Cybersecurity Project

Secure chat

Antonio Di Tecco, Luigi Treccozzi

Cybersecurity course A.Y. 2017-18

1. Overview

The purpose of our project is to develop a simple P2P chat application, where sensible messages are encrypted by means of AES-256.

To achieve this goal, we used C Programming Language, and OpenSSL 1.0.1t Library.

In our scenario a Server is in charge to distribute session keys to all the clients that are willing to join it in order to protect the messages that are exchanged between them. Additionally, the server is responsible to exchange in a safe fashion session keys between two peers that wish to communicate each other. Last, the Server has to store the hashed passwords associated to the peers with the scope of avoiding dangerous leak of data.

From a cryptographic perspective we implemented three protocols:

- **Ephemeral RSA**: to exchange temporary keys between Server and Clients
- **Needham–Schroeder Protocol (Denning-Sacco variant)**: to exchange keys between clients
- **Hash Functions**: to store passwords

In a typical execution of the application, a client registers at the server. Where he states that he wants to start a registering procedure a key is exchanged by means of Ephemeral RSA. Now, the session is protected with a key and the client can safely exchange the password with the Server. When the latter receives the encrypted password, first decrypts it, then stores its hash (SHA-256), combined with a *salting* quantity. In this way a malicious adversary that has granted the access to the Server Database can't steal the password and consequentially can't impersonate any legitimate user.

When a client wishes to communicate with another client, it forwards the request to the Server. If the contacted client exists and it is online, the Server advertises the requesting Client and, by means of Needham-Schroeder protocol, a key session is exchanged between the parties.

The following sections are devoted to better explain and analyse the protocols listed above.

2. Ephemeral RSA

The aim of Ephemeral Protocols is to provide the clients with session keys when they require them. In parallel, the protocol aims to provide the so-called Perfect Forward Secrecy, i.e. protecting all the message that clients exchanged with the Server in the past, even when a long-term secret (Server Private key in this case) gets compromised. Specifically, in our project, clients hold the server public key, whereas Server, obviously, holds its private one.

Note that clients could, for instance, generate their own session Key, send it to the Server and use it to encrypt further messages. This approach does not provide Perfect Forward Secrecy since an attacker that managed to compromised Server private key could easily decrypt all the messages that has been exchanged.

Ephemeral RSA, like all the Ephemeral Protocols (EDH for instance), prevent such possible attacks. The scheme is the following:

The messages carry the following information (k_{prv} and k_{pub} are the Server private Key and the Server public Key, respectively):

- **M_1 (A → S)**: $h(N_A)$

Where N_A is a nonce and h is a cryptographically secure hash function. Now S generates an RSA key pair: k^T_{prv}, k^T_{pub}, and reply with M2 message:

- **M_2 (S → A)**: k^T_{pub} || $\{k^T_{pub}$ ||$h(N_A)\}_{k_{prv}}$

Where || means concatenation and $\{\cdot\}_{k_{prv}}$ is the Server digital signature. Upon receiving M2, A verifies the digital signature and, if it succeeds, in its turn A generates k_{ses}, encrypt it through Server Public Key, along with N_A, and sends in M3:

- **M_3 (A → S)**: $E_{k^T_{pub}}\{N_A,\ k_{ses}\}$

At the end, the Server compute $h(N_A)$, and compare the result with the hash that has received earlier. If the check succeeds, it sends the encrypted nonce back to inform the client that the password:

- **M_4 (S → A)**: $E_{k_{ses}}\{N_A\}$

The client will verify that N_A is the same nonce it has generated at the beginning of this instance of the protocol.

Now we proceed with the BAN Logic based analysis. First, we need to introduce the idealized protocol (since M1 carry an un-encrypted information, that has no relevance in the protocol analysis):

- **M_2 (S → A)**: $\left\{\xrightarrow{k_{pub}^T} S, \#\left(\xrightarrow{k_{pub}^T} S\right)\ ^{1}\right\}$
- **M_3 (A → S)**: $\left\{\langle A \overset{k_{ses}}{\longleftrightarrow} S, \#\left(A \overset{k_{ses}}{\longleftrightarrow} S\right)^{2}\rangle_{N_A}\right\}_{k_{pub}}$
- **M_4 (S → A)**: $\left\{A \overset{k_{ses}}{\longleftrightarrow} S, \#\left(A \overset{k_{ses}}{\longleftrightarrow} S\right)\right\}_{k_{ses}}$

The objectives of the protocol are Key Authentication and Key Confirmation.

Now we proceed with the BAN Logic analysis.

Under the hypothesis that $A| \equiv \xrightarrow{k_{pub}} S$ (*Hp 1*), applying the Message Meaning Rule and Nonce Verification, after A received M2, we obtain that:

$$\frac{A| \equiv \xrightarrow{k_{pub}} S, A \lhd \{\xrightarrow{k_{pub}^T} S, \#\left(\xrightarrow{k_{pub}^T} S\right)\}_{k_{prv}}}{A| \equiv S| \sim \{\xrightarrow{k_{pub}^T} S, \#\left(\xrightarrow{k_{pub}^T} S\right)\}}$$

$$\frac{A| \equiv \#\left(\xrightarrow{k_{pub}^T} S\right), A| \equiv S| \sim \{\xrightarrow{k_{pub}^T} S, \#\left(\xrightarrow{k_{pub}^T} S\right)\}}{A| \equiv S| \equiv \{\xrightarrow{k_{pu}^T} S, \#\left(\xrightarrow{k_{pub}^T} S\right)\}}$$

Now we need to state two more hypothesis:

- *Hp 2*: $A| \equiv S \Rightarrow \xrightarrow{k_{pub}^T} S$
- *Hp 3*: $A| \equiv S \Rightarrow \#\left(\xrightarrow{k_{pub}^T} S\right)$

Applying the Jurisdiction Rule:

$$\frac{A| \equiv S| \equiv \left\{\xrightarrow{k_{pub}^T} S, \#\left(\xrightarrow{k_{pub}^T} S\right)\right\}, A| \equiv \{S \Rightarrow \xrightarrow{k_{pub}^T} S, S \Rightarrow \#\left(\xrightarrow{k_{pub}^T} S\right)\}}{A| \equiv \{\xrightarrow{k_{pub}^T} S, \#\left(\xrightarrow{k_{pub}^T} S\right)\}}$$

Now we pass to examine M3. M3 is encrypted with the Server Public key: anyone could have encrypted it. On the other hand, it carries N_A, that links this message to the current instance of the protocol (under the assumption that *h* is a cryptographically secure hash function and A generates unpredictable nonces), and S

[1] $\#\left(\xrightarrow{k_{pub}^T} S\right)$ is an implicit statement, not directly derived from the real protocol

[2] $\#\left(A \overset{k_{ses}}{\longleftrightarrow} S\right)$ is an implicit statement, not directly derived from the real protocol

is the only principal that can see it, since it is encrypted through k_{pub}^{T}. In other words: $S \triangleleft \{N_A\}$. Now, after M1, if we assume that $S|\equiv A|\sim \{h(N_A)\}$ then, after M3:

$$\frac{S|\equiv A|\sim \{h(N_A)\}, S \triangleleft \{N_A\}}{S|\equiv A|\sim \{N_A\}}$$

All this said, now N_A is a fresh (w.r.t the current instance of the protocol) secret only shared by A and S:

- *Hp 4*: $S| \equiv S \overset{N_A}{\Leftrightarrow} A$
- *Hp 5*: $S \mid \equiv \#\{N_A\}$

By virtue of this statements, we can proceed with the analysis:

$$\frac{S| \equiv \xrightarrow{k_{pub}} S, S \triangleleft \left\{\langle A \overset{k_{ses}}{\longleftrightarrow} S, \#\left(A \overset{k_{ses}}{\longleftrightarrow} S\right)^{\cdot}\rangle_{N_A}\right\}_{k_{pub}}}{S \triangleleft \langle A \overset{k_{ses}}{\longleftrightarrow} S, \#\left(A \overset{k_{ses}}{\longleftrightarrow} S\right)^{\cdot}\rangle_{N_A}}$$

And for the *Hp 4:*

$$\frac{S| \equiv S \overset{N_A}{\Leftrightarrow} A, S \triangleleft \langle A \overset{k_{ses}}{\longleftrightarrow} S, \#\left(A \overset{k_{ses}}{\longleftrightarrow} S\right)^{\cdot}\rangle_{N_A}}{S|\equiv A|\sim \left\{A \overset{k_{ses}}{\longleftrightarrow} S, \#\left(A \overset{k_{ses}}{\longleftrightarrow} S\right)^{\cdot}\right\}}$$

Now, by applying the Nonce Verification Rule and *Hp 5*:

$$\frac{S|\equiv A|\sim \left\{\#\left(A \overset{k_{ses}}{\longleftrightarrow} S\right), A \overset{k_{ses}}{\longleftrightarrow} S\right\}, S \mid \equiv \#\{N_A\}}{S|\equiv A| \equiv \{A \overset{k_{ses}}{\longleftrightarrow} S, \#\left(A \overset{k_{ses}}{\longleftrightarrow} S\right)^{\cdot}\}}$$

One more hypothesis is needed:

- *Hp 6*: $S| \equiv A \Rightarrow \left\{\#\left(A \overset{k_{ses}}{\longleftrightarrow} S\right), A \overset{k_{ses}}{\longleftrightarrow} S\right\}$

By this hypothesis, applying the Jurisdiction Rule:

$$\frac{S| \equiv A \Rightarrow \left\{\#\left(A \overset{k_{ses}}{\longleftrightarrow} S\right), A \overset{k_{ses}}{\longleftrightarrow} B\right\}, S| \equiv A \Rightarrow \left\{\#\left(A \overset{k_{ses}}{\longleftrightarrow} S\right), A \overset{k_{ses}}{\longleftrightarrow} B\right\}}{S| \equiv \{A \overset{k_{ses}}{\longleftrightarrow} S, \#\left(A \overset{k_{ses}}{\longleftrightarrow} S\right)^{\cdot}\}}$$

Last, we analyse M4 applying the Message Meaning:

$$\frac{A| \equiv A \overset{k_{ses}}{\longleftrightarrow} S, A \triangleleft \left\{A \overset{k_{ses}}{\longleftrightarrow} S, \#\left(A \overset{k_{ses}}{\longleftrightarrow} S\right)\right\}_{k_{ses}}}{A|\equiv S|\sim \{A \overset{k_{ses}}{\longleftrightarrow} S, \#\left(A \overset{k_{ses}}{\longleftrightarrow} S\right)\}}$$

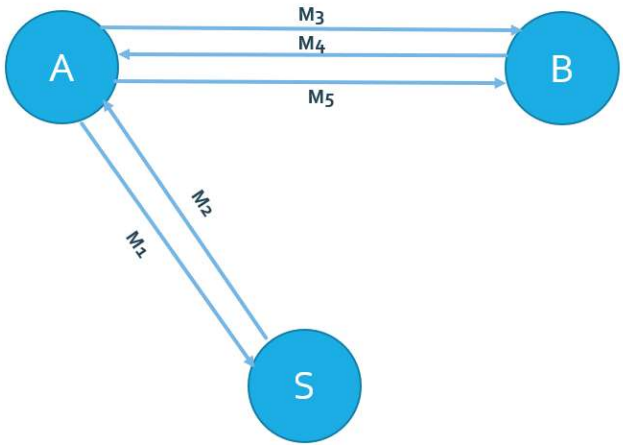
As last hypothesis *Hp 7*: $A| \equiv \left(A \overset{k_{ses}}{\longleftrightarrow} S\right)$ and *Hp 8*: $A| \equiv \#\left(A \overset{k_{ses}}{\longleftrightarrow} S\right)$, thanks to the Nonce Verification Rule:

$$\frac{A|\equiv S|\sim \left\{A \overset{k_{ses}}{\longleftrightarrow} S, \#\left(A \overset{k_{ses}}{\longleftrightarrow} S\right)\right\}, A|\equiv \#\left(A \overset{k_{ses}}{\longleftrightarrow} S\right)}{A|\equiv S|\equiv \{A \overset{k_{ses}}{\longleftrightarrow} S, \#\left(A \overset{k_{ses}}{\longleftrightarrow} S\right)\}}$$

This last assertion concludes the analysis. The protocol guarantees Key Authentication, Key Confirmation and Key Freshness.

3. Needham-Schroeder protocol

Like already mentioned, when to peers wish to communicate, they must share a secret key. Many protocols have been proposed in the past, with the scope to safely exchange symmetric key through a trusted third party. We choose to implement the well-known Needham-Schroeder Protocol, modified according the Denning-Sacco variant. The following picture describes the five steps of the original (1978) protocol.



The messages carry the following information:

- **M_1 (A → S)**: N_A, B, A
- **M_2 (S → A)**: $\{N_A, B, k_{AB}, \{k_{AB}, \mathrm{A}\}_{k_B}\}_{k_A}$
- **M_3 (A → B)**: $\{k_{AB}, \mathrm{A}\}_{k_B}$
- **M_4 (B → A)**: $\{N_B\}_{k_{AB}}$
- **M_5 (A → B)**: $\{N_B - 1\}_{k_{AB}}$

In order to demonstrate the correctness of the protocol, we exploited the BAN Logic. In particular, thanks to intensive use of the Message Meaning Rule, Nonce Verification and Jurisdiction Rule it is possible to claim that, under certain hypothesis, this protocol guarantees key authentication, key confirmation and key freshness.

First, we introduce the idealized protocol, i.e. replacing all the concrete message by the idealized message:

- **M_2 (S → A)**: $\{N_A, A \overset{k_{AB}}{\longleftrightarrow} B, \#\left(A \overset{k_{AB}}{\longleftrightarrow} B\right), \{A \overset{k_{AB}}{\longleftrightarrow} B\}_{k_B}\}_{k_A}$[3]
- **M_3 (A → B)**: $\{A \overset{k_{AB}}{\longleftrightarrow} B\}_{k_B}$

[3] $\#\left(A \overset{k_{AB}}{\longleftrightarrow} B\right)$ it's an implicit statement, not directly derived from the real protocol

- **M_4 ($B \rightarrow A$)**: $\{N_B, A \overset{k_{AB}}{\longleftrightarrow} B\}_{k_{AB}}$
- **M_5 ($A \rightarrow B$)**: $\{N_B - 1, A \overset{k_{AB}}{\longleftrightarrow} B\}_{k_{AB}}$

Now, let's make some hypothesis:

- *Hp 1*: $A| \equiv A \overset{k_A}{\leftrightarrow} S$
- *Hp 2*: $S| \equiv A \overset{k_A}{\leftrightarrow} S$
- *Hp 3*: $B| \equiv B \overset{k_B}{\leftrightarrow} S$
- *Hp 4*: $S| \equiv B \overset{k_B}{\leftrightarrow} S$
- *Hp 5*: $A| \equiv \#(N_A)$
- *Hp 6*: $B| \equiv \#(N_B)$

The objectives of the protocol are:

1) Key Authentication: $A| \equiv A \overset{k_{AB}}{\longleftrightarrow} B$ and $B| \equiv A \overset{k_{AB}}{\longleftrightarrow} B$
2) Key Confirmation: $A| \equiv B| \equiv A \overset{k_{AB}}{\longleftrightarrow} B$ and $B| \equiv A| \equiv A \overset{k_{AB}}{\longleftrightarrow} B$

Now we produce assertions and add some hypothesis.

Since M1 is not encrypted it is not relevant for the analysis, hence we start analysing M2.

Since *Hp 1*, according to the Message Meaning Rule:

$$\frac{A| \equiv A \overset{k_A}{\leftrightarrow} S, A \lhd \{N_A, A \overset{k_{AB}}{\longleftrightarrow} B, \#\left(A \overset{k_{AB}}{\longleftrightarrow} B\right), \{A \overset{k_{AB}}{\longleftrightarrow} B\}_{k_B}\}_{k_A}}{A| \equiv S| \sim \{N_A, A \overset{k_{AB}}{\longleftrightarrow} B, \#\left(A \overset{k_{AB}}{\longleftrightarrow} B\right), \{A \overset{k_{AB}}{\longleftrightarrow} B\}_{k_B}\}}$$

Now, using *Hp 5* and the Nonce Verification Rule:

$$\frac{A| \equiv \#(N_A), A| \equiv S| \sim \{N_A, A \overset{k_{AB}}{\longleftrightarrow} B, \#\left(A \overset{k_{AB}}{\longleftrightarrow} B\right), \{A \overset{k_{AB}}{\longleftrightarrow} B\}_{k_B}\}}{A| \equiv S| \equiv \{N_A, A \overset{k_{AB}}{\longleftrightarrow} B, \#\left(A \overset{k_{AB}}{\longleftrightarrow} B\right), \{A \overset{k_{AB}}{\longleftrightarrow} B\}_{k_B}\}}$$

If we add some other hypothesis:

- *Hp 7*: $S| \equiv A \overset{k_{AB}}{\longleftrightarrow} B$
- *Hp 8*: $A| \equiv S \Rightarrow A \overset{k_{AB}}{\longleftrightarrow} B$
- *Hp 9*: $A| \equiv S \Rightarrow \#(A \overset{k_{AB}}{\longleftrightarrow} B)$
- *Hp 10*: $S| \equiv \#(A \overset{k_{AB}}{\longleftrightarrow} B)$

using the previous result, Jurisdiction Rule, *Hp 8* and *Hp 9*:

$$\frac{A| \equiv S| \equiv A \overset{k_{AB}}{\longleftrightarrow} B, A| \equiv S \Rightarrow A \overset{k_{AB}}{\longleftrightarrow} B}{A| \equiv A \overset{k_{AB}}{\longleftrightarrow} B}$$

$$\frac{A|\equiv S| \equiv \#(A \overset{k_{AB}}{\longleftrightarrow} B), A| \equiv S \Rightarrow \#(A \overset{k_{AB}}{\longleftrightarrow} B)}{A| \equiv \#(A \overset{k_{AB}}{\longleftrightarrow} B)}$$

We now examine M3. Being valid *Hp 3*, using the Message Meaning Rule:

$$\frac{B| \equiv B \overset{k_{AB}}{\longleftrightarrow} S, B \lhd \{A \overset{k_{AB}}{\longleftrightarrow} B\}_{k_B}}{B| \equiv S| \sim A \overset{k_{AB}}{\longleftrightarrow} B}$$

We can't say anything else, because M3 is not protected by nonce, and therefore B has no way of being sure about the freshness of the key. We need further hypothesis:

- *Hp 11*: $B| \equiv \#(A \overset{k_{AB}}{\longleftrightarrow} B)$
- *Hp 12*: $B| \equiv T \Rightarrow A \overset{k_{AB}}{\longleftrightarrow} B$

Indeed, being valid *Hp 11*, by virtue of previous result, through the Nonce Verification Rule:

$$\frac{B| \equiv \#(A \overset{k_{AB}}{\longleftrightarrow} B), B| \equiv S| \sim A \overset{k_{AB}}{\longleftrightarrow} B}{B| \equiv S| \equiv A \overset{k_{AB}}{\longleftrightarrow} B}$$

Now, using *Hp 12* and the Jurisdiction Rule:

$$\frac{B| \equiv S| \equiv A \overset{k_{AB}}{\longleftrightarrow} B, B| \equiv S \Rightarrow A \overset{k_{AB}}{\longleftrightarrow} B}{B| \equiv A \overset{k_{AB}}{\longleftrightarrow} B}$$

We just achieved the mutual Key Authentication. We now pass to examine M4; if we first apply the Message Meaning Rule and then the Nonce Verification Rule we obtain:

$$\frac{A| \equiv A \overset{k_{AB}}{\longleftrightarrow} B, A \lhd \{N_B, A \overset{k_{AB}}{\longleftrightarrow} B\}_{k_{AB}}}{A| \equiv B| \sim \{N_B, A \overset{k_{AB}}{\longleftrightarrow} B\}}$$

$$\frac{A| \equiv \#(A \overset{k_{AB}}{\longleftrightarrow} B), A| \equiv B| \sim \{N_B, A \overset{k_{AB}}{\longleftrightarrow} B\}}{A| \equiv B| \equiv \{N_B, A \overset{k_{AB}}{\longleftrightarrow} B\}}$$

Last, we examine M5. Again, applying the previous results, Message Meaning Rule and Nonce Verification Rule we conclude that:

$$\frac{B| \equiv A \overset{k_{AB}}{\longleftrightarrow} B, A \lhd \{N_B - 1, A \overset{k_{AB}}{\longleftrightarrow} B\}_{k_{AB}}}{B| \equiv A| \sim \{N_B - 1, A \overset{k_{AB}}{\longleftrightarrow} B\}}$$

$$\frac{B| \equiv \#(A \overset{k_{AB}}{\longleftrightarrow} B)^{4}, B| \equiv A| \sim \{N_B - 1, A \overset{k_{AB}}{\longleftrightarrow} B\}}{B| \equiv A| \equiv \{N_B, A \overset{k_{AB}}{\longleftrightarrow} B\}}$$

[4] From *Hp 11*

We also achieved key confirmation and, by hypothesis, Key freshness too.

Unfortunately, *Hp 11* is a dangerous Hypothesis that cannot be validated. Indeed Needham-Schroeder protocol has a flaw deriving by this hypothesis. Consider a scenario in which an adversary that has gathered all the messages exchanged in past executions of the protocol managed to compromise one of those session keys. Now it is possible to wait for a next execution, intercept M3 and replace it by $M3^{old}$, that contains an old and compromised session key. B has no way to detect the non-freshness of that key, hence it behaves like $M3^{old}$ was part of the current and legitim execution of the protocol, replying with M4, encrypted with the old and compromised session key. Now the adversary can intercept this message and replying according to the protocol with M5. In other words, an adversary can now impersonate A, with no possibilities that B notices it is victim of an impersonation attack. D. E. Denning and G. M. Sacco in 1981 proposed to *complete* the protocol adding the timestamps to messages M3 and M4. In this way B can verify the freshness of the key (if M3 contains a fresh component, all the M3 component are fresh as well) and *Hp 11* becomes a valid hypothesis and the protocol is now safe.

We slavishly implemented the protocol just describes, taking care of adding the timestamp with the purpose of avoiding impersonation attacks.

4. Storing hash passwords

Like already mentioned, it is not a good practise to store clear passwords. An unexpected leak of information would reveal all the passwords that are stored in the system. To overcome this problem, a simple solution would be storing the hashed passwords, even better if combined with a secret known only to the server and with a random *salting* secret, associated to each password. This technique prevents a dictionary attack based on a popular (and therefore insecure) hashed passwords dictionary.

In our project we followed this principle. Specifically, we used SHA-256 hash function from an AES-128 in CBC mode cipher: the only information related to the password that will be stored is its hash.

When a user wished to log-in, the password will be encrypted and sent. Once received, it will be decrypted with the current session key, and its hash will be compared with the hash of the corresponding *salted* password.