I. J. Computer Network and Information Security, **2016, 5, 18-24**
Published Online May 2016 in MECS (http://www.mecs-press.org/)
DOI: 10.5815/ijcnis.2016.05.03

A Note on Group Authentication Schemes

Mohsen Pourpouneh
Department of Mathematical Sciences, Sharif University of Technology, Tehran, Iran
E-mail: m_pourpouneh@mehr.sharif.ir

Rasoul Ramezanian and Afshin Zarei
Department of Mathematical Sciences, Ferdowsi University of Mashhad, Mashhad, Iran
Department of Mathematical Sciences, Isfahan University of Technology, Isfahan, Iran
E-mail: {rramezanian@um.ac.ir, afshin.zarei@math.iut.ac.ir}

Abstract—In literature, there are many different forms of group authentication in conference key establishment protocols. The agents participating in a group need to authenticate each other in order to become assure that every agents that has access to the group key is an eligible member. In this paper, we informally classify different group authentication schemes, based on how the agents authenticate each other and provide examples of each class. We then improve one of the well-known key establishment protocol to an authenticated version according so that it meets one of our notions of group authentication.

Index Terms—Key agreement, Group authentication, Conference Key.

I. Introduction

In order to ensure secure communication, before communicating the main protocol messages, a key establishment protocol will distribute session keys to all participants. This needs to provide confidentiality and authentication for the session keys. Confidentiality ensures the sender that the message can be read only by an intended receiver and authentication ensures the receiver that the message is sent by a specified sender and the message is not altered by another party.

There are different definition for authentication in literature. Gollmann [1] has put forward a number of different options for what could be meant by authentication.

Syverson and van Oorschot [2], identify what they term six `Generic Formal Goals'. They stated that, it is not intended as a 'definitive list of the goals that a key agreement or key distribution protocol should meet'.

Menezes *et al*. [3] give a more comprehensive definition as follows: "Entity authentication is the process whereby one party is assured (through acquisition of corroborative evidence) of the identity of a second party involved in a protocol, and that the second has actually participated (i.e., is active at, or immediately prior to, the time the evidence is acquired)."

In 1997 [4], Lowe suggested that the appropriate authentication requirement will depend upon the use to which the protocol is put, and proposed an "authentication hierarchy".

In this paper, we aim to discuss group authentication in conference (group) key establishment protocols involving more than two party, where a group key is needed to be shared for all group members. Group key establishment protocols fall in to two main categories: Key transfer protocols and Key agreement protocols [5].

Most group key agreement protocols are generalization of the Diffie-Hellman [6] key agreement protocol, such as, Ingemarsson *et al*. [7], Steer *et al*. [8], Burmester and Desmedt [9], and Steiner *et al*. [10]. In

1996, Steiner *et al*. [10], proposed an extension of DH and in 2001, an authenticated version of it was proposed and has proved to be secure [11]. In 2007, Bresson *et al*. [12] constructed a generic authenticated group DH Key exchange. Also, in 2007, Katz and Yung [13] proposed the first constant-round and fully scalable group DH protocol.

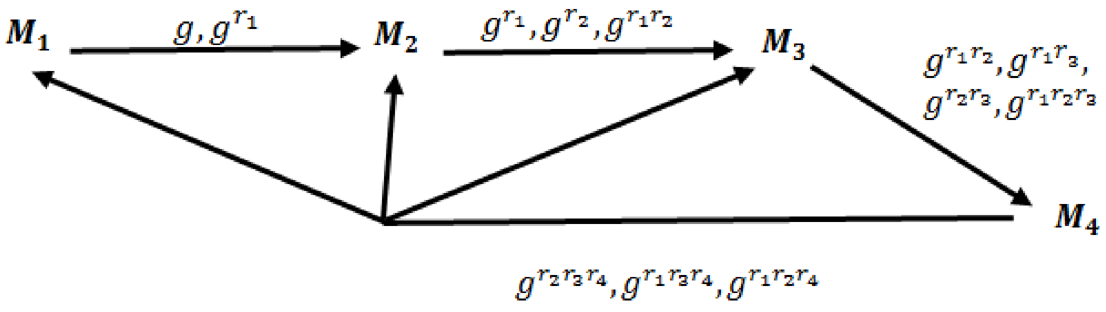
The generalization of what authentication is in a group key agreement protocol is different from two party protocols. The potential problem is that in a two-party protocol, authentication means that the intended communication partner are assured about each other identity. But in a group it may be quite difficult to know who the communication partners in the group are.

Saeednia and Safavi-Naini [14] suggest that for a key establishment protocol every principal should be sure that either the same key is shared with all other principals or that no two principals share the same key. In particular they consider that a situation in which a session key is established by a subset of the intended set of principles, but is not known to other members of the set, which is a major threat. In contrast Ateniese *et al*. [15], [16] note that key confirmation for all users requires' at the very least, one round of simultaneous broadcasts', implying that it may be too costly to justify.

In [17], key confirmation is also defined as "Let U be a set of principals with $U_i, U_j \in U$. Key confirmation of U_j to U_i is provided if U_i has assurance that key K is a good key to communicate with every principal in U, and principal U_j has received K. Complete key confirmation is provided to U_i if key confirmation of U_j is provided to U_i for all $U_j \in U \setminus \{U_i\}$."

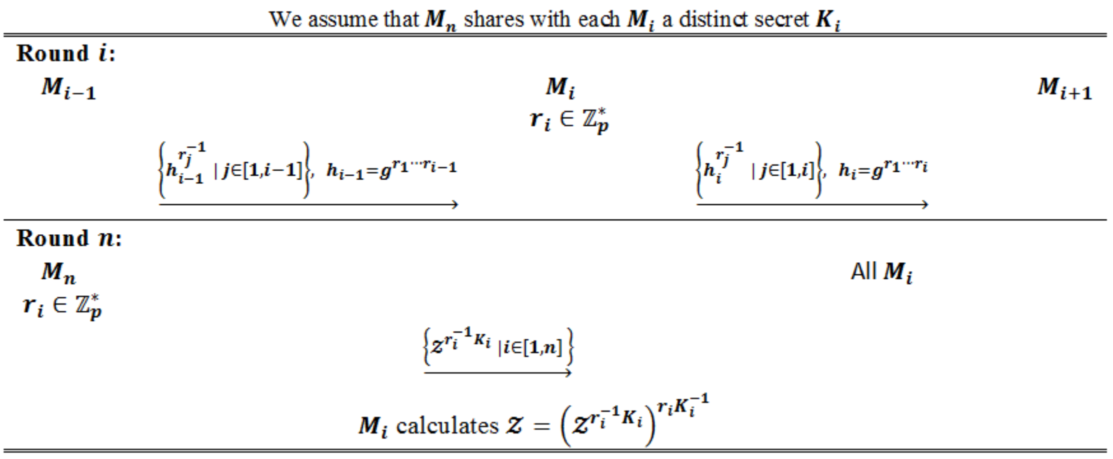
In this paper we informally introduce a hierarchy of different group authentication schemes and also provide examples for each of these schemes. We also improve the key establishment protocol proposed in [10] to an authenticated version.

The rest of this paper is organized as follows: First, in section *II* we define different group authentication schemes. In section *III*, we review "The Skinny TRee (STR)" protocol proposed by Steer *et al.* After that in section *IV* we propose an authenticated version of STR group key establishment protocol, and in section *V* we prove its correctness using Scyther. Finally, the conclusion is drawn is section *VI*.



M_1
g, g^{r_1}
M_2
$g^{r_1}, g^{r_2}, g^{r_1 r_2}$
M_3
$g^{r_1 r_2}, g^{r_1 r_3}, g^{r_2 r_3}, g^{r_1 r_2 r_3}$
M_4
$g^{r_2 r_3 r_4}, g^{r_1 r_3 r_4}, g^{r_1 r_2 r_4}$

Fig.1. Group Diffie-Hellman (GDH.2)



We assume that M_n shares with each M_i a distinct secret K_i
Round i:
M_{i-1}
M_i
$r_i \in \mathbb{Z}_p^*$
M_{i+1}
$\left\{h_{i-1}^{r_j^{-1}} \mid j\in[1,i-1]\right\}, h_{i-1}=g^{r_1 \cdots r_{i-1}}$
$\left\{h_i^{r_j^{-1}} \mid j\in[1,i]\right\}, h_i=g^{r_1 \cdots r_i}$
Round n:
M_n
$r_i \in \mathbb{Z}_p^*$
All M_i
$\left\{\mathcal{Z}^{r_i^{-1} K_i} \mid i\in[1,n]\right\}$
M_i calculates $\mathcal{Z} = \left(\mathcal{Z}^{r_i^{-1} K_i}\right)^{r_i K_i^{-1}}$

Fig.2. Authenticated GDH.2 Protocol.

II. Different Group Authentication Schemes

There exist many forms of group authentication schemes in the literature. In general in two party-protocols, authentication means to assure the receiver that the message is sent by a specified sender and the message is not altered by a third party. In its most basic form, authentication is a simple statement about the existence of communication partners. We can roughly say that *group authentication* means that to assure every legal party that there exists no illegal party in the group, and all expected legal parties are alive in the group, and agree on some specific values.

In this section we propose a hierarchy of *group authentication* schemes and also provide some examples for each scheme.

1) Authentication based on the Server

These types of protocols are based on a trusted server. The role of the server is authenticating and verifying the identity of each group member. In this way, everyone in the group would believe that there is no illegal principal in the group. Ateniese proposed a method to extend Group Diffie-Hellman key agreement. They proposed three version of it namely GDH.1, GDH.2, and GDH.3 [19]. After that they proposed an authenticated version of GDH.2 [15], [16] protocol to provide authenticated group key agreement. At the first we illustrate GDH.2 protocol, then explain the authenticated version of this protocol, AGDH.2.

Initialization: Let p be a prime and q a prime divisor of $p-1$. Let G be the unique cyclic subgroup of $\mathbb{Z}_p^*$ of order q, and let g be generator of G. Let $M = \{M1, M2, \ldots, Mn\}$ be the set of users who want to share a key $\mathcal{Z}$.

The protocol consists of two phase. In the first phase M_i receives i values from M_{i-1}. one of these values is the principal value $h_{i-1} = g^{r_1 r_2 \ldots r_{i-1}}$ while the remaining $i-1$ values consist of m_{i-1} with one of the exponents $r_1, r_2, \ldots, r_{i-1}$ `missing'. Initially M_1 starts Phase 1 by sending g^{r_1} and g to M_2. On receiving this message M_i raises all received values to its exponent r_i to form i new message components and also includes the principal value h_{i-1} in the message sent on to M_{i+1}.

The second phase of GDH.2 consists of a single message broadcast by M_n, which includes all the partial calculations necessary for each other M_i to find $\mathcal{Z}$ with a single exponentiation using r_i. On receiving the final message in the first phase, M_n can calculate the shared secret from the principal value h_{n-1} as $\mathcal{Z} = h_{n-1}^{r} = g^{r_1 r_2 \dots r_n}$. The final broadcast message can be calculated by M_n by raising each of the other $n-1$ components of its received message to its secret exponent r_n . An example of GDH.2 for four users is shown in fig 1.

In Authenticated GDH.2 protocol fig 2, M_n is assumed as a trusted server. M_n shares a unique secret with each member. In this way, only M_n authenticates other members of the group, and every other principal $M_{i \in \{1,2,\dots,n-1\}}$ believes that there is no illegal member in the group based on his trust on M_n.

2) Authentication all Member of the Group

In this scheme, every group member verifies the identity of all other members in the group at once. In other words, every user can verify the whole group and if there are some illegal users in the group we cannot find them; and the only thing we know is that there are some other users in the group. For example, Klein [20] proposed a protocol which is based on this type of authentication. In their protocol each message protected by digital signature of the sender, which also include a unique identifier for the protocol run. Also each intermediate value is calculated multiple times with the aim of detecting and recovering from errors caused by principals deviating from the protocols. Their protocol is as follow:

Suppose $M = \{M_1, M_2, \dots, M_n\}$ be the set of users who want to share a key K. There are $n-1$ rounds to the protocol during which messages are broadcast, via a write-only bulletin board, to all other principals. Messages all consist of triples of the form $(M_i, A, Sig_{M_i}(PID, g^A))$, where PID is an identifier, A is a set of users in M, and A is the set of exponent input by the users in A. For example, a particular message sent by M_2, might be $(M_2, \{M_3, M_2\}, \ Sig_{M_2}(PID, g^{r_2 r_3}))$.

Round 1: M_i chooses random ephemeral input x_i and broadcasts the triple $(M_i, \{M_i\}, Sig_{M_i}(PID, g^{xi}))$.

Round $(2 \le j \le n-1)$ **:** The following steps are taken.

a) M_i collects all messages from round $j-1$ that exclude x_i in the exponents. For each of these M_i, raises the g^A value to the exponent x_i, and form a new message triple (adding its identity to the set A) and broadcasts the result.
b) Each message with the same user set A is then compared. If there are any differences then the recovery process is invoked.
c) Once these are resolved, all duplicates are deleted and j is incremented.

The recovery protocol has as input a set of principals A and a set of message triples using set A but with differing values of g^A. All principals belonging to A are required to reveal their secret input x_i which allows checking against the signed inputs from the previous round. Principals found to have been cheated by a majority of the other principals are expelled from the conference. Those principals who have not cheated choose new x_i values and reconstruct their inputs for the current round.

For another example, propose a protocol based on the RSA cryptography. Suppose that $N = pq$ (p, q are tow distinct prime numbers) be an RSA modulo. Let $M = \{M_1, M_2, \dots, M_n\}$ be the set of users who want to share a key K . and (e_i, d_i) be public-private key of M_i and similar to GDH.2 M_n is assumed as a trusted server.

In this protocol Fig. 3, look like to process of GDH.2 users shared key with using their private key, d_i, instead of their random value, r_i . Therefore a shared key $K = g^{d_1 d_2 \dots d_n}$, in which g is the generator of group G.

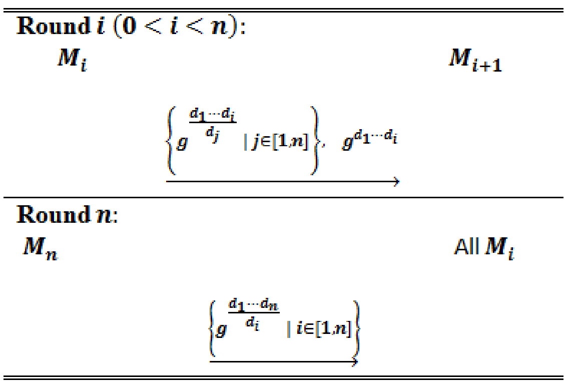

Round i $(0 < i < n)$:
M_i
M_{i+1}
$\left\{ g^{\frac{d_1 \cdots d_i}{d_j}} \mid j \in [1,n] \right\}, \ g^{d_1 \cdots d_i}$
Round n:
M_n
All M_i
$\left\{ g^{\frac{d_1 \cdots d_n}{d_i}} \mid i \in [1,n] \right\}$

Fig.3. Protocol based on RSA

For authentication every user can easily check that $K^{e_1 e_2 \dots e_n} = g$ and deduce that the other parties are the intended communication partners.

3) Authentication with dynamic trust

In this scheme, at first, it is assumed that the first member of the group is a trusted party, then he/she authenticates the second member of the group. If the first member of the group authenticates the second member, then the second member authenticates the third member of the group, and the protocol proceeds in this way.

In the next section we propose an authenticated version of STR protocol that meets this authentication scheme.

4) Complete Authentication:

In both above aforementioned schemes, the authentication is obtained based on having a trusted party. But in practice, it is a very ideal assumption to have a trusted party. In complete authentication schemes, each two member of the group authenticates each other. In comparison to **authentication all member of the group** scheme, in this case if there are some illegal users in the group we can detect them since, we are authenticating each member one by one.

Let $\{M_1, M_2, \dots, M_n\}$ be the group who are about to share a secret key. This group satisfies complete authentication if every pair $(M_i, M_j)_{0 \le i \ne j \le n}$ of the group members authenticate each other.

Ateniese *et al.* [15] proposed group Diffie-Hellman with complete key authentication (SA-GDH.2).

5) Common Authentication:

In complete authentication everyone authenticates separately all members of the group, but he/she has no information about the process of authentication of other members of the group. Common authentication satisfies this property. In other words, if $M = \{M_1, M_2, \dots, M_n\}$ is the group, then every member $M \in M$, knows that the $M - \{M_i\}$ group has complete authentication.

III. STR Protocol

The Skinny TRee (STR) protocol, proposed by Steer *et al.* [4] and undertaken by Kim *et al.* [18], is a contributive protocol using a tree structure. The notations used to describe this STR tree are shown in Table 1.

Table 1. Notation

Symbol	Definition
p	Large Prime number
g	Exponentiation Base
M_i	Member of the group
IN_j	Internal node at level j
r_i	M_i's session random
bk_i	M_i's blinded key (that is $g^{r_i} \bmod p$)
k_j	Secret key of internal node IN_j

An example of STR protocol tree is shown in Fig. 4. where, M_1 to M_4 are member nodes and IN_1 to IN_4 are internal nodes in tree structure. M_1 is the initiator of the group which is responsible for creating a new group. All internal nodes always have two children: one right leaf node and one left internal node. Each leaf node chose same large prime number p and exponentiation base g. Each leaf node has a session random r_i chosen and kept secret by M_i. The blinded version of this secret key is calculated as $bk_i = g^{r_i} \bmod p$.

The session random of first member acts as its secret key. In single node tree structure this session random r_1 acts as k_1 that is group key of single node group. The basic key agreement protocol is as follows:

Whenever M_1 is the only member in the group it generates its own session random and calculates the blinded key. When new node M_2 joins the group, both members M_1 and M_2 calculate the group key as:

M_1 calculates: $k_2 = (blinded\ key\ of\ M_2)^{r_1} \bmod p$
M_2 calculates: $k_2 = (blinded\ key\ of\ M_1)^{r_2} \bmod p$

Both of these calculate the same group key, and set the k_2 as their root secret key. This group key is used for the further group communication. Both members calculate the blinded group key and store in their root's blinded key field. In this tree structure any member can calculate the group key if it knows:

1. Its own session random
2. Blinded key of the sibling sub tree
3. Blinded session random of the member higher in the tree.

The group key can be calculated recursively as:

$$k_i = (bk_{i-1})^{r_i} \bmod p$$

Where, bk_{i-1} is the blinded group key. All blinded keys are assumed to be public.

In this protocol, adversary can easily masquerade as a member of the group to the other members, since the identity of group members are not checked in the protocol.

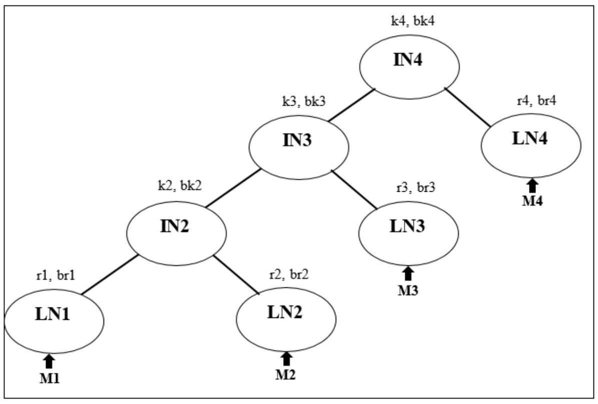
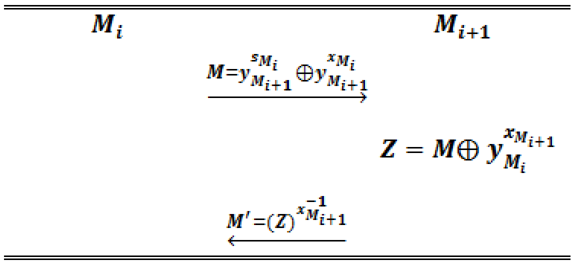


Fig.4. An example of STR

IV. Proposed Scheme

We propose an authentication protocol and add this to STR protocol to build Authenticated STR (ASTR) protocol which authentication is obtained with dynamic trust schema defined in previous section. Our proposed protocol is shown in Fig. 5.


M_i
M_{i+1}
$M = y_{M_{i+1}}^{s_{M_i}} \oplus y_{M_{i+1}}^{x_{M_i}}$
$Z = M \oplus y_{M_i}^{x_{M_{i+1}}}$
$M' = (Z)^{x_{M_{i+1}}^{-1}}$
Fig.5. The Proposed Protocol

The notations that used to describe ASTR protocol expressed in previous section, in addition the symbol T is the set of trust members in the group.

Our idea for authentication is to authenticate every one by previous user and then he/she add to the set of T. With this notations blinded key of every user is used for generate secret key if and only if this user is a member of the set T.

Suppose that, $\{M_1, \dots, M_n\}$ be the group that want to share a secret key, and suppose M_1 is trusted, therefore $T = M_1$. At the first M_1 runs authentication protocol with M_2. If this run was successful, then M_1 add M_2 to the set T, afterward run STR protocol between M_1 and M_2 to generate k_2 and bk_2 (look at Fig. 4.), and then M_2 similarly runs this protocol with M_3 and this protocol execute sequential between every M_i and M_{i+1}, for $1 \leq i \leq n$ (if all run of authentication protocol be successful).

If identity of M_2 not verified for M_1 (authentication protocol failed), then M_1 ejects M_2 from the group, and continue this protocol with next member of the group M_3. In other words M_1 pending to the first M_j for $2 \leq j \leq n$ that M_j's identity to be verified for him/her, then M_1 adds M_j to the set T and then protocol will be continued with M_j similarly.

When the last member of the group authenticates (successful or failed), protocol ends, then the set T consist of all trust member in the group. And secret key shared with all member of the set T, in other word shared secret key calculated by all blinded key of users that they are member of the set T (i.e $K = k_m = g^{r_m g^{r_{m-1} \dots g^{r_2 r_1}}}$ if and only if $T = M = \{M1, M2, \dots, Mm\}$).

Therefore T, the set of trust member in the group, is dynamic, as we say in the Section II in the case 3, authentication with dynamic trust.

According to the above description is sufficient to propose our authentication protocol for M_i and M_{i+1}.

Authentication Protocol: Suppose that G be a group with high order p that solving discrete logarithm problem is hard in G, and g be a generator of G.

Let $(x_{M_i}, y_{M_i} = g^{x_{M_i}})$ and $(x_{M_{i+1}}, y_{M_{i+1}})$ be private-public key of M_i and M_{i+1}. M_i choose random number $s_{M_i} \in \mathbb{Z}_p^*$ and send $M = y_{M_{i+1}}^{s_{M_i}} \oplus y_{M_{i+1}}^{x_{M_i}}$ to M_{i+1}, then M_{i+1} calculate $Z = M \oplus y_{M_i}^{x_{M_{i+1}}}$ and send back to M_i, $M' = (Z)^{x_{M_{i+1}}^{-1}}$.

Therefore, M_i can easily check that $M' = g^{s_{M_i}}$ or not.

V. Security Analysis

In this section, we model and analyze our proposed protocol in the group with five users, by the Scyther verification tool [20].

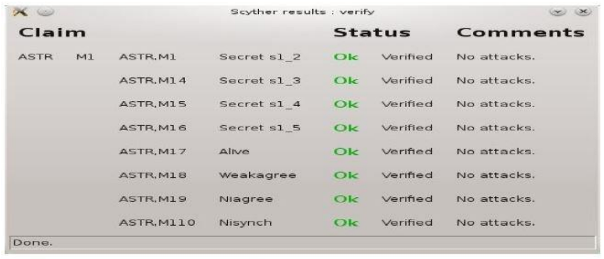
Due to some limitations of Scyther, we had to impose some level of abstraction on our protocol. We have analyzed the correctness of our protocol in the case that five users are participating in the protocol.

Since, every two users M_i and M_j, can calculate the value of $g^{x_i x_j}$ (by using his own secret key and the other parties public key), in order to simplify the protocol we consider this value can a shared key $k(M_i, M_j)$ between M_i and M_j. Since, Scyther does not provide exponentiation we abstracted the term $y_{M_j}^{s_{M_i}}$ by simply replacing this value by a freshly random number s_{i-j}.

If the other party is the claimed agent (i.e. he is user j), then he can extract s_{i-j} form $k(M_i, M_j) \oplus s_{i-j}$ and sends it back to the users i.

The Scyther code of our protocol is given in Appendix A.

The results of running Scyther for five users is shown in Fig. 6. It shows that the aliveness, Ni-synchronisation, Ni-agreement, and the secrecy of the value s_{i-j} are verified for three users and no attack is found within the two runs, in our protocol.

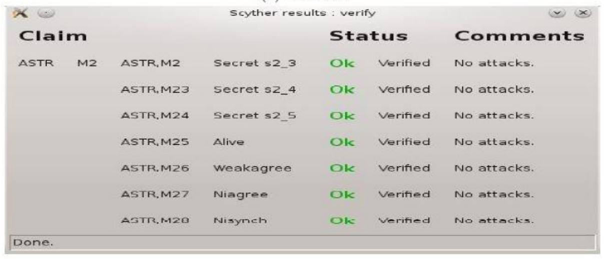


Scyther results : verify

Claim				Status		Comments
ASTR	M1	ASTR,M1	Secret s1_2	Ok	Verified	No attacks.
		ASTR,M14	Secret s1_3	Ok	Verified	No attacks.
		ASTR,M15	Secret s1_4	Ok	Verified	No attacks.
		ASTR,M16	Secret s1_5	Ok	Verified	No attacks.
		ASTR,M17	Alive	Ok	Verified	No attacks.
		ASTR,M18	Weakagree	Ok	Verified	No attacks.
		ASTR,M19	Niagree	Ok	Verified	No attacks.
		ASTR,M110	Nisynch	Ok	Verified	No attacks.

Done.

(a) User M1.

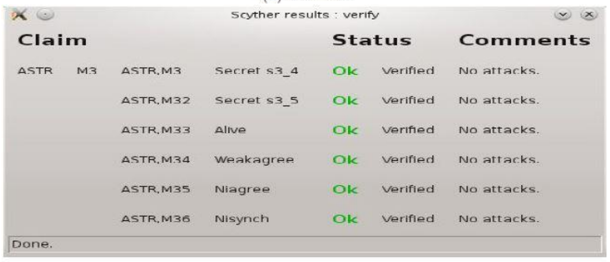


Scyther results : verify

Claim				Status		Comments
ASTR	M2	ASTR,M2	Secret s2_3	Ok	Verified	No attacks.
		ASTR,M23	Secret s2_4	Ok	Verified	No attacks.
		ASTR,M24	Secret s2_5	Ok	Verified	No attacks.
		ASTR,M25	Alive	Ok	Verified	No attacks.
		ASTR,M26	Weakagree	Ok	Verified	No attacks.
		ASTR,M27	Niagree	Ok	Verified	No attacks.
		ASTR,M28	Nisynch	Ok	Verified	No attacks.

Done.

(b) User M2.

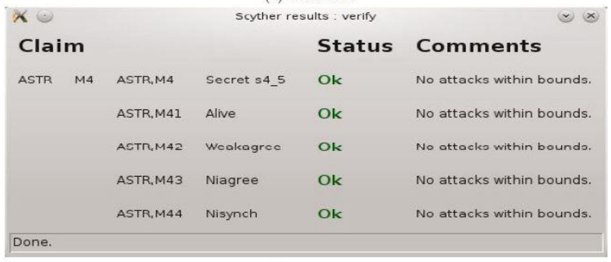


Scyther results : verify

Claim				Status		Comments
ASTR	M3	ASTR,M3	Secret s3_4	Ok	Verified	No attacks.
		ASTR,M32	Secret s3_5	Ok	Verified	No attacks.
		ASTR,M33	Alive	Ok	Verified	No attacks.
		ASTR,M34	Weakagree	Ok	Verified	No attacks.
		ASTR,M35	Niagree	Ok	Verified	No attacks.
		ASTR,M36	Nisynch	Ok	Verified	No attacks.

Done.

(c) User M3.

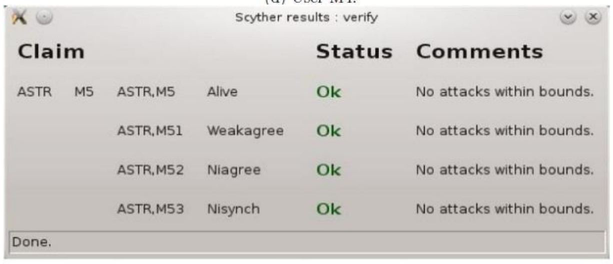


Scyther results : verify

Claim				Status	Comments
ASTR	M4	ASTR,M4	Secret s4_5	Ok	No attacks within bounds.
		ASTR,M41	Alive	Ok	No attacks within bounds.
		ASTR,M42	Weakagree	Ok	No attacks within bounds.
		ASTR,M43	Niagree	Ok	No attacks within bounds.
		ASTR,M44	Nisynch	Ok	No attacks within bounds.

Done.

(d) User M4.



Scyther results : verify

Claim				Status	Comments
ASTR	M5	ASTR,M5	Alive	Ok	No attacks within bounds.
		ASTR,M51	Weakagree	Ok	No attacks within bounds.
		ASTR,M52	Niagree	Ok	No attacks within bounds.
		ASTR,M53	Nisynch	Ok	No attacks within bounds.

Done.

(e) User M5.

Fig.6. Setup: Maximum number of runs = 2, Matching type = typed matching, Search pruning = Find all attacks.

VI. Conclusion

Authentication is one the most important concepts in design and verification of security protocols. There are different types of authentication in literature. In 1997, Lowe suggested an “authentication hierarchy” and identified several possible definitions of “authentication”.

In this paper we informally categorized different schemes for group authentication in conference (group) key establishment protocols involving more than two party, and provided some examples for each category. We also proposed an authenticated version of "The Skinny TRee" (STR) protocol which falls into the "Authentication with dynamic trust" category of our hierarchy and proved it's correctness via Scyther.

REFERENCES

[1] Dieter Gollmann. What do we mean by entity authentication? In IEEE Symposium on Security and Privacy, pages 46-54. IEEE Computer Society Press, 1996.

[2] Paul Syverson and Paul C. van Oorschot. On unifying some cryptographic protocol logics. In IEEE Symposium on Research in Security and Privacy, pages 14-28. IEEE Computer Society Press, 1994.

[3] Alfred J. Menezes, Paul C. van Oorschot, and Scott A. Vanstone. Handbook of Applied Cryptography. CRC Press, 1997.

[4] Gavin Lowe. A hierarchy of authentication specification. In 10th IEEE Computer Security Foundations Workshop, pages 31-43. IEEE Computer Society Press, June 1997.

[5] L. Harn, C. Lin, Authenticated Group Key Transfer Protocol Based on Secret Sharing, IEEE Trans. Computers, vol. 59, no. 6, pp. 842-846, June. 2010.

[6] W. Diffie and M.E. Hellman,\ New Directions in Cryptography, IEEE Trans. Information Theory, vol. IT-22, no. 6, pp. 644-654, Nov. 1976.

[7] I. Ingemarsson, D.T. Tang, and C.K. Wong, A Conference Key Distribution System, IEEE Trans. Information Theory, vol. IT-28, no. 5, pp. 714-720, Sept. 1982.

[8] D.G. Steer, L. Strawczynski, W. Diffie, and M.J. Wiener, A Secure Audio Teleconference System, Proc. Eighth Ann. International Cryptology Conf. Advances in Cryptology (Crypto 88), pp. 520-528, 1988.

[9] M. Burmester and Y.G. Desmedt, A Secure and Efficient Conference Key Distribution System, Proc. Eurocrypt 94 Workshop Advances in Cryptology, pp. 275-286, 1994.

[10] M. Steiner, G. Tsudik, and M. Waidner, Diffie-Hellman Key Distribution Extended to Group Communication, Proc. Third ACM Conf. Computer and Comm. Security (CCS 96), pp. 31-37, 1996.

[11] E. Bresson, O. Chevassut, D. Pointcheval, and J.-J. Quisquater, Provably Authenticated Group Diffie-Hellman Key Exchange, Proc. ACM Conf. Computer and Comm. Security (CCS 01), pp. 255-264, 2001.

[12] J.M. Bohli, A Framework for Robust Group Key Agreement, Proc. International Conf. Computational Science and Applications (ICCSA 06), pp. 355-364, 2006.

[13] J. Katz and M. Yung, Scalable Protocols for Authenticated Group Key Exchange, J. Cryptology, vol. 20, pp. 85-113, 2007.

[14] Shahrokh Saeednia and Rei Safavi-Naini. Efficient identity-based conference key distribution protocols. In C. Boyd et aI., editors, Information Security and Privacy - Third Australasian Conference, pages 320-331. Springer-Verlag, 1998. Lecture Notes in Computer Science Volume 1438.

[15] Giuseppe Ateniese, Michael Steiner, and Gene Tsudik. Authenticated group key agreement and friends. In 5th ACM Conference on Computer and Communications Security, pages 17-26. ACM Press, 1998.

[16] Giuseppe Ateniese, Michael Steiner, and Gene Tsudik. New multiparty authentication services and key agreement protocols. IEEE Journal on Selected Areas in Communications, 18(4):628-639, April 2000.

[17] Boyd, Colin A., and Anish Mathuria. Protocols for key establishment and authentication. Springer-Verlag New York, Inc., 2003.

[18] Michael Steiner, Gene Tsudik, and Michael Waidner. Diffie-Hellman key distribution extended to group communication. In 3rd ACM Conference on Computer and Communications Security, pages 31-37. ACM Press, 1996.

[19] Y. Kim, A. Perrig, and G. Tsudik. Communication Efficient group Key Agreement. IFIP SEC, June 2001.

[20] B. Klein, M. Otten, and T. Beth. Conference key distribution protocols in distributed systems. In P. G. Farrell, editor, Codes and Cyphers - Cryptography and Coding IV,page 225-241. IMA, 1995.

[21] Cremers, Cas JF. \textquotedblleft The Scyther Tool: Verification, falsification, and analysis of security protocols." In Computer Aided Verification, pp. 414-418. Springer Berlin Heidelberg, 2008.

[22] CH. V. Raghavendran,G. Naga Satish,P. Suresh Varma,"A Study on Contributory Group Key Agreements for Mobile Ad Hoc Networks", IJCNIS, vol.5, no.4, pp.48-56,2013.DOI: 10.5815/ijcnis.2013.04.07.

APPENDIX

The Scyther verification code for our authentication protocol:

```
#Authenticated-protocol
const Star: Function;
usertype XOR;
protocol ASTR(M1,M2,M3,M4,M5){
role M1
{
fresh s1-2,s1-3, s1-4, s1-5: XOR;
send_1(M1,M2, {k(M1,M2)}s1-2 );
recv_2(M2,M1, s1-2);
not match(s1-2, s1-2);
send_3(M1,M3, {k(M1,M3)}s1-3 );
recv_4(M3,M1, s1-3);
not match(s1-3, s1-3);
send_5(M1,M4, {k(M1,M4)}s1-4 );
recv_6(M4,M1, s1-4);
not match(s1-4, s1-4);
send_7(M1,M5, {k(M1,M5)}s1-5);
recv_8(M5,M1, s1-5);
claim_M1(M1, Secret, s1-2);
claim_M1(M1, Secret, s1-3);
claim_M1(M1, Secret, s1-4);
claim_M1(M1, Secret, s1-5);
claim_M1(M1, Alive);
claim_M1(M1, Weakagree);
claim_M1(M1, Niagree);
claim_M1(M1, Nisynch);
}
role M2
{
fresh s1-2, s2-3, s2-4, s2-5: XOR;
recv_1(M1,M2, {k(M1,M2)}s1-2);
send_2(M2,M1, s1-2);
send_9(M2,M3, {k(M2,M3)}s2-3);
recv_10(M3,M2, s2-3);
not match(s2-3, s2-3);
send_11(M2,M4, {k(M2,M4)}s2-4);
recv_12(M4,M2, s2-4);
not match(s2-4, s2-4);
send_13(M2,M5, {k(M2,M5)}s2-5);
recv_14(M5,M2, s2-5);
claim_M2(M2, Secret, s2-3);
claim_M2(M2, Secret, s2-4);
claim_M2(M2, Secret, s2-5);
claim_M2(M2, Alive);
claim_M2(M2, Weakagree);
claim_M2(M2, Niagree);
claim_M2(M2, Nisynch);
}
role M3
{
fresh s1-3,s2-3,s3-4,s3-5: XOR;
recv_3(M1,M3, {k(M1,M3)}s1-3 );
send_4(M3,M1, s1-3);
recv_9(M2,M3, {k(M2,M3)}s2-3);
send_10(M3,M2, s2-3);
send_15(M3,M4, {k(M3,M4)}s3-4);
recv_16(M4,M3, s3-4);
not match(s3-4,s3-4);
send_17(M3,M5, {k(M3,M5)}s3-5);
recv_18(M5,M3, s3-5);
claim_M3(M3, Secret, s3-4);
claim_M3(M3, Secret, s3-5);
claim_M3(M3, Alive);
claim_M3(M3, Weakagree);
claim_M3(M3, Niagree);
claim_M3(M3, Nisynch);
}
role M4
{
fresh s1-4,s2-4,s3-4,s4-5: XOR;
recv_5(M1,M4, {k(M1,M4)}s1-4 );
send_6(M4,M1, s1-4);
recv_11(M2,M4, {k(M2,M4)}s2-4);
send_12(M4,M2, s2-4);
recv_15(M3,M4, {k(M3,M4)}s3-4);
send_16(M4,M3, s3-4);
send_19(M4,M5, {k(M4,M5)}s4-5);
recv_20(M5,M4, s4-5);
claim_M4(M4, Secret,  s4-5);
claim_M4(M4, Alive);
claim_M4(M4, Weakagree);
claim_M4(M4, Niagree);
claim_M4(M4, Nisynch);
}
role M5
{
fresh s1-5,s2-5, s3-5, s4-5: XOR;
recv_7(M1,M5, {k(M1,M5)}s1-5);
send_8(M5,M1, s1-5);
recv_13(M2,M5, {k(M2,M5)}s2-5);
send_14(M5,M2, s2-5);
recv_17(M3,M5, {k(M3,M5)}s3-5);
send_18(M5,M3, s3-5);
recv_19(M4,M5, {k(M4,M5)}s4-5);
send_20(M5,M4, s4-5);
claim_M5(M5, Alive);
claim_M5(M5, Weakagree);
claim_M5(M5, Niagree);
claim_M5(M5, Nisynch);
}
}
```

Authors' Profiles

Mohsen Pourpouneh was born in 1989 in Isfahan. He got his B.Sc. (2011) and M.Sc. (2013) in Computer Science, from Shahid Beheshti University and Tehran University, respectively. He started his career as a Ph.D. student at Sharif University of Technology, Tehran, Iran in 2013. His research interest includes Formal method, Specifying and Verifying Security Protocols, Computational Number Theory, Electronic Voting.

Rasoul Ramezanian was born in Mashhad in 1979. He got his B.S. and M.S. in Mathematics. In 2008, he graduated from a Ph.D. program of Mathematical Science Department of Sharif University of Technology, Tehran, Iran. He is an assistant professor at the same department. His research interests include Formal method, Specifying and Verifying Security Protocols, Multi-Agent Systems, and Process Algebra.

Afshin Zarei was born in 1990 in Shiraz. He got his B.Sc. (2012) and M.Sc. (2014) in Mathematics, from Shiraz University and Sharif University of Technology, respectively. He started his career as a Ph.D. student at Isfahan University of Technology, Isfahan, Iran in 2014. His research interest includes Formal method, Specifying and Verifying Security Protocols, Cryptography, Logic, Access Control, Set Theory.

How to cite this paper: Mohsen Pourpouneh, Rasoul Ramezanian, Afshin Zarei,"A Note on Group Authentication Schemes", International Journal of Computer Network and Information Security(IJCNIS), Vol.8, No.5, pp.18-24, 2016.DOI: 10.5815/ijcnis.2016.05.03