View metadata, citation and similar papers at core.ac.uk

brought to you by CORE

provided by Loughborough University Institutional Repository

This item was submitted to Loughborough's Institutional Repository (https://dspace.lboro.ac.uk/) by the author and is made available under the following Creative Commons Licence conditions.

creative commons

COMMONS DEED

Attribution-NonCommercial-NoDerivs 2.5

You are free:

- to copy, distribute, display, and perform the work

Under the following conditions:

Attribution. You must attribute the work in the manner specified by the author or licensor.

Noncommercial. You may not use this work for commercial purposes.

No Derivative Works. You may not alter, transform, or build upon this work.

- For any reuse or distribution, you must make clear to others the license terms of this work.
- Any of these conditions can be waived if you get permission from the copyright holder.

Your fair use and other rights are in no way affected by the above.

This is a human-readable summary of the Legal Code (the full license).

Disclaimer

For the full text of this licence, please go to:
http://creativecommons.org/licenses/by-nc-nd/2.5/

Attacks on Improved Key Distribution Protocols with Perfect Reparability

Raphael C.-W. Phan[1], *Member, IEEE*; and Bok-Min Goi[2], *Member, IEEE*
[1]*Electronic & Electrical Engineering, Loughborough University, UK*
r.phan@lboro.ac.uk
[2]*Centre for Multimedia Security and Signal Processing,*
Cryptography & Information Security Research Group,
Faculty of Engineering, Multimedia University, Malaysia
bmgoi@mmu.edu.my

Abstract

In this paper, we present attacks on two improved key distribution protocol with perfect reparability that were presented at ICON 2000. First, we show that the two "attacks" described in their paper are trivial and do not count as attacks at all since they are well-known attacks that apply to any security system. Further, we describe several attacks on both improved protocols, and show that an illegitimate attacker could easily impersonate legitimate parties and have other parties think they are sharing keys with the impersonated party when in fact that party is not present at all.

1. Introduction

It is often desirable within a distributed computing environment that users who wish to communicate with each other be able to do so securely over the network, despite the fact that the communication channel might be insecure or easily tapped into. To achieve this, users typically make use of *encryption* [7, 9] to secure the information transmitted back and forth. Note that encryption only provides confidentiality. Integrity and authenticity employs other cryptographic primitives.

Secret-key encryption schemes [7, 9] are preferred over their public-key encryption counterparts as they are faster and more efficient. However, unlike the latter, secret-key encryption schemes require the same secret key to be used in the encryption and decryption process, hence the problem of key distribution is a vital issue, ie how do two communicating parties securely inform each other of what secret key to use in the encryption? This is solved by *key distribution protocols*.

In ICON 2000, Li et al. [5] presented weaknesses of Hwang and Ku [3] two reparable key distribution protocols and then proceeded to present improved versions of these two protocols.

One thesis that we maintain is that security-providing protocols, as is a key distribution protocol, should be designed carefully not only to withstand known attacks but also be robust against any future ones. Otherwise, the purpose of the protocol, that of achieving security against attacks in the first place, is lost. Yet in literature, many insecure protocols continue to be designed, and subsequently many more variants are proposed by tweaking the insecure protocols. This tweak-and-break cycle goes on ad infinitum because insecure variants keep triggering tweaks that often result again in insecure variants; this latter should be discouraged. To be precise, no more variants should be proposed by mere tweaks; rather they should be accompanied with rigorous security analysis within well defined security models.

In this paper, we show that the weaknesses presented by Li et al. are trivial. Further, we present several attacks on their two improved protocols, which for lack of better names we simply call as IKDP1 and IKDP2 respectively. Our results show that these protocols are extremely insecure and should not be used at all; thus supporting our above thesis about tweak-and-break cycles.

2. Previous Work

In this section, we review previous work by Li et al., namely the weaknesses they mentioned of Hwang and Ku's protocols, and their suggested improvements.

2.1 Hwang and Ku's Protocols

Hwang and Ku's [3] key distribution protocols are given by:

978-0-7695-3492-3/08 $25.00 © 2008 IEEE
DOI 10.1109/EUC.2008.161

IEEE computer society

Authorized licensed use limited to: LOUGHBOROUGH UNIVERSITY. Downloaded on December 16, 2009 at 04:48 from IEEE Xplore. Restrictions apply.

1: $A \rightarrow B$ A, EM_A
2: $B \rightarrow A$ $\{EM_A, SK\}_{K\_ab}$

A and B denote the two communicating parties, while EM_A denotes a certain event marker [1] generated by A and locally kept in synchronization at A. SK denotes the secret session key generated by B for use in the secret-key encryption of messages. Here, $\{x\}_K$ denotes the secret-key encryption of message x under control of the secret key, K.

Meanwhile, their second key distribution protocol is:

1: $A \rightarrow B$ A, EM_A
2: $B \rightarrow AS$ A, EM_A, B, EM_B
3: $AS \rightarrow A$ $\{EM_B, A, SK\}_{MK\_b}, \{EM_A, B, SK\}_{MK\_a}$
4: $A \rightarrow B$ $\{EM_B, A, SK\}_{MK\_b}$

MK_a and MK_b denote the master keys owned and known only by A and B respectively. Note also that AS denotes the authentication key server trusted by both parties, hence he also knows what MK_a and MK_b are.

2.2 Li et al.'s Improved Protocols

Li et al.'s proposed improvement of the Hwang and Ku's key distribution protocol, KDP1, is as follows:

1: $A \rightarrow B$ $A, \{EM_A\}_{K\_ab}$
2: $B \rightarrow A$ $\{EM_A, SK\}_{K\_ab}$

From now on, we will call this improvement the IKDP1. Note that the only difference between this "improvement" and the original KDP1 proposed by Hwang and Ku is that the EM_A is encrypted.

Meanwhile, Li et al.'s proposed improvement of Hwang and Ku's second key distribution protocol, KDP2, is as follows:

1: $A \rightarrow B$ $\{A, EM_A\}_{MK\_a}$
2: $B \rightarrow AS$ $\{A, EM_A\}_{MK\_a}$, $\{B, EM_B \oplus R_B\}_{MK\_b}$
3: $AS \rightarrow A$ $\{EM_B \oplus R_B, A, SK\}_{MK\_b}, \{EM_A, B, SK\}_{MK\_a}$
4: $A \rightarrow B$ $\{EM_B \oplus R_B, A, SK\}_{MK\_b}$

From now on, we will call this improvement the IKDP2. Here, R_B is the random number generated by B when he generates EM_B.

2.3 Remarks on Trivial Previous "Attacks" on Hwang and Ku's Protocols

Li et al. presented in [5] two "attacks" on the key distribution protocols proposed by Hwang and Ku [3].

Their first "attack", which they call an eavesdropping or passive attack, observes that the plaintext value of the event marker, EM_A is available in message 1, while its corresponding encrypted ciphertext value is available in message 2, assuming that encryption has been done in the cipher block chaining (CBC) mode [7, 9]. Therefore, they remarked that one could perform a *dictionary attack* (more popularly known as *brute-force exhaustive key search*) on the plaintext-ciphertext pair of EM_A and $\{EM_A\}_{K\_ab}$ by guessing all possible values of K_ab and encrypting EM_A to see if $\{EM_A\}_{K\_ab}$ is obtained.

However, we remark that this is very trivial since it is well-known that any system using a secret key would be susceptible to such a dictionary attack. In fact, dictionary attacks are not at all practical but are simply used as a benchmark to compare other more practical attacks on a system. Therefore, Li et al.'s first "attack" is no attack at all.

Li et al's second "attack" bases on the same assumption that an attacker has to mount a dictionary attack that requires guessing all possible secret key values. Again, this is trivial and not an attack at all.

3 Attacks on the Improved Protocols

We present in this section several attacks on Li et al.'s improved key distribution protocols.

3.1 Attacks on First Improved Protocol IKDP1

We consider the simplest type of attacks on protocols, namely *replay attacks* which work by simply replaying previous messages captured by an attacker, C, by merely eavesdropping on the communication channel.

Our first such attack, a special type of replay attack known as the *unknown key-share attack* [2, 4, 8] is as follows:

α.1: $A \rightarrow B$ $A, \{EM_A'\}_{K\_ab}$
α.2: $B \rightarrow A$ $\{EM_A', SK'\}_{K\_ab}$

β.1: $C_A \rightarrow B$ $A, \{EM_A'\}_{K\_ab}$
β.2: $B \rightarrow C_A$ $\{EM_A', SK^*\}_{K\_ab}$

Authorized licensed use limited to: LOUGHBOROUGH UNIVERSITY. Downloaded on December 16, 2009 at 04:48 from IEEE Xplore. Restrictions apply.

Here, C_A denotes the attacker, C impersonating *A* while EM_A' and *SK'* are the event marker and session key used in a previous protocol run. α denotes a valid previous protocol run while β denotes a false protocol run initiated by the attacker.

During a previous protocol run, the attacker, C has captured messages 1 and 2 being communicated between A and B. Now, after any valid protocol run between A and B, denoted by α in this case, the attacker, C impersonates A and replays a previously captured message 1 as a supposedly new message β.1 to B. B notices nothing amiss since this was a valid message generated by A, except that it is already outdated! B therefore responds with message β.2 to A, but this is intercepted by C.

B now thinks that he shares a new session key, SK* with A when in fact A does not even know of this protocol run, β and still thinks that he shares the old key, SK with B. This attack shows a total failure of the Li et al.'s first improved key distribution protocol.

Our second attack is again an unknown key-share attack but interleaves two protocol runs. Such attacks are sometimes also called *interleaving attacks* [10] or *parallel-session attacks*. It works as follows:

$\alpha.1: A \rightarrow C_B \quad A, \{EM_A\}_{K\_ab}$
$\beta.1: C_B \rightarrow A \quad B, \{EM_A\}_{K\_ab}$
$\beta.2: A \rightarrow C_B \quad \{EM_A, SK\}_{K\_ab}$
$\alpha.2: C_B \rightarrow A \quad \{EM_A, SK\}_{K\_ab}$

A desires to establish a new protocol run with B but this is intercepted by an attacker who impersonates B. This attacker is denoted as C_B. Then, C_B in continuing to impersonate B, immediately initiates another new protocol run with A by simply replaying the $\{EM_A\}_{K\_ab}$ component of message α.1 to form β.1. A is therefore led into thinking that as he is having the protocol run, α with B, B is starting another new protocol run, β with A. Since message β.1 is supposed to be a message generated by B for A, it is B's responsibility to check that the event marker contained within β.1 is in proper non-repeating order and thus A does not check it against his own record of previous event markers generated by A.

A, in response to message β.1, replies with message β.2 containing the new session key, SK that he would share with B. This is however received by C_B instead since he is impersonating B. Protocol run, β is now complete with A thinking he is sharing the key, SK with B when in fact B is not even involved. Further, C_B immediately replays message β.2 as its message α.2 to A. Upon receipt of this, A thinks that protocol run, α is also complete and that he shares another secret key, SK with B when again, B is not involved. This attack shows a total failure of authentication of the IKDP1 protocol since an attacker could by simple replaying of messages, lead a legitimate party, A into thinking he is communicating and establishing secret keys with another legitimate party, B when in fact he is doing so with an illegitimate party, C while B is not present at all! It also shows how C could manipulate A as an oracle into generating valid messages from itself to itself, and have A believe that the messages are generated from another legitimate party.

Our third attack is a type of replay attack known as the *multiplicity attack* [6]. It proceeds as follows:

$\alpha.1: A \rightarrow B \quad A, \{EM_A\}_{K\_ab}$
$\alpha.2: B \rightarrow A \quad \{EM_A, SK\}_{K\_ab}$

$\beta.1: C_A \rightarrow B \quad A, \{EM_A\}_{K\_ab}$
$\beta.2: B \rightarrow C_A \quad \{EM_A, SK\}_{K\_ab}$

Protocol run α is a normal previous run between legitimate parties A and B. The attacker, C has eavesdropped on this protocol run, and immediately after it ends, impersonates A and replays message α.1 as a new message β.1 to B. Thinking that this is a new initiation message from A, B responds with message β.2.

Therefore, although A only meant to execute one protocol run with B, B thinks that A has executed two protocol runs with it. Again, since the event marker is generated and sent by A, it is A's responsibility to check it for correct order and so B does not detect anything amiss. Multiplicity attacks are advantages in certain situations such as in banking applications where A and C could be account holders while B is the bank. A initiates one protocol run with B as a direction to B to transfer $1000 from A's account to C's account as a payment for something. C mounts this attack and causes B to transfer $1000 for two times instead of once so C has cheated A of an extra $1000. In fact, C could repeat this attack indefinitely and cause B to transfer the money as many times as C likes.

Our final attack is a denial of service (DoS) attack, as follows:

$\alpha.1: C_A \rightarrow B \quad A, \{EM_C\}_{K\_cx}$
$\alpha.2: B \rightarrow C_A \quad \{EM_y, SK\}_{K\_ab}$

Here, C simply impersonates A by sending the message α.1 to B. The encrypted component, $\{EM_C\}_{K\_cx}$ is generated by C and has been encrypted with the key K_Cx that C shares with any party, x. Upon receipt of

Authorized licensed use limited to: LOUGHBOROUGH UNIVERSITY. Downloaded on December 16, 2009 at 04:48 from IEEE Xplore. Restrictions apply.

this, B tries to decrypt $\{EM_C\}_{K\_cx}$ using the key, K_ab that he shares with A. However, since this key is different from the original key used by C, B will therefore obtain a value, EM_y that is different from EM_C. Nevertheless, this does not matter since B does not check its validity because the event marker is supposed to be checked by the initiator, A. Therefore, the protocol session successfully completes with B thinking he has executed a protocol run with A and now shares the secret key, SK with A. However, A is not present at all in the protocol.

Note that all our 4 attacks allow an attacker to cheat some legitimate party into thinking he is communicating with another legitimate party. This itself is already a proof that the IKDP1 protocol fails to achieve the basic requirement of mutual authentication, that legitimate parties are guaranteed that they are communicating with other legitimate parties who are really who they claim to be.

3.2 Attacks on Second Improved Protocol IKDP2

Our first attack is a replay attack that works as follows:

$\alpha.1: A \rightarrow C_B$	$\{A, EM_A\}_{MK\_a}$
$\alpha.2: C_B \rightarrow AS$	$\{A, EM_A\}_{MK\_a}, \{B, EM_B' \oplus R_B'\}_{MK\_b}$
$\alpha.3: AS \rightarrow A$	$\{EM_B' \oplus R_B', A, SK\}_{MK\_b}$, $\{EM_A, B, SK\}_{MK\_a}$
$\alpha.4: A \rightarrow C_B$	$\{EM_B' \oplus R_B', A, SK\}_{MK\_b}$

When A tries to initiate a protocol run with B by sending it message $\alpha.1$, C intercepts, leading A into thinking that he is B. C then impersonates B again and replays message $\alpha.1$ as the first component of message $\alpha.2$ to AS. The remaining component is obtained by replaying part of a previously captured message 2, namely $\{B, EM_B' \oplus R_B'\}_{MK\_b}$. AS responds with message $\alpha.3$ to A, who then forwards the first part of that as message $\alpha.4$ to B, which is immediately intercepted by C.

At the end of this protocol run, A thinks he shares a secret session key with B when in fact B is not be present at all. This is a failure of the IKDP2 protocol and is a type of unknown key-share attack.

Our second attack is also a replay attack that proceeds as follows:

$\alpha.1: A \rightarrow B$	$\{A, EM_A'\}_{MK\_a}$
$\alpha.2: B \rightarrow A$	$\{A, EM_A'\}_{MK\_a}$, $\{B, EM_B' \oplus R_B'\}_{MK\_b}$
$\alpha.3: AS \rightarrow A$	$\{EM_B' \oplus R_B', A, SK'\}_{MK\_b}$, $\{EM_A', B, SK'\}_{MK\_a}$
$\alpha.4: A \rightarrow B$	$\{EM_B' \oplus R_B', A, SK'\}_{MK\_b}$
$\beta.1: C_B \rightarrow A$	$\{B, EM_B' \oplus R_B'\}_{MK\_b}$
$\beta.2: A \rightarrow AS$	$\{B, EM_B' \oplus R_B'\}_{MK\_b}$, $\{A, EM_A \oplus R_A\}_{MK\_a}$
$\beta.3: AS \rightarrow C_B$	$\{EM_A \oplus R_A, B, SK\}_{MK\_a}$, $\{EM_B', A, SK\}_{MK\_b}$
$\beta.4: C_B \rightarrow A$	$\{EM_A \oplus R_A, B, SK\}_{MK\_a}$

Here, α denotes a valid previous protocol run between A and B, while the values EM_A', EM_B' and SK' denote the corresponding values in that previous protocol run. The attacker, C has eavesdropped on this communication. C then impersonates B and replays the second component of message, $\alpha.2$, namely $\{B, EM_B' \oplus R_B'\}_{MK\_b}$ as part of a new message $\beta.1$ to A. A, in thinking it is B trying to initiate a new protocol run with it, continues with the protocol by sending message $\beta.2$ to the authentication server, AS. AS responds by sending the message $\beta.3$ to B but this is in fact received by C since he is impersonating B. C then completes the protocol by forwarding the first component of message $\beta.3$ as message $\beta.4$ to A.

At the end of this protocol, β, the party A thinks he now shares the new secret key, SK with B when in fact B was not involved at all in this protocol run. To B, he still shares the secret key, SK' with A. This is also a type of unknown key-share attack.

4. Concluding Remarks

We have presented attacks on the two improved key distribution protocols, IKDP1 and IKDP2 proposed by Li et al. in [5]. Our attacks work because the designers of the IKDP1 and IKDP2 used event markers incorrectly. Namely, the event markers are synchronized locally meaning that only the party which generated them would check to make sure they are used in the correct order and remain unique. Other parties do not do this check and so as far as they are concerned, they would not be able to detect if such event markers are replayed and reused. However, in the IKDP1 and IKDP2, the event marker is not sent back to the party which generated it for rechecking and maintaining local synchronization. This is totally in contrast to the protocols in the paper [1] where event markers were first proposed.

Therefore, we stress that when protocol designers wish to use a certain technique to achieve the corresponding security that it is supposed to provide, then they should consult the original paper where the

Authorized licensed use limited to: LOUGHBOROUGH UNIVERSITY. Downloaded on December 16, 2009 at 04:48 from IEEE Xplore. Restrictions apply.

technique was proposed and thoroughly understand how that technique is to be used properly and correctly. Otherwise, it would fail to achieve the desired security objective.

Further, the IKDP2 protocol also falls to our attacks because there is no requirement on the initiator of the protocol to take part in any challenge-response. Referring to the 4 messages communicated in IKDP2, the initiator is involved in sending the messages 1 and 4. Message 1 can be anything generated by *A* while message 4 is simply a forward of the first component of message 3. Therefore, the initiator is not supposed to demonstrate his knowledge of any secret, which is common in challenge-response schemes for achieving mutual authentication of parties.

5. References

[1] R.C. Bauer, T.A. Berson, and R.J. Feiertag, "A Key Distribution Protocol Using Event Markers," ACM Transactions on Computer Systems, 1 (3), 1983, pp. 249-255.

[2] S. Blake-Wilson, and A. Menezes, "Unknown Key-Share Attacks on the Station-to-Station (STS) Protocol," Proceedings of Public Key Cryptography (PKC'99), Lecture Notes in Computer Science (LNCS), vol. 1560, 1999, pp. 154-170.

[3] T. Hwang, and W.C. Ku, "Reparable Key Distribution Protocols for Internet Environments," IEEE Trans. Comm., 43 (5), 1995, pp. 1947-1949.

[4] B.S. Kaliski, Jr, "An Unknown Key-Share Attack on the MQV Key Agreement Protocol," ACM Transactions on Information and System Security (TISSEC), 4 (3), 2001, pp. 275-288.

[5] H. Li, H. Chen, and H. Zhu, "An Improved Key Distribution Protocol with Perfect Reparability," Proceedings of International Conference on Networks (ICON 2000), 2000, pp. 273-277.

[6] G. Lowe, "A Family of Attacks Upon Authentication Protocols. [Online] Available at http://web.comlab.ox.ac.uk/oucl/work/gavin.lowe/Security/Papers/multiplicityTR.ps.

[7] B. Schneier, "Applied Cryptography: Protocols, Algorithms, and Source Code in C", U.S.A: John Wiley & Sons, 1996.

[8] K. Shim, "Unknown Key-Share Attack on Authenticated Multiple-Key Agreement Without One-Way Hash Functions," IEE Electronics Letters, 39 (1), 2003, pp. 38-39.

[9] W. Stallings, "Cryptography and Network Security: Principles and Practices", U.S.A: Prentice Hall, 1999.

[10] P. Syverson, "A Taxonomy of Replay Attacks," Proceedings of Computer Security Foundations Workshop, IEEE Computer Society Press, 1994.

Authorized licensed use limited to: LOUGHBOROUGH UNIVERSITY. Downloaded on December 16, 2009 at 04:48 from IEEE Xplore. Restrictions apply.