1 Attack of Needham-Schroeder protocol

Alice and Bob want to exchange a session key. Ivan shares a secret key A with Alice and a secret key B with Bob. THe protocol (of Needham-Schroeder) is as follow :

i. Alice sends $(Alice||Bob||Random_1)$ to Ivan.

ii. Ivan creates a session key K and sends $E_A(Random_1||Bob||K||E_B(K||Alice))$ to Alice.

iii. Alice sends $E_B(K||Alice)$ to Bob.

iv. Bob sends $E_K(Random_2)$ to Alice.

v. Alice sends $E_K(Random_2 - 1)$ to Bob.

1. What is the use of $Random_1$?
2. What is the use of $Random_2$ and why does Alice send $Random_2 - 1$? The security issue with this protocol is that old session keys have some value
3. If Eve is capable of listening to exchanged messages and was able to recover an old session key, show that she can convinve Bob that she is Alice.
4. What should added to the messages to fix this issue ? Why is it sufficient and to what does this new protocol resemble ?

(1)

The use of $random_1$ is to make sure that the answare is not pre computed and to avoid any replay attack

(2)

The use of $random_2$ and $random_2 - 1$ is to make sure that Alice has the key and she is able to decrypt the message

(3)

If Eve gets access to k she can do the following :

Bob sends $E_k(Random_3)$ → Eve (fake Alice)

Eve (fake Alice) sends back $E_k(Random_3 \pm 1)$ → Bob

Since there is no timestamp , Eve may even send the same message $E_k(random_2 - 1)$ to Bob .

(4)

It is sufficient to add a timestamp as follows :

$E_k(random_2 \,||\, timestamp)$ → Then we can verify timestamp

2 Third party protocols

1. Key server.
   In the system obtained above (question 4 of the previous exercise), a user doesn't need to authentify himself to the KDC each time he needs to access to a service. Why? Give a benefit and a drawback of this characteristic (about security) and justify them.

2. Key steal.
   In this authentication process, the symetric key of Alice is the hash print of her password. Ivan thus possesses the list of all the hashed passwords of his clients. If somedy steal this list, can he authentify himself as one of the clients? If not, why not? If yes, what could be done to fix this issue?

3. Signature.
   given a cryptographic hash function H and a system with shared symetric keys : e.g. K_{AK} between Alice and the KDC and K_{KB} between Bob and the KDC.
   — Alice --> KDC : $M||ID_A||E_{K_{AK}}(ID_B||H(M))$.
   — KDC --> Bob : $E_{K_{KB}}(ID_A||M||E_{K_{AK}}(ID_B||H(M))||Time)$.
   Modify this protocol so that Bob can check the signature.

4. Encryption : Public key system. (Supposing that $\mathcal{K}_{priv} \subseteq \mathcal{K}_{pub}$, i.e., that the encryption function E allows cihpering messages with both the public and privbate keys).
   — Alice --> KDC : $ID_A||E_{K_{privA}}(ID_A||ID_B||E_{K_{pubB}}(E_{K_{privA}}(M)))$.
   — KDC --> Bob : $E_{K_{privK}}(ID_A||E_{K_{pubB}}(E_{K_{privA}}(M))||Time)$
   $||E_{K_{privA}}(ID_A|||ID_B E_{K_{pubB}}(E_{K_{privA}}(M)))$.
   What is the insterest of such a protocol? Modify this protocole to avoid four crypting of the message.

5. Challenge.
   There are three ways of using a challenge (or nonce). Let N_a be a random number generated by Alice. Alice and Bob share a symmetric key K.
   — Alice --> Bob : N_a ; puis Bob --> Alice : $E_K(N_a)$.
   — Alice --> Bob : $E_K(N_a)$; puis Bob --> Alice : N_a.
   — Alice --> Bob : $E_K(N_a||\text{“}Alice\text{”})$; puis Bob --> Alice : $E_K(++N_a)$.
   Give an appropriate situation =for each case. Why add “Alice” in the last one?

(1)

This is because the user was granted with a "ticket" that has a certain validity and can be used for later accesses.

Advantages : fast and users may use the same key for longer

Drawbacks : no way to revoke the accesses

(2)

By using the correct hash anyone can identify himself as a client. A way to mitigate or to avoid the problem would be avoid storing the hash list in cleartext. Encrypt it using AES or other protocols.

(3)

The message should be encrypted with Bob's key :

$$\dots E_{K_{KB}}(ID_B \| H(M) \| Time) \dots$$

(4)

The interesting aspects of this protocol are related to the fact that Bob can be sure that Alice belongs to the KDC and that she was authorized by the KDC to send the message. To avoid multiple encryption Alice could send $E_{KpubB}(M)$ since KpubB is public. Or they could negotiate a session key

(5)

Each of them may be used at the beginning of an exchange. With the first two Bob has no guarantee while in the third we have some authentication by Alice

3 Kerberos pre-authentication

The exchanged messages within the Kerberos v5 protocol are given in Figure 1.
small

1. Explain the advantages of separating the authentication server, AS, from the ticket granting server, TGS.

Communication	Messages
1 : Alice → AS	$\text{KRB\_AS\_REQ} = (a \| \| tgs)$
2 : AS → Alice	$\text{KRB\_AS\_REP} = (E_{K_a}(K_{a,tgs} \parallel TGS \parallel \ldots) \parallel T_{a,tgs})$ $= E_{K_a}(K_{a,tgs} \parallel TGS \parallel \ldots) \parallel (id_{tgs} \parallel E_{K_{tgs}}(id_a \| \| t \| \| t_{end} \| \| K_{a,tgs}))$
3 : Alice → TGS	$\text{KRB\_TGS\_REQ} = (A_{a,tgs} \parallel Bob \parallel T_{a,tgs})$ $= E_{K_{a,tgs}}(id_a \| \| t) \parallel Bob \parallel (id_{tgs} \parallel E_{K_{tgs}}(id_a \| \| t \| \| t_{end} \| \| K_{a,tgs}))$
4 : TGS → Alice	$\text{KRB\_TGS\_REP} = (E_{K_{a,tgs}}(K_{a,b} \parallel Bob \parallel \ldots) \parallel T_{a,b})$ $= E_{K_{a,tgs}}(K_{a,b} \parallel Bob \parallel \ldots) \parallel (id_b \parallel E_{K_b}(id_a \| \| t \| \| t_{end} \| \| K_{a,b}))$
5 : Alice → Bob	$\text{KRB\_AP\_REQ} = (A_{a,b} \parallel T_{a,b})$ $= E_{K_{a,b}}(id_a \| \| t) \parallel (id_b \parallel E_{K_b}(id_a \| \| t \| \| t_{end} \| \| K_{a,b}))$
6 : Bob → Alice	$\text{KRB\_AP\_REP} = (E_{K_{a,b}}(t+1))$

TABLE 1 – Exchanged messages within Kerberos Version 5.

2. Why add an *authenticator*, $A_{a,tgs} = E_{K_{a,tgs}}(id_a \| \| t)$ in phase 3?
3. Describe a potential attack by *Password Guessing*, which could allow an intruder to obtain a session key (knowing that the user key in in general the hash of a password).
4. The [RFC4120] (The Kerberos Network Authentication Service, V5) states that pre-authentication addresses a weakness in initial implementations of Kerberos (see also the [RFC6113], A Generalized Framework for Kerberos Pre-Authentication). This pre-authentication adds a ciphered time-stamp to the first message, `KRB_AS_REQ`. This time-stamp is enciphered with the Key of the user. Which problem(s) is(are) solved by this pre-authentication?

(1)

There are various advantages related to the separation of TGS and AS. Here are some of them:
By having two distinct servers we avoid the single point of failure and we make sure that compromizing one of the two does not give access to the full architecture. Additionally, the TGS does not interact with sensitive information and can be secured in different ways compared to the AS

(2)

Because with $A_{a,tgs}$ Alice can prove her identity and prove the fact that she has been authenticated by AS and she is a user of the system. Only an authenticated and authorized user can have the $K_{a,tgs}$.

(3)

Since the user key is the hash of a password the intruder could perform a dictionary attack with rainbow tables to try to get the correct key. At that point he can start from point 2. of the protocol and retrive all the infos needed.

(4)

By adding this extra part in the initial request we avoid impersonation of another user. If we assume that all the users in the system have previously exchanged the key with AS; then an external user is not able to fake its identity and pretend to be Alice. Moreover, this avoids DoS on the AS since it will respond only to legit requests.

4 A toy protocol

Alice and Bob have invented the following protocol for sending a message securely from A to B. The protocol is based on the ideas of the one-time pad, but without a common, shared secret. Instead, for each message, both A and B invent a random nonce (N_A and N_B respectively) and execute the following protocol to send message M from A to B; here, in 3 rounds only the messages M_1, M_2 and M_3 in the right hand side are sent :

a. $A \longrightarrow B : M_1 = M \oplus N_A$
b. $B \longrightarrow A : M_2 = M_1 \oplus N_B$
c. $A \longrightarrow B : M_3 = M_2 \oplus N_A$

1. Show that B can recover M.
2. Is the system secure? Why or why not?

(1)

B receives : $M_3 = M_2 \oplus N_A$

$= M_1 \oplus N_B \oplus N_A$

$= M_0 \oplus N_A \oplus N_B \oplus N_A$

$= M_0 \oplus N_B$

Bob knows N_B so che can perform the following : $M_0 = M_3 \oplus N_B$

(2)

This system does not look secure. This are the problems that I can find :

There is no authentication or proof of identification. Moreover, the Nonces are always the same, this means that any MITM can intercept the traffic and so the communication

$A \rightarrow M_0 \oplus N_A \rightarrow$ Eve $\rightarrow$ NOTHING $\rightarrow B$

$A \leftarrow M_2 \leftarrow$

$A \rightarrow M_2 \oplus N_A \rightarrow$ Eve

M_2 is the same message by Alice $\Rightarrow M_2 = M_1 = M_0 \oplus N_A$

Eve can now retrive the message $\Rightarrow M_3 = M_2 \oplus N_A$

$= M_0 \oplus N_A \oplus N_A$

Actually Alice is doing it for her $= M_0$ Y

5 Public key Needham-Schroeder

We consider the following Public key Needham-Schroeder protocol :

i. $A \rightarrow B : Init = E_{K_{pubB}}(A||Na)$

ii. $B \rightarrow A : Challenge = E_{K_{pubA}}(Na||Nb)$

iii. $A \rightarrow B : Response = E_{K_{pubB}}(Nb)$

1. Now classical, attack
   (a) Give a generic attack in networks which applies perfectly to this protocol, so that Mallory can impersonate somebody.
   (b) What is the fix ?
2. Type Attack on Needham-Schroeder-Lowe.
   We consider a variant of this protocol where the identity of b is placed on the right of the message in the answer of B.
   (a) Show that an attack exists on the secret challenge N_b. *Indication : the attack is on the confusion betwen the type of an agent and of a challenge.*
   (b) Is this attack realistic ? Propose a counter-measure.

(1)

This is a great example of MITM attack :

$$A \rightarrow E_{KpubM}(A||Na) \rightarrow M \rightarrow E_{KpubB}(A||Na) \rightarrow B$$

$$A \leftarrow E_{KpubA}(Na||N_D) \leftarrow M \leftarrow E_{KpubM}(Na||N_B) \leftarrow B$$

$$A \rightarrow E_{KpubM}(N_D) \rightarrow M \rightarrow E_{KpubB}(N_D) \rightarrow B$$

the external actor goes un-noticed as can be seen by the red part

The fix is that the users have to properly authenticate and make correct use of the PKI system

(2)

The attack can be done if we change the order of the communication :

$$A \leftarrow E_{KpubA}(P||(Nb||B)) \qquad P \xrightarrow{1} E_{KpubB}(A||P) \rightarrow B$$

$$A \xrightarrow{4} E_{KpubP}((Nb||B)||Na||A) \rightarrow P \xrightarrow{3} \quad P \leftarrow E_{KpubA}(P||(Nb||B)) \xleftarrow{2} B$$

$$A \leftarrow E_{KpubA}(N_D) \qquad \xleftarrow{5} \qquad \xrightarrow{5} E_{KpubB}(N_D) \rightarrow B$$