WEAKNESSES IN EHA AUTHENTICATION AND KEY DISTRIBUTION PROTOCOL

Martin Stanek and Daniel Olejár
Department of Computer Science
Faculty of Mathematics and Physics
Comenius University
842 15 Bratislava, Slovakia
stanek@dcs.fmph.uniba.sk
olejar@dcs.fmph.uniba.sk

Abstract Recently, there was a new authentication and key distribution protocol presented in [EHA98]. In this paper we show that certain claims on its properties are not valid. We also suggest some modifications to strengthen this protocol.

Keywords: cryptographic protocols, protocol analysis

1. INTRODUCTION

Authentication and key distribution protocols constitute the basis of security in many distributed systems. Kerberos [KNT94], authenticated Diffie-Hellman [Fo94], SPX [TA91] and others are to the well-known authentication and/or key distribution protocols. Various additional requirements, such as performance, encryption systems used, security goals, lead to the design of new cryptographic protocols. Every (new) protocol has to be analysed carefully in great detail to avoid bugs, security weaknesses and redundancy – see e.g. [BAN90, Me92]. A good survey of formal methods for the analysis of authentication protocols is [RH93].

Recently, a new protocol for authentication and key distribution has been presented ([EHA98]). In this paper it will be referred to as EHA. In order to prove that the proposed protocol satisfies stated goals, the authors used the BAN logic, see e.g. [BAN90]. However, despite this formal analysis of the EHA, the protocol still contains several weaknesses

B. Preneel (ed.), *Secure Information Networks*

which do not stem from the BAN logic itself but rather from the context in which the logic was used. It is worth mentioning that the BAN logic does not provide the "proof" of security, it just increases confidence in protocols. Recently, more useful complexity-theoretic approaches have emerged, see e.g. [BR95].

First, we describe the protocol. Then, we show its weaknesses and propose some modifications of the EHA protocol to strengthen (and fix) it.

2. DESCRIPTION OF EHA PROTOCOL

The EHA protocol consists of three modules: the setup module, the login module and the authentication/key distribution module. It uses both symmetric encryption and public key encryption/key distribution algorithms. The protocol requires a trusted third party, the so-called Security Management Facility (SMF).

2.1 SETUP MODULE

The setup module is executed once, when a new station (participant) has joined the network. Let us suppose a principal C wants to join the network. His communication with the SMF consists of three steps.

Setup module:

1. $C \rightarrow SMF$: C, K_C
2. $SMF \rightarrow C$: $SMF, K_C, a, p, N_{SMF}, \{K_C, a, p, N_{SMF}\}_{K_{SMF}^{-1}}$, $\{K_{SMF,C}, \{K_{SMF,C}\}_{K_{SMF}^{-1}}\}_{K_C}$
3. $C \rightarrow SMF$: $C, \{N_{SMF}, a^z \bmod p\}_{K_{SMF,C}}$

K_X denotes the public key of a principal X (C or SMF). Message m encrypted with key K will be denoted by $\{m\}_K$. The protocol assumes that the participants know the public key of the SMF. First, C sends his identity and public key. The SMF generates a symmetric key $K_{SMF,C}$ (this key is supposed for use in subsequent modules for trusted communication between the SMF and C). It answers with its identity, generator a and modulus p in the Diffie-Hellman system, and nonce N_{SMF}. The SMF sends the signature of these parameters generated by means of its private key K_{SMF}^{-1}. It also sends key $K_{SMF,C}$ and its signature, both encrypted with the public key of C - K_C. Principal C verifies the signature, decrypts the $K_{SMF,C}$ key and checks its signature. C chooses his key z in Diffie-Hellman and replies with his public key ($a^z \bmod p$) and nonce both encrypted with $K_{SMF,C}$. The SMF decrypts the received message and checks the correctness of nonce. Having successfully finished the module, the SMF stores the public key ($a^z \bmod p$) and the

communication key $K_{SMF,C}$ for C. Participant C stores the communication key $K_{SMF,C}$, too.

2.2 LOGIN MODULE

This module is executed during the login to the system (network). The aim of this module is to change symmetric keys between the SMF and principals, and the public keys of principals in Diffie-Hellman to decrease the risk of their exposure. After executing the login module, both C and the SMF share a new symmetric key $K'_{SMF,C}$ (generated by SMF) and C has a new public key $a^{z'} \bmod p$ (generated by C) which is stored in the SMF.

Login module:

1. $C \rightarrow SMF$: $C, \{a^{z'} \bmod p\}_{K_{SMF,C}}$
2. $SMF \rightarrow C$: $SMF, N_{SMF}, \{a^{z'} \bmod p, K'_{SMF,C}\}_{K_{SMF,C}}$
3. $C \rightarrow SMF$: $C, \{N_{SMF}, a^{z'} \bmod p\}_{K'_{SMF,C}}$

N_{SMF} denotes (again) nonce generated by the SMF. Principal C checks the freshness of the received message in step 2 by comparing $a^{z'} \bmod p$ with the one already sent. The SMF ensures that the message in step 3 is fresh using the nonce N_{SMF}.

2.3 AUTHENTICATION AND KEY DISTRIBUTION MODULE

This module is executed when two principals A and B want to communicate securely. The symmetric key K is computed as $a^{xy} \bmod p$. The goals of the module are (as stated in [EHA98]):

> "As a result of execution of the authentication and key distribution module both A and B authenticate each other and establish the symmetric key $a^{xy} \bmod p$."

Symmetric key K is computed as $a^{xy} \bmod p$, just like in the Diffie-Hellman key exchange protocol, where $a^x \bmod p$ is the public key of A and $a^y \bmod p$ is the public key of B.

Authentication and Key Distribution module:

1. $A \rightarrow B$: A, N_A
2. $B \rightarrow A$: A, N_A, B, N_B
3. $A \rightarrow SMF$: A, N_A, B, N_B
4. $SMF \rightarrow A$: $\{B, N_A, a^y \bmod p\}_{K_{SMF,A}}$, $\{A, N_B, a^x \bmod p\}_{K_{SMF,B}}$
5. $A \rightarrow B$: $\{A, N_B, a^x \bmod p\}_{K_{SMF,B}}$, $\{N_B\}_K$
6. $B \rightarrow A$: $\{N_A\}_K$

Description of symbols used in the module:
N_A – nonce generated by A;
N_B – nonce generated by B;
x – private key of A for Diffie-Hellman;
$a^x \bmod p$ – public key of A in Diffie-Hellman;
y – private key of B for Diffie-Hellman;
$a^y \bmod p$ – public key of B in Diffie-Hellman.

Upon receiving a reply from the SFM in step 4 A decrypts the first part of the message and checks nonce N_A to ensure the freshness of the message. Then A computes the symmetric key $K = (a^y \bmod p)^x \bmod p$. Analogously, principal B decrypts the first part of the message in step 5 and checks nonce N_B. B computes key $K = (a^x \bmod p)^y \bmod p$ and verifies the second part of the received message. A performs an additional check (using nonce N_A) after step 5 of this module.

The BAN logic analysis performed in [EHA98] yielded the following results (described in the BAN logic syntax):

$$A \mid\equiv A \overset{K}{\longleftrightarrow} B \qquad A \mid\equiv B \mid\equiv A \overset{K}{\longleftrightarrow} B$$
$$B \mid\equiv A \overset{K}{\longleftrightarrow} B \qquad B \mid\equiv A \mid\equiv A \overset{K}{\longleftrightarrow} B$$

So, principal A believes that K is a good key for communication with B and B believes that K is a good key for communication with A. Moreover, A believes that B believes in "goodness" of K and vice-versa.

3. WEAKNESSES

In this section we present weaknesses in the EHA protocol. First notice the lack of "authenticity checks" in the setup module. The SMF doesn't authenticate principal C, and hence the SMF has no guarantee whatsoever about who it is that knows the shared secret key $K_{SMF,C}$, and whose public key K_C is.

3.1 FAKE AUTHENTICITY

The BAN logic analysis performed in [EHA98] deals only with the authentication and key distribution module. Therefore, initial assumptions in the analysis are based on the correctness and security of the setup and the login modules. Our attack is based on the observation that the SMF does not check in the login module whether the principal actually knows the private key corresponding to the submitted public key.

Let us denote an attacker by E. He wants to convince principal B that his identity is A. E waits until A initializes communication with

him, i.e. starts the authentication and key distribution module. We denote the steps in this instance of module with prefix "AE":

AE1. $A \rightarrow E$: A, N_A

Immediately, E finds out B's current public key. He obtains this key from the SMF through sending the message: $E \rightarrow SMF$: E, N'_E, B, N'_B, where N'_E and N'_B are nonces generated by E himself. The SMF assumes that step 3 in the authentication and key distribution module has been performed. Thus, the SMF answers with the message containing B's current public key. Nobody is affected by this "investigation". The attacker can begin his session with B (steps in this instance of the module will be denoted with prefix "EB").

EB1. $E \rightarrow B$: A, N_E
EB2. $B \rightarrow E$: A, N_E, B, N_B
EB3. $E \rightarrow SMF$: A, N_E, B, N_B
EB4. $SMF \rightarrow E$: $\{B, N_E, a^y \bmod p\}_{K_{SMF,A}}$,
$\{A, N_B, a^x \bmod p\}_{K_{SMF,B}}$

E changes his public key to $a^y \bmod p$ (B's public key) in the login module. He is able to do this because the SMF does not verify the knowledge of the private key corresponding to the submitted public key. Then, he proceeds in communication with A (notice, N_B is the same nonce as chosen by principal B and announced in EB2):

AE2. $E \rightarrow A$: A, N_A, E, N_B
AE3. $A \rightarrow SMF$: A, N_A, E, N_B
AE4. $SMF \rightarrow A$: $\{E, N_A, a^y \bmod p\}_{K_{SMF,A}}$,
$\{A, N_B, a^x \bmod p\}_{K_{SMF,E}}$
AE5. $A \rightarrow E$: $\{A, N_B, a^x \bmod p\}_{K_{SMF,E}}$, $\{N_B\}_K$,

where $K = a^{xy} \bmod p$. Now, the attacker can use the message $\{N_B\}_K$ to proceed with the step 5 in "EB" protocol:

EB5: $E \rightarrow B$: $\{A, N_B, a^x \bmod p\}_{K_{SMF,B}}$, $\{N_B\}_K$
EB6: $B \rightarrow E$: $\{N_E\}_K$

The authentication and key distribution module was finished successfully. Principal B believes that E is A and K is a good key for communication with A. In fact, E does not know the value of K, but it has convinced B of his fake identity. However, since the authenticity is often the only check performed in various situations, there is a serious risk in using this protocol in such contexts.

Moreover, it is reasonable to assume that E can continue to communicate with B for a while. Principal A will probably try to reestablish communication with E. This enables E to use key K to encrypt an

arbitrary plaintext (as he did with N_B) in this execution of the authentication and key distribution module.

The simplest method to avoid this threat is to enforce checking that the public key submitted to the SMF in the setup and login modules is unique. This solution, however, is purely implementational and requires further specification of how to deal with conflicts in the protocol. Moreover, it cannot ensure that the principal knows the private key corresponding to the sent public key. So, he can send a key which is, actually, different but related to the public key of some principal (such as its square etc.). Thus, constructed session keys are related as well.

Another method of avoiding this threat could be based on the usage of key K for authentication. That is, the last steps in the authentication and key distribution module should be modified to perform mutual authentication of A and B. On the other hand, one has to keep in mind the attacker's ability to encrypt the chosen text when A tries to reestablish the session.

A robust way to ensure the possession of the private key corresponding to the submitted public key is the zero-knowledge proof for discrete logarithm, see [CEGP88]. On the other hand, such solution substantially increases the communication overhead.

Another solution can be based on challenge-response schemes where the SMF challenges the principal to encrypt something (generated by the SMF itself) with his private key. Further check (decryption) ensures the possession of the private key by the principal.

3.2 COMPROMISING SMF

The authors of EHA protocol state that this protocol avoids security threats regarding compromising the SMF:

> "Moreover, in order to avoid the security threats resulting from compromising the SMF, only Diffie-Hellman components of the symmetric keys are stored in the SMF."

Let us briefly discuss the situation after the SMF security has been compromised. This means that all keys stored in the SMF are revealed. Actually though, no old communication is compromised – the private parts of principals for Diffie-Hellman are not communicated over the network at all, therefore, they are not compromised, either. However, such statements as the one above can lead to the misinterpretation of security threats. It is reasonable to assume that symmetric keys ($K_{SMF,X}$) and/or the private key of the SMF (K_{SMF}^{-1}) are revealed to an attacker. Due to the greater difficulty in protecting a table of keys than a single key, it can be worth distinguishing between compromising the symmet-

ric keys and compromising the SMF's private key. In both situations, as expected, the attacker can mount various attacks. The compromise of the SMF's private key affects the security of the setup module. In the case of the compromised symmetric keys, the "man in the middle" attack is the most efficient one.

Let us assume the authentication and key distribution module and denote the attacker as E. The first and the second steps of the module are executed as described. E intercepts the third message and, in step 4, sends A the message:

$$\{B, N_A, a^w \bmod p\}_{K_{SMF,A}}, \quad \{A, N_B, a^w \bmod p\}_{K_{SMF,B}},$$

where w is E's own private key (possibly generated only for this session). The principal A computes key $K_1 = a^{xw} \bmod p$ and uses it for constructing the second part of the message in step 5. E intercepts (again) this message and sends B the following:

$$\{A, N_B, a^w \bmod p\}_{K_{SMF,B}}, \quad \{N_B\}_{K_2},$$

where $K_2 = a^{yw} \bmod p$. The module is finished by replacing the last message with $\{N_A\}_{K_1}$ by E. After this, A believes that K_1 is a good key for communication with B and B believes that K_2 is a good key for communication with A. Now, E can act as the "man in the middle", reading and controlling all the communication.

It should be also noted that dependency of the new keys $K_{SMF,X}$ on the old ones is very straightforward. Thus, compromising any of the old keys immediately leads to compromising the current key $K_{SMF,X}$. As a consequence, the attacker can change the public key of principal X using the login module and act in the network with X's identity.

3.3 REDUNDANCY

Dealing with redundancy is at least questionable once other weaknesses in the protocol are detected. On the other hand, it is worth considering this aspect of the protocol to see the authors' effort in designing the latter. Although we didn't find any redundant steps in the modules, there is a remarkable redundancy in several messages:

1. Step 2 in the authentication and key distribution module: there is no need to repeat A's ID and its nonce (A already knows it);

2. Step 3 in the login module: new public key ($a^{z'} \bmod p$) is redundant in the encrypted message, the SMF knows the key and this message only has to convince the SMF of the identity of C and of the correct transmission of $K'_{SFM,C}$.

Looking at the authentication and key distribution module we can point out that using encryption to protect the public keys is unnecessary, since the SMF is not distributing secret information. The use of MACs (message authentication codes) for integrity and origin protection would be more appropriate.

Given that both A and B have to give their public key to the SMF in advance, it would be much simpler for the SMF to generate certificates for A's and B's public Diffie-Hellman keys, which they could then exchange as necessary without any further intervention by the SMF.

4. CONCLUSION

We discussed the properties of the recently proposed EHA protocol for authentication and key distribution. We also showed that, despite the authors' application of the BAN logic analysis, some of their claims on the protocol are not valid or, at least, should be treated with caution and require deeper evaluation. The paper presents solutions for the "fake authenticity" weakness. To conclude, one has to be very careful when choosing this protocol for practical use.

Acknowledgments

We would like to thank all anonymous referees for many helpful suggestions, especially regarding the authenticity problem in the setup module and the usage of MACs and certificates.

References

[BR95] Bellare M., Rogaway P.: Provably secure session key distribution – the three party case, Proceedings of the 27th Annual Symposium on the Theory of Computing, ACM, pp. 57-66, 1995.

[BAN90] Burrows M., Abadi M., Needham R.M.: A Logic of Authentication, ACM Transactions of Computer Systems, Vol. 8, No. 1, pp. 18-36, 1990.

[CEGP88] Chaum D., Evertse J.H., van der Graff J., Peralta R.: An Improved Protocol for Demonstrating Possesion of Discrete Logarithms and Some Generalizations, Advances in Cryptology – EUROCRYPT'87 Proceedings, Springer-Verlag, pp. 127-141, 1988.

[EHA98] El-Hadidi M.T., Hegazi N.H., Aslan H.K.: Logic-Based Analysis of a New Hybrid Encryption Protocol for Authentication and Key Distribution, IFIP 14th International Conference

on Information Security, 15th World Computer Congress, pp. 173–183, 1998.

[Fo94] Ford W.: Computer Communications Security: Principals, Standard Protocols and Techniques, Prentice-Hall, 1994.

[KNT94] Kohl J.T., Neuman B.C., Tso T.: The Evolution of the Kerberos Authentication System, Distributed Open Systems, IEEE Computer Society Press, pp. 78–94, 1994.

[Me92] Meadows C.: Applying Formal Methods to the Analysis of a Key Management Protocol, Journal of Computer Security, Vol. 1, No. 1, pp. 5–53, 1992.

[RH93] Rubin A.D., Honeyman P.: Formal Methods for the Analysis of Authentication Protocols, CITI Technical Report 93-7, Dept. of Electrical Engineering and Computer Science, University of Michigan, 1993.
www.citi.umich.edu/techreports/reports/citi-tr-93-7.pdf

[TA91] Tardo J.J., Alagappan K.: SPX: Global Authentication Using Public Key Certificates, IEEE Privacy and Security Conference, 1991.