: $e(aP, bQ) = (P, Q)^{ab}$ for $\forall P, Q \in \mathbb{G}$ and $\forall a, b \in Z_q^*$.
- Non-degeneracy: $e(P, P) \neq 1$ for $P \neq 0$.
- Computability: there exists an efficient algorithm to compute $e(P, Q)$ for $\forall P, Q \in \mathbb{G}$.

Computational Diffie–Hellman (CDH) assumption: Given P,aP and bP for some $a, b \in Z_q^*$, it is computationally intractable to compute the value abP.
Bilinear Diffie–Hellman (BDH) assumption: Given P,aP,bP and cP for some $a, b, c \in Z_q^*$, it is computationally intractable to compute the value $e(P, P)^{abc}$.

3. Security Model

Generally, there are two kinds of handovers: a hard handover and a soft handover. The difference between them is that in a hard handover, the former connection with AP1 is broken before the new connection is established between MN and AP2, while in a soft handover, MN can retain the connection with AP1 after building the new connection with AP2. For simplicity, it is assumed that there is no communication among APs and that handover authentication protocols perform in the hard handover model. In the following, we present the formal security model for handover authentication protocols, which follows the approach initiated by Bellare and Rogway [18,19] and modified by Bresson *et al.* [20].

3.1. Communication Model

Protocol participants: In the model, there are two kinds of participants: mobile node MN and access point AP, which have unique identities ID_{MN} and ID_{AP}, respectively. Each instance of a participant (U or V) is molded as an oracle, denoted by Π_{MN}^n (Π_{AP}^n, respectively), meaning the n-th running instance of the participant MN (AP, respectively).

Protocol execution: In the model, it is assumed that an adversary $\mathcal{A}$ fully controls over the communication channels and can create several concurrent instances of the protocol. The public parameters *params* and identity information are known to all participants, including the adversary. During the execution of the protocol, the interaction between the adversary and the protocol participants occurs only via oracle queries, which models the adversary capabilities in a real attack. At any time, the adversary makes the following queries:

(1) $Execute(\Pi_U^n, \Pi_V^m)$: This query models passive attacks, where the attacker gets access to honest executions between instances Π_U^n and Π_V^n by eavesdropping. The output of this query is the complete transcript that was transported during the honest execution of the protocol.

(2) $Send(\Pi_U^n, M)$: This query models an active attack against an MN or AP, in which the adversary sends a message to the instance Π_U^n. The output of this query is the message that the instance Π_U^n generates upon receipt of the message M.

(3) $Reveal(\Pi_U^n)$: This query models the misuse of session keys. Only if the session key of the instance Π_U^n is defined, the query is available and returns to the adversary the session key.

(4) $Test(\Pi_U^n)$: This query is to measure the semantic security of the session key of instance Π_U^n: If the session key is not defined, it returns $\perp$. Otherwise, it returns either the session key held by the instance if $b = 0$ or a random number of the same size if $b = 1$, where b is the hidden bit previously selected at random before the protocol runs.

(5) $Corrupt(ID_U)$: This query models the exposure of the long-term secret key. When the adversary makes this query, the oracle returns the private key corresponding to ID_U.

3.2. Security Definitions

Notation: An instance Π_U^n is said to be opened if the query $Reveal(\Pi_U^n)$ has been made by the adversary. We say an instance Π_U^n is unopened if it is not opened. An instance Π_U^n is said to be accepted if it goes into an accept state after receiving the last expected protocol message.

Partnering: We say two instances Π_U^n and Π_V^m are partners if the following conditions are met: (1) they are an MN and an AP, respectively; (2) both Π_U^n and Π_V^m are accepted; (3) both Π_U^n and Π_V^m share the same session ID *sid*; (4) the partner identification for Π_U^n is Π_V^m and *vice versa*; and (5) no instance other than Π_U^n and Π_V^m accepts with a partner identification equal to Π_U^n or Π_V^m.

Freshness: If an instance Π_U^n has been accepted, both the instance and its partner are unopened and they are both instances of honest clients, we say the instance Π_U^n is fresh.

Semantic security: The security notion is defined in the context of executing an ID-based handover authentication protocol P in the presence of an adversary $\mathcal{A}$. During the protocol execution, $\mathcal{A}$ is allowed to make multiple $Execute$, $Send$ and $Reveal$ queries, but at most, one $Test$ query, to a fresh instance of an honest participant. Finally $\mathcal{A}$ outputs a bit guess b' for the bit b hidden in the Testoracle. The adversary $\mathcal{A}$ is said to be successful if $b' = b$. We denote the event by $Succ$ and define the advantage of $\mathcal{A}$ in violating the semantic security of the protocol P as follows:

$$Adv_{\mathcal{A},P}(k, t) = 2 \cdot Pr[Succ] - 1$$

where k is the security parameter and t is the time parameter.

We say a handover authentication protocol P is semantically secure if the advantage $Adv_{\mathcal{A},P}(k, t)$ is negligible.

Authentication security: To measure the security of a handover authentication protocol resisting the impersonation attacks, we consider the mutual authentication between MN and AP. We denote by $Auth_{\mathcal{A},P}^{MN\to AP}(k, t)$ (or $Auth_{\mathcal{A},P}^{AP\to MN}(k, t)$, respectively) the probability that an adversary $\mathcal{A}$ successfully impersonates an MN instance during executing the protocol P, while the target AP (or MN, respectively) does not detect it, where k is the security parameter and t is the time parameter. We say a handover authentication protocol P is mutual authentication secure if both $Auth_{\mathcal{A},P}^{MN\to AP}(k, t)$ and $Auth_{\mathcal{A},P}^{AP\to MN}(k, t)$ are negligible in the security parameter.

4. Review of the Protocol

In this section, we review He *et al.*'s improved protocol [15], which is very similar to the original PairHand and consists of four phases: system initialization, handover authentication, batch authentication and denial-of-service (DoS) attack resistance. The only differences between the two versions appear at the selection of the group order in the system initialization phase and the computation of the hash value of the authentication message in the handover authentication phase, and our attack is exactly to address these two phases. Below, we only review the first two phases of the improved PairHand protocol. For more details, please refer to [14].

4.1. System Initialization

The AS randomly chooses a value $s \in \mathbb{Z}_q^*$ as the master key and a generator P of $\mathbb{G}$, computes the corresponding public key $P_{pub} = sP$ and selects two cryptographic hash functions H_1 and H_2, where $H_1 : \{0,1\}^* \rightarrow \mathbb{G}$ and $H_2 : \{0,1\}^* \rightarrow \mathbb{Z}_q^*$. The resulting public system parameter, *params*, is $\{\mathbb{G}, \mathbb{G}_T, q, P, P_{pub}, H_1, H_2\}$, and the private secret of AS is s. For each AP, AS computes $H_1(ID_{AP})$ and $sH_1(ID_{AP})$ as the public and private keys of that AP, respectively, and delivers them to the AP via a secure channel, where ID_{AP} is the identity of the AP.

For the registration of a qualified MN i with real identity ID_i, AS generates a family of unlinkable pseudo-IDs $PID = \{pid_1, pid_2, \cdots\}$, computes the public key $H_1(pid_j)$ and the corresponding private key $s \cdot H_1(pid_j)$ for each pseudo-ID $pid_j \in PID$ and, finally, securely sends to MN i all tuples $(pid_j, sH_1(pid_j))$. The use of shorter-lived pseudonyms is to protect each MN's privacy, preventing them from being traced.

4.2. Handover Authentication

When an MN, say i, moves into the communication range of a new AP (AP2), a handover authentication process, which is shown in Figure 1, will be performed between MN i and AP2 in the following steps.

(1) MN i firstly picks an unused pseudo-ID pid_i from his pseudo-ID family and the corresponding private key $sH_1(pid_i)$. Then, MN i generates an authentication message as $M_i = pid_i||ID_{AP2}||ts$, where ts is a timestamp, which is used to resist against replay attacks and "||" denotes the concatenation of messages and checks whether $H_2(M_i)$ and q are co-prime or not. If $H_2(M_i)$ and q are not co-prime, it does nothing; otherwise, it continues to append redundant bits rb into M_i until $H_2(M_i)$ and q are not co-prime. After that, MN i computes the signature $\sigma_i = H_2(M_i) \cdot sH_1(pid_i)$ and unicasts the access request message $\{M_i, \sigma_i\}$ to AP2. Finally, MN i computes the session key with AP2 as $K_{i-2} = e(sH_1(pid_i), H_1(ID_{AP2}))$.

(2) Upon receiving the request message $\{M_i, \sigma_i\}$, AP2 firstly checks whether the timestamp ts is valid. If invalid, the request is rejected. Otherwise, AP2 verifies if $e(\sigma_i, P) = e(H_2(M_i) \cdot H_1(ID_{pid_i}), P_{pub})$. If true, AP2 computes the session key $K_{2-i} = e(H_1(pid_i), sH_1(ID_{AP2}))$ and the authentication code $Aut = H_2(K_{2-i}||pid_i||ID_{AP2})$ and, then, sends the tuple $\{pid_i, ID_{AP2}, Aut\}$ to MN i.

(3) Upon receipt of the message $\{pid_i, ID_{AP2}, Aut\}$, MN i computes the verification code $Ver = H_2(K_{i-2}||pid_i||ID_{AP2})$ and compares it with Aut. If they are equal, MN i confirms that AP2 is legitimate, and the generated session key is valid. Otherwise, MN i cancels the connection.

(4) At last, AP2 securely transports $\{M_i, \sigma_i\}$ to AS. By receiving this message, AS can identify the real identity of MN i according to the pseudo-ID in M_i.

After successfully executing the handover protocol, MN i and AP2 share a session key, since $K_{i-2} = e(sH_1(pid_i)), H_1(ID_{AP2})) = e(H_1(pid_i), H_1(ID_{AP2}))^s = e(H_1(pid_i), sH_1(ID_{AP2})) = K_{2-i}$. Furthermore, the use of a pseudo-ID enables unilateral anonymous authentication for the MN i, and each session is uniquely identified by (pid_i, ID_{AP2}).

Figure 1. The handover authentication phase in He *et al.*'s improved PairHand protocol.

MN i		AP2
$\{(pid_1, sH_1(pid_1)), (pid_2, sH_1(pid_2)), ...\}$		$\{sH_1(ID_{AP2})\}$
$M_i = pid_i\|\|ID_{AP2}\|\|ts\|\|rb$ $\sigma_i = H_2(M_i) \cdot sH_1(pid_i)$	$\xrightarrow{M_i, \sigma_i}$	Check ts
$K_{i-2} = e(sH_1(pid_i), H_1(ID_{AP2}))$		Check $e(\sigma_i, P) \stackrel{?}{=} e(H_2(M_i) \cdot H_1(pid_i), P_{pub})$ $K_{2-i} = e(H_1(pid_i), sH_1(ID_{AP2}))$ $Aut = H_2(K_{2-i}\|\|pid_i\|\|ID_{AP2})$
$Ver = H_2(K_{i-2}\|\|pid_i\|\|ID_{AP2})$ Check $Ver \stackrel{?}{=} Aut$	$\xleftarrow{pid_i, ID_{AP2}, Aut}$	

5. Attack on the Protocol

For the original PairHand protocol, what enables an attacker to recover the private key $sH_1(pid_i)$ is that when $H_2(M_i)$ and q are co-prime, he can use the inverse u of $H_2(M_i)$ modulo q to get $u \cdot \sigma_i = u \cdot H_2(M_i) \cdot sH_1(pid_i) = sH_1(pid_i) (\mathrm{mod} q)$, since $u \cdot H_2(M_i) = 1 (\mathrm{mod} q)$. The countermeasures of He *et al.* [15] are to restrict the group order q to be a composite and to append redundant bits into the request message M_i to ensure that the resulting $H_2(M_i)$ and q are not co-prime. By doing this, it seems that the private key $sH_1(pid_i)$ will not be exposed by the signature σ_i, since there is no modular inverse for $H_2(M_i)$.

However, the following attack will show that He *et al.*'s improved protocol [15] does not eradicate the design weakness. Our attack is based on the assumption [6] that adversary $\mathcal{A}$ has total control over all communication channels between AP2 and MN i. This means that the adversary may intercept, delete or modify any message in the channels. Suppose that MN i requests the service of a new access point AP2 by sending the message (M_i, σ) (where $M_i = pid_i||ID_{AP2}||ts||rb$ and $\sigma_i = H_2(M_i) \cdot sH_1(pid_i)$) in a wireless channel, which is dominated by the adversary $\mathcal{A}$. $\mathcal{A}$ interrupts the request message, so that MN i will not receive the response from AP2. After a certain delay, MN i will regenerate a new request message (M_i', σ_i') and send it to AP2, where $M_i' = pid_i||ID_{AP2}||ts'||rb'$, $\sigma_i' = H_2(M_i') \cdot sH_1(pid_i)$, ts' denotes a new

timestamp and rb' is a new redundant bit string. Since $\mathcal{A}$ controls the whole network communication, it can easily capture the new message. Once the adversary $\mathcal{A}$ owns two authentication messages (M_i, σ) and (M_i', σ_i') corresponding to the same pseudo-ID pid_i, it randomly selects two values $x_1, x_2 \in \mathbb{Z}_q$ and computes:

$$\begin{aligned} \alpha &= (x_1 \cdot \sigma_i) + (x_2 \cdot \sigma_i') \\ &= (x_1 \cdot H_2(M_i) sH_1(pid_i)) + (x_2 \cdot H_2(M_i') sH_1(pid_i)) \\ &= (x_1 \cdot H_2(M_i) + x_2 \cdot H_2(M_i')) sH_1(pid_i). \end{aligned}$$

Then, $\mathcal{A}$ directly computes $\beta = (x_1 \cdot H_2(M_i) + x_2 \cdot H_2(M_i'))(\mathrm{mod} q)$ by using M_i and M_i' and checks whether β and q are co-prime. If yes, it generates $\gamma = (x_1 \cdot H_2(M_i) + x_2 \cdot H_2(M_i'))^{-1}(\mathrm{mod} q)$ and computes the private key $sH_1(pid_i) = \gamma \cdot \alpha$. Otherwise, it reselects random values x_1, x_2 and computes α and β again, until β and q are co-prime.

As q is a randomly generated system parameter, M_i and M_i' are random messages and x_1 and x_2 are randomly chosen from $\mathbb{Z}_p$, we can approximately view q and $\beta = (x_1 \cdot H_2(M_i) + x_2 \cdot H_2(M_i'))(\mathrm{mod} q)$ as two independent random numbers. Let $\{p_1, p_2, p_3, \cdots\}$ denote the ascending sequence of all prime numbers, and let p_k be the largest prime number less than q. The probability that β is divisible by a prime p_i is $\frac{1}{p_i}$, where $i \leq k$, and the same fact holds for q. Therefore, the probability that the two numbers β and q are both divisible by this prime number is $\frac{1}{p_i^2}$, whilst the probability that at least one of them is not is $1 - \frac{1}{p_i^2}$. Thus, the probability of the success of our attack for one time, $P_{success}$, which is equal to the probability that β and q are co-prime, is:

$$\begin{aligned} P_{success} &= \prod_{i=1}^{k} (1 - \frac{1}{p_i^2}) \\ &> \prod_{i=1}^{\infty} (1 - \frac{1}{p_i^2}) \\ &= (\prod_{i=1}^{\infty} \frac{1}{1 - p_i^{-2}})^{-1} \\ &= \frac{6}{\pi^2} \\ &\approx 0.6079. \end{aligned}$$

Obviously, a natural way to cope with the above attack is to ensure that each pseudo-ID is used only one time, regardless of whether the AP responds correctly, which will require different signatures to correspond to different pseudo-IDs. As a consequence, it is impossible for an adversary to compute out the private key by using linear combinations of two signatures. However, this countermeasure will largely reduce the availability of the handover authentication protocol and give rise to more serious security problems, as shown as follows. When an MN moves and leaves from the service range of its old AP, it will attempt to connect to and identify a new AP by instantly sending authentication messages. Once a pseudo-ID is used only one time, an attempt to connect will cost one pseudo-ID of the MN, which will cause a great waste on the pseudo-IDs and force the MN to store a much larger number of pseudo-IDs. However, mobile nodes are often lightweight devices and have limited storage spaces, this

makes them unable to afford a large number of redundant pseudo-IDs. Additionally, the increase of the number of pseudo-IDs will lead to the growth of the length of pseudo-IDs, which deeply affects the efficiency of the computation and the communication of the whole wireless network. More seriously, if there is an adversary who always interrupts the request authentication message of an MN, the MN will eventually use up all its pseudo-IDs and be out of the service of the system, due to instantly repeating the request. Such an attack can be avoided by using additional precautions, such as delaying the response or introducing exponentially increasing delays after failed attempts and switching to other AP after an excessive amount of failures. However, all of these measures are very costly and can cause more additional risks, which is contrary to the design rational of PairHand.

6. Our Handover Authentication Protocol

According to the above analysis, the point to overcome the security weakness of the two PairHand protocols is to provide a secure authentication mechanism for the first message transmission. Below, we provide a simple scheme, which not only eliminates the security risks mentioned above, but greatly preserves the desirable efficiency features of the original protocol. Similar to PairHand, the proposed scheme is composed of four phases: system initialization, handover authentication, batch authentication and DoS attack resistance, where the first phase and the fourth phase are the same as those of the PairHand protocol. For the sake of completeness, all of the four phases are fully described in the following.

6.1. System Initialization

Let $\mathbb{G}$ be a cyclic additive group of composite order q and $\mathbb{G}_T$ be a cyclic multiplicative group of the same order. Let P be a generator of $\mathbb{G}$ and e be a bilinear map $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$.

The AS randomly chooses a value $s \in \mathbb{Z}_q^*$ as the master key, computes the corresponding public key $P_{pub} = sP$ and selects two cryptographic hash functions H_1 and H_2, where $H_1 : \{0,1\}^* \rightarrow \mathbb{G}$ and $H_2 : \{0,1\}^* \rightarrow \mathbb{Z}_q^*$. The resulting public system parameter, $params$, is $\{\mathbb{G}, \mathbb{G}_T, q, P, P_{pub}, H_1, H_2\}$, and the private secret of AS is s. For each AP, AS computes $H_1(ID_{AP})$ and $sH_1(ID_{AP})$ as the public and private keys of that AP, respectively, and delivers them to the AP via a secure channel, where ID_{AP} is the identity of the AP.

For the registration of a qualified MN i with real identity ID_i, AS generates a family of unlinkable pseudo-IDs $PID = \{pid_1, pid_2, \cdots\}$, computes the public key $H_1(pid_j)$ and the corresponding private key $s \cdot H_1(pid_j)$ for each pseudo-ID $pid_j \in PID$ and, finally, securely sends to MN i all tuples $(pid_j, sH_1(pid_j))$.

6.2. Handover Authentication

When an MN, say i, moves into the communication range of a new AP (AP2), a handover authentication process will be performed between MN i and AP2 in the following steps.

(1) MN i firstly picks an unused pseudo-ID pid_i and the corresponding private key $sH_1(pid_i)$ and computes $M_i = (pid_i \| ID_{AP2} \| ts)$, where ts is the timestamp. Then, MN i chooses a random value

$r_i \in \mathbb{Z}_q^*$, which is a nonce, computes $R_i = r_i P$ and $\sigma_i = H_2(M_i||R_i) \cdot sH_1(pid_i) + r_i P_{pub}$ and unicasts the access request message M_i and its signature pair (R_i, σ_i) to AP2. Finally, it computes the session key with AP2 as $K_{i-2} = e(sH_1(pid_i), H_1(ID_{AP2}))$.

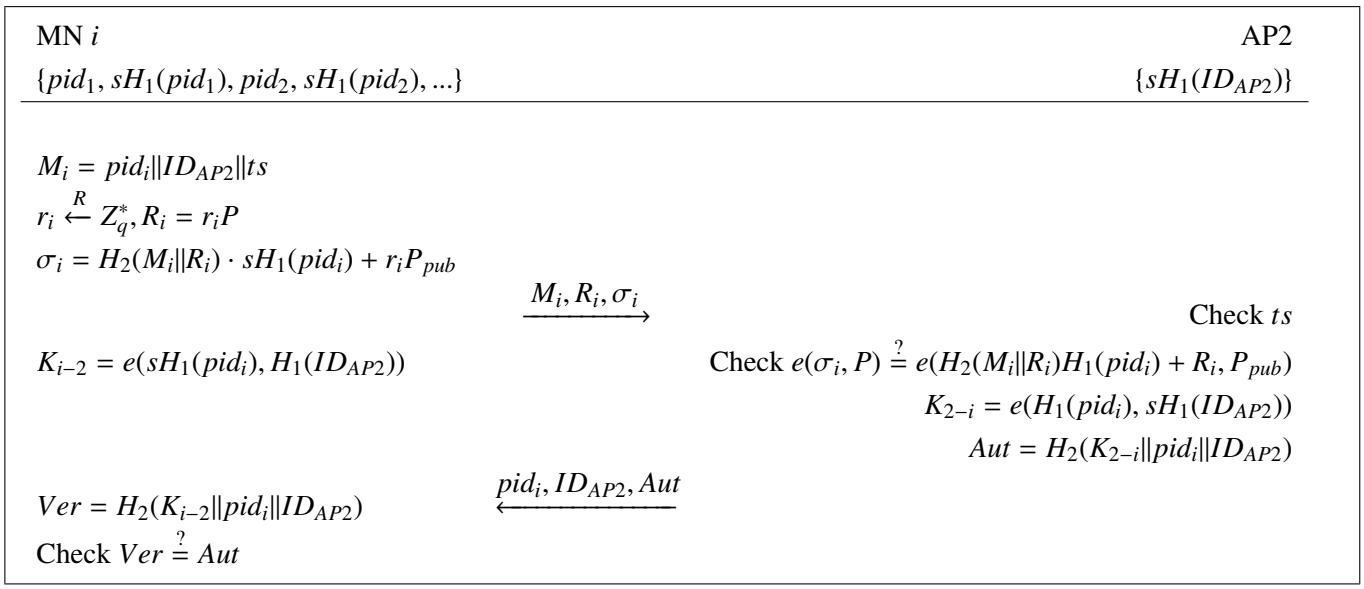
(2) Upon receiving the message $\{M_i, r_i, \sigma_i\}$, AP2 checks the timestamp ts. If invalid, the request is rejected. Otherwise, AP2 verifies if $e(\sigma_i, P)$=$e(H_2(M_i||R_i)H_1(pid_i) + R_i, P_{pub})$. If true, AP2 computes the session key $K_{2-i} = e(H_1(pid_i), sH_1(ID_{AP2}))$ and the authentication code $Aut = H_2(K_{2-i}||pid_i||ID_{AP2})$ and, then, sends the tuple $\{pid_i, ID_{AP2}, Aut\}$ to MN i.

(3) Upon receipt of the message $\{pid_i, ID_{AP2}, Aut\}$, MN i computes the verification code $Ver = H_2(K_{i-2}||pid_i||ID_{AP2})$ and compares it with Aut. If they are equal, MN i confirms that AP2 is legitimate and that the generated session key is valid. Otherwise, MN i cancels the connection.

(4) At last, AP2 securely transports $\{M_i, \sigma_i\}$ to AS. By receiving this message, AS can identify the real identity of MN i according to the pseudo-ID in M_i.

The handover authentication phase of the proposed scheme is also shown in Figure 2.

Figure 2. The handover authentication phase in our protocol.



6.3. Batch Authentication

A mass of signature verifications is likely to cause the potential bottleneck at APs. Batch authentication [14] is a desirable feature to solve the problem, which allows APs to verify multiple signatures simultaneously. Its advantage lies in that the total computation cost in the verification performed by APs can be apparently reduced.

Our protocol still enjoys the batch authentication feature. Suppose n request messages $\{M_1, R_1, \sigma_1\}$, $\{M_2, R_2, \sigma_2\}, \cdots$, $\{M_n, R_n, \sigma_n\}$, come simultaneously from n distinct MNs, MN 1, MN 2, $\cdots$, MN n, respectively. The target AP can perform a batch verification on these n signatures as follows:

$$\begin{aligned} & e(\sum_{i=1}^{n} \sigma_i, P) \\ = \; & e(\sum_{i=1}^{n} (H_2(M_i \| r_i) \cdot sH_1(pid_i) + r_i P_{pub}), P) \\ = \; & e(\sum_{i=1}^{n} (H_2(M_i \| r_i) \cdot sH_1(pid_i) + r_i sP), P) \\ = \; & e(\sum_{i=1}^{n} (H_2(M_i \| r_i) \cdot H_1(pid_i) + R_i), P_{pub}) \end{aligned}$$

From the above equation, it is obvious that the computation cost of verifying n signatures is dramatically reduced to n point multiplication and two pairing operations by using the batch processing.

6.4. DoS Attack Resistance

In the handover authentication circumstance, DoS attack is an attempt to exhaust the resources of AP and AS and make them unavailable to its intended partners. A usual manner adopted by the adversary is to inject bogus access requests to the networks, forcing the APs to perform expensive cryptographic verifications and eventually exhaust their resources.

The proposed scheme still adopts the polynomial-based lightweight verification of PairHand [14] to resist the DoS attack. In the system initialization phase, AS randomly generates a bivariate t-degree polynomial $f(x, y) = \sum_{i,j=0}^{t} a_{ij} x^i y^j$ over a prime field F_p, such that $f(x, y) = f(y, x)$. When MN i registers to AS, for each pseudo-ID pid_i, AS computes $f(pid_i, y)$, which is a polynomial share of $f(x, y)$, and then securely transmits them to MN i. Furthermore, AS computes and deliveries $f(ID_{AP}, y)$ to each AP, where ID_{AP} is the identity of the AP. As the evaluation of the polynomial is very fast [14], each AP can perform a lightweight verification on the access request from MN i by checking $f(pid_i, ID_{AP}) \stackrel{?}{=} f(ID_{AP}, pid_i)$, where the former is computed by MN i with $f(pid_i, y)$ at point ID_{AP} and the later is done by the AP with $f(ID_{AP}, y)$ at point pid_i. Once an AP is under attack, it starts the above measure, adding "Yes" and its identity into the beacon messages. As a result, DoS attack can be effectively mitigated, since each AP can promptly verify the authorized user with the communication key before conducting expensive cryptographic verifications.

6.5. Security Analysis

Theorem 1. *Assume hash functions H_1 and H_2 are random oracles. Let $\mathcal{A}$ be a probabilistic polynomial time Turing machine. Let Q_s, Q_1 and Q_2 respectively denote the number of queries that $\mathcal{A}$ can ask of the Sendoracle, the number of queries that $\mathcal{A}$ can ask of the H_1 random oracle and the number of queries that $\mathcal{A}$ can ask of the H_2 random oracle. If the attacker $\mathcal{A}$ can successfully violate the MN-to-AP authentication security of the protocol within time T, with probability $\varepsilon \geq 10(Q_s + 1)(Q_s + Q_2)/q$, then*

another probabilistic polynomial time Turing machine $\mathcal{B}$ can be built to utilize $\mathcal{A}$ to break the CDH problem in expected time $T' \leq 120686Q_1^2Q_2T/\varepsilon$.

Proof. Suppose that the attacker $\mathcal{B}$ is given a challenging CDH triple (P, aP, sP) and its goal is to compute asP. $\mathcal{B}$ runs $\mathcal{A}$ as a subroutine and simulates the environment for attacking the protocol. According to the challenging instance, $\mathcal{B}$ provides $\mathcal{A}$ the public parameters $(\mathbb{G}, \mathbb{G}_T, e, q, P, P_{pub}, H_1, H_2)$, such that $P_{pub} = sP$.

Without loss of generality, we assume that for any pseudo-ID, the adversary $\mathcal{A}$ invokes H_1, H_2, $Send$, $Execute$ and $Corrupt$ at most once. To provide consistent responses for these queries, $\mathcal{B}$ maintains two lists L_{H_1} and L_{H_2}, which are initially empty.

H_1-query: When $\mathcal{A}$ invokes an H_1 query for pid_i, $\mathcal{B}$ checks whether $pid_i = pid_U$. If yes, $\mathcal{B}$ returns aP. Otherwise, $\mathcal{B}$ returns a randomly selected value $h \in \mathbb{G}$ and appends $< pid_i, h >$ into the list L_{H_1}.

H_2-query: When $\mathcal{A}$ invokes an H_2 query for messages (m, R), $\mathcal{B}$ returns a random number $t \in \mathbb{Z}_q^*$ and stores $< m, R, t >$.

Corrupt-query: If the queried pseudo-ID is legal and is not equal to pid_U, $\mathcal{B}$ searches the corresponding item in the list L_{H_1} according to the pseudo-ID and then returns the secret key. Otherwise, $\mathcal{B}$ returns $\perp$.

Execute-query: This query is responded to by invoking the corresponding $Send$ queries.

Send-query: When $\mathcal{A}$ invokes a send(Π_i^n, m) query, simulator $\mathcal{B}$ extracts pid_i involved in the query and uses it to invoke query H_1. Then, $\mathcal{B}$ randomly chooses $r_i, t \in \mathbb{Z}_q^*$, computes $\sigma_i = r_iP_{pub}$, $R_i = r_iP - tH_1(pid_i)$ and stores the item $< m, R_i, t >$ in the list L_{H_2}. Finally, it outputs (pid_i, R_i, σ_i). If there is no collision of queries to the random oracle during the process, $\mathcal{B}$ can successfully simulate the protocol environment in front of $\mathcal{A}$, due to the fact that the probabilities of the duple $(\alpha, \beta, \gamma, \delta)$, such that $\beta \in \mathbb{Z}_q^*$, $\alpha, \gamma, \delta \in \mathbb{G}$ and $e(\gamma, P) = e(\beta\alpha + \delta, sP)$ appear, the following two distributions Γ and Γ' are the same.

$$\Gamma' = \left\{ (h, t, \sigma, R) \middle| \begin{array}{c} r, t \in_R \mathbb{Z}_q^* \\ h \in_R \mathbb{G} \\ R = rP - th \\ \sigma = rsP \end{array} \right\} \quad \text{and} \quad \Gamma = \left\{ (h, t, \sigma, R) \middle| \begin{array}{c} r, t \in_R \mathbb{Z}_q^* \\ h \in_R \mathbb{G} \\ R = rP \\ \sigma = tsh + rsP \end{array} \right\}$$

According to the Forking lemma [21], if $\mathcal{A}$ outputs a valid authentication message tuple (pid_U, m, σ, R), after a polynomial replay of the attacker $\mathcal{A}$ with the same tape, but different choices of H_2, $\mathcal{B}$ obtains two valid message tuples $(pid_U, m, \sigma = tsaP + rsP, R = rP)$ and $(pid_U, m, \sigma' = t'saP + rsP, R = rP)$ with $t \neq t'$ and eventually resolves the CDH challenge by computing $asP = (\sigma - \sigma')/(t - t')$. □

Additionally, the probability of that the two forged authentication messages correspond to pid_U is $1/Q_1^2$. As a result, the upper bound of the expected time for breaking the CDH problem will be expanded Q_1^2-times the one in the Forking lemma.

Theorem 2. *Assume hash functions H_1 and H_2 are random oracles. Let $\mathcal{A}$ be a probabilistic polynomial time Turing machine. If the attacker $\mathcal{A}$ can successfully violate the AP-to-MN authentication security of the protocol, then another probabilistic polynomial time Turing machine $\mathcal{B}$ can be built to utilize $\mathcal{A}$ to break the BDH problem.*

Proof. Let (P, aP, bP, sP) be the BDH instance provided to the simulator $\mathcal{B}$. To simulate the attacking environment for $\mathcal{A}$, $\mathcal{B}$ publishes the public parameters $(\mathbb{G}, \mathbb{G}_T, e, q, P, P_{pub}, H_1, H_2)$, such that $P_{pub} = sP$ and maintains two hash lists L_{H_1} and L_{H_2}, which are initially empty. Without loss of generality, we assume that for any pseudo-ID or AP identity, the adversary $\mathcal{A}$ invokes H_1, H_2, $Send$, $Execute$ and $Corrupt$ at most once. Let Q_M and Q_A be the number of queries that $\mathcal{A}$ can ask of the random oracle H_1 for MN nodes and the number of queries that $\mathcal{A}$ can ask of the random oracle H_1 for AP nodes, respectively. $\mathcal{B}$ guesses the target session between the MN pid_U and the AP ID_V, which are randomly chosen.

H_1-query: If $\mathcal{A}$ makes an H_1 query for pid_U, $\mathcal{B}$ returns aP. If the query is for the AP identification ID_V, $\mathcal{B}$ returns bP. Otherwise, $\mathcal{B}$ returns a randomly selected value $h \in \mathbb{G}$ and adds $< pid_i, h >$ or $< ID_V, h >$ into the list L_{H_1}.

H_2-query: When $\mathcal{A}$ invokes an H_2 query for messages M, $\mathcal{B}$ chooses a random number $t \in \mathbb{Z}_q$, stores $< M, t >$ the list L_{H_2} and then returns it.

Corrupt-query: If the queried identity is legal and is not equal to pid_U and ID_V, $\mathcal{B}$ searches the corresponding item in the list L_{H_1} according to the identity and then returns the secret key. Otherwise, $\mathcal{B}$ returns $\perp$.

Execute-query: This query is responded to by invoking the corresponding $Send$ queries.

Send-query: There are two types of $Send$ queries: MN-to-AP and AP-to-MN, denoted by $Send1$ and $Send2$, respectively. $\mathcal{B}$ answers them by invoking the H_1 and H_2 queries.

- If the query is $Send1$, simulator $\mathcal{B}$ randomly chooses $r_i, t \in \mathbb{Z}_q^*$ and computes $\sigma_i = r_i P_{pub}$, $R_i = r_i P - tH_1(pid_i)$ by invoking H_1 queries with ID_i. Then, it adds the item (m, R_i, t) in the list L_{H_2} and outputs message tuple (pid_i, R_i, σ_i).
- If the query is $Send2$, simulator $\mathcal{B}$ checks whether $e(\sigma_i, P)$=$e(H_2(M_i \| R_i)H_1(pid_i) + R_i, P_{pub})$. If false, $\mathcal{B}$ outputs "$\perp$". Otherwise, $\mathcal{B}$ chooses a random value $k \in \mathbb{G}_T$ and computes $aut = H_2(k \| pid_i \| ID_j)$ by making the H_2 query. Finally, it returns the message tuple $(aut \| pid_i \| ID_j)$.

The success of $\mathcal{B}$ breaking the BDH problem denotes the event that pid_U and ID_V are partners and $\mathcal{A}$ asks the H_2 query with a tuple $(K \| pid_U \| ID_V)$ where $K = e(P, P)^{abs}$. According to the above simulation, the probability that pid_U and ID_V are partners is $1/Q_M Q_A$. Therefore, if $\mathcal{A}$ outputs a valid authentication message with probability ε, the probability of the success of $\mathcal{B}$ is less than $\varepsilon / Q_M Q_A$. □

Theorem 3. *Assume hash functions H_1 and H_2 are random oracles. If the protocol enjoys the mutual authentication security, it is also semantically secure.*

Proof. To prove the semantic security for the protocol, we apply the same simulation way used in proving Theorem 2. Let F_1 (or F_2, respectively) denote the event that the attacker successfully forges an MN-to-AP (or AP-to-MN, respectively) authentication message. Let S_0 (or S_1, respectively) denote the event that in the real (or simulated, respectively) attacking game, the attacker successfully guess the challenge bit involved in the $Test$ oracle. If both the events F_1 and F_2 do not happen, the real and simulated games proceed identically, and we get the following equation:

$$Pr[S_0 \wedge \neg F_1 \wedge \neg F_2] = Pr[S_1 \wedge \neg F_1 \wedge \neg F_2].$$

On the other hand, it is obvious that in the simulation, the attacker cannot obtain any information about the protocol session key, since the session key is a randomly chosen value not related to any message transported in the public information channel. This means that the attacker can only guess the hidden bit, so that $Pr[S_1] = 1/2$. Following the difference lemma [22], we get

$$|Pr[S_0] - Pr[S_1]| \leq Pr[F_1 \vee F_2] \leq Pr[F_1] + Pr[F_2].$$

According to the definition of the semantic security for the protocol, then we have $Adv_{\mathcal{A},P}(k,t) \leq 2 \cdot (Pr[F_1] + Pr[F_2]) = 2 \cdot Auth_{\mathcal{A},P}^{MN \to AP}(k,t) + 2 \cdot Auth_{\mathcal{A},P}^{AP \to MN}(k,t)$, □

6.6. Performance Comparison

Compared with the existing handover authentication protocols, the proposed protocol has the advantage in communication, computation and security. For those protocols prior to PairHand [1–4,6–10,12], its superiority comes through the low burden on AS, the two-run handshakes between MN and AP, the batch authentication and the privacy protection for MN. To evaluate its advantage over the post-PairHand protocols [14,15,17], we mainly consider its performance superiority on secret key size, computational cost and security features. In Table 1, we present the comparison results on these aspects among He *et al.*'s improved PairHand [15], Tsai *et al.*'s [17] protocol and our scheme. For computational cost, we focus on the time spent on the high cost operations, such as the time spent on the paring operations (T_p), the time spent on the multiplications on the elliptic curve (T_m) and the time spent on the search for non-co-primes (T_s), while the time spent on highly efficient operations, such as the hash function and the scalar addition on the elliptic curve, is neglected. The estimate of the time consumption at an MN is based on He *et al.*'s work in [14,15], where by using the MNT curve with the order of 160 bits and the degree $k = 6$ and the MIRACL and PBC libraries (c/c++), an MN runs on an 800 MHz processor. To evaluate the length of the messages transmitted in the protocol execution, we assume that the lengths of pid_i, ts and ID_{AP2} are four, two and four bytes, respectively. We notice that the computational time of our protocol and Tsai *et al.*'s protocol are much lower than He *et al.*'s protocol, due to their prime-order work groups. This is because the composite order in He *et al.*'s protocol should be at least 1024-bit to be infeasible to factorize, while a 160-bit prime order is enough to achieve the same security level. An estimation [23] shows that the composite-order pairing is roughly 50-times slower than its prime-order counterpart. For security, both our scheme and Tsai *et al.*'s protocol enjoy provable security, but He *et al.*'s protocol does not. In terms of the secret key size, our protocol is superior to Tsai *et al.*'s protocol and is the same as the original PairHand protocol [14]. As a result, our scheme can be easily implanted to the running environment of the original PairHand protocol without any change to the public and private parameters.

Table 1. Protocol comparisons. MN, mobile node; AP, access point.

	He *et al.* [15]	**Tsai *et al.* [17]**	**Ours**
The number of private keys	1	1	1
The number of Public keys	1	2	1
Provably secure	No	Yes	Yes
MN Anonymity	Yes	Yes	Yes
MN unlinkability	Yes	Yes	Yes
MN computational cost	$1T_p + 1T_m + 1T_s$	$1T_p + 1T_m$	$1T_p + 1T_m$
The computation time consumption at an MN	$\approx$ 299.332 ms	$\approx$ 7.564 ms	$\approx$ 7.564 ms
AP computational cost	$1T_p$	$1T_p$	$1T_p$
The length of the transmitted messages	166 bytes	78 bytes	78 bytes
Work group	composite order	prime order	prime order

7. Conclusions

In this paper, in reviewing the PairHand family protocols, we present a stronger key recovery attack on an improved PairHand protocol, which requires less signatures to be generated with the same private key compared with the existing attacks. Consequently, we present a new handover authentication protocol and prove its security in the random oracle model. Compared with the two latest handover authentication protocols, the proposed protocol has the advantages of efficiency and security.

Acknowledgements

The work is partially supported by the Fundamental Research Funds for the Central Universities under Grant No.2012JBM104.

Author Contributions

Weijia Wang is the principal researcher of this study and main author of this work. Lei Hu has contributed to the revision of this paper and provided insightful comments and suggestions.

Conflicts of Interest

The authors declare no conflict of interest.

References

1. Choi, J.; Jung, S. A secure and efficient handover authentication based on light-weight Diffie-Hellman on mobile node in FMIPv6. *IEICE Trans. Commun.* **2008**, *E-91B*, 605–608.
2. Choi, J.; Jung, S. A handover authentication using credentials based on chameleon hashing. *IEEE Commun. Lett.* **2010**, *14*, 54–56.

3. Choi, J.; Jung, S.; Kim, Y.; Yoo, M. A Fast and Efficient Handover Authentication Achieving Conditional Privacy in V2I Networks. In *Smart Spaces and Next Generation Wired/Wireless Networking*; Balandin, S., Moltchanov, D., Koucheryavy, Y., Eds.; Springer: Berlin, Germany, 2009; pp. 291–300.
4. Chang, C.-C.; Lee, C.-Y.; Chiu, Y.-C. Enhanced authentication scheme with anonymity for roaming service in global mobility networks. *Comput. Commun.* **2009**, *32*, 611–618.
5. Chang, C.-C.; Tsai, H.-C. An anonymous and self-verified mobile authentication with authenticated key agreement for large-scale wireless networks. *IEEE Trans. Wirel. Commun.* **2010**, *9*, 3346–3353.
6. He, D.; Bu, J.; Chan, S.; Chen, C.; Yin, M. Privacy-preserving universal authentication protocol for wireless communications. *IEEE Trans. Wirel. Commun.* **2011**, *10*, 431–436.
7. He, D.; Ma, M.; Zhang, Y.; Chen, C.; Bu, J. A strong user authentication scheme with smart cards for wireless communications. *Comput. Commun.* **2011**, *34*, 367–374.
8. Hsiang, H.-C.; Shih, W.-K. Improvement of the secure dynamic ID based remote user authentication scheme for multi-server environment. *Comput. Stand. Interfaces* **2009**, *31*, 1118–1123.
9. Kim, Y.; Ren, W.; Jo, J.; Yang, M.; Jiang, Y.; Zheng, J. SFRIC: A secure fast roaming scheme in wireless LAN using ID-based cryptography. In Proceedings of the IEEE International Conference on Communications (ICC '07), Glasgow, UK, 24–28 June 2007; pp. 1570–1575.
10. Liao, Y.-P.; Wang, S.-S. A secure dynamic ID based remote user authentication scheme for multi-server environment. *Comput. Stand. Interfaces* **2009**, *31*, 24–29.
11. Yang, G. Comments on "An anonymous and self-verified mobile authentication with authenticated key agreement for large-scale wireless networks". *IEEE Trans. Wirel. Commun.* **2011**, *10*, 2015–2016.
12. Yang, G.; Huang, Q.; Wong, D.S.; Deng, X. Universal authentication protocols for anonymous wireless communications. *IEEE Trans. Wirel. Commun.* **2010**, *9*, 168–174.
13. Munir, S.; Ren, B.; Jiao, W.; Wang, B.; Xie, D.; Ma, J. Mobile Wireless sensor Network: Architecture and Enabling Technologies for Ubiquitious Computing. In Proceedings of the 21st International Conference on Advanced Information Networking and Applications Workshops (AINAW'07), Ontario, Canada, 21–23 May 2007; pp. 113–120.
14. He, D.; Bu, J.; Chan, S.; Chen, C. Secure and efficient handover authentication based on bilinear pairing functions. *IEEE Trans. Wirel. Commun.* **2012**, *11*, 48–53.
15. He,D.; Chen, C.; Chan, S.; Bu, J. Analysis and Improvement of a Secure and Efficient Handover Authentication for Wireless Networks. *IEEE Commun. Lett.* **2012**, *16*, 1270–1273.
16. Yeo, S.; Yap, W.-S.; Liu, J.; Henricksen, M. Comments on " Analysis and improvement of a secure and efficient handover authentication based on bilinear pairing functions". *IEEE Commun. Lett.* **2013**, *17*, 1521–1523.
17. Tsai, J. ; Lo, N.; Wu, T. Secure handover authentication protocol based on bilinear Pairings. *Wirel. Pers. Commun.* **2013**, *73*, 1037–1047.
18. Bellare, M.; Rogaway, P. Entity authentication and key distribution. In *Advances in Cryptology—CRYPTO'93*; Springer: Berlin/Heidelberg, Germany, 1993; pp. 232–249.

19. Bellare, M.; Rogaway, P. Provably-Secure Session Key Distribution: The Three Party Case. In Proceedings of the Twenty-Seventh Annual ACM Symposium on Theory of Computing (STOC'95), Las Vegas, NV, USA, 29 May–1 June 1995; pp. 57–66.
20. Bresson, E.; Chevassut, O.; Pointcheval, D. Provably Authenticated Group Diffie-Hellman Key Exchange-The Dynamic Case. In *Advances in Cryptology—ASIACRYPT 2001*; Boyd, C., Ed.; Springer: Berlin, Germany, 2001; pp. 290–309.
21. Pointcheval, D.; Stern, J. Security arguments for digital signatures and blind signatures. *J. Crytol.* **2000**, *13*, 361–396.
22. Shoup, V. Sequences of games: A tool for taming complexity in security proofs. 2004. Available online: http://www.shoup.net/papers/games.pdf (accessed on 26 June 2014).
23. Freeman, D. Converting pairing-based cryptosystems from composite-order groups to prime-order groups. In *Advances in Cryptology—EUROCRYPT 2010*; Gilbert, H., Ed.; Springer: Berlin, Germany, 2010; pp. 44–61.