Journal of Applied Sciences

ISSN 1812-5654

science alert

ANSInet
an open access publisher
http://ansinet.com

Journal of Applied Sciences 5 (2): 281-283, 2005
ISSN 1812-5654

Cryptanalysis of an Authentication Key Exchange Protocol

[1]Chou-Chan Yang, [2]Ya-Wen Yang and [3]Ting-Yi Chang
[1]Department of Management Information System, National Chung Hsing University,
250 Kuo Kuang Road, Taichung County, Taiwan 402, Republic of China
[2]Graduate Institute of Networking and Communication Engineering,
Chaoyang University of Technology, 168 Gifeng E. Rd., Wufeng,
Taichung County, Taiwan 413, Republic of China
[3]Department of Computer and Information Science,
National Chiao-Tung University, 1001 Ta Hsueh Road,
Hsinchu 300, Taiwan, Republic of China

Abstract: An authenticated multiple-key agreement protocol uses a digital signature to sign the Diffie-Hellman public keys which can generate multiple session key in a single round of message exchange and achieve mutual authentication. Recently, Hwang *et al.* proposed an authentication key exchange protocol which takes less computation time than Harn and Lin's protocol. However, this study, shows that Hwang *et al.*'s protocol is vulnerable to forgery attacks. An attacker can fool one communication part into believing the forged short-term public keys and share session keys with him.

Key words: Authentication, cryptanalysis, key agreement, multiple-key

INTRODUCTION

In 1976, the key agreement protocol was introduced by Diffie and Hellman[1]. The two parties were able to establish a secret session key over an insecure channel, such that the confidential information was transmitted securely. However, the Diffie and Hellman protocol could not provide authentication of the two parties. In other words, the two parties could not authenticate each other.

To solve this problem, there are two ways to integrate authentication into a key agreement protocol. One approach uses a pre-shared password[2]. With this pre-shared password, the session key can be established with user authentication. The other approach uses certificates (e.g. digital signatures), which provides authentication of the session key in key agreement protocols.

In 1995, the MQV key agreement protocol was proposed by Menezes *et al.*[3] The MQV key agreement protocol was the first key agreement protocol to use a digital signature to sign the Diffie-Hellman public keys without using one-way hash functions. Moreover, the MQV key agreement protocol was adopted as a standard by the IEEE P1363 committee[4].

In 1998, based on the MQV protocol, Harn and Lin[5] proposed an authenticated key agreement protocol without using one-way functions. Summarily, Harn and Lin's protocol contain the authentication for the Diffie-Hellman protocol. The two communication entities can establish multiple session keys in one round of interaction and use simple key computations.

Unfortunately, Yen and Joye[6] pointed out that Harn and Lin's protocol had a security flaw; it suffered from forgery attacks. If a valid short-term public key pair is given, an attacker can forge a new short-term public key pair and pass the verification procedure. In 2001, Harn and Lin[7] further proposed an improved protocol to avoid forgery attacks by modifying the signature signing equation. However, there was still a weakness in Harn and Lin's improved protocol. The common session keys generated by two parties were limited to the use of if two parties send n Diffie-Hellman public keys at a time. The purpose of this limit was to prevent the known-key attacks.

Recently, Hwang *et al.*[8] modified Harn and Lin's protocol using the XOR operation to decrease computational time and to use n^2-1 to provide perfect forward secrecy. In this study, we will show that

Corresponding Author: Ya-Wen Yang, Graduate Institute of Networking and Communication Engineering, Chaoyang University of Technology, 168 Gifeng E. Rd., Wufeng, Taichung County, Taiwan 413, Republic of China Tel: 886 4 23323000 7293 E-mail: s9230603@mail.cyut.edu.tw

Hwang *et al.*'s protocol suffers from forgery attacks. An attacker can fool one communication part into believing the forged short-term public keys and share session keys with him.

Review of Hwang *et al.*'s protocol: In the Diffie-Hellman scheme, the system publishes two values p and g, where, p is a large prime and g is a generator with order p-1 in GF(p). Each user in the system selects a long-term secret key $x \in GF$ and computes a corresponding long-term public key $y=g^x \bmod p$. Assume the two communication parties are Alice and Bob. The long-term secret key and the long-term public key for Alice is (x_A and y_A) and for Bob, it is (x_B and y_B), respectively. The following are the steps needed to establish multiple common session keys.

Step 1: Alice privately selects two random integers k_{A1} and k_{A2} to be short-term secret keys and computes short-term public keys $r_{A1}=g^{k_{A1}} \bmod p$ and $r_{A2}= g^{k_{A2}} \bmod p$. The signature value s_A can be obtained by computing the following equation:

$$S_A = x (r_{A1} \oplus r_{A2}) + k_{A1} \bmod (p-1)$$

Alice then sends the authenticated message $\{r_{A1}, r_{A2}, s_A, cert(y_A)\}$ where, $cert(y_A)$ is the certificate of Y_A to Bob.

Step 2: Bob follows the same procedure as Alice. He chooses two integers, k_{B1} and k_{B2} and computes $\{r_{B1}, r_{B2}, s_B\}$. Then Bob sends $\{r_{B1}, r_{B2}, s_B, cert(y_B)\}$ to Alice.

Step 3: After receiving the message $\{r_{B1}, r_{B2}, s_B, cert(y_B)\}$ sent from Bob, Alice checks the following verification equation:

$$g^{s_B} = y_B^{(r_{B1} \oplus r_{B2})} . r_{B1} \bmod p$$

Once the verification is valid, Alice uses r_{B1} and r_{B2} to compute four common session keys.
-pjas

$$K_1 = r_{B1}^{k_{A1}} \bmod p \frac{\Delta y}{\Delta x}$$

$$K_2 = r_{B1}^{k_{A2}} \bmod p$$

$$K_3 = r_{B2}^{k_{A1}} \bmod p$$

$$K_4 = r_{B2}^{k_{A2}} \bmod p$$

Step 4: Bob still uses the same procedure as Alice. After receiving the message $\{r_{A1}, r_{A2}, s_A, cert(y_A)\}$, he checks the following verification equation:

$$g^{s_A} = y_A^{(r_{A1} \oplus r_{A2})} . r_{A1} \bmod p$$

If the verification equation holds, Bob also uses r_{A1} and r_{A2} to compute four common session keys using the following equation:

$$K_1 = r_{A1}^{k_{B1}} \bmod p$$

$$K_2 = r_{A2}^{k_{B1}} \bmod p$$

$$K_3 = r_{A1}^{k_{B2}} \bmod p$$

$$K_4 = r_{A2'}^{k_{B2}} \bmod p$$

The four common session keys have been established successfully by Alice and Bob.

Forgery attack by an attacker

Step 1: An attacker intercepts the message $\{r_{A1}, r_{A2}, s_A, cert(y_A)\}$ from Alice. The attacker chooses a random integer k_{C1} and the corresponding $r_{C1}= g^{k_{c1}}$ and computes 'r_{A2} as follows:

$$r_{A2}' = r_{A2} \oplus r_{C1}$$

Step 2: The attacker impersonates Alice in order to share keys with Bob. He/she sends $\{r_{A1}, r_{A2}', r_{C1}, s_A, cert(y_A)\}$ to Bob. When Bob receives the message, he thinks that Alice wants to share 9 keys with him and verifies $\{r_{A1}, r_{A2}', r_{C1}\}$ by checking $g^{S_A} = y_A^{(r_{A1} \oplus r_{A2} \oplus r_{c1})} . r_{A1} \bmod p$

In fact,

$$y_A^{(r_{A1} \oplus r_{A2}' \oplus r_{C1})} . r_{A1} \bmod p = y_A^{(r_{A1} \oplus r_{A2} \oplus r_{C1} \oplus r_{C1})} . r_{A1} \bmod p = g^{S_A}$$

The attacker can then successfully share $K_1' = r_{B1}^{k_{C1}} \bmod p, K_2' = r_{B2}^{k_{C1}} \bmod p$ and $K_3' = r_{B3}^{k_{C1}} \bmod p$ with Bob.

With this attack, since Alice's short term public key can be forged, the attacker can fool Bob into believing that he has shared 9 keys with Alice. However, Alice has actually only shared 4 keys with Bob. The attacker can therefore use forged short term public keys to compute 3 session keys and share them with Bob.

CONCLUSION

We have pointed out that Hwang *et al.*'s protocol is insecure, since an attacker can easily share some session keys with others only if he/she gets an old message from a legitimate user.

REFERENCES

1. Diffie, W. and M. E. Hellman, 1976. New directions in cryptography. IEEE Transaction on Information Theory, IT-22: 644-654.
2. Seo, D. and P. Sweeney, 1999. Simple authenticated key agreement algorithm. IEE Electronics Lett., 35: 1073-1074.
3. Menezes, A.J., M. Qu and S.A. Vanstone, 1995. Some key agreement protocols providing implicit authentication. Proceeding of the Second Workshop on Selected Areas in Cryptography (SAC'95), pp: 22-32.
4. IEEE P1363 Working Group, 2001. IEEE p1363a D10 (Draft Version 10): Standard Specifications for Public Key Cryptography: Additional Techniques, IEEE P1363 WorkingGroup, Workin draft, http://grouper.ieee.org/groups/1363/.
5. Harn, L. and H.Y. Lin, 1998. An authenticated key agreement protocol without using one-way function. In Proc. 8th Natl. Conf. Information Security, Kaohsiung, Taiwan, May, pp: 155-160.
6. Yen, S. and M. Joye, 1998. Improved authenticated multiple-key agreement protocol. IEE Electronics Lett., 34: 1738-1739.
7. Harn, L. and H.Y. Lin, 2001. Authenticated key agreement protocol without using one-way function. IEE Electronics Lett., 37: 629-630.
8. Ren-Junn, H. S. Sheng-Hua and L. Chih-Hua, 2003. An enhanced authentication key exchange protocol. In: Proceedings of the 17th Intl. Conf. Advanced Information Networking and Applications, pp: 202-205, March 2003.