A Promising Key Agreement Protocol

Eun-Kyung Ryu, Kee-Won Kim, and Kee-Young Yoo

Department of Computer Engineering, Kyungpook National University,
Daegu 702-701, Republic of Korea
{ekryu, nirvana}@infosec.knu.ac.kr, yook@knu.ac.kr

Abstract. In 1999, Seo and Sweeney proposed a simple authenticated key agreement protocol(SAKA) that was designed to act as a Diffie-Hellman scheme with user authentication. However, the protocol was subsequently found to have security flaws and enhanced in the literature. Recently, Ku and Wang showed a variant of SAKA. This paper shows that the Ku and Wang's scheme is still vulnerable to an off-line password guessing attack. The attack illustrates that extreme care must be taken when passwords are combined to provide user authentication in the key agreement protocols. This paper also presents a new key agreement protocol that resists the guessing attacks mounted by either passive or active network attackers, allowing low-entropy passwords to be used safely. The protocol has more efficient performance by reducing the number of protocol steps.

1 Introduction

Key agreement protocol is the process in which two communication parties establish a shared secret key using information contributed by both of them. The key may subsequently be used to achieve some cryptographic goal, such as confidentiality or data integrity. Secure authenticated key agreement protocols are important as effective replacements for traditional key establishment achieved using expensive and inefficient couriers[1]. The Diffie-Hellman(DH) scheme[2] is a well-known key agreement protocol whose security is based on the difficulty of computing discrete logarithms over a finite field. Unfortunately, the main problem of the DH key agreement protocol is that it cannot withstand *the man-in-the-middle attack*.

In the literature, a number of techniques[1], [3]-[7] have been shown for solving the problem of Diffie-Hellman key agreement. They can be classified into two principal techniques, symmetric protocols and asymmetric protocols, by information contributed by legitimate parties that is used to derive a shared secret key. In symmetric protocols the two parties possess common secret information in advance, while in asymmetric protocols the two parties share only public information that has been authenticated. In particular, many works have focused on symmetric setting in which the parties only use a pre-shared secret, such as password; no supplementary keys or certificates are required. This paper is concerned with key agreement protocols in the symmetric setting.

T. Ibaraki, N. Katoh, and H. Ono (Eds.): ISAAC 2003, LNCS 2906, pp. 655–662, 2003.

Simple authenticated key algorithm(SAKA) [3] is a symmetric key agreement scheme in which the communication parties use only a pre-shared password. Since the password-based mechanism allows people to choose their own passwords with no assistant device to generate or store, it is the most widely used method for user authentication. SAKA is considered as an efficient one in terms of computational time and exchanged messages[4]. However, the protocol was found to have security flaws, and then was modified to eliminate such problems in [4][5][6]. Recently, Ku and Wang's protocol [6] proposed a variant of SAKA that they claimed their protocol can solve the problems.

In this paper, we will show an off-line password guessing attack on Ku and Wang's protocol whereby an attacker can easily obtain a legitimate communication parties' password. The attack illustrates that extreme care must be taken when passwords are combined to provide user authentication in the key agreement protocols. We also present a new key agreement protocol that resists the guessing attacks mounted by either passive or active network attackers, allowing low-entropy passwords to be used safely. The protocol has more efficient performance by reducing the number of protocol steps.

The remainder of this paper is organized as follows. In section 2, we begin by describing desirable properties for key agreement protocols. Then, briefly review the Ku and Wang's protocol and explain security weaknesses on their protocol. In section 3, we demonstrate a new key agreement protocol that can solve such problems. In section 4, we analyze the security of the new scheme. Finally, conclusion is given in section 5.

2 Background

In this section, we describe some desirable properties for key agreement protocols. Then, briefly review Ku and Wang's protocol and discuss security weaknesses on their protocol.

2.1 Desirable Properties for Key Agreement Protocols

A secure protocol should be able to withstand both *passive* attacks(where an attacker attempts to prevent a protocol from achieving its goals by merely observing honest entities carrying out the protocol) and *active* attacks(where an attacker additionally subverts the communications by injecting, deleting, altering or replaying messages). In addition to implicit key authentication and key confirmation, the following *security properties* of authenticated key agreement protocols should be considered since they are often desirable in some environments. In the following, A and B are honest parties.

1. *Known-key security.* Each run of a key agreement protocol between two entities A and B should produce a unique secret keys; such keys are called *session keys.* A protocol should still achieve its goal in the face of an attacker who has learned some other session keys.

2. *Perfect forward secrecy.* If long-term private keys of one or more entities are compromised, the secrecy of previous session keys established by honest entities is not affected.

Desirable *performance properties* of authenticated key agreement protocols include a minimal number of passes(the number of messages exchanged in a run of the protocol), low communication overhead(total number of bits transmitted), and low computation overhead. Other properties that may be desirable in some circumstances include role-symmetry(the message transmitted between entities have the same structure), non-interactiveness(the messages transmitted between the two entities are independent of each other), and the non-reliance on encryption(to meet possible export requirements), hash functions(since these are notorious hard to design), and timestamping(since it is difficult to implement securely in practice).

2.2 Review of Ku and Wang's Protocol

The system parameters are n and g, where n is a large prime and g is a generator with order $n-1$ in GF(n) as the original Diffie-Hellman scheme. All exponentiations are performed modulo n. Assume that two communication parties, called Alice and Bob, have a common pre-shared secret password P. Alice and Bob can pre-compute two integers Q and $Q^{-1} \pmod{n-1}$ from P in any predetermined way before performing the key agreement protocol. In their protocol, two parities exchange two messages to establish a shared secret session key in the protocol. Then, the exchange of two more messages allows the two parties to mutually authenticate. The description of each step in their protocol is as follows:

Key Establishment Phase:

1. Alice selects a random integer a and sends X_1 to Bob, where $X_1 = g^{aQ}$
2. Bob also selects a random integer b and sends Y_1 to Alice, where $Y_1 = g^{bQ}$
3. Alice computes the session key as follows:

$$Y = Y_1^{Q^{-1}} = g^b$$
$$Key_1 = Y^a = g^{ab}$$

4. Bob computes the session key Key_2 as follows:

$$X = X_1^{Q^{-1}} = g^a$$
$$Key_2 = X^b = g^{ab}$$

Key Validation Phase:

1. Alice computes $Y_2 = (Key_1)^Q = g^{abQ}$ and then sends Y_2 to Bob.

2. Bob checks whether $Y_2^{Q^{-1}} = Key_2$ holds or not. If it holds, Bob believes that he has obtained the correct X_1 and Alice has obtained the correct Y_1. That is, Bob is convinced that Key_2 is validated, and then sends X to Alice.
3. Alice checks whether $X = g^a$ holds or not. If it holds, Alice believes that she has obtained the correct Y_1 and Bob has obtained the correct X_1. Alice is convinced that Key_1 is validated.

2.3 Security Weaknesses

Although the Ku and Wang's protocol tried to solve the security problems in SAKA, their scheme is still vulnerable to an off-line password guessing attack. An attacker who captured messages exchanged over network can easily obtain a legitimate communication parties' password P. The attack proceeds as follows: Firstly, an attacker records a pair of information $< X_1, X >$ exchanged in a valid key agreement session. It is not difficult to obtain the information since they all are exposed over the open network. Then, the attacker obtains the password P shared two legitimate parties as follows: (1) The attacker makes a guess at the secret password P' and derives corresponding Q' and Q'^{-1} mod $n-1$. (2) Computes $X^{Q'}$ and checks if $X_1 = X^{Q'} \bmod n$, where X_1 and X are the information that he or she captured. (3) If it is not correct, the attacker repeatedly performs it until $X_1 = X^{Q'} \bmod n$.

Unlike typical private keys, the password has limited entropy, constrained by the memory of the user. Roughly speaking, the entropy of human memorable password is about 2 bits per character. Therefore, the goal of obtaining a legitimate communication parties' password by the attacker can be achieved within a reasonable time. Thus, the guessing attack on Ku and Wang's protocol should be considered as the realistic one.

In addition, suppose that the password P is compromised. The secrecy of previous session keys established by the legitimate parties is also affected. Since $Y_2^{Q^{-1}} = Key_1 = Key_2 \pmod n$, the attacker can easily recover the previous session keys by computing $Y_2^{Q^{-1}}$, where Y_2 is obtainable information over the network. Therefore, the protocol do not provide the perfect forward secrecy.

The weakness of the Ku and Wang's protocol is due to the two values X_1 and X in their key establishment and key validation phase, respectively. Since the values are publicly visible, an attacker capturing them can easily make a guess at legitimate communication parties' password by judging the correctness of the guess. Thus, the most important requirement to prevent the guessing attack is to eliminate the information that can be used to verify the correctness of the guess. The main idea of our scheme is to isolate such information by using asymmetric structure in the exchanged messages.

3 PKA: A Promising Key Agreement Protocol

In this section we demonstrate a new key agreement protocol resistant to the guessing attacks in which two users Alice and Bob can authenticate each other

and establish a common session key $K = h(V||W||g^{r_A r_B})$ for a secure communication.

3.1 System Setup

Two communication parties, called *Alice* and *Bob*, are assumed to share the common secret *password* π in a secure way. In protocol, all computations are performed in a finite field GF(n). In other words, a large prime number n is chosen ahead of time, and all multiplications and exponentiations are performed modulo n. Table 1 shows the notation used in this section. The value n and g are well-known values, agreed to beforehand.

Table 1. Notation

n	A large prime number. All computations are performed modulo n
g	A generator(primitive root) with order $n-1$
π	The user's password
r_A, r_B	Ephemeral private keys, generated randomly and not publicly revealed
$h()$	One-way hash function
$\|\|$	Concatenation
V, W	Corresponding public keys
K	Session key

3.2 Protocol Run

To establish a session key, Alice and Bob engage in the protocol as shown in Figure 1. The description of each step is as follows:

1. Alice chooses a random integer r_A, $1 < r_A < n$, computes $V = g^{r_A} h(\pi)$, and sends it to Bob.

2. Bob chooses his own random number r_B, $1 < r_B < n$, computes W, α, and X_B, where $W = g^{r_B}$, $\alpha = (V/h(\pi))^{r_B}$ by using the password $h(\pi)$ of Alice, and $X_B = h(\alpha)$. Then, sends W and X_B back to Alice.

3. Alice verifies if X_B is equal to $h(\beta)$, where $\beta = W^{r_A}$. If they match each other, Alice authenticates Bob. Then computes $X_A = h(\beta^{h(\pi)})$ and send it to Bob.

4. Bob verifies if X_A is equal to $h(\alpha^{h(\pi)})$. If they match each other, Bob authenticates Alice.

5. Finally, Alice and Bob agree on the common session key $K = h(V||W||\alpha) = h(V||W||\beta) = h(V||W||g^{r_A r_B})$.

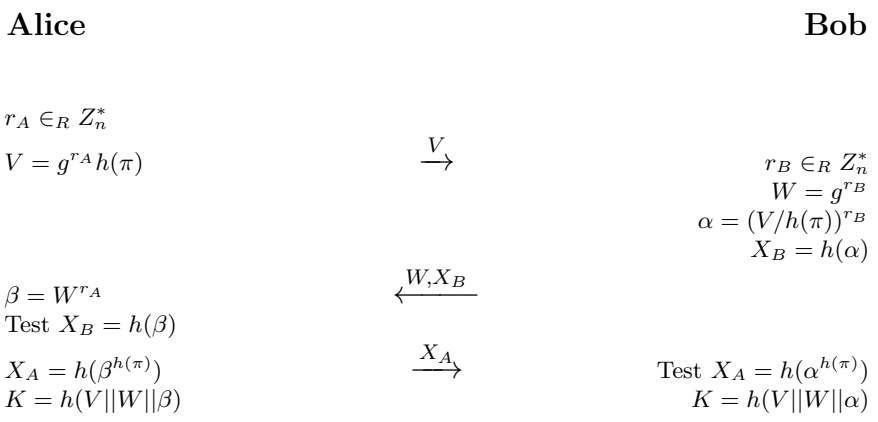

Fig. 1. PKA Protocol

Both sides will agree on the session key $K = h(V||W||g^{r_A r_B})$ if all steps are executed correctly. Once the protocol run completes successfully, both parties may use K to encrypt subsequent session traffic.

4 Security Analysis

In this section, we discuss the security of the proposed scheme by focusing on how the protocol protects against passive and active attack, to either obtain information about the password, or to obtain a shared session key without using the password.

Passive attack. If an attacker, called *Eve*, who eavesdrops on a successful PKA run can make a guess at the session key using only information obtainable over network and a guessed value of π, she could break a DH key exchange. The reason will be clear. Such a problem can be reduced to compute a keying material $g^{r_A r_B}$ from the value $V = g^{r_A} h(\pi)$ and $W = g^{r_B}$ in Figure 1. Thus, we claim that it is as difficult as to break the DH problem. Without the ability to compute the keying material $g^{r_A r_B}$, the message X_A and X_B leak no information to the passive off-line attacker. Since Alice and Bob do not leak any information either, a passive attacker cannot verify guesses at the user's password. Thus, PKA resists passive password guessing attacks.

Active attack. Active attacks can take many different forms, depending on what information is available to the attacker. An attacker who knows Alice's password π can obviously pretend to be Alice and communicate with Bob. Similarly, an attacker with π can masquerade as Bob when Alice tries to contact him. A man-in-the middle attack, which requires an attacker to fool both sides

of a legitimate conversation, cannot be carried out by an attacker who does not know Alice's password. For example, suppose that an attacker, Eve wants to fool Bob into thinking he is talking to Alice. First, she can compute $V' = g^{r_E} h(\pi')$ and send it to Bob. Bob computes $W = g^{r_B}$ and $X_B = h((V'/h(\pi))^{r_B})$ and send them to Eve. When Eve receives W and X_B from Bob, she has to make X_E and send it to Bob. Since the problem is combined with discrete logarithm and a secret password, she can not guess g^{r_B} and $h(\pi)$. Thus, the PKA withstands the man-in-the-middle attack.

Known-key attack. If K is revealed to a passive eavesdropper Eve, she does not learn any new information from combining K with publicly-visible information. This is true because the message X_A or X_B leak no information to the attacker. We have already established that Eve cannot make meaningful guesses at the session key K from guessed passwords, and there does not appear to be any easier way for her to carry out a brute-force attack. It means that the attacker, having obtained some past session keys, cannot compromise the current or future session keys. Thus, it resists the known-key attack.

Perfect forward secrecy. If the user's password itself is compromised, it does not allow the attacker to determine the session key K for past sessions and decrypt them. Since the attacker is still faced with the DH problem. Therefore, the PKA protocol satisfies the property of perfect forward secrecy.

5 Conclusion

In this paper, we presented that the Ku and Wang's protocol is vulnerable to an off-line password guessing attack and do not provide perfect forward secrecy. The attack illustrates that extreme care must be taken when passwords are used to provide user authentication in key agreement protocols. We also proposed a new key agreement protocol that resists the guessing attacks mounted by either passive or active network attackers, allowing low-entropy passwords to be used safely. The protocol has more efficient performance by reducing the number of protocol steps.

Acknowledgement. This work was supported by the Brain Korea 21 Project in 2003.

References

1. S.Blake-Wilson, A. Menezes: Authenticated Diffie-Hellman key agreement protocols. Proceedings of the 5th Annual Workshop on Selected Areas in Cryptography (SAC '98), Lecture Notes in Computer Science, 1556 (1999) 339–361
2. W.Diffie, M.E.Hellman: New directions in cryptography, IEEE Transaction on Information Theory, IT-22 (1976) 644–654

3. Seo D.H, Sweeney, P: Simple authenticated key agreement algorithm. Electronics Letters. Vol. 35 (1999) 1073–1074
4. Yuh-Min Tseng: Weakness in simple authenticated key agreement protocol. Electronics Letters. Vol. 36 (2000) 48–49
5. Iuon-Chang Lin, Chin-Chen Chang, Min-Shiang Hwang: Security enhancement for the simple authentication key agreement algorithm. Computer Software and Applications Conference. (2000) 113–115
6. Wei-Chi Ku, Sheng-De Wang: Cryptanalysis of modified authenticated key agreement protocol. Electronics Letters. Vol. 36 (2000) 1770–1771
7. T. Wu: The secure remote password protocol. In Proceedings of Internet Society Network and Distributed System Security Symposium. (1998) 97–111