Design of improved DH (Diffie-Hellman) key agreement protocol based on distance bounding for peer-to-peer wireless networks

Seon-Yeong Park[O], Ju-Young Kim, Hong-Yeop Song

Coding and Information Theory Lab,
Department of Electrical and Electronics Engineering, Yonsei University
134 Shinchon-dong, Seodaemun-gu, Seoul, Korea, 120-749
{sy.park, jy.kim, hysong}@yonsei.ac.kr

Abstract

We propose the improved DH key agreement protocol over a radio link in peer-to-peer networks. The proposed protocol ensures the secure establishment of a shared key between two parties through DB (distance bounding). In this way, we can ensure the integrity of exchanged data under the limited power and limited capacity of memory. We analyze the inefficiency and problems of existing DH key agreement protocols. Based on the analysis we improve the DH key agreement protocol. Proposed protocol reduces the two messages exchanged, the four memory required, and (7682(k/64)-64)*2 operations, where k means the length of random sequence.

1. Introduction

As the data communication is possible between personal device (e.g., a PDAs, laptops, and mobile phones), the peer-to-peer communication frequently occurs. Also, the communication systems are scattered on the fields. Therefore, the establishment of system requires auto configuration of mobile routers.

In this situation, the communication between devices must be properly secured. For this work, DH (Diffie-Hellman) key agreement protocol [1] is conventionally used. It achieves key agreement by calculating simple integer parameter without shared secret. It is appropriate for systems which have limited-power and limited-memory. However, it is vulnerable to an active adversary who uses a MITM (man-in-the-middle) attack. Also, it can be attacked by mathematical methods such as degenerate message attack. Theses attacks are easily done as the computing powers are raised. However, devices should have limited-power and limited-memory.

Recently proposed protocol cope with these attacks by using various methods. In case of DH-SC (DH based on Short String Comparison) [2], it has more resistance against the MITM attack, but it needs an involvement of user. DH-DB (DH based on distance bounding) [3] checks integrity without involvement of user, but it needs a physical device to measure the distance between two peers. DH-IC (DH using Integrity Codes) [4] reduces the involvement of the user but it is inappropriate for key agreement.

Especially, we focused on DH-DB protocol. DH-DB protocol ensures the secure establishment of a shared key between two peers through distance bounding. That is, pair of devices has the means to accurately estimate the distance between them. Based on the distance, each device upper-bounds its distance to the device of peer. If there is no other user in the boundary, the exchanged DH public parameters are accepted. However, existing DH-DB protocol still has weakness for security. Also, it has inefficiency when it checks the integrity.

Through this research, we analyze the complexity and problems of existing DH-DB. Based on the analysis we improve the DH-DB. Finally we compare the complexity and security of proposed and existing protocol.

The paper is organized as follows. In section 2 we analysis the existing DH-DB. In section 3 we present our protocol. In section 4 we provide analysis of our protocols. Finally, we conclude the paper in section 5.

2. Existing DH-DB

We analyze the following protocol. Two users, A (Alice) and B (Bob), each equipped with a personal device capable of communicating over a radio link, get together and want to establish a shared key. We assume that they do not share any authenticated cryptographic information (e.g., public keys or a shared secret) prior to this meeting. Also, we assume that each device has the means to accurately estimate the distance between them.

2.1. Symbol and Notation

The following symbols and notations are used through this paper.

p : large prime number
q : prime number that divides $p-1$
Z_p^* : multiplicative group
g : generator of Z_p^*, $(2 \le g \le p-2)$
$X \| Y$: concatenation operation of X and Y
$X \oplus Y$: XOR operation of X and Y
$(c,d) \leftarrow Commit(m)$: the commitment/opening pair (c,d) for m
$m' \leftarrow Open(c,d)$: opens the commitment with the opening key d

We assume that p and g are selected and published.

2.2. Commitment Scheme

In this paper we will make use of a collision-free has function based commitment scheme [5]. This scheme is a very practical commitment scheme based solely on collision-free hashing. To commit to a message m, the sender picks at random a string x and a universal has function f so that $f(x) = m$. Then the user applies the collision-free hash function h (e.g., SHA-1) to the random string x to get $y = h(x)$ and sends the commit string $c = (y, f)$ to the intended receiver. To open the commit string the sender simply sends the random string x. The efficiency of this commitment scheme comes from the fact that it makes use of inexpensive hash functions only.

2.3. Protocol Description

The DH-DB protocol is shown in Fig. 1. The protocol is divided into three steps: initialization, distance-bounding, and verification. In the initialization step, A and B select their secret exponents X_A and X_B randomly from Z_q^* (q = large prime) and calculate DH public parameters g^{X_A} and g^{X_B}, respectively. A and B generate k-bit random string N_A and N_B, respectively. A and B concatenate $0 \| ID_A \| g^{X_A} \| N_A$ and $1 \| ID_B \| g^{X_B} \| N_B$, respectively. Here, 0 and 1 are used to prevent a reflection attack. Then, A and B compute commitment/opening pairs, respectively. A and B also concatenate $0 \| R_A$ and $1 \| R_B$ and calculate (c'_A, d'_A) and (c'_B, d'_B). A sends the commitment c_A and c'_A to B. B responds with his own commitment c_B and c'_B. A send out d_A. B opens $(\hat{c}_A, \hat{d}_A)$ and get $\hat{m}_A$. B checks the correctness of $(\hat{c}_A, \hat{d}_A)$ by verifying that 0 appears at the beginning of $\hat{m}_A$. If it is successful, B sends d_B. A checks $(\hat{c}_B, \hat{d}_B)$ by verifying that 1 appears at the beginning of $\hat{m}_B$. If it is successful, A and B generate the verification string i_A and i_B.

In the distance-bounding step, A and B execute distance bounding by exchanging bit by bit all the bits of R_A, R_B, i_A and i_B. Here, A and B execute XOR operation before exchange the bit. This work protects the verification string by giving dependency to exchanged bits. During distance bounding time the devices measure round-trip times between sending a bit and receiving a response bit. The device estimates the distance-bound to the other device by multiplying the round trip time by the speed of light in the case of the radio or by the speed of sound in the case of ultrasound communication.

In the verification step, A and B retrieve $\hat{R}_B, \hat{i}_B$ and $\hat{R}_A, \hat{i}_A$, respectively. Then, A and B verify $\hat{i}_B$ and $\hat{i}_A$ against i_A and i_B. (By the devices A and B, not users A and B) If it is successful, devices A and B display the measured distance bounds on their screens. The users A and B then visually verify that there are no other users/devices in their integrity regions. Then the users accept the exchanged DH public parameters and IDs as being authentic.

2.4. Analysis of the complexity and problems of DH-DB

We analyze the vulnerability against the MITM attack and the complexity of DH-DB protocol. Active adversary M tries to collect information exchanged between A and B. Since the existence of adversary is checked at last step in DH-DB protocol, adversary can get m_A and m_B, which contains DH public parameter in readable manners, through collected information. Therefore, this protocol is not secure in the situation where DH public parameters are frequently reused.

In the aspect of complexity, this protocol performs complicated procedures for measuring the round-trip times. It generates random sequence and performs XOR operation with bits used for measuring the round-trip times. We can hide the verification string from adversary and obtain security through this procedure. However, we can reduce the overhead from random generator and XOR operation by designing new commitment scheme and reordering the procedure of protocol.

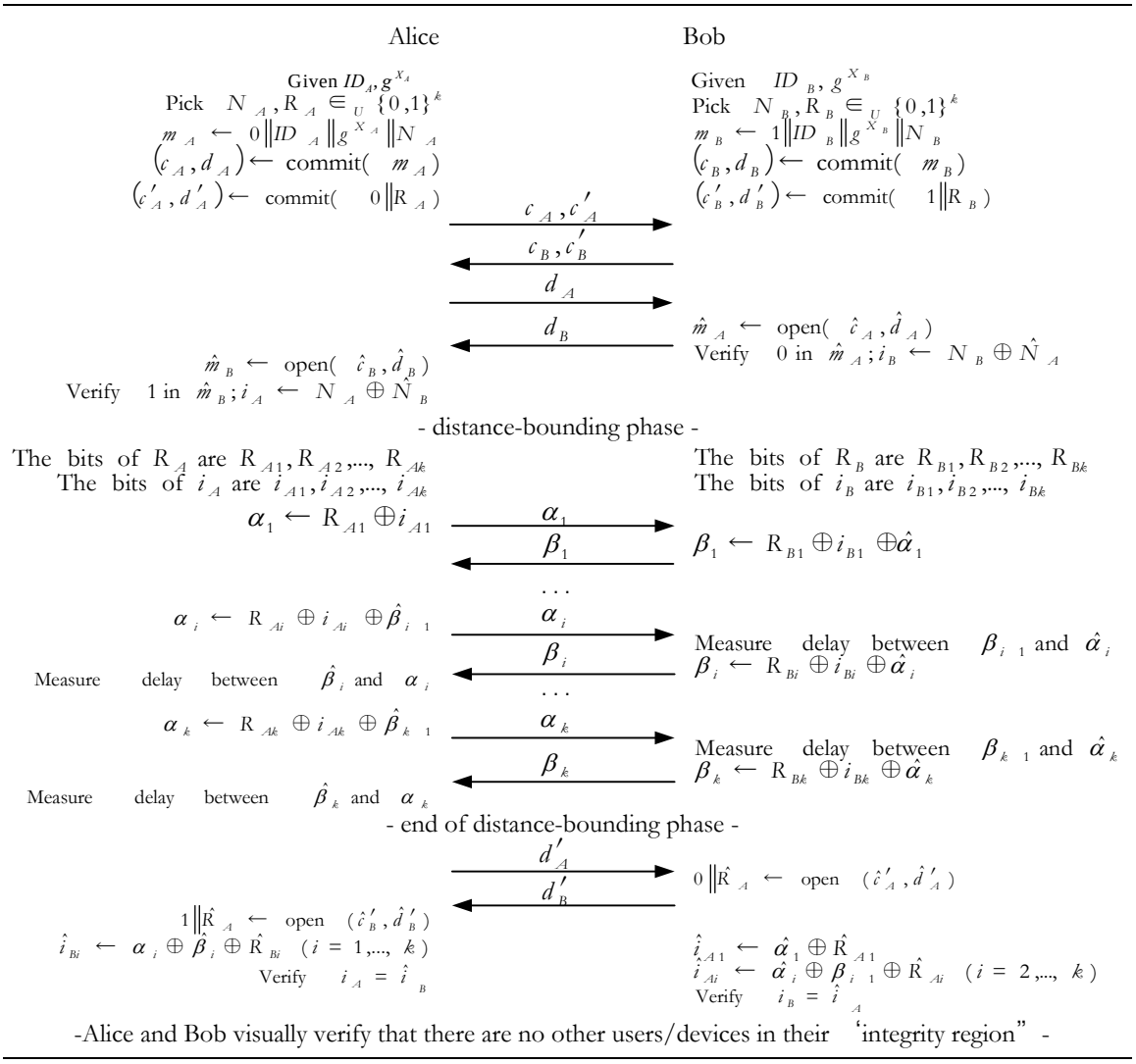

Fig. 1. Operation of DH-DB

3. Improved DH-DB

In this chapter we propose improved DH-DB protocol.

3.1. New Commitment Scheme

We define commitment/opening triplet (c, b, d). Sender picks collision-free hash function whose output y is b, k-bit string. c means a universal has function f and d means the random string x, where f and x are referred at chapter 2. Since many hash functions are used as random Oracle, these hash function can ensure randomness of b [6]. String b is used as exchanged bits for measuring round-trip time in proposed protocol. Since probabilities of 0 and 1 are equally likely, adversary can make attack at most $1/2^k$ of probability. Therefore, we can secure the integrity against the MITM attack without additional use of random generator.

3.2. Protocol Description

Fig.2 shows proposed protocol. It is also divided into three steps: initialization, distance-bounding, opening. The initialization step is similar to the initialization step of existing protocol. However, proposed protocol does not generate k-bit random string N_A and N_B.

In the distance-bounding step A and B exchange b_A and b_B without XOR operation. Therefore, we can reduce the computational complexity. Also, since we use collision-free hash function, adversary rarely get pre-image of b_A and b_B.

It is different from existing protocol that A and B visually verify that there are no other users/devices in their integrity regions between second step and third step. Even though adversary exists in integrity regions, adversary cannot open triplet with collected information. Therefore, adversary cannot get DH public parameters. With this, we can ensure the secure reuse of DH public parameters.

In third step, A sends d_A to B and B opens commitment $\hat{c}_A$ and examines that $\hat{m}_A$'s first bit is 0. If

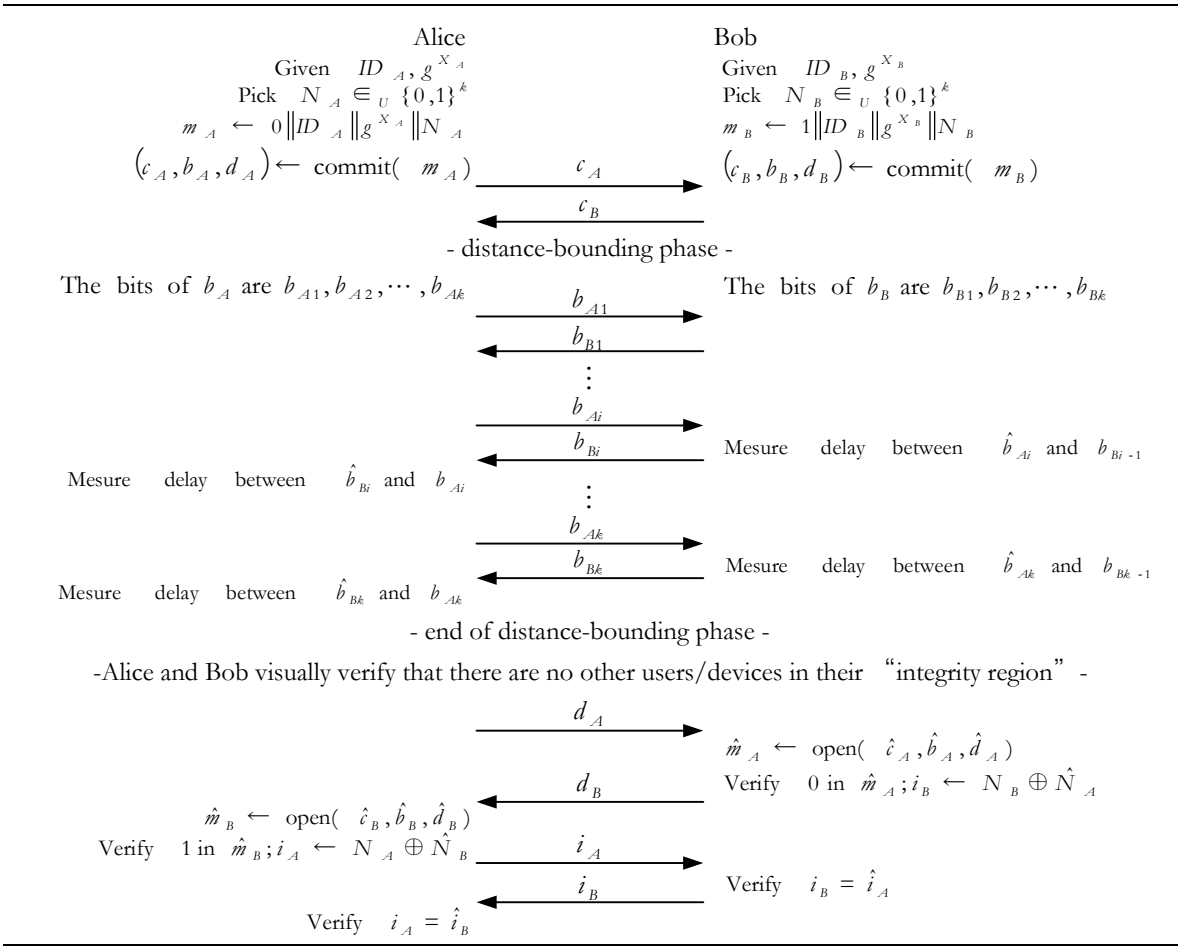

Fig. 2. Operation of improved DH-DB

it is successful, B sends d_B and A does similar procedure. After that A and be generate verification string i_A and i_B, respectively. Since we check the existence of other user before the third step, it is possible to send i_A and i_B in readable forms. Then, A and B verify $\hat{i}_B$ and $\hat{i}_A$ against i_A and i_B. If it is successful the users accept the exchanged DH public parameters and IDs as being authentic.

4. Performance Analysis

In this chapter, we confirm improvement by comparing security, the number of messages exchanged, required amount of memory, and number of operation of proposed protocol and existing protocol.

In the existing protocol, adversary can obtain m_A and m_B, even though A and B discontinue the communication. On the other hand, proposed protocol adversary cannot get m_A and m_B. It means that the security of reused DH parameter depends on the powerfulness of commitment scheme.

In the aspect of the number of messages exchanged 2k+6 messages are exchanged in the existing protocol if the protocol is finished successfully and 2k+4 messages are exchanged if the protocol is discontinued due to the other users in the integrity region. In proposed protocol, it also uses 2k+6 messages when the protocol is finished successfully. However, it exchanges 2k+2 messages when the protocol is discontinued. Therefore, two messages are reduced.

Existing protocol needs 18 memories for buffering nine parameters ID, X, g^X, N, R, c, d, α, and i of A and B. Since proposed protocol does not need R and α, 14 memories are required.

Finally we compare the computational complexity between two protocols. For fair comparison, we assume that two protocols use same universal hash function and collision-free hash function. Also, we count the number of XOR operation. We do not consider complexity due to memory access and table lookup. We assume that the random generator defined at ANSI X9.17 [7] is used.

The random generator generates 64-bit random string by using 3-DES which uses two keys. For generating 64-bit random string, 3-DES is used twice and additional 64 XOR operations are needed for every used of 3-DES expect final use. 3840 XOR (=16 [round/DES] *(32+48) [XOR/round] *3 [DES]) operations are needed for use of 3-DES. Therefore, total number of operations is as follow:

$$2\lceil k/64\rceil \times 3840 + 2\lceil k/64\rceil - 64 = 7628\lceil k/64\rceil - 64 \qquad (1)$$

In the case of k=64, 7618 XOR operations are needed.

Table 1 summarizes the comparison of two protocols. It is important to note that we improve the resistance against the MITM attack without increasing computational power or complexity.

	DH-DB	Proposed DH-DB
Exchanged message (Success)	2k+6	2k+6
Exchanged message (fail)	2k+4	2k+2
Required memory	18	14
XOR operation	-	(7682(k/64)-64)*2 are reduced
Reusability of DH public parameter	Vulnerable against MITM attack	Ensured

Table 1. Comparison between DH-DB and improved DH-DB

5. Conclusion

In the paper we provide a solution to the fundamental problem of key agreement over a radio link. We improve the existing DH-DB protocol. We confirm that its resistance against the MITM attack is raised and its computational complexity and required memory are reduced. Therefore, proposed protocol is appropriate for devices which have limited power, limited memory, and limited computational power.

6. Reference

[1] W. Diffie and M. Hellman, "New Directions in Cryptography," *IEEE Transaction on Information Theory*, 1976.

[2] M. Cagalj and J.-P. Hubaux, "Key agreement over a radio link," EPFL-IC-ICA, Tech. Rep. IC/2004/16, Jan. 2004.

[3] S. Brands and D. Chaum, "Distance-bounding protocols," in EUROCRYPT. Heidelberg, Germany: Springer-Verlag, 1993, vol. 765, Lecture Notes in Computer Science, pp. 344–359.

[4] Jean-Francois Raymond, Anton Stiglic, "Security Issues in the Diffie-Hellman Key Agreement Protocol", September 2000. (http://citeseer.nj.nec.com/453885.html)

[5] S. Halevi and S. Micali, "Practical and Provably-Secure Commitment Schemes from Collision-Free Hashing," In N. Koblitz, editor, *Advances in Cryptology-CRYPTO 96*, pages 201-215, Lecture Notes in Computer Science, Springer-Verlag, 1996.

[6] Mihir Bellare and Phillip Rogaway, *Random Oracles are Practical*: A Paradigm for Designing Efficient Protocols, ACM Conference on Computer and Communications Security 1993.

[7] A. J. Menezes, P. C. van Oorschot, and S. A. Vanstone, *Handbook of Applied Cryptography*, CRC press, 1997.