One-Round Protocols for Two-Party Authenticated Key Exchange

Ik Rae Jeong[1⋆], Jonathan Katz[2⋆⋆], and Dong Hoon Lee[1⋆]

[1] Center for Information Security Technologies (CIST),
Korea University, Seoul, Korea. jir@cist.korea.ac.kr, donghlee@korea.ac.kr
[2] Dept. of Computer Science, University of Maryland, College Park, MD, USA.
jkatz@cs.umd.edu

Abstract. Cryptographic protocol design in a two-party setting has often ignored the possibility of *simultaneous* message transmission by each of the two parties (i.e., using a duplex channel). In particular, most protocols for two-party key exchange have been designed assuming that parties alternate sending their messages (i.e., assuming a bidirectional half-duplex channel). However, by taking advantage of the communication characteristics of the network it may be possible to design protocols with improved latency. This is the focus of the present work.
We present a number of provably-secure protocols for two-party authenticated key exchange (AKE) which require only a single round. Our first protocol provides *key independence* only, and is analyzed in the random oracle model. This scheme matches the most efficient AKE protocols among those found in the literature. Our second scheme additionally provides *forward secrecy*, and is also analyzed in the random oracle model. Our final protocol provides the same strong security guarantees, but is proven secure in the standard model. This scheme is only slightly less efficient (from a computational perspective) than the previous ones. These last two schemes are the first provably-secure *one-round* protocols for authenticated 2-party key exchange which provide forward secrecy.

Keywords: Authenticated key exchange, Forward secrecy, Round complexity, Diffie-Hellman key exchange.

1 Introduction

Key-exchange protocols are among the most basic and widely used cryptographic protocols. Such protocols are used to derive a common *session key* between two (or more) parties; this session key may then be used to communicate securely over an insecure public network. Thus, secure key-exchange protocols serve as basic building blocks for constructing secure, complex, higher-level protocols. For this reason, the computational efficiency, communication requirements, and

⋆ Work supported by the Ministry of Information & Communications, Korea, under the Information Technology Research Center (ITRC) Support Program.
⋆⋆ Work supported by NSF Trusted Computing Grant #ANI-0310751.

M. Jakobsson, M. Yung, J. Zhou (Eds.): ACNS 2004, LNCS 3089, pp. 220–232, 2004.

round complexity of key-exchange protocols are very important and have received much attention, both in the two-party [16,22,5,17,4,3,6,7,14] and multi-party (i.e., group) [18,13,26,20,2,12,10,9,21] settings.

This paper concerns protocols for *authenticated* key exchange (AKE); achieving such authentication is only possible if some out-of-band initialization phase is assumed prior to execution of the protocol. One common assumption is that each communicating party has an associated public-/private-key pair, with the public key known to all other parties in the network (of course, this includes the adversary). We assume this model here.

Most protocols for two-party key exchange have been designed and analyzed assuming that parties alternate sending messages (equivalently, that the parties communicate over a bidirectional *half-duplex* channel). However, in many common applications parties can actually transmit messages simultaneously (i.e., they have access to a bidirectional *duplex* channel). Of course, any protocol designed and proven secure in the former model may be used in the latter; however, it may be possible to design protocols with improved round complexity by fully exploiting the communication characteristics of the underlying network, and in particular the possibility of simultaneous message transmission.

As a simple example, consider the traditional Diffie-Hellman key-exchange protocol [16] (which does *not* provide any authentication). Traditionally, this is presented as a two-round protocol in which Alice first sends g^a and Bob then replies with g^b. However, in this particular case Alice and Bob can send their messages simultaneously, thereby "collapsing" this protocol to a single round. However, the situation is more complex when authentication is required. For instance, *authenticated* Diffie-Hellman typically involves one party signing messages sent by the other party; this may be viewed as a type of "challenge-response" mechanism. (For example, the work of Bellare, et al. [3] suggests implementing "authenticated channels" in exactly this way.) When this is done, it is no longer possible to collapse the protocol to a single round.

Motivated by the above discussion, we explore the possibility of designing protocols for authenticated key exchange which can be implemented in only a single round (assuming simultaneous message transmission). Of course, we will also ensure that our protocols are efficient with respect to other measures, including communication complexity and computational efficiency.

1.1 Our Work in Relation to Prior Work

Before relating our work to prior works, we briefly recall various notions of security for key exchange protocols (formal definitions are given below). At the most basic level, an authenticated key-exchange scheme must provide secrecy of a generated session key. Yet to completely define a notion of security, we must define the class of adversarial behaviors tolerated by the protocol. A protocol achieving *implicit authentication* simply ensures secrecy of session keys for an adversary who passively eavesdrops on protocol executions and may also send messages of its choice to the various parties. A stronger notion of security (and the one that is perhaps most often considered in the cryptographic literature) is

key independence, which means that session keys are computationally independent from each other. A bit more formally, key independence protects against "Denning-Sacco" attacks [15] involving compromise of multiple session keys (for sessions other than the one whose secrecy must be guaranteed). Lastly, protocols achieving *forward secrecy* maintain secrecy of session keys even when an adversary is able to obtain long-term secret keys of principals who have previously generated a common session key (in an honest execution of the protocol, without any interference by the adversary).

The original two-party key-exchange scheme of Diffie and Hellman [16] is secure against passive eavesdroppers, but not against active attacks; indeed, that protocol provides no authentication at all. Several variations of the scheme have been suggested to provide security against active attacks [22,23,24,7], but these schemes have either been found to be flawed or have not yet been proven secure. There are only a few provably secure schemes in the literature which provide both key independence and forward secrecy. Most such schemes seem to be "overloaded" so as to provide explicit authentication along with key independence and forward secrecy. (For example, the schemes of [1,6,3] use signatures and/or message authentication codes to authenticate messages in a way that achieves explicit authentication.) However, in some cases explicit authentication may be unnecessary, or may be provided anyway by subsequent communication. Thus, one may wonder whether more efficient protocols (say, with reduced round complexity) are possible if explicit authentication is not a requirement.

We first propose and analyze a very simple one-round scheme, $\mathcal{TS}1$, which provides key independence but not forward secrecy (security is based on the computational Diffie-Hellman assumption in the random oracle model). In Table 1 we compare our scheme to a scheme of Boyd and Nieto [9] which achieves the same level of security in the same number of rounds. (Boyd and Nieto actually propose a protocol for group AKE, but their protocol can of course be instantiated for the case of two parties.) Our scheme is (slightly) more efficient than the scheme of Boyd and Nieto and has other advantages as well: our protocol is simpler and is also symmetric with respect to the two parties.

Table 1. Comparison of the Boyd-Nieto scheme [9] to $\mathcal{TS}1$. Efficiency of the Boyd-Nieto scheme depends on the instantiation of its generic components; the above are rough estimates assuming the random oracle model and "discrete-log-based" components using an order-q subgroup of $\mathbb{Z}_p^*$.

	Boyd-Nieto*	$\mathcal{TS}1$
Modular exponentiations (per party)	2	1
Communication (total)	$2\|p\| + \|q\|$	$2\|q\|$
Security	KI	KI
Assumptions	(varies)	CDH in random oracle model

Table 2. Comparison of key-exchange protocols achieving key independence and forward secrecy. Efficiency of some schemes depends on instantiation details; the above represent rough estimates assuming "discrete-log-based" instantiations using an order-q subgroup of $\mathbb{Z}_p^*$.

	[1,6]	Auth. DH (cf. [3])	Auth. DH (cf. [21])	$\mathcal{TS}2$	$\mathcal{TS}3$
Modular exponentiations (par party)	3	4	4	3	3
Rounds	3	3	2	1	1
Communication (total)	$2\|p\| + 2\|q\|$	$4\|p\|$	$4\|p\| + 2\|q\|$	$2\|p\|$	$2\|p\| + 2\|q\|$
Model	R.O.	standard	standard	R.O.	standard

We next propose a modification of this scheme, $\mathcal{TS}2$, which provides both key independence and forward secrecy, yet still requires only a single round of communication (security is again proved based on the CDH assumption in the random oracle model). We are not aware of any previous one-round protocol achieving this level of security. $\mathcal{TS}2$ requires only 3 modular exponentiations per party and uses neither key confirmation nor digital signatures, and hence the protocol is more efficient than previous schemes in terms of computation and communication as well. A drawback of $\mathcal{TS}2$ is that its security is analyzed only in the random oracle model. For this reason, we propose a third protocol, $\mathcal{TS}3$, which provides the same level of security in the same number of rounds but whose security can be analyzed in the standard model based on the stronger, but still standard, *decisional* Diffie-Hellman assumption. This protocol is only slightly less efficient than $\mathcal{TS}2$ (it uses message authentication codes, whose efficiency is negligible compared to modular exponentiations). We compare both of these protocols to previous work in Table 2.

1.2 Outline

In Section 2 we define our security model for authenticated key exchange. We present our two-party authenticated key-exchange protocols in Section 3. Proofs of security for each of our protocols are deferred to the full version of this paper.

2 Security Model for Authenticated Key Exchange

We use the standard notion of security as defined in [4] and used extensively since then. We assume that there are N parties, and each party's identity is denoted as P_i. Each party P_i holds a pair of private and public keys. We consider a key-exchange protocol in which two parties want to exchange a session key using their public keys. Π_i^k represents the k-th instance of player P_i. If a key-exchange protocol terminates, then Π_i^k generates a session key $sk_{\Pi_i^k}$. A session identifier of an instance, denoted $sid_{\Pi_i^k}$, is a string different from those of all other sessions

in the system (with high probability). We assume that $sid_{\Pi_i^k}$ is a concatenation of all transmitted messages of a session in Π_i^k, where the sequence of messages is determined *by the (lexicographic, say) ordering of the owners*. Note that ordering messages by their appearance cannot be used in our setting, because two parties may send their messages simultaneously.

We denote the identity set of the communicating parties in a session $sid_{\Pi_i^k}$ by $C_{\Pi_i^k}$, where $|C_{\Pi_i^k}| = 2$ in our case, and the index set of identities of the communicating parties in a session $sid_{\Pi_i^k}$ is denoted by $I_{\Pi_i^k} = \{i | P_i \in C_{\Pi_i^k}\}$. We say that Π_i^k and Π_j^l are *matching* if i and $j(\neq i)$ are in $I_{\Pi_i^k}$, and $sid_{\Pi_i^k}$ and $sid_{\Pi_j^l}$ are equal. Any protocol should satisfy the following correctness condition: if two instances are matching, then the session keys computed by those instances are equal.

To define a notion of security, we define the capabilities of an adversary. We allow the adversary to potentially control all communication in the network via access to a set of oracles as defined below. We consider an *experiment* in which the adversary asks queries to oracles, and the oracles answer back to the adversary. Oracle queries model attacks which an adversary may use in the real system. We consider the following types of queries in this paper.

- A query $\mathsf{Initiate}(C)$ models an invocation of a key-exchange protocol in the real system in which each $P_i \in C$ initiates a key exchange protocol with other entities in C and sends the first message of the protocol.
- A query $\mathsf{Send}(\Pi_i^k, M)$ is used to send a message M to instance Π_i^k. When Π_i^k receives M, it responds according to the key-exchange protocol. An adversary may use this query to perform *active* attacks by modifying and inserting the messages of the key-exchange protocol. Impersonation attacks and man-in-the-middle attacks are also possible using this query.
- A query $\mathsf{Execute}(C)$ represents passive eavesdropping of the adversary on an execution of the protocol by the parties in C. Namely, the parties specified in C execute the protocol without any interference from the adversary, and the adversary is given the resulting transcript of the execution. (Although the output of an Execute query can be simulated via repeated Initiate and Send oracle queries, this particular query is needed to define forward secrecy.)
- A query $\mathsf{Reveal}(\Pi_i^k)$ models *known key* attacks (or Denning-Sacco attacks) in the real system. The adversary is given the session key for the specified instance.
- A query $\mathsf{Corrupt}(P_i)$ models exposure of the long-term key held by player P_i. The adversary is assumed to be able to obtain long-term keys of players, but cannot control the behavior of these players directly (of course, once the adversary has asked a query $\mathsf{Corrupt}(P_i)$, the adversary may impersonate P_i in subsequent Send queries.)
- A query $\mathsf{Test}(\Pi_i^k)$ is used to define the advantage of an adversary. When an adversary $\mathcal{A}$ asks a *test* query to an instance Π_i^k, a coin b is flipped. If b is 1, then the session key $sk_{\Pi_i^k}$ is returned. Otherwise, a random string is

returned. The adversary is allowed to make a single Test query, at any time during the experiment.

At the end of the experiment, the adversary $\mathcal{A}$ outputs a bit b'. The advantage of $\mathcal{A}$, denoted $\mathsf{Adv}_{\mathcal{A}}(\cdot)$, is defined as $|2 \cdot \Pr[b' = b] - 1|$.

To define a meaningful notion of security, we must first define *freshness*.

Definition 1. An instance Π_i^k is *fresh* if both the following conditions are true at the conclusion of the experiment described above:

(a) For all $P_j \in C_{\Pi_i^k}$, the adversary has not queried $\mathsf{Corrupt}(P_j)$.
(b) The adversary has not queried $\mathsf{Reveal}(\Pi_i^k)$, nor has it queried $\mathsf{Reveal}(\Pi_j^\ell)$ where Π_j^ℓ and Π_i^k are matching.

In all cases described below, the adversary is only allowed to ask its Test query to a fresh instance. Generically speaking, a protocol is called "secure" if the advantage of any PPT adversary is negligible. The following notions of security may then be considered, depending on the types of queries the adversary is allowed to ask:

(1) IA (Implicit Authentication): An adversary $\mathcal{A}$ can ask neither Reveal nor Corrupt queries.
(2) KI (Key Independence): An adversary $\mathcal{A}$ can ask Reveal queries, but can not ask Corrupt queries.
(3) FS (Forward Secrecy): An adversary $\mathcal{A}$ can ask *corrupt* queries, but can not ask *reveal* queries. The freshness condition (a) in this case is changed as follows: either the adversary did not query $\mathsf{Corrupt}(P_j)$ for any $P_j \in C_{\Pi_i^k}$, or the adversary did not query $\mathsf{Send}(P_j, \star)$ for any $P_j \in C_{\Pi_i^k}$ (and thus must have instead queried $\mathsf{Execute}(C_{\Pi_i^k})$).

Of course, the strongest notion of security requires both key independence and forward secrecy.

If a key exchange scheme satisfies (1), it is called a IA-secure key exchange scheme. If a key exchange scheme satisfies (2), it is called a KI-secure key exchange scheme. If a key exchange scheme satisfies (3), it is called a FS-secure key exchange scheme. If a key exchange scheme satisfies both (2) and (3), it is called a KI&FS-secure key exchange scheme.

For an adversary $\mathcal{A}$ attacking a scheme in the sense of XX (where XX is one of IA, KI, FS, or KI&FS), we denote the advantage of this adversary (as a function of k) by $\mathsf{Adv}_{\mathcal{A}}^{XX}(k)$. For a particular protocol P, we may define its security via:

$$\mathsf{Adv}_P^{XX}(k,t) = \max_{\mathcal{A}}\{\mathsf{Adv}_{\mathcal{A}}^{XX}(k)\},$$

where the maximum is taken over all adversaries running in time t. A scheme P is said to be XX-secure if $\mathsf{Adv}_P^{XX}(k,t)$ is negligible (in k) for any $t = \mathrm{poly}(k)$.

$C = \{P_1, P_2\}; I = \{1, 2\}$

	$P_1(x_1)$	$P_2(x_2)$
Round 1	r_1	r_2

$sid = r_1 || r_2$
$sk = H(sid || g^{x_1 x_2})$

Fig. 1. An example of an execution of $\mathcal{TS}1$

3 One-Round Protocols for Authenticated Key Exchange

We assume that parties can be ordered by their names (e.g., lexicographically) and write $P_i < P_j$ to denote this ordering. Let k be a security parameter, and let G be a group of prime order q (where $|q| = k$) with generator g. Let H be a hash function such that $H : \{0,1\}^* \to \mathbb{Z}_q$. We assume that each party P_i has a public-/private-key pair $(y_i = g^{x_i}, x_i)$ which is known to all other parties in the network (alternately, these keys may be certified by a central CA). Recall that the standard definition of security (discussed above) does not include the possibility of "malicious insiders"; thus, in particular, we assume that all public-/secret-keys are honestly generated.

We now present our first protocol $\mathcal{TS}1$:

$\mathcal{TS}1$

Setup: Assume P_i wants to establish a session key with $P_j \neq P_i$, and $P_i < P_j$. Let (y_i, x_i) (respectively, (y_j, x_j)) denote the public-/private-keys of player P_i (respectively, P_j).

Round 1: P_i selects a random number $r_i \in_R \{0,1\}^k$ and transmits it (and P_j acts analogously).

Computation of session key: P_i forms a session identifier by concatenating the messages according to the ordering of P_i, P_j. That is, $sid_{\Pi_i} = sid_{\Pi_j} = r_i || r_j$. Party P_i computes the session key $sk_{\Pi_i} = H(sid_{\Pi_i} || y_j^{x_i})$ (and P_j acts analogously).

An example of an execution of $\mathcal{TS}1$ is shown in Fig. 1. In the example we assume that $P_1 < P_2$. The following theorem states the security achieved by this protocol.

Theorem 1. Under the CDH assumption, $\mathcal{TS}1$ is a KI-secure key-exchange protocol when H is modeled as a random oracle. Concretely,

$$\mathsf{Adv}^{KI}_{\mathcal{TS}1}(k, t, q_{re}, q_H) \leqslant 2 \cdot q_H \cdot N^2 \cdot \mathsf{Adv}^{CDH}(k, t) + \frac{4q_s^2}{2^k},$$

where t is the maximum total experiment time including the adversary's execution time, and the adversary makes q_{re} Reveal queries and q_H hash queries. Here, N is an upper bound on the number of parties, and q_s is an upper bound on the number of the sessions an adversary initiates.

The proof of this theorem appears in the full version of this paper [19].

It is easy to see that $\mathcal{TS}1$ does not provide forward secrecy. To provide forward secrecy, we add an ephemeral Diffie-Hellman exchange to $\mathcal{TS}1$. The resulting protocol, $\mathcal{TS}2$, is given below:

$\mathcal{TS}2$

Setup : Same as in $\mathcal{TS}1$.

Round 1 : P_i selects a random number $\alpha_i \in_R \mathbb{Z}_q$ and sends $B_i = g^{\alpha_i}$ to the other party. (Party P_j acts analogously.)

Computation of session key : P_i forms a session identifier by concatenating the messages according to the ordering of P_i, P_j. That is, $sid_{\Pi_i} = sid_{\Pi_j} = B_i||B_j$. P_i computes the session key $sk_{\Pi_i} = H(sid_{\Pi_i}||B_j^{\alpha_i}||y_j^{x_i})$. (Party P_j acts analogously.)

An example of an execution of $\mathcal{TS}2$ is shown in Fig. 2. In the example we assume that $P_1 < P_2$.

$C = \{P_1, P_2\}; I = \{1, 2\}$

	$P_1(x_1)$	$P_2(x_2)$
Round 1	g^{α_1}	g^{α_2}

$sid = g^{\alpha_1}||g^{\alpha_2}$

$sk = H(sid||g^{\alpha_1\alpha_2}||g^{x_1x_2})$

Fig. 2. An example of an execution of $\mathcal{TS}2$

The following characterizes the security of $\mathcal{TS}2$.

Theorem 2. Under the CDH assumption, $\mathcal{TS}2$ is a KI&FS-secure key-exchange protocol when H is modeled as a random oracle. Concretely,

$$\mathsf{Adv}^{KI\&FS}_{\mathcal{TS}2}(k, t, q_{re}, q_{co}, q_H) \leqslant 2 \cdot q_H \cdot (N^2 + q_s) \cdot \mathsf{Adv}^{CDH}(k, t) + \frac{4q_s^2}{q},$$

where t is the maximum total experiment time including an adversary's execution time, and an adversary makes q_{re} Reveal queries, q_{co} Corrupt queries, and q_H hash queries. N is an upper bound of the number of parties, and q_s is the upper bound on the number of the sessions an adversary initiates.

The proof of this theorem appears in the full version of this paper [19].

The security of $\mathcal{TS}2$ (and $\mathcal{TS}1$, for that matter) is proven in the random oracle model. Next, we present protocol $\mathcal{TS}3$ which may be proven secure in the standard model (under the stronger DDH assumption):

$\mathcal{TS}3$

Setup: Same as in $\mathcal{TS}1$.

Round 1: P_i computes $k_{i,j} = k_{j,i} = y_j^{x_i}$ which it will use as a key for a secure message authentication code. (Of course, $k_{i,j}$ may need to be hashed before being used; we ignore this technicality here.) Next, P_i chooses a random number $\alpha_i \in_R \mathbb{Z}_q$, computes $\tau_i \leftarrow \mathsf{MAC}_{k_{i,j}}(P_i||P_j||g^{\alpha_i})$, and sends $B_i = g^{\alpha_i}||\tau_i$ to the other party. (Party P_j acts analogously.)

Computation of session key: P_i verifies the MAC of the received message. If verification fails, no session key is computed. Otherwise, P_i computes a session key $sk_{\Pi_i} = (g^{\alpha_j})^{\alpha_i}$. The session identifier, computed by concatenating the messages, is $sid_{\Pi_i} = B_i||B_j$. (Party P_j acts analogously.)

An example of an execution of $\mathcal{TS}3$ is shown in Fig. 3. In the example we assume that $P_1 < P_2$.

$C = \{P_1, P_2\}; I = \{1, 2\}$

	$P_1(x_1)$	$P_2(x_2)$
Round 1	$g^{\alpha_1}\|\|\tau_1$	$g^{\alpha_2}\|\|\tau_2$

$k_{1,2} \leftarrow g^{x_1 x_2}$

$\tau_1 \leftarrow \mathsf{MAC}_{k_{1,2}}(P_1||P_2||g^{\alpha_1}); \tau_2 \leftarrow \mathsf{MAC}_{k_{1,2}}(P_2||P_1||g^{\alpha_2})$

$sid = g^{\alpha_1}||\tau_1||g^{\alpha_2}||\tau_2$

$sk = g^{\alpha_1 \alpha_2}$

Fig. 3. An example of an execution of $\mathcal{TS}3$

The following characterizes the security of $\mathcal{TS}3$.

Theorem 3. Let M be an unforgeable MAC scheme. Then $\mathcal{TS}3$ is a KI&FS-secure key exchange scheme under the DDH assumption. Concretely,

$$\mathsf{Adv}^{KI\&FS}_{\mathcal{TS}3}(k, t, q_{re}, q_{co}) \leqslant (q_s + 2 \cdot N^2 + 2 \cdot q_s^2) \cdot \mathsf{Adv}^{DDH}(k, t) + N^2 \cdot \mathsf{Adv}^{SUF}_{M}(k, t, 2 \cdot q_s) + \frac{4q_s^2}{q},$$

where t is the maximum total experiment time including an adversary's execution time, and an adversary makes q_{re} Reveal queries and q_{co} Corrupt queries. N is an upper bound on the number of parties, and q_s is an upper bound of the number of the sessions an adversary initiates.

The proof of this theorem appears in the full version of this paper [19]. We note that the concrete security bound given here can be improved using random self-reducibility of the DDH problem.

A variant. In the above description of $\mathcal{TS}3$, each party computes a key $k_{i,j}$ which it then uses to authenticate its message using a message authentication code. It is also possible to have each party P_i *sign* its messages using, for example, its public key y_i as part of a Schnorr signature scheme. In this case, the party should sign (P_i, P_j, g^{α_i}) (in particular, it should sign the recipient's identity as well) to ensure that the signed message will be accepted only by the intended partner. The proof of security for this modified version is completely analogous to (and, in fact, slightly easier than) the proof of $\mathcal{TS}3$.

References

1. R. Ankney, D. Johnson, and M. Matyas. The Unified Model. Contribution to ANSI X9F1, October 1995.
2. G. Ateniese, M. Steiner, and G. Tsudik. New Multi-Party Authentication Services and Key Agreement Protocols. *IEEE Journal of Selected Areas in Communications*, volume 18, No. 4, pages 628–639, 2000.
3. M. Bellare, R. Canetti, and H. Krawczyk. A Modular Approach to the Design and Analysis of Authentication and Key Exchange Protocols. *Proc. 30th Annual Symposium on the Theory of Computing*, pages 419–428, ACM, 1998.
4. M. Bellare and P. Rogaway. Entity Authentication and Key Distribution. *Advances in Cryptology-CRYPTO 1993*, volume 773 of *Lecture Notes in Computer Science*, pages 232–249, Springer Verlag, 1993.
5. R. Bird, I. Gopal, A. Herzberg, P. Janson, S. Kutten, R. Molva, and M. Yung. Systematic Design of Two-Party Authentication Protocols. *IEEE Journal on Selected Areas in Communications* 11(5): 679–693 (1993).
6. S. Blake-Wilson, D. Johnson, and A. Menezes. Key Agreement Protocols and their Security Analysis. *Sixth IMA International Conference on Cryptography and Coding*, volume 1335, pages 30–45, ACM, 1997.
7. S. Blake-Wilson and A. Menezes. Authenticated Diffie-Hellman Key Agreement Protocols. *Selected Areas in Cryptography*, volume 1556 of *Lecture Notes in Computer Science*, pages 339–361, Springer Verlag, 1998.
8. C. Boyd. On Key Agreement and Conference Key Agreement. *ACISP 1997*, volume 1270 of *Lecture Notes in Computer Science*, page 294–302, Springer Verlag, 1997.
9. C. Boyd and J.M.G. Nieto. Round-Optimal Contributory Conference Key Agreement. *Public Key Cryptography*, volume 2567 of *Lecture Notes in Computer Science*, pages 161–174, Springer Verlag, 2003.
10. E. Bresson, O. Chevassut, and D. Pointcheval. Provably Authenticated Group Diffie-Hellman Key Exchange — The Dynamic Case. *Advances in Cryptology-ASIACRYPT 2001*, volume 2248 of *Lecture Notes in Computer Science*, pages 290–309, Springer Verlag, 2001.

11. E. Bresson, O. Chevassut, and D. Pointcheval. Dynamic Group Diffie-Hellman Key Exchange under Standard Assumptions. *Advances in Cryptology-EUROCRYPT 2002*, volume 2332 of *Lecture Notes in Computer Science*, pages 321–336, Springer Verlag, 2002.
12. E. Bresson, O. Chevassut, D. Pointcheval, and J.-J. Quisquater. Provably Authenticated Group Diffie-Hellman Key Exchange. *ACM Conference on Computer and Communications Security*, pages 255–264, 2001.
13. M. Burmester and Y. Desmedt. A Secure and Efficient Conference Key Distribution System. *Advances in Cryptology-EUROCRYPT 1994*, volume 950 of *Lecture Notes in Computer Science*, pages 275–286, Springer Verlag, 1994.
14. R. Canetti and H. Krawczyk. Universally Composable Notions of Key Exchange and Secure Channels. *Advances in Cryptology-Eurocrypt 2002*, volume 2332 of *Lecture Notes in Computer Science*, pages 337–351, Springer Verlag, 2002.
15. D. Denning and G. M. Sacco. Timestamps in Key Distribution Protocols. *Comm. ACM* 24(8): 533–536, 1981.
16. W. Diffie and M. Hellman. New Directions in Cryptography. *IEEE Transactions on Information Theory*, volume 22, Issue 6, pages 644–654, 1976.
17. W. Diffie, P. van Oorschot, and M. Wiener. Authentication and Authenticated Key Exchanges. *Designs, Codes, and Cryptography* 2(2): 107–125 (1992).
18. I. Ingemarasson, D.T. Tang, and C.K. Wong. A Conference Key Distribution System. *IEEE Transactions on Information Theory*, volume 28, Issue. 5, pages 714–720 , 1982.
19. I.R. Jeong, J. Katz, and D.H. Lee. Full version of this paper. Available at http://cist.korea.ac.kr/e_cist/e_index.htm.
20. M. Just and S. Vaudenay. Authenticated Multi-Party Key Agreement. *ASIACRYPT 1996*, volume 1163 of *Lecture Notes in Computer Science*, page 36–49, Springer Verlag, 1996.
21. J. Katz and M. Yung. Scalable Protocols for Authenticated Group Key Exchange. *Advances in Cryptology — CRYPTO 2003*.
22. L. Law, A. Menezes, M. Qu, J. Solinas, and S. Vanstone. An Efficient Protocol for Authenticated Key Agreement. Technical report CORR 98-05, University of Waterloo, 1988.
23. T. Matsumoto, Y. Takashima, and H. Imai. On Seeking Smart Public-Key Distribution Systems. *The Transactions of the IECE of Japan*, E69, pages 99–106, 1986.
24. National Security Agency. SKIPJACK and KEA algorithm specification. Version 2.0, May 29, 1998.
25. V. Shoup. On Formal Models for Secure Key Exchange. Available at http://eprint.iacr.org.
26. M. Steiner, G. Tsudik, and M. Waidner. Diffie-Hellman Key Distribution Extended to Group Communication. *ACM Conference on Computer and Communications Security*, page 31–37, 1996.
27. W.-G. Tzeng. A Practical and Secure-Fault-Tolerant Conference-Key Agreement Protocol. *Public Key Cryptography 2000*, volume 1751 of *Lecture Notes in Computer Science*, page 1–13, Springer Verlag, 2000.

A Primitives

A.1 Computational Diffie-Hellman Problem

Let $\mathcal{GG}$ be a group generator which generates a group G whose prime order is q and a generator g. Let $k \in N$ be a security parameter. Consider the following experiment:

$\mathbf{Exp}^{\mathrm{CDH}}_{\mathcal{A}_{\mathrm{CDH}}}(k)$
$(G, q, g) \leftarrow \mathcal{GG}(k)$
$u_1, u_2 \in_R [1, q-1]$
$U_1 \leftarrow g^{u_1}; U_2 \leftarrow g^{u_2}$
$W \leftarrow \mathcal{A}_{\mathrm{CDH}}(U_1, U_2)$
if $W = g^{u_1 u_2}$ return 1
else return 0

The advantage of an adversary $\mathcal{A}_{\mathrm{CDH}}(k)$ is defined as follows:

$$Adv^{\mathrm{CDH}}_{\mathcal{A}_{\mathrm{CDH}}}(k) = Pr[\mathbf{Exp}^{\mathrm{CDH}}_{\mathcal{A}_{\mathrm{CDH}}}(k) = 1]$$

The advantage function is defined as follows:

$$Adv^{\mathrm{CDH}}(k, t) = \max_{\mathcal{A}} \{Adv^{\mathrm{CDH}}_{\mathcal{A}_{\mathrm{CDH}}}(k)\},$$

where $\mathcal{A}_{\mathrm{CDH}}$ is any adversary with time complexity t. The CDH assumption is that the advantage of any adversary $\mathcal{A}_{\mathrm{CDH}}$ with time complexity polynomial in k is negligible.

For simplicity we consider a subgroup G, whose prime order is q and a generator is g, of a cyclic group Z_p^* where p is a prime.

A.2 Decisional Diffie-Hellman Problem

Let $\mathcal{GG}$ be a group generator which generates a group G whose prime order is q and a generator g. Let $k \in N$ be a security parameter. Consider the following experiment:

$\mathbf{Exp}^{\mathrm{DDH}}_{\mathcal{A}_{\mathrm{DDH}}}(k)$
$(G, q, g) \leftarrow \mathcal{GG}(k)$
$u_1, u_2, w \in_R [1, q]$
$U_1 \leftarrow g^{u_1}; U_2 \leftarrow g^{u_2}$
$d \stackrel{R}{\leftarrow} \{0, 1\}$
if $d = 1$ then $W \leftarrow g^{u_1 u_2}$
else $W \leftarrow g^{w}$
$d' \leftarrow \mathcal{A}_{\mathrm{DDH}}(U_1, U_2, W)$

The advantage of an adversary $\mathcal{A}_{\mathrm{DDH}}(k)$ is defined as follows:

$$Adv^{\mathrm{DDH}}_{\mathcal{A}_{\mathrm{DDH}}}(k) = 2 \cdot Pr[d = d'] - 1.$$

The advantage function is defined as follows:

$$Adv^{\mathrm{DDH}}(k,t) = \max_{\mathcal{A}}\{Adv^{\mathrm{DDH}}_{\mathcal{A}_{\mathrm{DDH}}}(k)\},$$

where $\mathcal{A}_{\mathrm{DDH}}$ is any adversary with time complexity t. We assume that the advantage of any adversary $\mathcal{A}_{\mathrm{DDH}}$ with time complexity polynomial in k is negligible.

For simplicity we consider a subgroup G, whose prime order is q and a generator is g, of a cyclic group Z_p^* where p is a prime.

A.3 Strong Unforgeability (SUF) of MAC

A MAC scheme consists of $M = (M.key, \mathsf{MAC}, \mathsf{Vrfy})$. $M.key$ generates a MAC key for the users. MAC computes a MAC for the message using the MAC key. Vrfy verifies the message-MAC pair with the MAC key and returns 1 if valid or 0 otherwise.

Let $k \in N$ be a security parameter. Let M be a MAC scheme. Consider the following experiment:

$\mathbf{Exp}^{\mathrm{SUF}}_{M,\mathcal{A}}(k)$
  $sk \leftarrow M.key(1^k)$
  $(M,\tau) \leftarrow \mathcal{A}^{\mathsf{MAC}_{sk}(\cdot)}(1^k)$
  if $\mathsf{Vrfy}_{sk}(M,\tau) = 1$ and oracle $\mathsf{MAC}_{sk}(\cdot)$
    never returned τ on input M then return 1
  else return 0

The advantage of an adversary $\mathcal{A}_{\mathrm{SUF}}(k)$ is defined as follows:

$$\mathsf{Adv}^{\mathrm{SUF}}_{M,\mathcal{A}}(k) = \Pr[\mathbf{Exp}^{\mathrm{SUF}}_{M,\mathcal{A}_{\mathrm{SUF}}}(k) = 1]$$

The advantage function of the scheme is defined as follows:

$$\mathsf{Adv}^{\mathrm{SUF}}_{M}(k,t,q_g) = \max_{\mathcal{A}}\{\mathsf{Adv}^{\mathrm{SUF}}_{M,\mathcal{A}}(k)\},$$

where $\mathcal{A}_{\mathrm{SUF}}$ is any adversary with time complexity t and making at most q_g MAC queries. The scheme M is SUF secure if the advantage of any adversary $\mathcal{A}_{\mathrm{SUF}}$ with time complexity polynomial in k is negligible.