Authenticated key agreement without using one-way hash functions

L. Harn and H.-Y. Lin

The MQV key agreement protocol has been adopted by the IEEE P1363 Committee to become a standard. The MQV protocol uses a digital signature to sign the Diffie-Hellman public keys without using any one-way function. Here, the MQV protocol is generalised in three respects. First, signature variants for Diffie-Hellman public key systems have been previously employed in the new protocol. Second, two communication entities are allowed to establish multiple secret keys in a single round of message exchange. Thirdly, the key computations are simplified.

However, there exists a major difference of security assumptions between digital signature schemes and conventional one-way hash functions. The security assumption of most signature schemes are based on some well-known computational problems, such as the discrete logarithm problem [4] and the factorising problem. The complexities of these problems have been well studied and the difficulties of solving them are recognised. In contrast, the security of one-way hash function is based on the complexity of analysing a simple iterated function. A one-way hash function may seem very difficult to analyse at the beginning, but it may turn out to be vulnerable to some special attacks later, e.g. recent advancement in cryptanalytic research has found that MD5 is at the edge of risk.

In 1998, we published a key agreement protocol [8] that generalised the MQV protocol in three respects. First, a signature for the Diffie-Hellman public key without using one-way hash functions.

<table>
<thead>
<tr>
<th>Signature equation</th>
<th>Signature verification</th>
</tr>
</thead>
<tbody>
<tr>
<td>( r = k + s \mod p - 1 )</td>
<td>( y = r^2 \mod p )</td>
</tr>
<tr>
<td>( s = k + r \mod p - 1 )</td>
<td>( y = r^x \mod p )</td>
</tr>
<tr>
<td>( x = k + s \mod p - 1 )</td>
<td>( y = r \mod p )</td>
</tr>
<tr>
<td>( x = s \mod p - 1 )</td>
<td>( y = r \mod p )</td>
</tr>
</tbody>
</table>

Key agreement protocols: In the following discussion we assume that A and B want to establish a secret key(s) using the key agreement protocol. The short-term secret key and long-term public key for A are \( k_A \) and \( y_A \), and the long-term secret key and long-term public key for B are \( y_B \) and \( k_B \). Similarly, B has \( k_B \), \( r_B \), \( s_B \), and \( y_B \). The following key agreement protocol enables A and B to establish an authenticated secret key \( K \).

(1) A generates a random short-term secret key \( r_A \) and its corresponding public key \( r_A \), and computes the signature \( s_B \) based on any variant as listed in Table 1. A sends \( \{r_A, s_B, \text{cert}(y_B)\} \) to B, where \( \text{cert}(y_B) \) is the public-key certificate of \( y_B \) signed by a trusted party.

(2) Similarly, B generates \( k_B \), \( r_B \), \( s_B \), and \( y_B \) and sends \( \{r_B, s_B, \text{cert}(y_B)\} \) to A.

(3) A verifies \( r_B \) based on the signature \( s_B \) and B's public key \( y_B \). Then A computes the shared secret key \( K = k_B^r \mod p \).

(4) Similarly, B verifies \( r_A \) based on the signature \( s_A \) and A's public key \( y_A \). Then B computes the shared secret key \( K = k_A^r \mod p \).

Possible attacks: One drawback of the above protocol is that it does not offer perfect forward secrecy [12], i.e. if an adversary learns one shared secret key they can deduce all shared secret keys between A and B. We illustrate this attack in the following discussion. Assume that the protocol uses \( s = k + r \mod p - 1 \) to sign each Diffie-Hellman public key. We then have the following two equations:

\[
\begin{align*}
\lambda_A &= r_A k_B + s_B \mod p - 1 \quad \text{and} \quad \lambda_B = r_B k_A + s_A \mod p - 1
\end{align*}
\]

By multiplying these two equations, we obtain

\[
\begin{align*}
\lambda_A \lambda_B &= r_A k_B + s_B \mod p - 1 \quad \text{and} \quad \lambda_B = r_B k_A + s_A \mod p - 1
\end{align*}
\]

In other words,

\[
K_{AB} = \left( r_A^{\lambda_B} \right) \left( r_B^{\lambda_A} \right) \mod p
\]

where \( K_{AB} = \alpha^{\lambda_B} \mod p \) is the long-term shared secret key. Assume that the adversary knows one short-term shared secret key \( K \). The adversary can then solve the long-term shared secret key \( K_{AB} \) from the above equation.

Improved protocol: In the two-pass MQV key agreement protocol, instead of using \( K = \alpha^{\lambda_B} \mod p \) as the shared secret key, it uses \( K = \alpha^{\lambda_B} \mod p \) as the shared secret key. The MQV protocol can provide perfect forward secrecy.
exchange. For simplicity, we assume that A and B want to share four secrets.

(i) A generates two random short-term secret keys, \( k_A \) and \( k_B \), and two corresponding public keys, \( r_A \) and \( r_B \), where \( r_A < r_B \). Then, A computes the signature \( s_A \) for \( \{ r_A, r_B \} \) based on any signature variant as listed in Table 1. For example, A obtains \( s_A \) by solving the following equation

\[
x_A = r_A k_A + r_B k_B + s_A \mod p - 1
\]

A sends \( \{ r_A, r_B, s_A, \text{cert}(y_A) \} \) to B, where \( \text{cert}(y_A) \) is the public-key certificate of \( y_A \) signed by a trusted party.

(ii) Similarly, B generates \( k_B \), \( r_B \), \( r_B' \), \( s_B \), and \( \{ r_B, r_B', s_B, \text{cert}(y_B) \} \) to A.

(iii) A verifies \( \{ r_B, r_B' \} \) based on the signature \( s_B \) and B’s public key \( y_B \) by checking

\[
y_B = r_B^{y_B^{x_A r_B' r_B}} \mod p
\]

Then A computes the shared secret keys as

\[
K_1 = r_B^s_A \mod p
\]

\[
K_2 = r_B^s_A r_B' \mod p
\]

\[
K_3 = r_B^s_A r_B^{s_B} \mod p
\]

\[
K_4 = r_B^s_A r_B^{s_B} r_B' \mod p
\]

(iv) Similarly, B computes \( \{ r_A, r_A' \} \mod p \) first and verifies \( \{ r_A, r_A' \} \) based on the signature \( s_A \).

Then, B computes the shared secret keys as

\[
K_1 = r_A^s_B \mod p
\]

\[
K_2 = r_A^s_B r_B \mod p
\]

\[
K_3 = r_A^s_B r_B s_B \mod p
\]

\[
K_4 = r_A^s_B r_B s_B r_B' \mod p
\]

Discussion: We point out here that we have modified the original protocol in signature signing and verification equations. Two recent attacks [10, 11] on the original protocol cannot work successfully in this modified protocol. This modified protocol does not increase any computational load and the key agreement protocol does not involve any additional one-way hash function.

The signatures, \( x_A \) and \( x_B \), satisfy the following equations as

\[
x_A = r_A k_A + r_B k_B + s_A \mod p - 1
\]

\[
x_B = r_B k_B + r_B k_B s_B + s_B \mod p - 1
\]

By multiplying these two equations together, we obtain

\[
x_A x_B = \text{cert}(y_A) \cdot \text{cert}(y_B)
\]

In other words, we have

\[
K_{AB} = K_A x_B r_A k_A r_A k_B r_A k_B r_A k_B + r_A k_B + r_A k_B s_A + s_A r_B k_B + s_A r_B k_B s_B + s_B r_B k_B + s_B r_B k_B s_B \mod p
\]

If the adversary knows four consecutive shared secret keys, he can solve the long-term shared secret \( K_{AB} \). Thus, to achieve the perfect forward secrecy, we should limit ourselves to use only three out of the four shared secret keys. The protocol can be generalised to enable A and B to share \( n - 1 \) secrets if each user computes and sends \( n \) Diffie-Hellman public keys in each pass. Since each user only needs to generate (verify) one signature for \( n \) different Diffie-Hellman public keys to establish \( n - 1 \) shared secret keys, this new protocol is very efficient.

Conclusion: We have proposed an authenticated key agreement protocol that utilises a digital signature to authenticate Diffie-Hellman public keys. We summarise features in this new protocol as follows:

(i) Since we integrate the Diffie-Hellman public key in the signature scheme, this approach reduces overall computation.

(ii) Since the protocol does not use any one-way hash function, the security assumption relies solely on solving the discrete logarithm problem.

(iii) This protocol allows two communication parties to share multiple secret keys in two-pass interaction.

(iv) The computation for shared secret keys is simpler than the MQV protocol.

References


Cryptanalysis on improved user efficient blind signatures

C.-I. Fan and C.-L. Lei

Shao proposed a blind signature scheme based on the Fan-Lei scheme. It is shown here that Shao’s scheme is not secure. Also, Shao claimed that the Fan-Lei scheme is not really blind, however this claim is demonstrated as not being true.

Introduction: In 1996, Fan and Lei proposed a blind signature scheme based on quadratic residues (QRs) [1], and they also presented an enhanced version of the scheme to reduce the computation for requesters or users [2]. In [3], Shao proposed a blind signature scheme based on quadratic residues (QRs) [1]. However, we find that Shao’s scheme cannot withstand Pollard-Schnorr attacks [4]. Besides, Shao claimed that the Fan-Lei blind signature scheme [2] is not really blind. In this Letter, we also show that Shao’s claim is not true.

Attacks on Shao’s blind signature scheme: Shao proposed a blind signature scheme based on the Fan-Lei scheme in [3]. We show that Pollard-Schnorr attacks [4] are valid on Shao’s scheme as follows. In the scheme of [3], the tuple \((e, s)\) is a signature of \(m\) and they can be verified by checking if

\[
H(m)^2(c^2+1) \equiv 1 \mod n
\]

An attacker can choose a message \(m\) and then derive \((w, y)\) in polynomial time such that

ELECTRONICS LETTERS 10th May 2001 Vol. 37 No. 10

© IEE 2001

DOI: 10.1049/el:20010441

L. Harn (Department of Computer Networking, University of Missouri, Kansas City, MO 64110, USA)

H.-Y. Lin (Computer Science Department, California State University, San Marcos, CA 92096-0001, USA)