COMP00043: Cryptography and Security

Week 11 WorkshopActivity

Before we begin, take a few minutes to discuss the following:

1. What are the steps involved in an authentication process?

   Two steps are involved in an authentication process:

   (a) Identification step: Presenting an identifier to the security system.

   (b) Verification step: Presenting or generating authentication information that corroborates the binding between the entity and the identifier.

2. List three general approaches to dealing with replay attacks.

   (a) Attach a sequence number to each message used in an authentication exchange. A new message is accepted only if its sequence number is in the proper order.

   (b) Party A accepts a message as fresh only if the message contains a timestamp that, in A's judgment, is close enough to A's knowledge of current time. This approach requires that clocks among the various participants be synchronized.

   (c) Party A, expecting a fresh message from B, first sends B a nonce (challenge) and requires that the subsequent message (response) received from B contain the correct nonce value.

3. What is a suppress-replay attack?

   When a sender's clock is ahead of the intended recipient's clock, an opponent can intercept a message from the sender and replay it later when the timestamp in the message becomes current at the recipient's site. This replay could cause unexpected results.

Now try the following questions:

1. Consider Mutual Authentication proposed by Woo and Lam in Section 15.4. The protocol referred presented there can be reduced from seven steps to five, having the following sequence:

   (a) $A \rightarrow B$:

   (b) $B \rightarrow KDC$:

   (c) $KDC \rightarrow B$:

   (d) $B \rightarrow A$:

   (e) $A \rightarrow B$:

   Show the message transmitted at each step. Hint: The final message in this protocol is the same as the final message in the original protocol.

   Solution:

   (a) $A \rightarrow B : ID_A||N_a$

   (b) $B \rightarrow KDC : ID_A||ID_B||N_a||N_b$

   (c) $KDC \rightarrow B : E(PR_{auth}, [ID_A||PU_a])||E(PU_b, E(PR_{auth}, [N_a||N_b||K_s, ||ID_A||ID_B]))$

(d) $B \rightarrow A : E(PU_a, E(PR_{auth}, [N_a||N_b||K_s, ||ID_A||ID_B]))$

(e) $A \rightarrow B : E(K_s, N_b)$

2. Reference the suppress-replay attack described in Section 15.2 to answer the following.

   (a) Give an example of an attack when a party's clock is ahead of that of the KDC.

   An unintentionally post-dated message (message with a clock time that is in the future with respect to the recipient's clock) that requests a key is sent by a client. An adversary blocks this request message from reaching the KDC. The client gets no response and thinks that an omission or performance failure has occurred. Later, when the client is off-line, the adversary replays the suppressed message from the same workstation (with the same network address) and establishes a secure connection in the client's name.

   (b) Give an example of an attack when a party's clock is ahead of that of another party.

   An unintentionally post-dated message that requests a stock purchase could be suppressed and replayed later, resulting in a stock purchase when the stock price had already changed significantly.

3. There are three typical ways to use nonces as challenges. Suppose is a nonce generated by A, A and B share key K, and f() is a function (such as an increment). The three usages are

Usage 1	**Usage 2**	**Usage 3**
(1) $A \rightarrow B : N_a$	(1) $A \rightarrow B : E(K, N_a)$	(1) $A \rightarrow B : E(K, N_a)$
(2) $B \rightarrow A : E(K, N_a)$	(2) $B \rightarrow A : N_a$	(2) $B \rightarrow A : E(K, f(N_a))$

   Describe situations for which each usage is appropriate.

   Solution:

   All three really serve the same purpose. The difference is in the vulnerability. In Usage 1, an attacker could breach security by inflating N_a and withholding an answer from B for future replay attack, a form of suppress-replay attack. The attacker could attempt to predict a plausible reply in Usage 2, but this will not succeed if the nonces are random. In both Usage 1 and 2, the messages work in either direction. That is, if N is sent in either direction, the response is E[K,N]. In Usage 3, the message is encrypted in both directions; the purpose of function f is to assure that messages 1 and 2 are not identical. Thus, Usage 3 is more secure.

Home Work:

1. In addition to providing a standard for public-key certificate formats, X.509 specifies an authentication protocol. The original version of X.509 contains a security flaw. The essence of the protocol is as follows:

$$A \rightarrow B : At_A, r_A, ID_B$$

$$B \rightarrow A : Bt_B, r_B, ID_A, r_A$$

$$A \rightarrow B : Ar_B$$

Where t_A and t_B are timestamps, r_A and r_B are nonces and the notation X{Y} indicates that the message Y is transmitted, encrypted, and signed by X.

The text of X.509 states that checking timestamps t_A and t_B is optional for three-way authentication. But consider the following example: Suppose A and B have used the preceding protocol on some previous occasion, and that opponent C has intercepted the preceding three messages. In addition, suppose that timestamps are not used and are all set to 0. Finally, suppose C wishes to impersonate A to B. C initially sends the first captured message to B:

$$C \rightarrow B : A0, r_A, ID_B$$

B responds, thinking it is talking to A but is actually talking to C:

$B \rightarrow C : B0, r`_B, ID_A, r_A$

C meanwhile causes A to initiate authentication with C by some means. As a result, A sends C the following:

$A \rightarrow C : A0, r`_A, ID_C$

C responds to A using the same nonce provided to C by B:

$C \rightarrow A : C0, r\acute{}_B, ID_A, r`_A$

A responds with

$A \rightarrow C : Ar`_B$

This is exactly what C needs to convince B that it is talking to A, so C now repeats the incoming message back out to B.

$C \rightarrow B : Ar`_B$

So B will believe it is talking to A whereas it is actually talking to C. Suggest a simple solution to this problem that does not involve the use of timestamps.

Solution:

The problem has a simple fix, namely the inclusion of the name of B in the signed information for the third message, so that the third message now reads:

$A \rightarrow B : Ar`_B, B$