Integrating Diffie–Hellman Key Exchange into the Digital Signature Algorithm (DSA)

Lein Harn, Manish Mehta, *Student Member, IEEE*, and Wen-Jung Hsin

***Abstract*—The National Institute of Standards and Technology (NIST) has published a series of security standards under Federal Information Processing Standard (FIPS). Standard for key agreement is still missing in the current standards. Arazi proposed integrating Diffie–Hellman (DH) key exchange into the Digital Signature Algorithm (DSA). However, the protocol was attacked by Nyberg *et al.* We propose three different protocols that securely integrate DH key exchange into DSA for authenticated key distribution.**

***Index Terms*—Diffie–Hellman (DH) key, digital signature, Digital Signature Algorithm (DSA).**

I. Introduction

A KEY agreement protocol establishes secret communication key(s) among all parties involved. In 1976, Diffie and Hellman (DH) [1] proposed the well-known public-key distribution scheme, based on the discrete logarithm problem, to enable two parties to establish a common secret key based on their exchanged public keys. However, their scheme did not provide authentication mechanism for the exchanged public keys. In past years, NIST has published a series of security standards under Federal Information Processing Standard (FIPS) [2]. FIPS 186-2 Digital Signature Standard (DSS) [3] introduces Digital Signature Algorithm (DSA). So far, there is no FIPS standard for key agreement between two parties. In 1993, Arazi [4] suggested replacing the *message* in the DSA algorithm with DH *exchange key* to achieve key authentication. Later, Nyberg and Rueppel [5] pointed out a weakness in Arazi's scheme: if one secret session key is compromised, then the others will be disclosed as well. This attack is known as *known key attack*. In another kind of attack known as *unknown key-share attack* [6], an opponent coerces honest parties into establishing a secret key where at least one of honest parties does not know that the secret key is shared with the other. Another common attack discussed in the literature is the *key replay attack*. In this attack, the attacker records the information of the on-going session and then replays it to impersonate a party in future. In this letter, we extend Arazi's approach to securely integrate the DH key exchange into the DSA. We propose three alternative protocols for a variety of applications and show how they can prevent the attacks discussed above. We call this approach as secure DH + DSA.

Manuscript received August 20, 2003; The associate editor coordinating the review of this letter and approving it for publication was Dr. C.-K. Wu.

The authors are with the School of Computing and Engineering, University of Missouri, Kansas City, MO 64110 USA (e-mail: mmmef7@umkc.edu).

Digital Object Identifier 10.1109/LCOMM.2004.825705

II. Review of DSA

The parameters of DSA are two primes $p(2^{L-1} < p < 2^L)$ where $512 \leq L \leq 1024$ and L a multiple of 64, and $q(2^{159} < q < 2^{160})$, and an integer g, where q is a divisor of $p-1$, and $g = h^{(p-1)/q} \bmod p > 1$ for some random integer h with $1 < h < p-1$. The private key of the user is a random value $x(0 < x < q)$. y is the corresponding public key, where $y = g^x \bmod p$. H is a secure hash function on message m, yielding a 160-bit $H(m)$. $\{p, q, g, y\}$ are public values and $\{x\}$ is user's private key. k is a randomly chosen integer such that $0 < k < q$. The signature of a message m is the pair of numbers r and s computed as $r = ((g^k \bmod p) \bmod q)$ and $s = (k^{-1}(H(m) + xr)) \bmod q$. Here, k^{-1} is the multiplicative inverse of $(k \bmod q)$. i.e. $(k^{-1}k) \bmod q = 1$. On the receiver end, let m', r' and s' be the received versions of m, r, and s, respectively. To verify the signature, the verifier first checks to see if $0 < r' < q$ and $0 < s' < q$; if either condition is violated, the signature is rejected. Otherwise, the verifier computes $a = (s')^{-1} \bmod q$, $u1 = (H(m')a) \bmod q$, $u2 = (r'a) \bmod q$ and $b = ((g^{u1}y^{u2}) \bmod p) \bmod q$. If $b = r'$, the signature is verified.

III. Secure DH + DSA Key Exchange Protocols

In the following discussions, || is the concatenation operation.

A. One-Round Protocol

Fig. 1 shows the one-round protocol. This protocol supports noninteractive applications, such as secure e-mail transmission. Let us say user A wants to securely send an email to user B. Key K_{AB} is the shared secret key.

B. Two-Round Protocol

Fig. 2 shows the two-round protocol. This protocol supports interactive applications. Let us say user A wants to communicate with user B interactively. Here, K_{AB} and K_{BA} are the shared secret keys for directions A to B and B to A, respectively.

C. Three-Round Protocol

This protocol supports interactive applications. Fig. 3 shows the three-round protocol. Here, user A wants to communicate with user B interactively. This protocol differs from the two-round protocol in that it includes *key confirmation* in the third round. Here, K_{AB} and K_{BA} are the shared secret keys for directions A to B and B to A respectively.

IV. Security Analysis

As the three-round protocol encompasses the features of the other two protocols, we consider only the three-round protocol

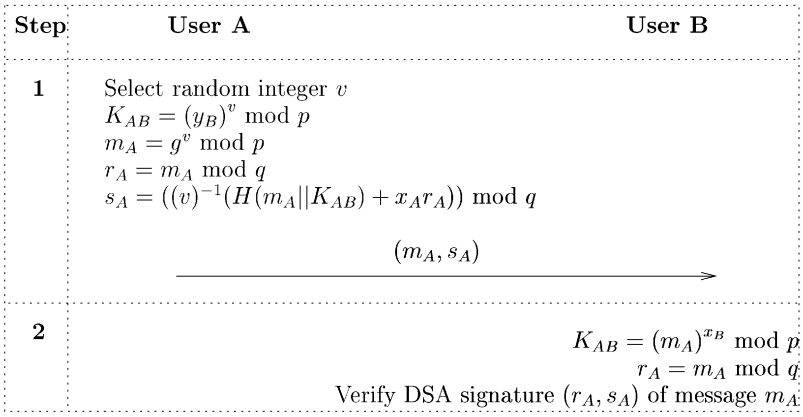

Fig. 1. One-round protocol.

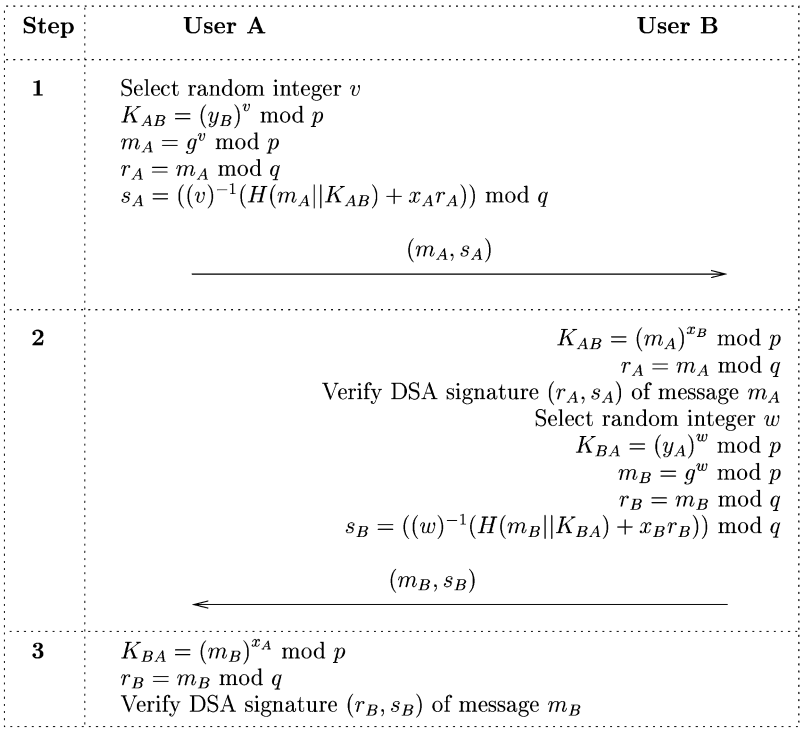

Fig. 2. Two-round protocol.

for security analysis. Specifically, we show that the three-round protocol can prevent known key attack, key replay attack and unknown key-share attack. Note that one-round protocol and two-round protocol do not have key confirmation and hence can not prevent key replay attacks.

The following discussion shows how known key attack is prevented. By rewriting the signature equations in steps 2 and 3 of the three-round protocol, we obtain

$$x_B r_B = w s_B - H\left(m_B||K_{BA}||K_{AB}\right) \bmod q \quad (1)$$

$$x_A r_A = v s_A - H\left(m_A||K_{AB}||K_{BA}\right) \bmod q. \quad (2)$$

Cross multiplying (1) and (2), and raising g to the power of both sides, we obtain

$$g^{w x_A r_A s_B + x_B r_B H(m_A||K_{AB}||K_{BA})} = g^{v x_B r_B s_A + x_A r_A H(m_B||K_{BA}||K_{AB})} \bmod p.$$

That is,

$$(K_{BA})^{r_A s_B} (y_B)^{r_B H(m_A||K_{AB}||K_{BA})} = (K_{AB})^{r_B s_A} (y_A)^{r_A H(m_B||K_{BA}||K_{AB})} \bmod p. \quad (3)$$

From (3), given one of the shared secret keys (e.g., K_{AB}), as [7] pointed out, finding the other shared secret key (K_{BA}) is no easier than a discrete logarithm problem.

Also, by rewriting (2), multiplying both sides by x_B, and then raising g to the power of both sides, we obtain

$${K_{AB}}^{s_A} = {y_B}^{H(m_A||K_{AB}||K_{BA})} (g^{x_A x_B})^{r_A} \bmod p. \quad (4)$$

Suppose that an attacker can obtain both shared secret keys (e.g. K_{AB} and K_{BA}) for a session. The attacker can then compute long term value $g^{x_A x_B}$. However, with the knowledge of the long term value $g^{x_A x_B}$ and one of the shared secret keys (e.g. K_{AB} or K_{BA}) for a different session, finding the other

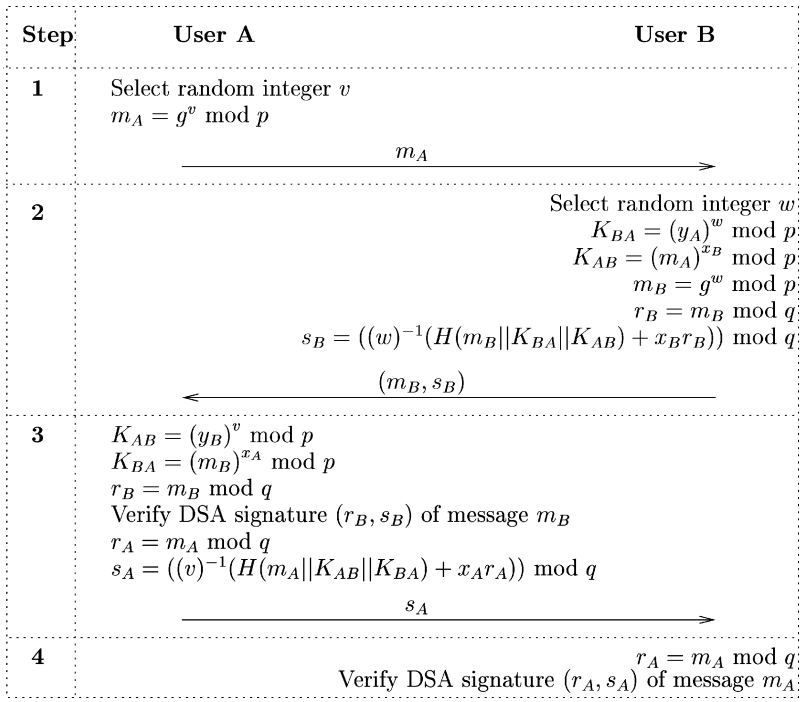

Fig. 3. Three-round protocol.

shared secret key for that session is no easier than solving discrete logarithm problem as both the shared secret keys appear in the exponent in (4). Thus, by including the shared secret keys in the signature equation we prevent *known key* attack in our protocol.

Key confirmation can prevent unknown key-share attack [6]. The following discussion shows how key confirmation is achieved and key replay attack is prevented: User B confirms with user A of receiving the shared secret key K_{AB} by signing this key along with m_B (step 2). Since this shared secret key K_{AB}, a function of a random integer that was originally selected by user A, acts as a nonce, after receiving the signature (m_B, s_B) from user B, user A is convinced that the message m_B is not a replayed one and knows that it is indeed from user B. Thus, K_{BA} (function of m_B) is not a replayed key. Similarly, user A confirms with user B of receiving the shared secret key K_{BA} by signing this key (acting as a nonce) along with m_A (step 3). After verifying the signature of user A, user B is also convinced that the shared secret key K_{AB} (function of m_A) is not a replayed key and knows that it is indeed from user A.

V. Summary of Contribution

Our protocols differ from that of Arazi's in that (1) Our protocols provide multiple secret keys, one for each direction. This arrangement conforms with most standard protocols, such as SSL and IPSec, which use different secret keys for different directions. (2) The shared secret key is included in the signature equation along with the message in our scheme. This arrangement prevents the *known key* attack and the *key replay* attack. (3) Our three-round protocol achieves key confirmation, which prevents the *unknown key-share* attack.

VI. Conclusion

We have proposed three protocols that securely integrate Diffie–Hellman key exchange into the DSA. One-round protocol can be used in secure e-mail transmission. Two-round protocol provides authenticated key exchange for interactive communications. Three-round protocol provides authenticated, key confirmation and nonplayback key exchange for interactive communications.

References

[1] W. Diffie and M. E. Hellman, "New directions in cryptography," *IEEE Trans. Inform. Theory*, vol. IT-22, pp. 644–654, Nov. 1976.

[2] Federal Information Processing Standards Publication, National Institute of Standards and Technology. [Online]. Available: http://www.itl.nist.gov/fipspubs/

[3] National Institute of Standards and Technology, Digital Signature Standard (DSS), "Federal Information Processing Standards Publication,", FIPS PUB 186-2, Reaffirmed, January 27, 2000.

[4] A. Arazi, "Integrating a key cryptosystem into the digital signature standard," *Electron. Lett.*, vol. 29, no. 11, pp. 966–967, 1993.

[5] K. Nyberg and R. A. Rueppel, "Weaknesses in some recent key agreement protocols," *Electron. Lett.*, vol. 30, no. 1, pp. 26–27, 1994.

[6] B. Kaliski, "An unknown key-share attack on the MQV key agreement protocol," *ACM Trans. Inform. Syst. Security*, vol. 4, no. 3, pp. 275–288, Aug. 2001.

[7] T. ElGamal, "A public key cryptosystem and a signature scheme based on discrete logarithms," in *Proc. Crypto '84*, pp. 10–18.