A simple deniable authentication protocol based on the Diffie-Hellman algorithm

Rongxing Lu∗†, Xiaodong Lin‡, Zhenfu Cao†, Linquang Qin†, and Xiaohui Liang†
†Department of Computer Science and Engineering, Shanghai Jiao Tong University
No. 800, Dongchuan Road, Shanghai, P. R. of China, 200240
‡Department of Electrical and Computer Engineering, University of Waterloo
Waterloo, Ontario, Canada N2L 3G1

(Accepted 6 August 2007)

Deniable authentication protocol is a new authentication mechanism in secure computer communication, which not only enables an intended receiver to identify the source of a received message but also prevents a third party from identifying the source of the message. In this paper, based on the Diffie-Hellman algorithm, we propose a new simple deniable authentication protocol from a provably secure simple user authentication scheme. Compared with other deniable authentication protocols, our proposed protocol not only achieves the property of deniable authenticity, but also provides the mutual authentication between the sender and the intended receiver and the confidentiality.

Keywords: User authentication scheme, Deniable authentication protocol, Diffie-Hellman algorithm, Bilinear pairing

C.R. Category: E.3

1. Introduction

Deniable authentication protocol is a new security authentication mechanism. Compared with traditional authentication protocols, it has the following two features: i) It enables an intended receiver to identify the source of a given message; ii) However, the intended receiver can not prove to any third party the identity of the sender [2]. Just due to these two features, deniable authentication protocol has become a solution to some special requirements for secure communication and a lot of relevant work [2, 7–14, 16] has been published in recent years.


*Corresponding author. Email: rxlu.cn@gmail.com
More recently, many non-interactive deniable authentication protocols were also proposed [10–14,16].

However, when we take a closer look at these protocols, we will find none of them has provided the property of confidentiality, while the confidentiality should be one important part of the overall solution for truly secure communications. Therefore, in this paper, to resolve this problem, we would like to propose a new simple interactive deniable authentication protocol based on the Diffie-Hellman algorithm [6]. Our proposed protocol will achieve the following properties.

- Deniable authentication: The intended receiver can identify the source of a given message, but cannot prove the source to any third party.
- Mutual authentication: During the protocol execution, the sender and the intended receiver can authenticate each other.
- Confidentialities: Any outside adversary has no ability to gain the deniable authentication message from the transmitted transcripts.

The rest of this paper is organized as follows. In the next section, we first review the bilinear pairings [3] and Diffie-Hellman problems [1]. In section 3, we present a simple user authentication scheme and propose our simple deniable authentication protocol. Then, we discuss the security and performance of our proposed protocol in section 4 and section 5, respectively. Finally, we draw our conclusions in section 6.

2. Bilinear Pairings and Diffie-Hellman Problems

More recently, bilinear pairings have allowed the opening of a new territory in modern cryptography [3]. In this section, we briefly review the necessary facts used for our protocol.

2.1. Bilinear Pairings

Let \( G, G_T \) be two cyclic groups of the same prime order \( q \). Let \( e \) be a computable bilinear map \( e : G \times G \rightarrow G_T \), which satisfies the following properties:

- Bilinear: \( e(P^a, Q^b) = e(P, Q)^{ab} \), where \( P, Q \in G \) and \( a, b \in \mathbb{Z}_q^* \).
- Non-degenerate: There exists \( P, Q \in G \) such that \( e(P, Q) \neq 1_{G_T} \).
- Computability: There exists an efficient algorithm to compute \( e(P, Q) \) for all \( P, Q \in G \).

We call such a bilinear map \( e \) an admissible bilinear pairing, and the Weil or Tate pairing in elliptic curve can give a good implementation of the admissible bilinear pairing [3].

**Definition 2.1 (Bilinear Parameter Generator)** A bilinear parameter generator \( \text{Gen} \) is a probabilistic algorithm that takes a security parameter \( k \) as input and outputs a 5-tuple \( (q, G, G_T, e, g) \) as the bilinear parameters, including a prime number \( q \) with \( |q| = k \), two cyclic groups \( G, G_T \) of the same order \( q \), an admissible bilinear map \( e : G \times G \rightarrow G_T \) and a generator \( g \) of \( G \).

2.2. Diffie-Hellman Problems

The problems relevant to our deniable authentication protocol are of the Diffie-Hellman type, namely the Computational Diffie-Hellman (CDH) problem, the De-
A Simple User Authentication Scheme

3.1. A simple deniable authentication protocol based on the Diffie-Hellman algorithm

Let \((g, \mathcal{G}, e, g)\) be a 5-tuple generated by the bilinear parameter generator \(\mathcal{G}en(k)\), and let \(H : \{0, 1\}^* \rightarrow \{0, 1\}^l\) be a secure cryptographic hash function, where \(l\) is a security parameter, and let \(x, y, h \in \{0, 1\}^l\), the HDDH problem in \(\mathcal{G}\) is as follows: Given \((g, g^x, g^y, g^h)\), decide whether it is a hash Diffie-Hellman tuple \((g, g^x, g^y, g^h)\).

Owing to the property of bilinear, we know \(e(g^x, g^y) = e(g, g^z)\) in \(\mathcal{G}\) if and only if \((g, g^x, g^y, g^z)\) is a Diffie-Hellman tuple. Therefore, the DDH problem in \(\mathcal{G}\) is easy.

Since the CDH problem in \(\mathcal{G}\) is hard and the DDH problem in \(\mathcal{G}\) is easy, the group \(\mathcal{G}\) is called as a Gap Diffie-Hellman Group.

According to the literature [1], the hash Diffie-Hellman assumption would seem to be a much weaker assumption than the decisional Diffie-Hellman assumption. Here, although the DDH problem is easy in \(\mathcal{G}\), we have the confidence that the HDDH assumption still holds in \(\mathcal{G}\) provided that the hash function \(H : \{0, 1\}^* \rightarrow \{0, 1\}^l\) is secure. Throughout the rest of this paper, we assume that the HDDH problem is hard in \(\mathcal{G}\).

3. New Deniable Authentication Protocol

The Diffie-Hellman algorithm [6] is a well-known key exchange protocol, which allows two entities to agree on a shared session key over an insecure channel. Although it is not secure (against person-in-the-middle attack), yet, the Diffie-Hellman algorithm broke a new path in cryptography [6] and has been used as a building block for constructing a lot of complex cryptographic systems. In this section, we will develop our new deniable authentication protocol based on the Diffie-Hellman algorithm over the above Gap Diffie-Hellman Group \(\mathcal{G}\). As a preliminary work, we first present a simple user authentication scheme in the following.

3.1. A Simple User Authentication Scheme

Let \((g, \mathcal{G}, e, g)\) be a 5-tuple generated by the bilinear parameter generator \(\mathcal{G}en(k)\), and let \(H : \{0, 1\}^* \rightarrow \{0, 1\}^l\) be a secure cryptographic hash function,
where $l$ is security parameter. Assume that a user $U$, who holds the public key and private key pair $(Y_u = g^{x_u}, x_u \in \mathbb{Z}_q^*)$, wants to enter a remote server $RS$, as shown in Fig. 1, they should execute the following steps:

**Step 1:** When the user $U$ wants to enter the remote server $RS$, he first sends his identity $ID_u$ to $RS$.

**Step 2:** $RS$ then chooses a random number $r \in \mathbb{Z}_q^*$, computes $R_0 = g^r$, $R_1 = Y_u^r$, and sends $R_0$ back to the user $U$.

**Step 3:** User $U$ then uses his private key $x_u$ to compute $R'_0 = R_0^{x_u}$ and $h = H(ID_u, R_0, R'_0)$. Finally, $U$ sends $h$ to $RS$.

**Step 4:** Upon receiving $h$, $RS$ verifies it by checking the equality $h = H(ID_u, R_0, R_1)$. If it does hold, the user $U$ will be authenticated; otherwise, rejected, since $R_1 = R'_0 = g^{x_u}r$.

### 3.2. Proposed Deniable Authentication Protocol

Based upon the simple user authentication scheme above, we now propose our simple deniable authentication protocol. Our proposed protocol involves two entities: a sender $S$ and an intended receiver $R$.

As in the user authentication scheme, let $(q, G, G_T, e, g)$ be a 5-tuple generated by the bilinear parameter generator $Gen(k)$, and let $H : \{0,1\}^* \rightarrow \{0,1\}^l$ be a secure cryptographic hash function, where $l$ is a security parameter. The public key and private key pairs of the sender $S$ and the receiver $R$ are $(Y_s, x_s)$ and $(Y_r, x_r)$ respectively, where $x_s, x_r \in \mathbb{Z}_q^*$ and $Y_s = g^{x_s}$, $Y_r = g^{x_r}$.

Our simple deniable authentication protocol, as shown in Fig. 2, is then described as follows:

![Figure 2. Proposed simple deniable authentication protocol](image-url)
Step 1: The sender $S$ chooses a random number $u \in \mathbb{Z}_q^*$ and computes $U = g^u$, $U' = Y_s^u$ and then sends his identity $ID_s$ and $U$ to the receiver $R$.

Step 2: $R$ chooses a random number $v \in \mathbb{Z}_q^*$ and computes $V = g^v$, $V' = Y_s^v$. $R$ also uses his private key $x_r$ to compute $U'' = U^{x_r}$, $h_1 = H(ID_r, U, U'', V)$ and sends $(V, h_1)$ to $S$.

Step 3: $S$ checks the equality $h_1 = H(ID_r, U, U'', V)$. If it holds, $S$ is authenticated and $V$ will be accepted; otherwise rejected, since $U' = U'' = g^{ux_r}$.

Step 4: When $S$ wants to send a deniable message $M \in \{0, 1\}^l$, he first computes $V'' = V^{x_s}$ and sends both $h_2$ and $h_3$ to $R$, where $h_2 = H(V'') \oplus M$ and $h_3 = H(ID_s, V, V'', M)$.

Step 5: $R$ recovers $M$ by computing $h_2 \oplus H(V')$ and verifies its validity by checking equality $h_3 = H(ID_s, V, V'', M)$. If it holds, $M$ will be accepted; otherwise rejected, since $V' = V'' = g^{x_s v}$.

4. Security Analysis

In this section, we will give security analysis of our proposed deniable authentication protocol and show that it satisfies the properties of mutual authentication, confidentiality and deniability.

4.1. Security Analysis on Simple User Authentication Scheme

To prove our proposed deniable authentication protocol achieves the mutual authentication between the sender and the intended receiver, we will first show that the simple user authentication scheme in section 3.1 is secure.

Formal Security Notion: According to the resources and capabilities that an adversary may gain in most practical application scenarios, we provide a formal definition of user authentication schemes. Concretely, it is defined by the following game between an adversary $A$ and a challenge $C$.

Setup: On input of security parameters, $C$ runs the algorithm to generate the system parameters and public key and private key pairs $(pk_i, sk_i)$, $1 \leq i \leq n$, of $n$ users $U = \{U_1, U_2, \ldots, U_n\}$, and sends the system parameters and all public keys $pk_1, \ldots, pk_n$ to $A$.

Corrupt Queries: $A$ can corrupt some users in $U$ and obtain their private keys.

User Authentication Queries: $A$ also can make several user authentication queries on some uncorrupted users in $U$.

Impersonate: In the end, $A$ impersonates an uncorrupted user in $U$ by outputting a valid login authentication message.

The success probability of $A$ to win the game is defined by $Succ(A)$.

Definition 4.1 We say that a user authentication scheme is secure if the probability of success of any polynomial bounded adversary $A$ in the above game is negligible.

Theorem 4.2 Assume that $H$ behaves as a random oracle. Then the simple user authentication scheme in section 3.1 is secure provided that the CDH assumption holds in $G$.

Proof Assume that $A$ is an adversary, who can, with non-negligible probability,
break the simple user authentication scheme in section 3.1. Then, we can use \( A \) to construct another algorithm \( B \), which is able to break the CDH problem in \( G \) with another non-negligible probability.

**Setup:** First, algorithm \( B \) is given the system parameters \((g, G, G_T, c, g, H)\), where 
\[
H : \{0, 1\}^* \to \{0, 1\}^L
\]
behaves a random oracle \([5]\), and a CDH instance \((g, g^x, g^y)\) as her challenge, and her task here is to compute \( g^{xy} \). Let \( \mathcal{U} = \{U_1, \ldots, U_n\} \) be a set of \( n \) users who may participate in the system. \( B \) first picks a random number \( j \) from \( \{1, 2, \ldots, n\} \), and sets the user \( U_j \)'s public key \( Y_j = g^x \). Then, \( B \) chooses another \( n - 1 \) random numbers \( x_i \in \mathbb{Z}_q^* \) as user \( U_i \)'s secret key, where \( 1 \leq i \leq n \) and \( i \neq j \), and computes the corresponding public key \( Y_i = g^{x_i} \). Finally, \( B \) sends all public key \( Y_1, Y_2, \ldots, Y_n \) to the adversary \( A \).

**Corrupt Queries:** When \( A \) wants to corrupt the user \( U_i \)'s secret key, \( B \) will process as follows:

- If \( i = j \), \( B \) has to terminate the game and reports failure, since she has no knowledge on user \( U_j \)'s secret key.
- If \( i \neq j \), \( B \) returns the corresponding \( x_i \) to \( A \).

Clearly, after \( q_c \) times corrupting queries, this game doesn’t terminate with probability \( 1 - \frac{q_c}{n} \), where \( q_c < n \).

**\( H \) Random Oracle Queries:** To avoid collision and consistently respond to these random oracle queries, \( B \) maintains list \( \Lambda_H \), which is initially empty. Assume that “\( * \)” is a special symbol. When \( A \) makes a query on \( \alpha_i = \langle ID, R, R' \rangle \), if \( (\alpha_i, h_i) \) exists in \( \Lambda_H \), \( h_i \) is returned; else if \( (\langle ID, R, \* \rangle, h_i) \) exists and \( e(g, R') = e(Y_{ID}, R) \) (it means we find the CDH solution), then “\( \* \)” is set to \( R' \) and \( h_i \) is returned; otherwise \( B \) chooses a random number \( h_i \in \{0, 1\}^L \), adds \((\alpha_i, h_i)\) to \( \Lambda_H \) and returns \( h_i \) to \( A \).

**User Authentication Queries:** Since only the unilateral authentication is required in the simple user authentication scheme, the adversary is allowed to make the user authentication queries.

When \( A \) chooses \( R = g^r \) and launches the user \( U_i \) authentication query, \( B \) will respond as follows:

- If \( i = j \), \( B \) chooses a random number \( h_i \in \{0, 1\}^L \), adds the entry \((\alpha_i = \langle ID_i, R, \* \rangle, h_i)\) in \( \Lambda_H \) and returns \( h_i \) to \( A \). According the simulation rules of Random Oracle \( H \) described above, \( h_i \) is appointed to the query \( \langle ID_i, R, R' \rangle \), so \( B \) can always simulate the game successfully.
- If \( i \neq j \), \( B \) directly computes \( R' = R^{x_i} \), makes a query of \( \langle ID_i, R, R' \rangle \) to Random Oracle of \( H \). Suppose \( h_i \) is returned from the query, \( B \) sends \( h_i \) to \( A \).

**Impersonate:** In the end, \( A \) is ready to impersonate an uncorrupted user \( U_i \) from \( (n - q_c) \) users to enter the remote server. \( B \) then responds as follows:

- If \( i = j \), \( B \) sets \( R = g^y \) and sends it back to \( A \). \( A \) then sends a login message \( h \) to \( B \). Then looks up the \( \Lambda_H \).
  (i) If for some \((\alpha = \langle ID_j, R, R' \rangle, h) \in \Lambda_H, e(Y_j, R) = e(g^x, g^y) = e(g, R')\), returns \( R' \) as the CDH challenge \( g^{xy} \).
  (ii) Else \( A \) couldn’t win the game, the probability that the hash value of \( \langle ID_j, g^u, g^{xk} \rangle \) equals to \( h \) is at most \( \frac{1}{n} \), when it hasn’t be queried and will be returned randomly from the space \( \{0, 1\}^L \).
- If \( i \neq j \), \( B \) has to terminate the game, because it doesn’t help \( B \) to solve the
CDH problem.

We define the success probability of \( \mathcal{A} \) in breaking the simple authentication scheme is \( \text{Succ}(\mathcal{A}) \) and the advantage of \( \mathcal{B} \) in solving the CDH problem is \( \text{Adv}^{\text{CDH}}(\mathcal{B}) \). Then, from the above game, we will have

\[
\text{Succ}(\mathcal{A}) \cdot (1 - \frac{q_c}{n}) \cdot \frac{1}{n - q_c} - \frac{1}{2^l} \leq \text{Adv}^{\text{CDH}}(\mathcal{B}),
\]

That is,

\[
\text{Adv}^{\text{CDH}}(\mathcal{B}) \geq \frac{\text{Succ}(\mathcal{A})}{n} - \frac{1}{2^l}.
\]

This completes the proof. \( \square \)

4.2. Security Analysis on Deniable Authentication Protocol

Since our proposed deniable authentication protocol is based upon the simple user authentication protocol, we can easily prove that our protocol can achieve the mutual authentication between the sender and the intended receiver.

**Theorem 4.3** Our proposed protocol achieves the mutual authentication between the sender and the intended receiver.

**Proof** According to the result of Theorem 4.2, the sender \( S \) can authenticate the receiver \( R \) in Step 3, and the receiver \( R \) also can authenticate the sender \( S \) in Step 5. Hence it follows that our proposed protocol achieves the mutual authentication between the sender and the intended receiver. \( \square \)

**Definition 4.4** Informally, a deniable authentication protocol is said to achieve the property of confidentiality, if there is no polynomial time algorithm that can distinguish the transcripts of two distinct messages.

**Theorem 4.5** Our proposed protocol achieves the property of confidentiality provided that the HDDH problem is hard in \( G \).

**Proof** Let us go back to observe our proposed protocol. In Step 3, the sender \( S \) can authenticate the receiver \( R \) and the transcript \( V = g^v \) as well. If we regard \( V = g^v \) as the temporal public key of \( R \), then \( (Y_s = g^{x_s}, h_2 = H(g^{x_s} \oplus M)) \) is actually a hashed ElGamal ciphertext [15]. Using the almost same proof technique in [15], we can easily prove that the hashed ElGamal encryption here is also semantic security under the HDDH assumption in \( G \). As a result, our proposed protocol can achieves the confidentiality. In addition, since \( h_3 \) behaves a secure MAC, our proposed protocol also provides the message authentication. \( \square \)

**Theorem 4.6** Our proposed protocol also achieves the property of deniability.

**Proof** To prove that our proposed protocol has the property of deniability, we should prove that all transcripts transmitted between the sender \( S \) and the receiver \( R \) could be simulated by the receiver \( R \) himself.
Transcript Simulation: To simulate the transcripts on message $M$, the receiver $R$ chooses two random numbers $u, v \in \mathbb{Z}_q^*$ and computes as follows:

$$
\begin{align*}
U &= g^u, V = g^v \\
U' &= U^u = U^{x_r} = g^{ux_r} \\
V' &= V^v = Y_s^v = g^{vx_s} \\
\end{align*}
$$

$$
\begin{align*}
h_1 &= H(ID_r, U, U'', V) \\
h_2 &= H(V'') \oplus M \\
h_3 &= H(ID_s, V, V'', M) \\
\end{align*}
$$

Clearly, the transcripts $(ID_s, U, h_2, h_3)$ in simulation are indistinguishable from those of the sender $S$. As a result, the receiver $R$ is not able to prove to a third party that the transcripts were produced by the sender $S$.

According to the receiver’s indistinguishable transcript simulation above, our proposed protocol also achieves the property of deniability. □

Based upon the analysis above, we can conclude that our proposed protocol can achieve the deniable authentication, mutual authentication and confidentiality. We compare our proposed protocol with other interactive deniable authentication protocols, and the result is shown in Table 1.

Table 1. Security properties comparisons

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Deniable authentication</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Mutual authentication</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>✓</td>
</tr>
<tr>
<td>Confidentiality</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>✓</td>
</tr>
</tbody>
</table>

5. Performance Analysis

To measure the performance of a protocol, the computational costs and the communication overheads are two useful criteria. In this section, we compare our proposed protocol with another Diffie-Hellman algorithm based deniable authentication protocol [9]. To achieve the same security level, when the modular $n$ in their protocol is 1024 bits, the group element of $G$ in our proposed protocol is about 171 bits (when employing any of the families of curves described in [4]).

The following notations are used for analyzing the computational costs: $\text{Exp}_G$ denotes the exponent operation in $G$, $\text{Exp}_n$ denotes the modular exponent operation in $\mathbb{Z}_n^*$, $\text{Hash}$ denotes the hash function operation, $\text{Enc}$ denotes the public key encryption operation and $\text{Dec}$ denotes the public key decryption operation. For the convenient comparison of communication overheads, we assume that the length of identity $ID_s$, message $M$ and the hash value $H(.)$ are both 160 bits, and the ciphertext in [9] is not expanded. Then, we use Table 2 to summarize the computational costs and the communication overheads of these two protocols. As we know, $\text{Exp}_G$ is faster than $\text{Exp}_n$, $\text{Enc}$, $\text{Dec}$, when achieving the same security level. Therefore, from Table 2, we can conclude that our proposed deniable authentication protocol is an efficient one.

Table 2. Comparisons of two deniable authentication protocols

<table>
<thead>
<tr>
<th></th>
<th>Protocol [9]</th>
<th>Our proposed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Computation costs</td>
<td>$4 \text{Exp}_n + 2 \text{Hash} + \text{Enc} + \text{Dec}$</td>
<td>$6 \text{Exp}_G + 6 \text{Hash}$</td>
</tr>
<tr>
<td>Communication overheads</td>
<td>2368 bits</td>
<td>989 bits</td>
</tr>
</tbody>
</table>
6. Conclusions

In this paper, based on the Diffie-Hellman algorithm, we first presented a provably secure simple user authentication scheme. Then, based upon the simple user authentication scheme, we proposed our simple deniable authentication protocol. Compared with other deniable authentication protocols, our proposed protocol can simultaneously achieve not only the deniable authentication, but also the mutual authentication between the sender and the intended receiver and the confidentiality.

Acknowledgment

The author would like to thank the anonymous referees for their valuable comments and suggestions to improve the presentation of this paper.

References