Journal of Mathematics Research; Vol. 6, No. 1; 2014
ISSN 1916-9795 E-ISSN 1916-9809
Published by Canadian Center of Science and Education

# Attack on "Strong Diffie-Hellman-DSA KE" and Improvement

Demba Sow[1], Mamadou Ghouraissiou Camara[1] & Djiby Sow[1]

[1] Ecole Doctorale de Mathématiques et Informatique, Laboratoire d'Algèbre de Cryptologie de Géométrie Algèbrique et Applications, Université Cheikh Anta Diop de Dakar, Sénégal

Correspondence: Mamadou Ghouraissiou Camara, Ecole Doctorale de Mathématiques et Informatique, Laboratoire d'Algèbre de Cryptologie de Géométrie Algèbrique et Applications, Université Cheikh Anta Diop de Dakar, Sénégal. E-mail: ghouraissiou@yahoo.fr

Received: November 11, 2013 Accepted: December 15, 2013 Online Published: January 9, 2014

doi:10.5539/jmr.v6n1p70 URL: http://dx.doi.org/10.5539/jmr.v6n1p70

**Abstract**

In this paper, we do a cryptanalyse of the so called "Strong Diffie-Hellman-DSA Key Exchange (briefly: SDH-DSA-KE)" and after we propose "Strong Diffie-Hellman-Exponential-Schnnor Key Exchange (briefly: SDH-XS-KE)" which is an improvement for efficiency and security. SDH-XS-KE protocol is secure against Session State Reveal (SSR) attacks, Key independency attacks, Unknown-key share (UKS) attacks and Key-Compromise Impersonation (KCI) attacks. Furthermore, SDH-XS-KE has Perfect Forward Secrecy (PFS) property and a key confirmation step. The new proposition is not vulnerable to Disclosure to ephemeral or long-term Diffie-Hellman exponents. We design our protocol in finite groups therefore this protocol can be implemented in elliptic curves.

**Keywords:** key exchange protocol, Diffie-Hellman key exchange, DSA signature, Schnorr protocol, attacks

## 1. Introduction

Diffie-Hellman (DH) protocol (Diffie & Hellman, 1976) is the most popular key exchange protocol. Since the basic protocol is vulnerable to a large class of attacks against protocols, many proposals were done to improve the security of DH protocol (see Nyberg & Rueppel, 1994; Krawczyk, 2005). But, most of the proposals have been broken or shown to suffer from weaknesses.

In 2007, in IEEE Communications letters journal (see Jeong, Kwon, & Lee, 2007), Jeong et al. prove that the previous scheme is insecure against session state reveal attack. After, the authors propose the "Strong Diffie-Hellman-DSA Key Exchange" (briefly: SDH-DSA-KE) where the mutual authentication is done by DSA signatures but it use 5 exponents and is vulnerable to some attacks.

In this paper, we propose a cryptanalyse of SDH-DSA-KE by showing that it is insecure against KCI attacks and is vulnerable to Disclosure to ephemeral and long-term CDH exponents. After, we propose "Strong Diffie-Hellman-Exponential-Schnorr Key Exchange" (briefly: SDH-XS-KE) which is an improvement of SDH-DSA-KE for efficiency and security. Our protocol use 4 exponents and is secure against Session State Reveal (SSR) attacks, Key independency attacks, Unknown-key share (UKS) attacks and Key-Compromise Impersonation (KCI) attacks. Furthermore, SDH-XS-KE has Perfect Forward Secrecy (PFS) property. For the mutual authentication, instead of DSA signatures, we use a modified Exponential Schnorr protocol.

Note that SDH-DSA-KE was designed only over $\mathbb{Z}/p\mathbb{Z}$ but our protocol is designed over an arbitrary finite (multiplicative) group therefore our protocol can be implemented in elliptic curves.

## 2. Preliminaries

### *2.1 Discrte Logarithm Problem*

The Discrete Logarithm Problem (DLP) is the following: given a finite group $G$ of order $n$ and a cyclic subgroup $\langle g\rangle$ of prime order $q$ generated by $g$. If $y \overset{Rand}{\longleftarrow} \langle g\rangle$, find the integer $x$, $0 \leq x \leq q-1$, such that $g^x = y$.

The Computational Diffie-Hellman Problem (CDH) is the following: given a finite group $G$ of order $n$ and a cyclic subgroup $\langle g\rangle$ of prime order $q$ generated by $g$. If $y = g^a \overset{Rand}{\longleftarrow} \langle g\rangle$ and $z = g^b \overset{Rand}{\longleftarrow} \langle g\rangle$, find the group element $g^{ab}$.

*2.2 Diffie-Hellman Key Exchange*

The key exchange Diffie-Hellman protocol was developed in 1976 and published in the paper: *New directions in cryptography*.

**Diffie-Hellman Protocol**

Public data: $G$ a finite group, and $\langle g\rangle$ a cyclic subgroup of $G$ generated by $g$ with prime order $q$.

- $A$ selects an integer $a$ such that $1 < a < q-1$, keeps it secret and sends $g^a$ to $B$.
- $B$ selects an integer $b$ such that $1 < b < q-1$, keeps it secret and sends $g^b$ to $A$.
- Both $A$ and $B$ compute $k = (g^b)^a = (g^a)^b$.

*2.3 Exponential Schnorr Identification Protocol*

Let $G$ be a multiplicative group and $\langle g\rangle$ a cyclic subgroup of prime order $q$ with generator $g \in G$. The secret key $sk$ is an integer $x$ in $[1, q]$. Put $y = g^x$, the public key $pk$ is $(G, g, y)$.

Let $G$ be a multiplicative group and $\langle g\rangle$ a cyclic subgroup of prime order $q$ with generator $g \in G$.

In this protocol the prover is $\mathcal{P}$ and the verifier is $\mathcal{V}$.

The secret key $sk$ of $\mathcal{P}$ is an integer $x$ in $[1, q]$. Put $y = g^x$, the public key $pk$ of $\mathcal{P}$ is $(G, g, y)$.

(1) $\mathcal{V}$ chooses a random $w \overset{Rand}{\longleftarrow} ]1, q[$ and sends the "challenge" $W = g^w$ to $\mathcal{P}$.

(2) $\mathcal{P}$ chooses a random $v \overset{Rand}{\longleftarrow} ]1, q[$ and sends $V = g^v$ to $\mathcal{V}$.

(3) $\mathcal{V}$ chooses a random "challenge" $e \overset{Rand}{\longleftarrow} ]1, q[$ and sends $e$ to $\mathcal{P}$.

(4) $\mathcal{P}$ computes $s = v + xe \pmod q$ and sends $S = W^s$ to $\mathcal{V}$.

$\mathcal{V}$ accepts if $S = (Vy^e)^w$.

It is known that this protocol is a proof of the ability of $\mathcal{V}$ to compute $CDH(y, V)$ for any value $V \in G$. Moreover, the protocol is zero-knowledge against a verifier $\mathcal{V}$ that chooses $e$ at random (while $V$ may be chosen arbitrarily). (see Krawczyk, 2005).

*2.4 Security Notions*

Let us recall some security notions used in key exchange protocols.

(1) **Key independency.** This is a stronger notion of security and means that session keys are computationaly independent from each other.

(2)**Session state reveal attack.** The protocols providing security against session state reveal attacks maintain the secrecy of session keys even when an adversary is able to obtain the random numbers used to make the session keys.

(3) **Perfect forward secrecy (PFS).** a key-exchange protocol is said to have the PFS property if the leakage of the long-term key of a party does not compromise the security of session keys established by that party and erased from memory before the leakage occurred.

(4) **Resistance to key-compromise impersonation (KCI) attacks.** it provides the assurance that sessions established by a party Alice while not being actively controlled by the attacker, remain secure even if her private key is learned by the attacker.

(5) **The case of Diffie-Hellman key exchange protocol** (see Krawczyk, 2005). Consider a session $(id_A, id_B, V_A = g^{v_A}, V_B = g^{v_B})$ between two parties $A$ and $B$ with the following pair of private/public key $(x_A, g^{x_A})$ and $(x_B, g^{x_B})$; the computation of the session key involves the four secret values $x_A, x_B, v_A, v_B$. Obviously the disclosure of $\{x_A, v_A\}$, or $\{x_B, v_B\}$, allows the attacker to learn the session key.

For the secure of the communication between $A$ and $B$, one must prove that the disclosure of any other pair of values (except $\{x_A, v_A\}$ and $\{x_B, v_B\}$) in the set $\{x_A, x_B, v_A, v_B\}$ is insufficient for the attacker to success in any kind of attack. This includes the cases in that the attacker learns:

- $\{x_A, x_B\}$ and try to compute the session key of old sessions: this is the PFS property;

- $\{v_A, v_B\}$ : this follows from the security to State reveal attack;
- $\{x_A, v_B\}$ or $\{x_B, v_A\}$ : this follows from the security to KCI attacks;
- $g^{x_A x_B}$ without learning $(x_A, x_B)$: this follows from the security of the disclosure of long-term DH exponents;
- $g^{v_A v_B}$, without learning $\{v_A, v_B\}$: this follows from the security of the disclosure of ephemeral DH exponents.

## 3. Attacks on Strong DH-DSA Key Exchange

### *3.1 SDH-DSA-KE Protocol*

Let us recall the design of the Strong DH-DSA key exchange protocol.

Let $p, q$ be two sufficiently large primes such $q$ divides $p-1$ and $g \in \mathbb{Z}/p\mathbb{Z}$ is an element of order $q$. Let $H$: $\{0,1\}^* \to \mathbb{Z}/q\mathbb{Z}$ be a hash function.

We assume that Alice (respectively: Bob) has a pair of private/public key $(x_A, y_A = g^{x_A})$ (respectively: $(x_B, y_B = g^{x_B})$ ).

1) Alice generates $v_A \overset{Rand}{\longleftarrow} ]1, q[$, computes $m_A = g^{v_A} \mod p$ and sends $m_A$ to Bob.

2) • Bob generates $v_B \overset{Rand}{\longleftarrow} ]1, q[$ and computes $m_B = g^{v_B} \mod p$;
- Bob computes $DH1 = m_A^{v_B} \mod p = g^{v_A v_B} \mod p$, $DH2 = y_A^{x_B} \mod p = g^{x_A x_B} \mod p$, $r_B = m_B \mod p$;
- Bob signs $s_B = (v_B^{-1}(H(m_B||DH1||DH2) + x_B r_B)) \mod q$;
- Bob sends $(m_B, s_B)$ to Alice.

3) • Alice computes $DH1 = m_B^{v_A} \mod p = g^{v_A v_B} \mod p$, $DH2 = y_B^{x_A} \mod p = g^{x_A x_B} \mod p$, $r_B = m_B \mod q$, $r_A = m_A \mod q$;
- Alice check $\text{DSA.ver}_{y_B}(m_B||DH1||DH2, r_B, s_B) \overset{?}{=} 1$;
- Alice computes $K_{AB} = H(A||B||DH1||DH2)$ and $K_{BA} = H(B||A||DH1||DH2)$;
- Alice signs $s_A = (v_A^{-1}(H(m_A||DH1||DH2) + x_A r_A)) \mod q$;
- Alice sends $s_A$ to Bob.

4) • Bob computes $r_A = m_A \mod q$;
- Bob checks if $\text{DSA.ver}_{y_A}(m_A||DH1||DH2, r_A, s_A) \overset{?}{=} 1$;
- Bob computes $K_{AB} = H(A||B||DH1||DH2)$ and $K_{BA} = H(B||A||DH1||DH2)$.

### *3.2 Cryptanalysis of SDH-DSA-KE Protocols*

#### 3.2.1 KCI Attack on SDH-DSA-KE

**Theorem 3.1** *SDH-DSA-KE is insecure against Key-Compromise Impersonation (KCI) attack.*

*Proof.* Assume that the attacker knows $x_A$ and $v_B$, since $y_B = g^{x_B} \mod p$ and $m_A = g^{v_A} \mod p$ are public then the attacker can easily compute $DH1 = m_A^{v_B} \mod p = g^{v_A v_B} \mod p$, $DH2 = y_B^{x_A} \mod p = g^{x_A x_B} \mod q$ and deduce the key. We have the same result if the attacker knows ($x_B$ and $v_A$), Hence this protocol is vulnerable to Key-Compromise Impersonation (KCI) attack. □

#### 3.2.2 Disclosure to Ephemeral or Long-Term CDH Exponents

In SDH-DSA-KE protocol, the keys are computed as: $K_{AB} = \overline{H}_1(A||B||DH1||DH2)$ and $K_{BA} = \overline{H}_1(B||A||DH1||DH2)$. Therefore, in this protocol the value $DH2 = g^{x_A x_B} \mod p$ serves as a long-term shared key between parties Alice and Bob, and therefore its disclosure suffices for impersonating Alice to Bob, and vice versa.

We will see that for our improvement, the disclosure of $DH2 = g^{x_A x_B} \mod p$ does not allow impersonation Alice or Bob.

## 4. Design of SDH-XS-KE Protocol

In our protocol the mutual authentications is done via a modified Exponential Schnorr protocol where each party challenge his public key.

*4.1 Design SDH-XS-KE*

Let $G$ be a multiplicative group and $\langle g \rangle$ a cyclic subgroup of prime order $q$ with generator $g \in G$. Let $H$: $G \times G \times \overline{\mathcal{P}} \rightarrow \{0,1\}^l$ be a hash function (where $l \geq 224$ and $\overline{\mathcal{P}}$ is the set of all parties which are allowed to participate to the protocol). Let $\mathcal{G}$: $G \rightarrow \{0,1\}^l$ be a randomness extractor on $G$ and and $\overline{H}$: $\{0,1\}^l \times \overline{\mathcal{P}} \times \overline{\mathcal{P}} \times \{0,1\} \rightarrow \{0,1\}^l$ be a hash function and $\mathbf{MAC}_K$: $G \times G \times \overline{\mathcal{P}} \rightarrow \{0,1\}^l$ be a keyed hash function for Mac authentication.

We assume that Alice (respectively: Bob) has a pair of private/public key $(x_A, y_A = g^{x_A})$ (respectively: $(x_B, y_B = g^{x_B})$) where $x_A, x_B < q$ are random integers.

4.1.1 Protocol

1) Alice (the initiator) selects a secret session's random $v_A < q$, computes $V_A = g^{v_A}$, $\delta_{AB} = y_B^{v_A + x_A}$ and $h_{AB} = H(\delta_{AB}, V_A, id_A)$, discards $\delta_{AB}$ and sends $(V_A, h_{AB})$ to Bob;

2) • Bob (the responder) verifies if $V_A \neq 1$, computes $\lambda_{BA} = (V_A y_A)^{x_B}$ and $H(\lambda_{BA}, V_A, id_A)$ and discards $\lambda_{BA}$; verifies if $H(\lambda_{BA}, V_A, id_A) \neq h_{AB}$;

• If one of the above validations fail then Bob terminates the protocol run with failure;

• Bob selects a secret session's random $v_B < q$, computes $V_B = g^{v_B}$ and $K_{mac} = \overline{H}(K_{B2}, id_A, id_B, 1)$ where $K_{B2} = \mathcal{G}(g_{KBs}, l)$ and $g_{KBs} = (V_A y_A)^{v_B + x_B}$;

• Bob computes $\delta_{BA} = y_A^{v_B + x_B}$, $h_{MAC_B} = \mathbf{MAC}_{K_{mac}}(\delta_{BA}, V_B, id_B)$, discards $\delta_{BA}$ and sends $(V_B, h_{MAC_B})$ to Alice.

3) • Alice verifies if $V_B \neq 1$, computes $K_{mac} = \overline{H}(K_{A2}, id_A, id_B, 1)$ where $K_{A2} = \mathcal{G}(g_{KAs}, l)$ and $g_{KAs} = (V_B y_B)^{v_A + x_A}$;

• Computes $\lambda_{AB} = (V_B y_B)^{x_A}$ and $h'_{MAC_B} = \mathbf{MAC}_{K_{mac}}(\lambda_{AB}, V_B, id_B)$, and discards $\lambda_{AB}$; verifies if $h'_{MAC_B} \neq h_{MAC_B}$;

• If one of the above validations fail then Alice terminates the protocol run with failure; otherwise,

• Alice computes $h_{MAC_A} = \mathbf{MAC}_{K_{mac}}(g_{KAs}, V_A, id_A)$, and sends it to Bob;

• Alice computes and stores $K_{As} = \overline{H}(K_{A2}, id_A, id_B, 0)$ as her current session key.

4) • Bob computes $h'_{MAC_A} = \mathbf{MAC}_{K_{mac}}(g_{KBs}, V_A, id_A)$, and verifies if $h'_{MAC_A} \neq h_{MAC_A}$;

• If the validation fails then Alice terminates the protocol run with failure, otherwise, Bob computes and stores the key $K_{Bs} = \overline{H}(K_{B2}, id_A, id_B, 0)$ as his current session key.

*4.2 Security and Performance of SDH-XS-KE*

In our protocol the mutual authentication is done via a modified Exponential Schnorr protocol where each party challenge his public key. This protocol involves a key confirmation step. Note that the correctness of the protocol depends on the honestness of the parties establishing a session. They must choose, compute and store correctly all the values appearing in the protocol and also they must follow the steps in the good way. Therefore, in the sequel, we assume that the parties are honest.

4.2.1 Performance

Our protocols use only 4 exponents and 4 pass with a key confirmation step, therefore it is more speed than SDH-DSA-KE which use 5 exponents (3 exponents in the key generation and 2 exponents in the DSA verification process) and 4 pass.

4.2.2 Security

We have proven above that SDH-DSA-KE is vulnerable to KCI attacks. We will see in the following that our proposition is secure if CDH$(V_A, V_B)$, CDH$(y_A, y_B)$, CDH$(V_A, y_B)$, CDH$(V_B, y_A)$ are computationally infeasible and the hash function is robust.

**Theorem 4.1** *SDH-XS-KE key exchange protocol posses the Perfect Forward Secrecy property.*

*Proof.* This protocol has a key confirmation step then the attacker cannot be active when the key session is building by the two parties. The only way for the attacker, is to try to compute the session key directly, assuming that he know the long-term secret keys (namely $x_A$, $x_B$) of the parties. Since the session key is $K_{Bs} = \overline{H}(K_{B2}, id_A, id_B, 0)$ where $K_{B2} = \mathcal{G}(g_{KBs}, l)$ and $g_{KB} = g^{(v_A + x_A)(v_B + x_B)} = g^{v_A v_B} g^{x_A x_B} V_A^{x_B} V_B^{x_A}$ and the attacker knows $x_A$ and $x_B$ then he can compute $g^{x_A x_B} V_A^{x_B} V_B^{x_A}$. Therefore the attacker can compute $g_{KBs}$ if and only if he can compute $g^{(v_A v_B)} = CDH(V_A, V_B)$ which is computationally infeasible if the two parties are honest. □

**Theorem 4.2** *SDH-XS-KE key exchange protocol is secure against Key-Compromise Impersonation (KCI) attack and unknown-key share (UKS) attack.*

*Proof.* 1) Security against Key-Compromise Impersonation (KCI) attack: This protocol use a mutual authentication which is done via the hash of the output of a modified Exponential Schnorr protocol for mutual identification, where each party challenge his public key: Alice computes $\delta_{AB} = y_B^{v_A+x_A}$ and sends $h_{AB} = H(\delta_{AB}, V_A, id_A)$ to Bob and discards $\delta_{AB}$; Bob computes $\delta_{BA} = y_A^{v_B+x_B}$ and sends $h_{MAC_B} = \mathbf{MAC}_{K_{mac}}(\delta_{BA}, V_B, id_B)$ to Alice and discards $\delta_{BA}$. Hence the authentication fails if the attacker is active and doesn't know simultaneously $v_A$ and $x_A$ or $v_B$ and $x_B$. Therefore, the only way for the attacker, is to try to compute the session key directly, assuming that he knows the long-term secret key of Alice (namely $x_A$) and the session's random of Bob (namely $v_B$). Since the session key is $K_{Bs} = \overline{H}(K_{B2}, id_A, id_B, 0)$ where $K_{B2} = \mathcal{G}(g_{KB}, l)$ and $g_{KB} = g^{(v_A+x_A)(v_B+x_B)} = V_A^{v_B} y_B^{x_A} V_B^{x_A} V_A^{x_B}$ and the attacker knows $x_A$ and $v_B$ then he can compute $V_A^{v_B} y_B^{x_A} V_B^{x_A}$. Therefore the attacker can compute $g_{KBs}$ if and only if he can compute $V_A^{x_B} = CDH(V_A, y_B) = DLP_{V_A}(V_A^{x_B})$ which is computationally infeasible if the two parties are honest.

2) Security against unknown-key share (UKS) attack: in the authentication process, Alice computes $\delta_{AB} = y_B^{v_A+x_A}$ and sends $h_{AB} = H(\delta_{AB}, V_A, id_A)$ to Bob and discards $\delta_{AB}$; Bob computes $\delta_{BA} = y_A^{v_B+x_B}$ and sends $h_{MAC_B} = \mathbf{MAC}_{K_{mac}}(\delta_{BA}, V_B, id_B)$ to Alice and discards $\delta_{BA}$. Hence the public keys and the identities of the parties $(id_A, id_B)$ are hashed. This fact prevent from UKS attacks. □

**Theorem 4.3** *SDH-XS-KE key exchange protocol is secure against Session State Reveal (SSR) attack.*

*Proof.* Since the session key is $K_{Bs} = \overline{H}(K_{B2}, id_A, id_B, 0)$ where $K_{B2} = \mathcal{G}(g_{KBs}, l)$ and $g_{KBs} = g^{(v_A+x_A)(v_B+x_B)} = g^{v_A v_B} g^{x_A x_B} y_A^{v_B} y_B^{v_A}$ and the attacker knows $v_A$ and $v_B$ then he can compute $g^{v_A v_B} y_A^{v_B} y_B^{v_A}$. Therefore the attacker can compute $g_{KBs}$ if and only if he can compute $g^{x_A x_B} = CDH(y_A, y_B)$ which is computationally infeasible if the two parties are honest. □

**Theorem 4.4** *SDH-XS-KE key exchange protocol posses the key independency property.*

*Proof.* Since the session key is $K_{Bs} = \overline{H}(K_{B2}, id_A, id_B, 0)$ where $K_{B2} = \mathcal{G}(g_{KBs}, l)$ and $g_{KBs} = g^{(v_A+x_A)(v_B+x_B)} = g^{v_A v_B} g^{x_A x_B} V_A^{x_B} V_B^{x_A}$. Then key independency property is guaranteed by the properties of the hash function and the usage of the identities $id_A$ and $id_B$ and the session's random $v_A$ and $v_B$. □

**Theorem 4.5** *SDH-XS-KE key exchange protocol is secure against attack based on "disclosure to ephemeral and long-term CDH exponents".*

*Proof.* Since the session key is $K_{Bs} = \overline{H}(K_{B2}, id_A, id_B, 0)$ where $K_{B2} = \mathcal{G}(g_{KBs}, l)$ and $g_{KBs} = g^{(v_A+x_A)(v_B+x_B)} = g^{v_A v_B} g^{x_A x_B} y_A^{v_B} y_B^{v_A}$ and the attacker knows $g^{v_A v_B}$ and $g^{x_A x_B}$ then he can compute $g^{v_A v_B} g^{x_A x_B}$. Therefore the attacker can compute $g_{KBs}$ if and only if he can compute $y_A^{v_B} y_B^{v_A} = CDH(y_A, V_B) CDH(V_A, y_B)$ which is computationally infeasible if the two parties are honest. □

## 5. Conclusion

We have successfully described in this paper an attack on SDH-DSA-KE protocol and proposed a new protocol key exchange named SDH-XS-KE which resist for all attacks known on key exchange protocols.

## References

Arazi, A. (1993). Inegrating a key cryptosystem into the digital signature standard. *Electron. Lett., 29*, 966-967.

Canetti, R., & Krawczyk, H. (2001). Analysis of Key-Exchange Protocols and Their Use for Building Secure Channels. *Advances in Cryptology–EUROCRYPT 2001, Lecture Notes in Computer Science, 2045*, 453-474. http://dx.doi.org/10.1007/3-540-44987-6_28

Diffie, W., & Hellman, M. E. (1976). New Directions in Cryptography. *IEEE Trans. on Info. Theory, IT-22*, 644-654. http://dx.doi.org/10.1109/TIT.1976.1055638

Harn, L., Mehta, M., & Hsin, W. J. (2004). Integratin Diffie-Hellman key exchange into a digital signature algorithm (DSA). *IEEE Commun. Lett., 8*(3), 198-200. http://dx.doi.org/10.1109/LCOMM.2004.825705

Jeong, I. R., Kwon, J. O., & Lee, D. H. (2007). Strong Diffie-Hellman DSA Key Exchange. *IEEE Communications Letters, 11*(5), 432-433. http://dx.doi.org/10.1109/LCOMM.2007.070004

Krawczyk, H. (2005). HMQV: A high-performance secure Diffie-Hellman Protocol. *Advances in Cryptology–CRYPTO 2005, Lecture Notes in Computer Science, 3621*, 546-566. http://dx.doi.org/10.1007/1153521833

Law, L., Menezes, A., Qu, M., Solinas, J., & Vanstone, S. (2003). An efficient Protocol for Authenticated Key

Agreement. *Designs, Codes and Cryptography, 28*, 119-134.

Menezes, A. J., van Oorschot, P. C., & Vanstone, S. A. (1996). *Handbook of Applied Cryptography*. CRC Press.

Menezes, A., Qu, M., & Vanstone, S. (1995). *Some new key agreement protocols providing mutual implicit authentication*. Second Workshop on Selected Areas in Cryptography (SAC 95).

Nyberg, K., & Rueppel, R. A. (1994). Weaknesses in some recent key agreement protocols. *Electron. Lett., 30*, 26-27. http://dx.doi.org/10.1049/el:19940052

Phan, R. C. W. (2005). Fixing the integrated Diffie-Hellman-DSA key exchange protocol. *IEEE Commun. Lett., 9*, 570-572.