Conference or Workshop Item:
Vasilakis, Vasileios orcid.org/0000-0003-4902-8226, Alohali, Bashar, Moscholios, Ioannis et al. (1 more author) (Accepted: 2017) Security Analysis of Integrated Diffie-Hellman Digital Signature Algorithm Protocols. In: IEICE Information and Communication Technology Forum, 04-06 Jul 2017. (In Press)

Security Analysis of Integrated Diffie-Hellman Digital Signature Algorithm Protocols

Vassilios G. Vassilakis*, Bashar A. Alohali†, Ioannis D. Moscholios‡, Michael D. Logothetis§
* School of Computer Science, University of York, York, United Kingdom
† School of Computing & Mathematical Sciences, Liverpool John Moors University, Liverpool, United Kingdom
‡ Dept. of Informatics & Telecommunications, University of Peloponnese, Tripolis, Greece
§ Dept. of Electrical & Computer Engineering, University of Patras, Patras, Greece

***Abstract*—Diffie-Hellman (DH) key exchange is a well known method for secure exchange of cryptographic keys and has been widely used in popular Internet protocols, such as IPsec, TLS, and SSH. To enable authenticated key establishment, the DH protocol has been integrated with the digital signature algorithm (DSA). In this paper, we analyze three variants of the integrated DH-DSA protocol. We study the protocol variants with respect to known types of attacks and security features. In particular, the focus is on the properties of forward secrecy, known-key security, and replay attack resilience.**

Keywords—Diffie-Hellman protocol; digital certificate algorithm; key agreement protocol; network security.

I. INTRODUCTION

Today, the public-key (or, asymmetric) cryptosystems are mainly used for the establishment and distribution of secret keys over insecure channels [1]. These secret keys are subsequently used by symmetric cryptographic protocols to encrypt the actual data and to enable secure communication of two or more parties. Due to the inherent slowness of public-key cryptosystems in comparison with the symmetric cryptosystems, in typical scenarios, the public-key cryptosystems are not the preferred method for the distribution of the actual data. On the other hand, public-key cryptosystems are an attractive solution when the secret key distribution is required. This is due to the fact that they do not rely on the existence of a hidden or covert channel in order to enable secure communication.

The first and widely used public-key cryptosystem today is the Diffie-Hellman (DH) key agreement protocol. Its inventors, Whitfield Diffie and Martin Hellman received the 2015 ACM Turing Award for their seminal paper that introduced the concepts of public-key cryptography and digital signatures [2]. The DH protocol enables two communicating parties that have no prior knowledge of each other to jointly establish a secret key by exchanging a number of messages over an insecure channel. Thanks for this feature, the DH method has been adopted by many popular Internet protocols, such as the Internet protocol security (IPsec), the transport layer security (TLS), and the secure shell (SSH), for a secure exchange of cryptographic keys.

The joint key generation in the DH protocol relies on the difficulty of computing (by an attacker) discrete logarithms. One of the limitations of the DH protocol in its classical form is that it provides no authentication. Hence, to verify the identities of the communicating parties, the DH protocol must be integrated with a suitable authentication protocol. In this work, to ensure user authentication, we consider the integration of the DH protocol with the digital signature algorithm (DSA). The latter has been proposed in 1991 by the National Institute of Standards and Technology (NIST) and is part of their digital signature standard (DSS). The DSA was designed to sign messages that are sent in clear. It was not intended to provide data encryption. An integrated DH-DSA protocol was first introduced in [3] and aims at providing both secure and authenticated communication. To achieve authentication, this approach suggests replacing the messages exchanged in the DSA with the DH keys. Since then a number of weaknesses have been identified [4], [5] and several modifications of this protocol have been proposed [6]–[9].

In this paper, we analyze three variants of the integrated DH-DSA protocol from the security perspective. In particular, we study the protocols with respect to three important security properties, namely *forward secrecy* [5], *known-key security* [10], and *replay attack resilience* [11].

II. SECURITY PROPERTIES

In this section we briefly introduce and define three important properties of secure communication protocols.

Forward secrecy (or, perfect forward secrecy) is a feature of (specific) key agreement protocols in which compromising the long-term cryptographic keys does not compromise the past session keys. That is, the forward secrecy feature ensures that a user's session key will not be compromised if, for example, the private key of the server is compromised. It also protects past sessions against future compromises of secret keys or passwords.

Known-key security is a feature of (specific) key agreement protocols in which compromising one session key does not compromise other session keys. That is, assume that two communicating parties have established two session keys - one key in each direction. Even if the adversary obtains the session key for one direction, it will still be computationally hard to derive the session key for the opposite direction.

Replay attack resilience is a feature of (specific) secure communication protocols to resist against the replay attacks in which a valid message is captured and later maliciously repeated. To protect against the replay attacks, there must be some mechanism for the communicating parties to verify that the received message is fresh.

Step	Alice (x_A, y_A)		Bob (x_B, y_B)
1	Generate random $v \in Z_q$ $m_A = g^v \bmod p$ $r_A = m_A \bmod q$ $s_A = v^{-1}[H(m_A) + x_A r_A] \bmod q$	(m_A, s_A) $\longrightarrow$	
2		(m_B, s_B) $\longleftarrow$	Generate random $w \in Z_q$ $m_B = g^w \bmod p$ $r_B = m_B \bmod q$ $s_B = w^{-1}[H(m_B) + x_B r_B] \bmod q$
3	Verify signature (r_B, m_B) of message m_B Compute $K = m_B^v \bmod p$		
4			Verify signature (r_A, m_A) of message m_A Compute $K = m_A^w \bmod p$

Fig. 1. Arazi's Protocol.

III. Arazi's Protocol

One of the first attempts to integrate the DH protocol with the DSA was made by Arazi [3]. The aim is to enable two communicating parties, Alice and Bob, to establish a shared session key, K. In this section, we initially present Arazi's protocol. We then discuss its weaknesses. In particular, it is demonstrated that the protocol provides no known-key security and it is vulnerable to replay attacks. The basic protocol steps are also shown in Fig. 1.

A. The Protocol

Initialization:

At the beginning, Alice and Bob jointly select the protocol parameters: L, p, q, and g. These parameters need not be kept secret (from an adversary). Afterwards, Alice and Bob generate their own public and private cryptographic keys, which are referred to as the long-term keys. The generation of the protocol parameters and the long-term keys is explained below.

(i) Alice and Bob select

- a number L that is multiple of 64 and $512 \leq L \leq 1024$
- a large prime p, such that $2^{L-1} < p < 2^L$
- a prime q that is divisor of $p-1$ and $2^{159} < q < 2^{160}$
- g an element of multiplicative order q in Z_p (that is, $g = h^{(p-1)/q} \bmod p > 1$ for some random integer h with $1 < h < p-1$), where $Z_p = \{0, 1, \ldots, p-1\}$.

(ii) Alice generates

- her private key x_A, which is a random number $0 < x_A < q$
- her public key $y_A = g^{x_A} \bmod p$.

(iii) Bob generates

- his private key x_B, which is a random number $0 < x_B < q$
- his public key $y_B = g^{x_B} \bmod p$.

Message exchange:

Having established the protocol parameters and the long-term keys, Alice and Bob exchange appropriately crafted messages, m_A and m_B, that are accompanied by their respective signatures, s_A and s_B. Below we describe the required sequence of events and the generation of the signed messages.

(iv) Alice

- generates a random number $v \in Z_q$
- computes $m_A = g^v \bmod p$
- computes $r_A = m_A \bmod q$
- computes the signature s_A for message m_A as: $s_A = v^{-1}[H(m_A) + x_A r_A] \bmod q$, where v^{-1} is the multiplicative inverse of $v \bmod q$ (i.e., $v^{-1}v \bmod q = 1$) and H is a secure hash function on message m that produces a 160-bit hash value, $H(m)$
- sends (m_A, s_A) to Bob.

(v) Bob

- generates a random number $w \in Z_q$
- computes $m_B = g^w \bmod p$

- computes $r_B = m_B \bmod q$
- computes $s_B = w^{-1}[H(m_B) + x_B r_B] \bmod q$
- sends (m_B, s_B) to Alice.

Message verification and key derivation:

Once the aforementioned messages are received, both parties are able to independently derive a shared session key K. Note that an adversary that is observing the exchanged messages is not able to determine the key K due to the assumption that the discrete logarithm problem is hard to solve (Assumption 1) [12].

(vi) Alice

- receives (m_B, s_B) from Bob
- computes $r_B = m_B \bmod q$
- verifies the DSS signature (r_B, s_B) of message m_B
- computes the (secret) shared session key $K = m_B^v \bmod p$.

(vii) Bob

- receives (m_A, s_A) from Alice
- computes $r_A = m_A \bmod q$
- verifies the DSS signature (r_A, s_A) of message m_A
- computes the (secret) shared session key $K = m_A^w \bmod p$.

Note that the key, K, computed by Alice and Bob is the same. That is, the key computed by Alice is $K = m_B^v \bmod p = (g^w \bmod p)^v \bmod p = g^{wv} \bmod p$. Similarly, the key computed by Bob is $K = m_A^w \bmod p = (g^v \bmod p)^w \bmod p = g^{vw} \bmod p$. Finally, due to Assumption 1, an adversary by observing $(g^v \bmod p)$ and $(g^w \bmod p)$, is not able to determine neither v and w nor $(g^{vw} \bmod p)$.

B. Security Analysis

In this subsection we present a security analysis of Arazi's protocol, focusing on the known-key security property. Below it is shown that if one of the two private keys (x_A or x_B) will be compromised, then the adversary will be able to compute all the previous shared keys, K. Similarly to [4], this can be demonstrated as follows.

Recall that the signatures of the exchanged messages are:

$$s_A = v^{-1}[H(m_A) + x_A r_A] \bmod q \tag{1}$$

$$s_B = w^{-1}[H(m_B) + x_B r_B] \bmod q \tag{2}$$

From (1) and (2) we get:

$$v = s_A^{-1}[H(m_A) + x_A r_A] \bmod q \tag{3}$$

$$w = s_B^{-1}[H(m_B) + x_B r_B] \bmod q \tag{4}$$

Multiplying (3) and (4) we get:

$$vw = s_A^{-1} s_B^{-1}[H(m_A) + x_A r_A][H(m_B) + x_B r_B] \bmod q \tag{5}$$

Hence, the shared key, K, can be expressed as follows:

$$\begin{aligned} K &= g^{vw} \bmod p \\ &= g^{s_A^{-1} s_B^{-1}[H(m_A)+x_A r_A][H(m_B)+x_B r_B]} \bmod p \end{aligned} \tag{6}$$

By performing some manipulations in (6) we get:

$$K^{s_A s_B} = g^{[H(m_A)+x_A r_A][H(m_B)+x_B r_B]} \bmod p \tag{7}$$

$$\begin{aligned} K^{s_A s_B} =& g^{H(m_A)H(m_B)} g^{x_A r_A H(m_B)} \\ & g^{H(m_A) x_B r_B} g^{x_A r_A x_B r_B} \bmod p \end{aligned} \tag{8}$$

Recall that the public keys of Alice and Bob are:

$$y_A = g^{x_A} \bmod p \tag{9}$$

$$y_B = g^{x_B} \bmod p \tag{10}$$

Hence, due to (9) and (10), from (8) we get:

$$\begin{aligned} K^{s_A s_B} =& g^{H(m_A)H(m_B)} y_A^{r_A H(m_B)} \\ & y_B^{H(m_A) r_B} (g^{x_A x_B})^{r_A r_B} \bmod p \end{aligned} \tag{11}$$

In (11) we observe that the shared key, K, can be expressed in terms of publicly known quantities and the quantity $g^{x_A x_B} \bmod p$. This means that if the adversary obtains one session key, K, he/she can then compute the quantity $g^{x_A x_B} \bmod p$. Hence, the adversary will be able to compute all the previous sessions keys, K, and to decrypt all the encrypted messages, m_A and m_B. This means that the protocol provides no known-key security.

Finally, Arazi's protocol is vulnerable to replay attacks. That is, an adversary may intercept, for example, the message m_A with its signature s_A and replay them later to Bob. On such occasions, there is no way for Bob to determine whether the received message is fresh or not. An approach to mitigate replay attacks in this protocol using timestamps is presented in [9].

IV. Harn's Protocol

In this section we present and analyze a variant of the DH-DSA three-round protocol proposed by Harn *et al.* [6].

A. The Protocol

Harn's protocol shares some similarities with Arazi's protocol, with the differences that are discussed below. The basic steps of the protocol are also shown in Fig. 2.

Similarly to the popular Internet security protocols, such as TLS and IPsec, Harn's protocol uses two different shared session keys - one key for each direction. In particular, the messages that are sent from Alice to Bob are encrypted using the session key K_{AB}, whereas the messages that are sent from Bob to Alice are encrypted using the session key K_{BA}. The aforementioned keys are generated via (12) and (13) by Alice and Bob, respectively:

$$\begin{aligned} K_{AB} &= y_B^v \bmod p \\ K_{BA} &= m_B^{x_A} \bmod p \end{aligned} \tag{12}$$

$$\begin{aligned} K_{AB} &= m_A^{x_B} \bmod p \\ K_{BA} &= y_A^w \bmod p \end{aligned} \tag{13}$$

Step	**Alice** (x_A, y_A)	**Bob** (x_B, y_B)
1	Generate random $v \in Z_q$ $m_A = g^v \bmod p$ $\xrightarrow{(m_A)}$	
2		Generate random $w \in Z_q$ $K_{AB} = m_A^{x_B} \bmod p$ $K_{BA} = y_A^w \bmod p$ $m_B = g^w \bmod p$ $r_B = m_B \bmod q$ $s_B = w^{-1}[H(m_B\|\|K_{BA}\|\|K_{AB}) + x_B r_B] \bmod q$ $\xleftarrow{(m_B, s_B)}$
3	$K_{AB} = y_B^v \bmod p$ $K_{BA} = m_B^{x_A} \bmod p$ $r_B = m_B \bmod q$ Verify signature (r_B, m_B) of message m_B $r_A = m_A \bmod q$ $s_A = v^{-1}[H(m_A\|\|K_{AB}\|\|K_{BA}) + x_A r_A] \bmod q$ $\xrightarrow{(s_A)}$	
4		$r_A = m_A \bmod q$ Verify signature (r_A, m_A) of message m_A

Fig. 2. Harn's Protocol.

Recall that (y_A, x_A) and (y_B, x_B) are the public-private key pairs of Alice and Bob, respectively. Also, m_A and m_B are the messages that are exchanged between Alice and Bob and are defined as in Arazi's protocol:

$$\begin{aligned} m_A &= g^v \bmod p \\ m_B &= g^w \bmod p \end{aligned} \tag{14}$$

By using (9), (10), and (14), equations (12) and (13) are transformed to:

$$\begin{aligned} K_{AB} &= g^{x_B v} \bmod p \\ K_{BA} &= g^{x_A w} \bmod p \end{aligned} \tag{15}$$

Due to use of two session keys, the signatures s_A and s_B are defined (differently to Arazi's protocol) as follows:

$$s_A = v^{-1}[H(m_A||K_{AB}||K_{BA}) + x_A r_A] \bmod q \tag{16}$$

$$s_B = w^{-1}[H(m_B||K_{BA}||K_{AB}) + x_B r_B] \bmod q \tag{17}$$

where $||$ is the concatenation operator.

Note in (16) and (17) that the message signatures, s_A and s_B, depend on the session keys, which in turn depend on the messages m_A and m_B. Therefore, contrary to Arazi's protocol, the message m_A is initially sent without its signature (Step 1). This is because at the beginning Alice does not know m_B which is required to compute K_{BA}. Hence, sending the signature s_A has to wait until Step 3.

B. Security Analysis

In this subsection we discuss the security properties of the Harn's protocol. In particular, we present the protocol analysis with respect to the properties of known-key security, forward secrecy, and attack resilience.

1) Known-key security: With regard to the know-key security property, we are interested in identifying how much information about the session keys, K_{AB} and K_{BA}, is leaked to an adversary that observes the signatures s_A and s_B, which are exchanged between Alice and Bob. To this end, below, we transform (16) and (17) in a way that reveals the relation between K_{AB} and K_{BA}.

Begin by multiplying both sides of (16) and (17) by v and w, respectively:

$$s_A v = [H(m_A||K_{AB}||K_{BA}) + x_A r_A] \bmod q \tag{18}$$

$$s_B w = [H(m_B||K_{BA}||K_{AB}) + x_B r_B] \bmod q \tag{19}$$

Solve (18) and (19) for $x_A r_A$ and $x_B r_B$, respectively:

$$x_A r_A = [s_A v - H(m_A||K_{AB}||K_{BA})] \bmod q \tag{20}$$

$$x_B r_B = [s_B w - H(m_B||K_{BA}||K_{AB})] \bmod q \tag{21}$$

Cross multiply (20) and (21):

$$x_A r_A[s_B w - H(m_B||K_{BA}||K_{AB})] = \\ x_B r_B[s_A v - H(m_A||K_{AB}||K_{BA})] \bmod q \tag{22}$$

The above equation is equivalent to:

$$x_A r_A s_B w + x_B r_B H(m_A||K_{AB}||K_{BA}) = \\ [x_B r_B s_A v + x_A r_A H(m_B||K_{BA}||K_{AB})] \bmod q \tag{23}$$

Raise g to the power of both sides of (23) modulo p:

$$g^{x_A r_A s_B w + x_B r_B H(m_A||K_{AB}||K_{BA})} = \\ g^{x_B r_B s_A v + x_A r_A H(m_B||K_{BA}||K_{AB})} \bmod p \tag{24}$$

Substitute (9), (10), and (15) into (24):

$$(K_{BA})^{r_A s_B}(y_B)^{r_B H(m_A||K_{AB}||K_{BA})} = \\ (K_{AB})^{r_B s_A}(y_A)^{r_A H(m_B||K_{BA}||K_{AB})} \bmod p \tag{25}$$

In (25) by observing the relationship between the two session keys, it can be concluded that if the adversary knows one of the shared keys, the problem of computing the other shared key is at least of the same difficulty as solving the discrete logarithm problem for which no efficient method for computing on conventional computers is known [6], [12].

2) Forward secrecy: Below, we present a security analysis of Harn's protocol with regard to the forward secrecy property [7], [13]. The forward secrecy requires that if a long-term private key (i.e., x_A or x_B) has been compromised, then the adversary still is not able to determine the previously established session keys (i.e., K_{AB} and K_{BA}).

Recall from (12) and (13), that the session keys can be expressed as follows:

$$K_{AB} = m_A^{x_B} \bmod p \\ K_{BA} = m_B^{x_A} \bmod p \tag{26}$$

where m_A and m_B are quantities exchanged between Alice and Bob over an insecure channel, and, hence, are known to the adversary. Assume that the adversary obtains x_A. Then K_{AB} can be easily computed via (26). Similarly, when x_B is obtained, then K_{BA} can be computed. Hence, this approach provides no forward secrecy.

3) Replay attack resilience: Recall that during a replay attack, the adversary captures valid messages and then re-sends them to the intended recipient. Harn's protocol is immune to replay attacks for the following reason. Consider Alice sending to Bob the signed message (m_A, s_A). The message and its signature are given by (14) and (16), respectively. Note that the derivation of s_A depends on K_{BA} which in turn depends on w as shown in (15). However, as shown in Section III, w is generated by Bob for the establishment of the particular session key. Hence, Bob can be sure that the received message m_A is fresh. In a similar way, Alice can verify the freshness of the message m_B sent by Bob.

V. Phan's Protocol

In this section we present and analyze a variant of the DH-DSA protocol proposed by Phan [7]. The basic protocol steps are also shown in Fig. 3.

A. The Protocol

Phan's protocol can be seen as a modification of Harn's protocol, with the differences discussed below. Alice and Bob generate additional random quantities, n_A and n_B, respectively. These accompany the sent messages m_A, m_B and are used in the process of the shared keys generation. In particular:

$$n_A = y_A^v \bmod p \\ n_B = y_B^w \bmod p \tag{27}$$

The shared keys generated by Alice are:

$$K_{AB} = n_B^v \bmod p = g^{x_B v w} \bmod p \\ K_{BA} = m_B^{x_A w} \bmod p = g^{x_A v w} \bmod p \tag{28}$$

The shared keys generated by Bob are:

$$K_{AB} = m_A^{x_B w} \bmod p = g^{x_B v w} \bmod p \\ K_{BA} = n_A^w \bmod p = g^{x_A v w} \bmod p \tag{29}$$

By comparing (28) with (12), we observe that Phan's protocol uses n_B instead of y_B. As will be shown in the following subsection, this provides stronger security features since y_B is publicly known but n_B is not. For the same reason, the key generation by Bob via (29) relies on n_A instead of y_A.

B. Security Analysis

With regard to the known-key security, Phan's protocol has the same properties with Harn's protocol. In particular, if the adversary knows one shared session key (e.g., K_{AB}), it is computationally hard to infer the other shared session key (i.e., K_{BA}).

Below we discuss the forward secrecy property of Phan's protocol. Recall that this property requires that if a long-term private key has been compromised, the secrecy of the previously generated session keys must be preserved. Hence, let us assume that x_A has been compromised. In (28) and (29) we observe that in order for the adversary to compute K_{BA}, he/she must know w. However, recall from Section III that w is generated by Bob and is never sent to Alice (so, it cannot be intercepted). Note, that the adversary may still intercept m_B which is equal to g^w. However, by knowing m_B and g, it computationally hard to derive w (the discrete logarithm problem).

Similarly, if x_B has been compromised, the adversary will still need to know v in order to compute K_{AB}. However, v is generated by Alice, is never sent to Bob, and, hence, cannot be intercepted. Hence, we can conclude that Phan's protocol provides forward secrecy.

VI. Conclusion and Future Work

In this paper, we have presented a security analysis of three variants of the integrated DH-DSA protocol. In particular, we have focused on the properties of forward secrecy, known-key security, and replay attack resilience. We observe that Arazi's protocol provides no forward secrecy and is vulnerable to replay attacks. This is because the generated session keys are not mutually independent. Hence, if the adversary compromises one session key, other session keys can be easily determined based on the publicly exchanged messages. Harn's protocol provides known-key security and replay-attack resilience, but

Step	**Alice** (x_A, y_A)		**Bob** (x_B, y_B)
1	Generate random $v \in Z_q$ $m_A = g^v \bmod p$ $n_A = y_A^v \bmod p$	$\xrightarrow{(m_A, n_A)}$	
2		$\xleftarrow{(m_B, n_B, s_B)}$	Generate random $w \in Z_q$ $K_{AB} = m_A^{x_B w} \bmod p$ $K_{BA} = n_A^w \bmod p$ $m_B = g^w \bmod p$ $n_B = y_B^w \bmod p$ $r_B = m_B \bmod q$ $s_B = w^{-1}[H(m_B \| \| K_{BA} \| \| K_{AB}) + x_B r_B] \bmod q$
3	$K_{AB} = n_B^v \bmod p$ $K_{BA} = m_B^{x_A v} \bmod p$ $r_B = m_B \bmod q$ Verify signature (r_B, m_B) of message m_B $r_A = m_A \bmod q$ $s_A = v^{-1}[H(m_A \| \| K_{AB} \| \| K_{BA}) + x_A r_A] \bmod q$	$\xrightarrow{(s_A)}$	
4			$r_A = m_A \bmod q$ Verify signature (r_A, m_A) of message m_A

Fig. 2. Phan's Protocol.

no forward secrecy. Finally, Phan's protocol has all three aforementioned security properties. In our future work, we intend to study the DH-DSA protocol and its variants with respect to other security properties, such as the *key freshness*, and threats, such as the *unknown-key share attacks* [14].

REFERENCES

[1] W. Stallings, Cryptography and Network Security: Principles and Practices, 7th edition, Pearson, 2016.

[2] W. Diffie and M. Hellman, "New directions in cryptography," IEEE Transactions on Information Theory, vol. 22, no. 6, 1976, pp. 644-654.

[3] B. Arazi, "Integrating a key distribution procedure into the digital signature standard," Electronics Letters, vol. 29, no. 11, 1993, pp. 966-967.

[4] K. Nyberg and R. A. Rueppel, "Weaknesses in some recent key agreement protocols," Electronics Letters, vol. 30, no. 1, 1994, pp. 26-27.

[5] D. Adrian, *et al.*, "Imperfect forward secrecy: How Diffie-Hellman fails in practice," Proc. 22nd ACM SIGSAC Conference on Computer and Communications Security (CSS), Denver, USA, Oct. 2015, pp. 5-17.

[6] L. Harn, M. Mehta, and W.-J. Hsin, "Integrating Diffie-Hellman key exchange into the digital signature algorithm (DSA)," IEEE Communications Letters, vol. 8, no. 3, 2004, pp. 198-200.

[7] R. C.-W. Phan, "Fixing the integrated Diffie-Hellman-DSA key exchange protocol," IEEE Communications Letters, vol. 9, no. 6, 2005, pp. 570-572.

[8] J. Liu and J. Li, "A better improvement on the integrated Diffie-Hellman-DSA key agreement protocol," International Journal of Network Security, vol. 11, no. 2, Sept. 2010, pp. 114-117.

[9] S.-K. Chong, S.-F. Chiou, and M.-S. Hwang, "A simple method to secure the integrating a key distribution into digital signature standard," Proc. 8th International Conference on Computing Technology and Information Management, Seoul, Korea, April 2012, pp. 729-731.

[10] E. Andreeva, A. Bogdanov, and B. Mennink, "Towards understanding the known-key security of block ciphers," Proc. International Workshop on Fast Software Encryption, Springer Berlin Heidelberg, 2013, pp. 348-366.

[11] M. Zhu and S. Martínez, "On the performance analysis of resilient networked control systems under replay attacks," IEEE Transactions on Automatic Control, vol. 59, no. 3, 2014, pp. 804-808.

[12] T. ElGamal, "A public key cryptosystem and a signature scheme based on discrete logarithms," Proc. Workshop on the Theory and Application of Cryptographic Techniques, pp. 10-18. Springer Berlin Heidelberg, 1984.

[13] A. J. Menezes, P. C. Van Oorschot, and S. A. Vanstone, Handbook of Applied Cryptography, CRC press, 1996.

[14] J. Kaliski and S. Burton, "An unknown key-share attack on the MQV key agreement protocol," ACM Transactions on Information and System Security, vol. 4, no. 3, 2001, pp. 275-288.