Available online at www.sciencedirect.com
SCIENCE DIRECT®
Electronic Notes in Theoretical Computer Science
ELSEVIER Electronic Notes in Theoretical Computer Science 135 (2005) 79–94
www.elsevier.com/locate/entcs

Design and Analysis of Diffie-Hellman-Based Key Exchange Using One-time ID by SVO Logic

Kenji Imamoto[1,2]

Graduate School of Information Science and Electrical Engineering
Kyushu University
Fukuoka, Japan

Kouichi Sakurai[1,3]

Faculty of Information Science and Electrical Engineering
Kyushu University
Fukuoka, Japan

Abstract

Authenticated key exchange protocols have been developed to establish secure channel on the Internet. In this paper, we consider following attacks against an authenticated key exchange using shared secret: eavesdropping, DoS attack, replay attack, and impersonation. Besides prevention from all these attacks, efficiency is also important. In this paper, we propose a three-party authenticated key exchange protocol based on Diffie-Hellman key exchange with one-time ID, which is a user's extraordinary identity used only once [2,3]. Moreover, we analyze our proposal by SVO Logic, which is one of formal methods to analyze cryptographic protocols [7,8], and show what assumptions are needed.

Keywords: Diffie-Hellman based key exchange, SVO Logic, Pre-shared key model, One-time ID

[1] The authors would like to thank Dr. Wang Guilin and the anonymous referees for their useful comments. This research was partly supported from the grant of Secom Science and Technology Foundation, and the 21st Century COE Program 'Reconstruction of Social Infrastructure Related to Information Science and Electrical Engineering'. Also, the first author was partly supported from JSPS Research Fellowships for Young Scientists, 2004, 06737.

[2] Email:imamoto@itslab.csce.kyushu-u.ac.jp
[3] Email: sakurai@csce.kyushu-u.ac.jp

doi:10.1016/j.entcs.2005.06.003

1 Introduction

Today, information technology has developed as one of the most important techniques in the rationalization of industry. This has improved the infrastructure of industrial foundation. Namely, many resources have been changed into electronic data, and transferred via computer networks. To achieve secure and relievable electronic data exchange such as electronic commerce, it is important to provide authentication, confidentiality, integrity, and user's privacy. The subject of our study is a channel in which adversary can modify and eavesdrop on all transferred messages (e.g, the Internet, Bluetooth, radio frequency Identification (RFID), etc.). In such a communication channel, it is essential to prevent adversary from detecting user's identity [1]. Especially, we focus on pre-shared key model in which it is difficult to solve this problem [10].

In this paper, we consider the problem of authenticated key exchange with one-time ID. One-time ID [2,3] is a user's extraordinary identity, which has two properties: (1) An adversary cannot specify who is communicating even when he eavesdrops on the communication channel, but legitimate participants can specify the interlocutor; and (2) Any one-time ID can be used only once and the adversary, who does not know a long-term secret, cannot guess unused one-time ID. By substituting one-time ID for user's fixed identity (e.g., name, address), both leakage of user's identity and DoS attack can be prevented. However, because the system in this previous paper is designed to use one-time ID in two parties, it is not suitable for large system. This means if a user wants to communicate with n partners then he needs to prepare and store n shared secret keys beforehand. To overcome this shortcoming, we propose a new extensible system with one-time ID by using *Trusted Third Party (TTP)*. In this system, if a user already shared a secret with only one TTP, he can establish secure communication with any other users. We analyse the proposed protocol by *SVO Logic*, which is suitable for analysis of Diffie-Hellman based key exchange, and show it can achieve desired authentication goals. Finally, we consider a practical problem such as synchronization of one-time ID, and propose new calculation method of one-time ID to solve this problem.

2 Related Works

Almost all of the key establishment protocols using pre-shared key are based on challenge-response mechanism. The goal of challenge-response protocols is that the claimant proves its identity to the verifier by demonstrating knowledge of a secret known to be associated with that entity, without revealing the secret itself to the verifier during the protocol. This is done by providing

Table 1
Comparison of key establishment protocols using pre-shared key (TS: Time-Stamp, RN: Random Number, SN: Sequence Number, PN: Pre-chosen Number)

	Time-variant Parameters	Involvement of TTP	One-time ID	Number of Rounds
SIGNAL [2]	RN,SN,PN	No	Yes	2
P-SIGMA [3]	RN	No	Yes	3 or 4
Kerberos [4]	TS, RN	Yes	No	4
Otway-Rees [5]	RN	Yes	No	4
3PAK [6]	RN	Yes	No	5
Our proposal	RN,SN,PN	Yes	Yes	5

a response to a challenge using time-variant parameters, where the response depends on both the entity's secret and the challenge. Time-variant parameters can be divided into four types as follows: random number (RN), sequence number (SN), time-stamp (TS), and a number chosen before a protocol run (we call it pre-chosen number (PN)). In Table 1, we compare and classify the key establishment protocols based on pre-shared key ([2,3,4,5,6] and proposal of this paper), viewed from the type of time-variant parameters, the involvement of third party, the usage of one-time ID, and the total number of messages.

Besides, there are many requirements for authentication or key exchange protocol, e.g., key confirmation ([2] and [5] cannot provide this property), anonymity (Only [2,3] and this paper consider this requirement), prevention of DoS attack (especially, 3PAK [6] is fragile).

Because only a few attempts have been made on anonymous key exchange based on pre-shared key in three-party model, we propose a key exchange system using one-time ID with trusted server. This system is based on Diffie-Hellman key exchange [9] to realize *Perfect Forward Secrecy*. Perfect Forward Secrecy is a requirement to achieve following situation: Even if a long-term secret is revealed, security of other secret keys used previously can be guaranteed. When using sequence number or pre-chosen number, the synchronization problem can occur because any systems using one-time ID need to update them in each protocol run (the system using time-stamp also has similar problem). We consider this problem in section 4.

3 Preliminaries of SVO Logic

In this section, we give a brief review on the SVO logic [7,8]: the notations, rules, and axioms. However, we omit some definitions unused in our analysis such as signature or public key.

3.1 Notation

The entities who want to perform an authenticated key exchange are traditionally named *Alice* and *Bob*. The other main principal is the trusted server. Alice and Bob are assumed to share long-term keys with the server. We use the obvious symbols for Alice (A), Bob (B), the trusted server (S), and the shared keys (k_{AS}, k_{BS}, k_{AB}).

P ***believes*** X : The principal P may act as though X is true.

P ***received*** X : The principal P received a message containing X to P, who can read and repeat X.

P ***said*** X : The principal P at some time sent a message including X.

P ***says*** X : P must have said X since the beginning of current epoch.

P ***has*** X :

- Initially available to P,
- Received by P,
- Freshly generated by P,
- Constructible by P from the above.

P ***controls*** X : P has jurisdiction over X. The principal P is an authority on X and should be trusted on this matter.

$fresh(X)$: X has not been sent in any message prior to the current protocol run.

$P \overset{k}{\longleftrightarrow} Q$: P and Q may use the shared key k to communicate. k will never be discovered by any principal but P, Q, or a principal trusted by P or Q.

$PK_{\delta}(P, k)$: k is a public key-agreement key of P. A Diffie-Hellman key formed with k is shared with P.

$\{M\}_k$: Encryption of message, M, using key k. Encrypted messages are uniquely readable and verifiable as such by holders of the right keys.

$\langle X \rangle_{*P}$: This is used for messages that P doesn't know or recognize (e.g., $\{X\}_k$ where P does not know k).

$P \overset{k-}{\longleftrightarrow} Q \equiv (P \overset{k}{\longleftrightarrow} Q \wedge P\ has\ k)$

3.2 SVO Rules

SVO logic has two inference rules:

Modus Ponens : From φ and $\varphi \rightarrow \psi$ infer ψ

Necessitation : From $\vdash \varphi$ infer $\vdash P$ *believes* φ

'$\vdash$' is a metalingusitic symbol. '$\Gamma \vdash \varphi$' means that φ is deriable from the set of formulae Γ (and the axioms as started below) using the above rules. '$\vdash \varphi$' is a thorem, i.e., derivable from axioms alone without any additional assumptions.

3.3 SVO Axioms

Here, we introduce some axioms of SVO.

Belief Axioms

1. $(P$ *believes* $\varphi \wedge P$ *believes* $(\varphi \rightarrow \psi)\ ) \rightarrow P$ *believes* ψ
2. P *believes* $\varphi \rightarrow P$ *believes* $(P$ *believes* $\varphi)$

Source Association Axiom

3. $(P \overset{k}{\longleftrightarrow} Q \wedge R$ *received* $\{X$ *from* $Q\}_k) \rightarrow (Q$ *said* $X \wedge Q$ *has* $X)$

Key Agreement Axiom

4. $(PK_\delta(P, k_P) \wedge PK_\delta(Q, k_Q)) \rightarrow P \overset{F_0(k_P, k_Q)}{\longleftrightarrow} Q$

   $F_0(k, k')$ implicitly names the (Diffie-Hellman) function that combines k with k^{-1} (or, k' with k^{-1}) to form a shared key.

Receiving Axioms

5. P *received* $(X_1, \cdots, X_n) \longrightarrow P$ *received* X_i, *for* $i = 1, \cdots, n$
6. $(P$ *received* $\{X\}_{k^+} \wedge P$ *has* $k^-) \longrightarrow P$ *received* X

   Here k^+ and k^- are used to abstractly represent congnate keys, whether for symmetric or asymmetric cryptography. In the symmetric case, $k^+ = k^- = k$. In the asymmetric case, k^+ is a public key and k^- is the associated private key.

Posession Axioms

7. P *receivede* $X \longrightarrow P$ *has* X
8. P *has* $(X_1, \cdots, X_n) \longrightarrow P$ *has* X_i, *for* $i = 1, \cdots, n$
9. $(P$ *has* $X_1 \wedge \cdots \wedge P$ *has* $X_n) \longrightarrow P$ *has* $F(X_1, \cdots, X_n)$

   F is meta-notation for any function computable in practice by P, e.g., encryption with known keys.

Comprehension Axiom

10. $P\ believes\ (P\ has\ F(X)) \longrightarrow\ P\ believes\ (P\ has\ X)$

F is meta-notation for any function that is effectively one-one and such that F^+ or F^- is computable in practice by P.

Saying Axioms

11. $P\ said\ (X_1, \cdots, X_n)\ \longrightarrow\ P\ said\ X_i\ \wedge P\ has\ X_i,\ for\ i = 1, \cdots, n.$
12. $P\,says\,(X_1, \cdots, X_n)\ \longrightarrow\ (P\,said\,(X_1, \cdots, X_n)\ \wedge\ P\,says\,X_i),\ for\,i\ = 1, \cdots, n.$

Freshness Axioms

13. $fresh(X_i) \longrightarrow fresh(X_1, \cdots, X_n), for\ i = 1, \cdots, n.$
14. $fresh(X_1, \cdots, X_n) \longrightarrow fresh\,F(X_1, \cdots, X_n)$

F must genuinely depend on all component arguments. This means that it is infeasible to compute value of F without value of all the X_i.

Jurisdiction and Nonce-Verification Axioms

15. $(P\ controls\ \varphi\ \wedge\ P\ says\ \varphi)\ \longrightarrow \varphi$
16. $(fresh(X) \wedge P\ said\ X) \longrightarrow P\ says\ X$

Symmetric Goodness Axiom

17. $P \overset{k}{\longleftrightarrow} Q \equiv Q \overset{k}{\longleftrightarrow} P$

3.4 Authentication Goals

We list six generic goals that our protocols are desired to meet.

G1. Ping Authentication captures situations where a principal P wants to know whether an interlocutor Q is alive. It is expressed as "P *believes* Q *says* X".

G2. Entity Authentication further requires that P's interlocutor Q said something relevant to their present conversation. Given some information Y_P known to be fresh to P, entity authentication mandates that Q recently sent a message $F(X, Y_P)$ from which it is manifest that Q has seen Y_P and has processed it. It is expressed as "P *believes* $(Q\ says\ F(X, Y_P) \wedge fresh(Y_P))$".

G3. Secure key establishment indicates that a principal P believes that he has a good key k to communicate with a counterpart Q. Given the above notion of unconfirmed secret, this is expressed as "P *believes* $P \overset{k-}{\longleftrightarrow} Q$".

G4. Key freshness simply requires that a principal P believes a key k to be fresh, that is, "P *believes* $fresh(k)$".

G5. Mutual understanding of shared keys applies to situations where a principal P can establish that an interlocutor Q has sent a key k as an unconfirmed secret between the two of them (from Q's point of view). This is formalized by "P *believes* Q *says* $(Q \overset{k-}{\longleftrightarrow} P)$".

G6. Key confirmation is intended to describe scenarios in which a principal P believes that an interlocutor Q has proved to have received and successfully proccessed a previously unconfirmed secret key k between the two of them. It is expressed as "P *believes* $(P \overset{k-}{\longleftrightarrow} Q \ \wedge \ Q$ *says* $F(k))$".

Above authentication goals are listed in [7,8]. In addition to them, we further consider the following requirements.

Anonymity: An adversary could not guess who is communicating even if he eavesdropped on transferred messages.

Scalability: Even if any principals want to perform key exchange and mutual authentication with a lot of different interlocutor, the cost, which is needed to establish the system, is not directly proportional to the number of partners.

4 Proposed System

In the previous one-time ID system [2], secret key is assumed to have been already shared between Alice and Bob before a key exchange begining. Because of this assumption, if a principal wants to communicate with other user, he needs to prepare shared secrets with all partners beforehand. Therefore, this system cannot achieve good scalability. To provide high scalability for a system with one-time ID, we propose a new system that uses trusted server. In the proposed system, a secret key is assumed to have been already shared between a user and a server beforehand, and the user can communicate securely with any other users.

We design a key agreement protocol, and reason, by using SVO Logic, that it can achieve authentication goals, which are defined in section 3.4, from both Alice and Bob's views. Furthermore, we consider one-time ID synchronization problem and computation methods of one-time ID, and propose the solution for this problem.

4.1 Protocol

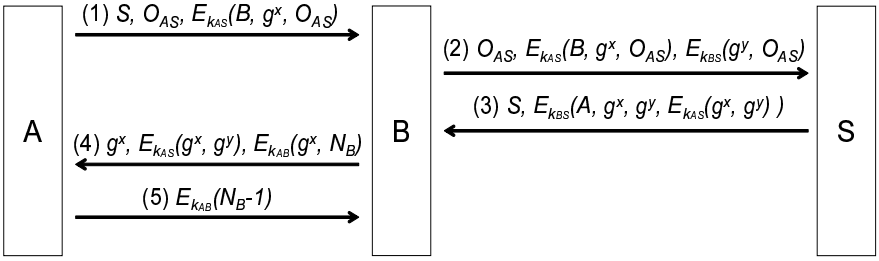
(i) $A \rightarrow B$: $S, O_{AS}, \{B, g^x, O_{AS}\}_{k_{AS}}$

(ii) $B \rightarrow S$: $O_{AS}, \{B, g^x, O_{AS}\}_{k_{AS}}, \{g^y, O_{AS}\}_{k_{BS}}$

(iii) $S \rightarrow B$: $S, \{A, g^x, g^y, \{g^x, g^y\}_{k_{AS}}\}_{k_{BS}}$

(iv) $B \rightarrow A$: $g^x, \{g^x, g^y\}_{k_{AS}}, \{g^x, N_b\}_{k_{AB}}$

(v) $A \rightarrow B$: $\{N_b - 1\}_{k_{AB}}$



Here, O_{AS} denotes an one-time ID for current session between Alice and server, and N_b denotes a nonce generated by Bob. Moreover, x, (y) is Alice's (Bob's) private key for Diffie-Hellman key exchange, and g^x (g^y) denotes the corresponding Diffie-Hellman public key. O_{AS} is calculated and shared as follows. It is assumed that Alice and trusted server have a synchronized sequence number i, which increases one by one in each protocol run. Each principal (that is, Alice and server) substitutes i and a shared secret for a function which cannot be calculated inversly by adversary (e.g., one-way hash function). Then, they can use this function's output as Alice's one-time ID in i session. By receiving one-time ID calculated beforehand, the server can know who the interlocutor is. Consequently, adversary cannot guess who is communicating even if he eavesdrops on one-time ID, but for server, it is the same information as Alice's fixed identity. Furthermore, one-time ID changes in every session because of using sequence number i, and adversary cannot guess unused one-time ID.

In section 4.4, we consider a calculation method of one-time ID, O_{AS}, for the proposed system.

4.2 Analysis using SVO approach

Our approach is based on the analysis of [7,8]. The protocol analysis steps are as follows: (1) Write assumptions about initial state, (2) Annotate protocol, (3) Assert comprehensions of received messages, (4) Assert interpretations of comprehended messages, and (5) Use the logic to derive beliefs held by protocol principals.

4.2.1 Initial State Assumptions

Here, we present initial state assumptions of our proposal using SVO.

P1. *A believes* $A \overset{k_{AS}}{\longleftrightarrow} S$

P2. *B believes* $B \overset{k_{BS}}{\longleftrightarrow} S$

P3. *A believes* $PK_\delta(A, g^x)$

P4. *B believes* $PK_\delta(B, g^y)$

P5. *A believes* $fresh(g^x)$

P6. *B believes* $fresh(g^y)$

P7. *A believes A has* (g^x, x)

P8. *B believes B has* (g^y, y)

P9. *S believes A controls* $PK_\delta(A, k_a)$

P10. *S believes B controls* $PK_\delta(B, k_b)$

P11. *A believes* $A \overset{O_{AS}}{\longleftrightarrow} S$

P12. *A believes* $fresh(O_{AS})$

P13. *S believes* $fresh(O_{AS})$

P14. *B believes* $fresh(N_b)$

P1 and P2 denote each principal shares a long-term secret with the server beforehand. P3 and P4 denote each principal knows own Diffie-Hellman public key. P5 and P6 denote each principal knows own Diffie-Hellman public key is fresh. P7 and P8 denote each principal knows own Diffie-Hellman private key. P9 and P10 denote the server believes that each principal controls their own Diffie-Hellman public key. P11 denotes Alice and the server share one-time ID used in the session beforehand. P12 and P13 denote Alice and the server believe that one-time ID is fresh. These assumptions are derived from the property of one-time ID, that is, one-time ID can be used only once. P14 denotes Bob believes his nonce is fresh.

4.2.2 Received Message Assumptions

In this step, for each message "$P \rightarrow Q : M$" of the proposed protocol, we assert "*Q received M*".

P15. *B received* $(S, O_{AS}, \{B, g^x, O_{AS}\}_{k_{AS}})$

P16. *S received* $(O_{AS}, \{B, g^x, O_{AS}\}_{k_{AS}}, \{g^y, O_{AS}\}_{k_{BS}})$

P17. *B received* $(S, \{A, g^x, g^y, \{g^x, g^y\}_{k_{AS}}\}_{k_{BS}})$

P18. *A received* $(g^x, \{g^x, g^y\}_{k_{AS}}, \{g^x, N_b\}_{k_{AB}})$

P19. *B received* $\{N_b - 1\}_{k_{AB}}$

4.2.3 Comprehension Assumptions

In this step, we express that which a principal comprehends of a received message.

P20. B *believes* B *received* $(S, \langle O_{AS} \rangle_{*B}, \langle \{B, g^x, O_{AS}\}_{K_{AS}} \rangle_{*B})$

P21. S *believes* S *received* $(O_{AS}, \{B, \langle g^x \rangle_{*S}, O_{AS}\}_{k_{AS}}, \{\langle g^y \rangle_{*S}, O_{AS}\}_{k_{BS}})$

P22. B *believes* B *received* $(S, \{A, \langle g^x \rangle_{*B}, g^y, \langle \{g^x, g^y\}_{k_{AS}} \rangle_{*B}\}_{k_{BS}})$

P23. A *believes* A *received* $(g^x, \{g^x, \langle g^y \rangle_{*A}\}_{k_{AS}}, \{g^x, \langle N_b \rangle_{*A}\ from\ B\}_{\langle k_{AB} \rangle_{*A}})$

P24. B *believes* B *received* $\{N_b - 1\}_{\langle k_{AB} \rangle_{*B}}$

4.2.4 Interpretation Assumptions

In this step, we are asserting how a principal interprets a received message (as that principal understands it).

P25. S *believes* S *received* $(O_{AS}, \{B, \langle g^x \rangle_{*S}, O_{AS}\}_{k_{AS}}, \{\langle g^y \rangle_{*S}, O_{AS}\}_{k_{BS}})$
$\longrightarrow$ S *believes* S *received* $(\{B, PK_\delta(A, g^x), O_{AS}\}_{k_{AS}}, \{PK_\delta(B, g^y), O_{AS}\}_{k_{BS}})$

P26. B *believes* S *says* $(A, \langle g^x \rangle_{*B}, g^y) \longrightarrow$ B *believes* $PK_\delta(A, g^x)$

P27. A *believes* S *says* $(g^x, \langle g^y \rangle_{*A}) \longrightarrow$ A *believes* $PK_\delta(B, g^y)$

P28. $(A$ *believes* A *received* $(\{g^x, \langle N_b \rangle_{*A}\}_{\langle k_{AB} \rangle_{*A}})) \wedge (A$ *believes* $A \overset{\langle k_{AB} \rangle_{*A}}{\longleftrightarrow} B)$
$\longrightarrow$ A *believes* A *received* $(\{g^x, \langle N_b \rangle_{*A}, A \overset{\langle k_{AB} \rangle_{*A}}{\longleftrightarrow} B\}_{\langle k_{AB} \rangle_{*A}})$

P29. $(B$ *believes* B *received* $\{N_b - 1\}_{\langle k_{AB} \rangle_{*B}}) \wedge (B$ *believes* $A \overset{\langle k_{AB} \rangle_{*B}}{\longleftrightarrow} B)$
$\longrightarrow$ B *believes* B *received* $\{N_b - 1, A \overset{\langle k_{AB} \rangle_{*B}}{\longleftrightarrow} B\}_{\langle k_{AB} \rangle_{*B}})$

4.2.5 Derivations

Here, we derive the beliefs each principal can obtain in proposed protocol. Then, we analyse which authentication goal can be achieved.

- **For Alice**

(i) A *believes* A *received* $(\{g^x, \langle g^y \rangle_{*A}\}_{k_{AS}})$
by Receiving Axioms, P23

(ii) A *believes* S *says* $(g^x, \langle g^y \rangle_{*A})$
by 1, Source Association Axiom, P5, Freshness Axioms, Nonce-Verification Axiom

(iii) *A believes* $PK_\delta(B, g^y)$
by Modus Ponens using 2 and P27

(iv) *A believes* $A \overset{\langle k_{AB}\rangle_{*A}}{\longleftrightarrow} B$
by 3, P3, Key Agreement Axiom (where $k_{AB} = F_0(g^x, \langle g^y\rangle_{*A})$)

(v) *A believes A has* k_{AB}
by 1, Receiving Axioms, Posession Axioms (where $k_{AB} = F_0(g^x, \langle g^y\rangle_{*A})$)

(vi) *A believes* $A \overset{\langle k_{AB}-\rangle_{*A}}{\longleftrightarrow} B$
by 4, 5, and def. of $P \overset{k-}{\longleftrightarrow} Q$

(vii) *A believes* $fresh(k_{AB})$
by P5, Freshness Axioms, Modus Ponens (where $k_{AB} = F_0(g^x, \langle g^y\rangle_{*A})$)

(viii) *A believes B said* $(\{g^x, \langle N_b\rangle_{*A}, A \overset{\langle k_{AB}\rangle_{*A}}{\longleftrightarrow} B\}_{\langle k_{AB}\rangle_{*A}})$
by P23, Receiving Axioms, P28, 4, Source Association Axioms, Belief Axixoms, Modus Ponens

(ix) *A believes B has* k_{AB}
by 8, Saying Axioms

(x) *A believes B says* $(\{g^x, \langle N_b\rangle_{*A}, A \overset{\langle k_{AB}\rangle_{*A}}{\longleftrightarrow} B\}_{\langle k_{AB}\rangle_{*A}})$
by P5, 8, Freshness Axioms, Nonce-Verification Axiom, Modus Ponens

From above analysis, we can derive following conclusion. Both G1 and G2 for Bob are derived in line 10. Moreover, both G1 and G2 for the server are derived in line 2. For Bob, G3 is derived in line 6, G4 in line 7, G5 in line 9 and 10, and G6 in line 6 and 10.

• **For Bob**

(i) *B believes B received* $(\{A, \langle g^x\rangle_{*B}, g^y\}_{k_{BS}})$
by Receiving Axioms, P22

(ii) *B believes S says* $(A, \langle g^x\rangle_{*B}, g^y)$
by 1, Source Association Axiom, P6, Freshness Axioms, Nonce-Verification Axiom

(iii) *B believes* $PK_\delta(A, g^x)$
by Modus Ponens using 2 and P26

(iv) *B believes* $A \overset{\langle k_{AB}\rangle_{*B}}{\longleftrightarrow} B$
by 3, P4, Key Agreement Axiom (where $k_{AB} = F_0(\langle g^x\rangle_{*B}, g^y)$)

(v) *B believes B has* k_{AB}
by 1, Receiving Axioms, Posession Axioms (where $k_{AB} = F_0(\langle g^x\rangle_{*B}, g^y)$)

(vi) *B believes* $B \overset{\langle k_{AB}-\rangle_{*B}}{\longleftrightarrow} A$
by 4, 5, and def. of $P \overset{k-}{\longleftrightarrow} Q$

(vii) B *believes* $fresh(k_{AB})$
by P6, Freshness Axioms, Modus Ponens (where $k_{AB} = F_0(\langle g^x \rangle_{*B}, g^y)$)

(viii) B *believes* A *said* $\{N_b - 1, A \overset{\langle k_{AB} \rangle_{*B}}{\longleftrightarrow} B\}_{\langle k_{AB} \rangle_{*B}}$
by P24, Receiving Axioms, P29, 4, Source Association Axiom, Belief Axixoms, Modus Ponens

(ix) B *believes* A *has* k_{AB}
by 8, Saying Axioms

(x) B *believes* A *says* $\{N_b - 1, A \overset{\langle k_{AB} \rangle_{*B}}{\longleftrightarrow} B\}_{\langle k_{AB} \rangle_{*B}}$
by P6, 8, Freshness Axioms, Nonce-Verification Axiom, Modus Ponens

From above analysis, we can derive following conclusion as well as Alice. Both G1 and G2 for Alice are derived in line 10. Moreover, both G1 and G2 for the server are derived in line 2. For Alice, G3 is derived in line 6, G4 in line 7, G5 in line 9 and 10, and G6 in line 6 and 10.

• **For Server**

(i) S *believes* S *received* $(\{B, PK_\delta(A, g^x), O_{AS}\}_{k_{AS}}, \{PK_\delta(B, g^y), O_{AS}\}_{k_{BS}})$
by Modus Ponens using P21 and P25

(ii) S *believes* A *said* $(\{B, PK_\delta(A, g^x), O_{AS}, \}_{k_{AS}})$
by 1, Receiving Axioms, P1, Source Association Axiom

(iii) S *believes* A *says*$(\{B, PK_\delta(A, g^x), O_{AS}, \}_{k_{AS}}$
by 2, P13, Freshness Axioms, Nonce-Verification Axiom

(iv) S *believes* $PK_\delta(A, g^x)$
by 3, Receiving Axioms, P9, Jurisdiction Axiom

(v) S *believes* B *said* $\{PK_\delta(B, g^y), O_{AS}\}_{k_{BS}}$
by 1, Receiving Axioms, P2, Source Association Axiom

(vi) S *believes* B *says* $\{PK_\delta(B, g^y), O_{AS}\}_{k_{BS}}$
by 5, P13, Freshness Axioms, Nonce-Verification Axiom

(vii) S *believes* $PK_\delta(B, g^y)$
by 6, Receiving Axioms, P10, Jurisdiction Axiom

G1 and G2 for both Alice and Bob are derived in line 3 and 6. By this goal, the server can update Alice's one-time ID. Remained authentication goals do not need to be realized because each principal does not share new session key with the server.

4.3 Analysis on Anonymity

The first message of the proposed protocol includes the server's identity and one-time ID, O_{AS}. An adversary who eavesdrops on this message can know only who the server is. In this step, Bob also cannot know who the interlocutor is. Moreover, because second message uses only one-time ID as identity, the server can understand that Alice is the initiator of the protocol, but the adversary cannot. However, both first and second messages include the same information, such as O_{AS} and $\{B, g^x, O_{AS}\}$, therefore, an eavesdropper can guess that first and second messages are performed in the same session's run. Third message includes the server's identity and new encrypted information. Hence, Bob can decrypt the encrypted message by k_{BS}, but the adversary cannot obtain any information except for the server's identity. That is, the adversary cannot guess whether third message is relational with first or second message or not. By seeing a plaintext of Alice's Diffie-Hellman public key of fourth message, Alice can realize this message was produced in the session she initiated. Namely, g^x can be used as session ID for her. The adversary who eavesdrops on fourth message can obtain Alice's Diffie-Hellman public key, but cannot relate with any other information.

According to the above analysis, we can recognize that our protocol can achieve anonymity of both Alice and Bob. Any eavesdropper can only know the server's identity and the fact that first and second messages are in the same protocol execution. This paper assumes that the leakage of the server's identity is not an issue. However, there might be the situations where it becomes significant problem. For example, if a server serves for limited users and the purpose of using the server was very restricted, then a message with its identity might have much meaning. This problem is our future work.

4.4 Analysis on Calculation of One-time ID

Here, we analyse the calculation of one-time ID, and propose a solution for synchronization problem of one-time ID.

One-time ID needs two properties. One is that one-time ID can be used only once, and the other is that an adversary, who does not know secret information, cannot guess unused one-time ID. Hence, there are two calculation methods of one-time ID as follows.

(i) $O_{ASi} := h(SO_{AS}, i)$

(ii) $O_{ASi} := \{A, SO_{AS}, i\}_{k_S}$

O_{ASi} denotes an one-time ID used in ith protocol run. Moreover, SO_{AS} is a secret for one-time ID between Alice and the server, and k_S is the server's

public key. This can be denoted as $'PK_{\psi}(S, k_S)'$ by SVO Logic. And $h(\cdot)$ denotes one-way hash function. Both methods can calculate different value in every session by using sequence number i, and the adversary cannot guess unused one-time ID unless SO_{AS} is revealed. Thus, both methods can satisfy the requirements for one-time ID. Incidentally, the systems proposed in [2,3] use first method.

By preparing one-time ID beforehand, although it is the same information as Alice's fixed identity for server, the adversary cannot understand who the sender is. However, this method causes the synchronization problem of one-time ID. For example, suppose Alice and the server share 5 as a sequent number. Then, Alice performs two protocols at the same time and sends O_{AS5} and O_{AS6} to the server. If the server can obtain them in order (i.e., $O_{AS5} \rightarrow O_{AS6}$), problem does not occur. But otherwise (i.e., $O_{AS6} \rightarrow O_{AS5}$), the server might discard O_{AS6} because he does not still prepare. This problem also happens when O_{AS5} is missing. We call this problem the synchronization problem of one-time ID.

The differences of above calculation methods are as follows. First method needs few computation because it uses only hash function. However, if the synchronization problem of one-time ID occurs, the server is difficult to solve this problem. On the other hand, second message needs more computation because it uses the server's public key. Moreover, new assumption is needed, that is, *A and B believes* $PK_{\psi}(S, k_S)$. In return for that, if the synchronization problem occurs, the server can solve it by decrypting O_{ASi} and seeing Alice's identity included in it.

Two-party model has the synchronization problem as well as three-party model, but it is more likely to occur this problem in three-party model because Bob intervenes. Therefore, second method is more suitable for our proposed system.

4.5 Differences of One-time ID's goals between P-SIGMA[3] and Our Proposal

Our proposal needs an assumption that S believes O_{AS} is fresh (i.e., **P13**) to derive each user's ping authentication (**G1**) and entity authentication (**G2**) by S, while P-SIGMA can derive all requirements introduced in Section 3.4 without freshness of one-time ID.

From the viewpoint of anonymity of both A and B, one-time ID of P-SIGMA has to be fresh. On the other hand, the freshness of one-time ID is needed to provide A's anonymity in our proposal.

As described above, each protocol has different security goals achieved by freshness of one-time ID (see Table 2). However, since a formal analysis of

Table 2
Security goals achieved by freshness of One-time ID

	A	B	S
P-SIGMA [3]	Anonymity	Anonymity	S does not exist (Two party model)
Proposal	G1(by S), G2(by S), Anonymity	G1(by S), G2(by S)	Nothing

anonymity has never developed so far, we cannot derive the necessity of one-time ID's freshness in P-SIGMA. It is our future work to analyze anonymity in formal way.

5 Conclusion and Future Work

In this paper, we presented a scalable authenticated Diffie-Hellman based key agreement protocol using trusted third party, which uses one-time ID to achieve anonymity in application layer. We analysed this protocol using SVO Logic [7,8], and showed that our protocol can achieve considered authentication goals. Furthermore, we considered one-time ID synchronization problem and computation methods of one-time ID, and proposed the solution for this problem.

References

[1] R. Perlman, C. Kaufman, Analysis of IPSec Key Exchange Standard, WETICE2001, 2001.

[2] K. Imamoto, K. Sakurai, Notes on Dynamic Information Management for Authenticated Key Exchange, ISEC, March 2003.

[3] H. Krawczyk, The IKE-SIGMA Protocol, Internet Draft, Nov 2001. http://www.ee.technion.ac.il/ hugo/draft-krawczyk-ipsec-ike-sigma-00.txt

[4] C. Neuman and T. Ts'o, Kerberos: an authentication service for computer networks, IEEE communications, 1994.

[5] D. Otway and O. Rees. Efficient and timely mutual authentication. Operating Systems Review, 21(1):8–10, 1987.

[6] C. Lin, H. Wen, T. Hwang, and H. Sun, Provably Secure Three-Party Password-Authenticated Key Exchange, IEICE TRANS. Fundamentals, Vol.E85.A, No.1 2002.

[7] P. Syverson and P. C. van Oorschot. A Unified Cryptographic Protocol Logic. NRL CHAOS Report, 5540-227, 1996.

[8] P. Syverson and I. Cervesato. The Logic of Authentication Protocols. FOSAD'00, LNCS2171, pp.63-137, 2001.

[9] W. Diffie and M. E. Hellman. New directions in cryptography. IEEE Trans. Inform. Theory, IT-22:644–654, Nov 1976.

[10] D. Harkins and D. Carrel. The Internet Key Exchange (IKE), RFC2409, 1998. http://www.ietf.org/rfc/rfc2409.txt

[11] V. Shoup, On formal models for secure key exchange, IBM Research Report RZ3120, April 1999.

[12] V. Boyko, P. MacKenzie, and S. Patel, Provably Secure Password-Authenticated Key Exchange Using Diffie-Hellman, EUROCRYPT'2000, 2000.

[13] H. Krawczyk, SIGMA: the 'SIGn-and-MAc' Approach to Authenticated Diffie-Hellman and its Use in the IKE Protocols, 2003.