An Application of Group Theory in Confidential Network Communications *

J.A. López-Ramos†, J. Rosenthal‡, D. Schipani‡, R. Schnyder‡

October 28, 2016

Abstract

A new proposal for group key exchange is introduced which proves to be both efficient and secure and compares favorably with state of the art protocols.

1 Introduction

Group Key Exchange (GKE) is currently a highly relevant topic due to the explosion of group communications in many applications that provide information exchange (cf. [7] and [14] and their references). A very recent application can be found in wireless sensor networks (WSNs), core of the so-called Internet of Things, that consists of tiny autonomous low-cost low-power devices that carry out monitoring tasks. WSNs can be found in many civil applications. The devices are called sensor nodes and the monitored data is typically collected at a base station, that will be later processed in data mining servers.

Since Ingemarsson et al. in [6] made an attempt to extend Diffie-Hellman two-party key exchange given in [5], many works have dealt with this issue, i.e., providing protocols for a group of communication nodes that allow this group to build a common key in a distributed and collaborative manner. In [11] the authors gave a distributed protocol for GKE that does not run efficiently in the Initial Key Agreement (IKA), which is, in most cases, the main problem. Probably two of the best known distributed protocols are given in [12] and [13], and [3] and [4], respectively, and both extend naturally the foundational Diffie-Hellman key exchange.

The protocol introduced in [3] and [4] proves to be very efficient in the IKA, using just two rounds. However, further rekeying operations require executing

*The Research was supported in part by the Swiss National Science Foundation under grant No. 169510. First author is partially suppported by Ministerio de Economia y Competitividad grant MTM2014-54439 and Junta de Andalucia (FQM0211). The last author is supported by Armasuisse.

†University of Almeria

‡University of Zurich

the protocol completely as in the initial phase (IKA) and produce a big number of messages, using a large bandwidth as the number of communication nodes grows.

On the other hand, the protocol introduced in [12] and [13] is quite efficient in the rekeying processes, i.e., rekeying operations once the group has shared a first key. The authors give an Auxiliary Key Exchange (AKA) that makes use of a single broadcasted message, although the IKA protocol is considerably slower than the preceding one since it requires as many rounds as the number of participants.

But the main issue in both efficient proposals is their security. In [10] and [1] active attacks to the systems proposed in [12] and [13], and [3] and [4] respectively are introduced. The authors show the possibility of sharing a common key with the components of a communication group, without letting them notice anything. However, one of the IKA proposals introduced in [12] and [13] (the so-called IKA.1) avoids this attack, but, as pointed out above, it requires a big numbers of rounds as the number of users grows.

In this paper we are introducing a new proposal that avoids both attacks ([1] and [10]) and shows the best characteristics of the aforementioned proposals: on one hand, the key is obtained in a distributed IKA with just two rounds and, on the other, the AKA protocol that provides rekeying operations is developed by means of a single message. Our protocol extends naturally the Diffie-Hellman key exchange as well and we show that its security is based on the difficulty of problems that refer to decisions in the group where the two party Diffie-Hellman key exchange takes place. Namely, the Decisional Diffie-Hellman (DDH) problem in a group G is the problem to decide whether, given g in G and random x, y, z in G, z equals $x^{\log_g y}$.

2 The Initial Key Agreement

Let us start by establishing the general setting for the GKE protocol. Participants in the communication process will be given by the set $\{U_1, \dots U_n\}$. The group of users agree on a cyclic group G of order a prime q and a generator g of G.

Every participant $\mathcal{U}_i$ holds two pairs of private-public keys, say (r_i, g^{r_i}) and (x_i, g^{x_i}). One of these users will be the group controller that we will denote by $\mathcal{U}_{c_1}$, for some c_1 in the set $\{1, \dots, n\}$. He will be in charge of sending the initial keying information as well as the following rekeying messages in case we wish to define a centralized protocol. However, as we will see in the following section, the character of the protocol can change from centralized to distributed (and vice versa) at any point of the following rekeying stages. The protocol that describes the initial key agreement is given by the following steps.

Protocol 1:

First Round:

1. Every user $\mathcal{U}_i$ publishes his pair of public keys (g^{r_i}, g^{x_i}), $i = 1, \ldots, n$, $i \neq c_1$.

2. The group controller $\mathcal{U}_{c_1}$ computes the key $K_1 = g^{r_{c_1} \sum_{j=1, j\neq c_1}^{n} r_j}$.

3. The group controller takes a new pair of private elements (r'_{c_1}, x'_{c_1}), which becomes his new private information. This will be used in the case of a rekeying operation at a later stage.

   Second Round:

4. Every user, using the public information, $\mathcal{U}_i$, $i = 1, \ldots, n$, $i \neq c_1$, computes $g^{\sum_{j=1, j\neq c_1, i}^{n} r_j}$ and sends this value to U_{c_1}.

5. The group controller $\mathcal{U}_{c_1}$ broadcasts the keying message

$$\{Y_{1,1}, \ldots, Y_{1,c_1}, \ldots, Y_{1,n}, R_1, S_1\}$$

   where $Y_{1,i} = g^{-x_{c_1} x_i} \left(g^{r_{c_1} \sum_{j\neq c_1, i} r_j} \right)$, for $i = 1, \ldots, n$, $i \neq c_1$,

   $Y_{1,c_1} = K_1 g^{-r'_{c_1} r_{c_1}} g^{-x'_{c_1} x_{c_1}}$, and $R_1 = g^{r_{c_1}}$ and $S_1 = g^{x_{c_1}}$.

6. Every user $\mathcal{U}_i$ computes $K_{1,i} = Y_{1,i} S_1^{x_i} R_1^{r_i}$, $i = 1, \ldots, n$, $i \neq c_1$.

The proof of the following Lemma is straightforward and shows the correctness of the protocol.

Lemma 2.1. *$K_{1,i} = K_1$ for every $i = 1, \ldots, n$, $i \neq c_1$.*

Remark 2.2. Let us assume that the number of users is $n = 2$ and that $g^{x_i} = e$ for $i = 1, 2$, where $e \in G$ is the neutral element. Now if $\mathcal{U}_1$ makes public g^{r_1} in the first round, $\mathcal{U}_2$ will send the keying message $\{e, R_1 = g^{r_2}\}$ in the second round. Thus our protocol is a natural extension of the classical Diffie-Hellman key exchange in the group G.

It can be observed in the preceding protocol that user $\mathcal{U}_{c_1}$ bears most of the workload. The protocol is designed in such a way that every node publishes just a pair of public keys, while $\mathcal{U}_{c_1}$ computes what is required for the first keying. This could be the case when $\mathcal{U}_{c_1}$ is a server that processes the pieces of information transmitted by every user. However, in case every user has similar capabilities, we can slightly modify the preceding protocol and distribute the computation requirements. As previously, every user holds a pair of private keys (r_i, x_i).

Protocol 2:

First Round:

1. Every user $\mathcal{U}_i$ publishes his public key g^{r_i}, $i = 1, \ldots, n$, $i \neq c_1$.

2. The group controller $\mathcal{U}_{c_1}$ computes the key $K_1 = g^{r_{c_1} \sum_{j=1, j\neq c_1}^{n} r_j}$.

3. The group controller takes a new pair of private elements (r'_{c_1}, x'_{c_1}), which becomes his new private information.

   Second Round:

4. Every user, using the public information, $\mathcal{U}_i$, $i = 1, \ldots, n$, $i \neq c_1$, computes $g^{\sum_{j=1, j\neq c_1, i}^{n} r_j} g^{-x_i}$ and sends this value to U_{c_1}.

5. The group controller $\mathcal{U}_{c_1}$ broadcasts the keying message

$$\{Y_{1,1}, \ldots, Y_{1,c_1}, \ldots, Y_{1,n}, R_1\}$$

   where $Y_{1,i} = \left(g^{r_{c_1} \sum_{j=1, j\neq c_1, i} r_j}\right) g^{-r_{c_1} x_i}$, for $i = 1, \ldots, n$, $i \neq c_1$,

   $Y_{1,c_1} = K_1 g^{-r'_{c_1} r_{c_1}} g^{-x'_{c_1} r_{c_1}}$, and $R_1 = g^{r_{c_1}}$.

6. Every user $\mathcal{U}_i$ computes $K_{1,i} = Y_{1,i} R_1^{x_i} R_1^{r_i}$, $i = 1, \ldots, n$, $i \neq c_1$.

We will now state the security of the preceding protocols. To this end let us now recall the following definition.

Definition 2.3. [4, Definition 2.2] Let $\mathcal{P}$ be a GKE protocol and $\mathcal{A}$ a passive adversary. Assume that $\mathcal{A}$ has witnessed polynomially-many instances of $\mathcal{P}$ and let K be the key output by the last instance.

We will say that $\mathcal{P}$ guarantees secrecy if $\mathcal{A}$ cannot distinguish K from a random bit string of the same length with probability better than $1/2 + \varepsilon$, where ε is negligible

Theorem 2.4. *If the DDH problem is intractable, then Protocols 1 and 2 provide secrecy.*

Proof. We observe that we can see the broadcasted message in Protocol 1 as a multiple ElGamal type of encryption in the following way. For $i \neq c_1$ we first encrypt K_1 using the public value g^{r_i} and r_{c_1} as random parameter, obtaining $\left(X_{1,i}, R_1\right) = \left(g^{r_{c_1} \sum_{j\neq c_1, i} r_j}, g^{r_{c_1}}\right)$ and then we encrypt X_i using the public key g^{x_i} and x_{c_1} as a random parameter, obtaining the pair $\left(Y_{1,i}, S_1\right)$.

The case of Y_{1,c_1} is analogous using the elements $g^{r'_{c_1}}$ and $g^{x'_{c_1}}$, that, although unknown to a passive adversary, could also be made public.

Now using Lemma 1 and Theorem 1 of [2] we can deduce the thesis.

The security of Protocol 2 follows similarly.

□

Remark 2.5. For ease of notation we have presented the preceding protocols using the action $\Phi(y, g^x) = (g^x)^y$, but more general scenarios based on linear actions can be considered too, as in [8].

Remark 2.6. We could wonder about the necessity of using two different keys for every user. To clarify this suppose only one key is used. Then we would

share a key of the form $K = g^{k_j \sum_{r=1, r\neq j}^{n} k_r}$. Without the x_i, an adversary could access the messages

$$D_i = g^{k_j \sum_{r=1, r\neq i,j}^{n} k_r},\ i \neq j,\ i = 1, \ldots, n,$$

from which she can compute $\prod_{r=1, r\neq j}^{n} D_r = K^{n-2}$. In the case where the order q of S is known, the adversary can now recover the key K from K^{n-2} by inverting $n-2$ modulo q. This is in particular the case where G is a subgroup of a finite field, or where it is the group of points of an elliptic curve. Using two different keys for every user avoids the above situation in these cases. However, the use of a single key for every user could still apply in other settings [8].

3 The Auxiliary Key Agreement

In the preceding section we have introduced a protocol to build a common key based on the information held by every user. As we can observe this may require some computational resources on one of the participants. On the other hand, this session key may expire due simply to key caducity or to changes in the communication group, i.e., users may join or leave the group and we are concerned about preserving secrecy of previous, respectively, later communications. In this section we provide an Auxiliary Key Agreement that solves this matter very efficiently and, moreover, allows either to keep the centralized aspect of the GKE, or to change to a distributed scheme, allowing any user to provide a new key for the remaining members of the group.

In the more general setting, we are assuming that some rekeying rounds have occurred, and in the next step keys are to be renewed again possibly by a new controller. The following protocol shows the Auxiliary Key Agreement after $t-1$ rekeying rounds, $t > 1$, whereby K_t denotes the last common key shared by the group. The user in charge of the t-th rekeying will be user $\mathcal{U}_{c_t}$, distinct from the preceding controller, and thus, rekeying information of this will be needed. Without loss of generality, we may assume that the precedent controller was user $\mathcal{U}_{c_1}$, the key shared was K_{t-1} and the last rekeying message was

$$\{Y_{t-1,1}, \ldots, Y_{t-1,c_1}, \ldots, Y_{t-1,n}, R_{t-1}, S_{t-1}\}$$

where $Y_{t-1,c_1} = K_{t-1} g^{-r'_{c_1} r_{c_1}} g^{-x'_{c_1} x_{c_1}}$, following Protocol 1. The proposed AKA is then as follows: *Protocol 3:*

1. User $\mathcal{U}_{c_t}$ computes two new elements r'_{c_t} and $x'_{c_t} \in G$ that become his new private information.

2. User $\mathcal{U}_{c_t}$ computes the new session key $K_{t-1}^{r'_{c_t}}$.

3. User $\mathcal{U}_{c_t}$ broadcasts the rekeying message

$$\{Y_{t,1}, \ldots, Y_{t,c_t}, \ldots, Y_{t,n}, R_t, S_t\}$$

where $Y_{t,i} = Y_{t-1,i}^{r'_{c_t}}$, $i \neq c_t$,

$Y_{t,c_t} = K_t R_{t-1}^{-r'_{c_t} r'_{c_t}} S_{t-1}^{-r'_{c_t} x'_{c_t}}$,

and $R_t = R_{t-1}^{r'_{c_t}}$ and $S_t = S_{t-1}^{r'_{c_t}}$.

4. Every user $\mathcal{U}_i$ computes $K_{t,i} = Y_{t,i} S_t^{x_i} R_t^{r_i}$, $i = 1, \ldots, n$, $i \neq c_t$.

Lemma 3.1. *$K_{t,i} = K_t$ for every $i = 1, \ldots, n$.*

Proof. For $i = 1, \ldots, n$, $i \neq c_t$

$$\begin{aligned} K_{t,i} &= Y_{t,i} S_t^{x_i} R_t^{r_i} \\ &= Y_{t-1,i}^{r'_{c_t}} S_{t-1}^{x_i r'_{c_t}} R_{t-1}^{r_i r'_{c_t}} \\ &= (Y_{t-1,i} S_{t-1}^{x_i} R_{t-1}^{r_i})^{r'_{c_t}} \\ &= K_{t-1}^{r'_{c_t}} \\ &= K_t. \end{aligned}$$

□

3.1 Altering the membership

As it was previously pointed out members of the group can be constantly changing: some of them may leave the group and other may wish to join it. For this reason it is convenient that those joining the group should not be able to get previously distributed keys and those leaving it should not get future distributed keys in order to preserve confidentiality of former and future communications.

In the case of a set of members leaving the group, rekeying is made naturally following Protocol 2, but erasing those positions in the rekeying message corresponding to those users leaving the group.

On the other hand joining operation can be carried out in a massive way and by any of the users. Let us assume that after the t-th rekeying message, l users wish to join the group. Then the corresponding values R_t and S_t are made public and without loss of generality, we may suppose that these l users sent a petition to join the group to a user that will be $\mathcal{U}_{c_{t+1}}$ for some c_{t+1} in the set $\{1, \ldots, n\}$.

In case the petitions are sent to different users, each of them may collect the received petitions and send a rekeying message including the petitioners after a period of time. The protocol is as follows:
Protocol 4.

1. Every new user $\mathcal{U}_{n+j}$, $j = 1, \ldots, l$, sends a petition to user $\mathcal{U}_{c_{t+1}}$ jointly with the pair $R_t^{r_{n+j}}, S_t^{x_{n+j}}$, where $r_{n+j}, x_{n+j} \in G$ is the user $\mathcal{U}_{n+j}$'s private information.

2. User $\mathcal{U}_{c_{t+1}}$ computes two new elements $r'_{c_{t+1}}, x'_{c_{t+1}} \in G$ that become his new private information.

3. User $\mathcal{U}_{c_{t+1}}$ computes the new key $K_{t+1} = \left(K_t R_t^{\sum_{i=1}^{l} r_{n+i}}\right)^{r'_{c_{t+1}}}$.

4. User $\mathcal{U}_{c_{t+1}}$ broadcasts the rekeying message

$$\{Y_{t+1,1}, \ldots, Y_{t+1,c_{t+1}}, \ldots, Y_{t+1,n}, Y_{t+1,n+1}, \ldots, Y_{t+1,n+l}, R_{t+1}, S_{t+1}\}$$

where $Y_{t+1,i} = \left(Y_{t,i} R_t^{\sum_{i=1}^{l} r_{n+i}}\right)^{r'_{c_{t+1}}}$, for $i = 1, \ldots, n$, $i \neq c_{t+1}$,

$Y_{t+1,c_{t+1}} = K_{t+1} R_t^{-r'_{c_{t+1}} r'_{c_{t+1}}} S_t^{-r'_{c_{t+1}} x'_{c_{t+1}}}$,

$Y_{t+1,i} = K_{t+1} R_t^{-r_i r'_{c_{t+1}}} S_t^{-x_i r'_{c_{t+1}}}$, for $i = n+1, \ldots, n+l$,

$R_{t+1} = R_t^{r'_{c_{t+1}}}$ and $S_{t+1} = S_t^{r'_{c_{t+1}}}$.

5. Every user $\mathcal{U}_i$ computes $K_{t+1,i} = Y_{t+1,i} S_{t+1}^{x_i} R_{t+1}^{r_i}$, $i = 1, \ldots, n+l$, $i \neq c_{t+1}$.

Analogously to Lemma 3.1 we get the following.

Lemma 3.2. *After Protocol 3,* $K_{t+1,i} = K_{t+1}$ *for every* $i = 1, \ldots, n+l$, $i \neq c_{t+1}$.

Also, using the same argument as in the IKA protocol, we may show that protocols for rekeying and altering the membership provide secrecy for the former and future keys distributed.

References

[1] M. Baouch, J.A. Lopez-Ramos, R. Schnyder, B. Torrecillas, An active attack on a distributed Group Key Exchange system, preprint.

[2] M. Bellare, A. Boldyreva, J. Staddon, Randomness Re-use in Multi-recipient Encryption Schemeas, Y.G. Desmedt (Ed.): PKC 2003, LNCS 2567, 85–99, Springer-Verlag, Berlin Heidelberg 2003

[3] M. Burmester, I. Desmedt, A secure and efficient conference key distribution system, in: Proc. Eurocypt'94, in: Lecture Notes in Comput. Sci., vol. 950, Springer-Verlag, Berlin, 1995, 275-286.

[4] M. Burmester, I. Desmedt, A secure and scalable Group Key Exchange system, Information Proc. Letters 94, 2005, 137-143.

[5] W.D. Diffie, M.E. Hellman, New directions in cryptography, IEEE Transactions on Information Theory, vol. 22(6), 644–654, 1976.

[6] I. Ingemarsson, D. Tang, C. Wong. A Conference Key Distribution System, IEEE Trans. Information Theory, vol. 28, no. 5, 714–720, 1982.

[7] P.P.C. Lee, J.C.S. Lui, D.K.Y. Yau, Distributed Collaborative Key Agreement and Authentication Protocols for Dynamic Peer Groups, IEEE/ACM Trans. Networking 14(2), 263-276, 2006.

[8] J. A. Lopez Ramos, J. Rosenthal, D. Schipani, R. Schnyder, Group key management based on semigroup actions, arXiv preprint, to appear in Journal of Algebra and its applications.

[9] G. Maze, C. Monico, J. Rosenthal, Public key cryptography based on semigroup actions, Advances of Mathematics of Communications, vol. 1(4), 489–507, 2007.

[10] R. Schnyder, J.A. Lopez-Ramos, J. Rosenthal, D. Schipani, An active attack on a multiparty key exchange protocol, Journal of Algebra Combinatorics Discrete Structures and Applications 3(1), 31–36, 2016.

[11] D. G. Steer, L. Strawczynski, W. Diffie, M. Wiener, A Secure Audio Teleconference System, Advances in Cryptology "CRYPTO" 88, Lecture Notes in Computer Science, vol. 403, 520–528, 2000.

[12] M. Steiner, G. Tsudik, M. Waidner, Diffie-Hellman key distribution extended to group communication, Proceedings of the 3rd ACM Conference on Computer and Communications Security, ACM: New York, NY, 31–37, 1996.

[13] M. Steiner, G. Tsudik, M. Waidner, Key agreement in dynamic peer groups. IEEE Transactions of Parallel and Distributed Systems, 11(8), 769–780, 2000.

[14] J. Van der Merwe, D. Dawoud, S. McDonald, A survey on peer-to-peer key management for mobile ad hoc networks, ACM Computing Surveys 39 (1) 2007.