

# **Design Principles for Cryptography Protocols**

Chiara Braghin

Seminario del corso di  
Sistemi di Elaborazione dell'Informazione (Sicurezza),  
Venezia, 28 Febbraio, 2001

# Criptographic Protocols

---

- Def. A *protocol* is a set of rules or conventions defining an exchange of messages between a set of two or more partners.
- In a *cryptographic protocol* the whole or part of the message is encrypted.

# Types of Protocols

---

- Authentication protocol
  - to assure the identity of the principals
- Key establishment protocol
  - to establish a shared secret
- Authenticated key establishment protocol
  - to establish a shared secret with an authenticated principal

# Types of Attacks

---

- **Passive Attack**
  - monitoring of data exchanged
  
- **Active Attack**
  - modification of messages
  - injection of new messages

# Distinguishing protocols characteristics

---

- 1 . *Nature* of the authentication.
- 2 . *Reciprocity* of authentication.
- 3 . *Key freshness*.
- 4 . *Efficiency*.
- 5 . *Third party* requirement.
- 6 . Type of *certificate* used.
- 7 . *Key control*.
- 8 . *Nonrepudiaton*.

# Protocol execution properties

---

- Many protocols executions may happen simultaneously
- The same principal may participate in several such executions, playing different roles
- Principals can be on-line or off-line
- Some of the principals may not be fully trusted

# Notations

---

- $A, B$  represent arbitrary principals,  $S$  a server,  $C$  the attacker,  $T$  a timestamp,  $N$  a nonce
- $K_a$  is  $A$ 's public key,  $K_a^{-1}$  is  $A$ 's private key
- $\{X\}_K$  represents  $X$  encrypted under  $K$ ; anyone who knows  $\{X\}_K$  and the inverse of key  $K$  can obtain  $X$
- *Message 4*  $B \rightarrow A: \{T_a + 1\}_{K_{ab}}$  describes the forth message in a protocol
- $H$  is a one-way hash function

# Is the protocol robust?

---

- Needham-Schroeder protocol

Message 1  $A \rightarrow S: A, B, N_a$

Message 2  $S \rightarrow A: \{ N_a, B, K_{ab}, \{K_{ab}, A\}_{K_{bs}} \}_{K_{as}}$

Message 3  $A \rightarrow B: \{ K_{ab}, A \}_{K_{bs}}$

Message 4  $B \rightarrow A: \{ N_b \}_{K_{ab}}$

Message 5  $A \rightarrow B: \{ N_b + 1 \}_{K_{ab}}$

# Verification with BAN logic (1)

---

- Derive the idealized protocol

Message 2  $S \rightarrow A: \{ N_a, (A \leftrightarrow^{K_{ab}} B), \#(A \leftrightarrow^{K_{ab}} B), \{A \leftrightarrow^{K_{ab}} B\}_{K_{bs}} \}_{K_{as}}$

Message 3  $A \rightarrow B: \{A \leftrightarrow^{K_{ab}} B\}_{K_{bs}}$

Message 4  $B \rightarrow A: \{ N_b, (A \leftrightarrow^{K_{ab}} B) \}_{K_{ab}}$  *from B*

Message 5  $A \rightarrow B: \{ N_b, (A \leftrightarrow^{K_{ab}} B) \}_{K_{ab}}$  *from A*

- Assumptions about the initial state:

–  $A \models A \leftrightarrow^{K_{as}} S$

$B \models B \leftrightarrow^{K_{bs}} S$

$S \models A \leftrightarrow^{K_{as}} S$

$S \models B \leftrightarrow^{K_{bs}} S$

$S \models A \leftrightarrow^{K_{ab}} B$

–  $A \models (S \Rightarrow A \leftrightarrow^K B)$

$B \models (S \Rightarrow A \leftrightarrow^K B)$

$A \models (S \Rightarrow \#(A \leftrightarrow^K B))$

# Verification with BAN logic (2)

---

- Assumptions (cont.):

$$A \models \#(N_a)$$

$$B \models \#(N_b)$$

$$S \models \#(A \leftrightarrow^{K_{ab}} B)$$

$$B \models \#(A \leftrightarrow^K B)$$

- Discover beliefs held by the parties:

$$A \models A \leftrightarrow^{K_{ab}} B$$

$$B \models A \leftrightarrow^{K_{ab}} B$$

$$A \models B \models A \leftrightarrow^{K_{ab}} B$$

$$B \models A \models A \leftrightarrow^{K_{ab}} B$$

# (1) Freshness of the key

---

- A key used recently can yet be quite old and possibly compromised
- Example: Needham-Schroeder

Message 3  $A \rightarrow B: \{ K_{ab}, A \}_{K_{bs}}$

Message 4  $B \rightarrow A: \{ N_b \}_{K_{ab}}$

Message 5  $A \rightarrow B: \{ N_b + 1 \}_{K_{ab}}$

## (2) Compromised session keys

---

- Session keys can be compromised: the protocol should minimize the effect of such events:
  - certificates should expire
  - when one key is expiring, it should not be used for encrypting the new key that will replace it

# (3) Explicit Communication

---

- Every message should say what it means: the message *interpretation* should depend only on its *content*
- If the *identity* of a principal is essential to the meaning of a message, it is prudent to mention its name explicitly

# example: SSH and SSL

---

- SSH:

Message 1  $A \rightarrow B: N_a$

Message 2  $B \rightarrow A: N_b$

Message 3  $B \rightarrow A: K_{bh}, K_{bs}$

Message 4  $A \rightarrow B: \{ \{ H(\text{prev. msgs}), K \}_{K_{bs}} \}_{K_{bh}}$

Message 5  $A \rightarrow B: \{ A, K_a, \{ H(A, N_a, N_b) \}_{K_a^{-1}} \}_{K'}$

- SSL:

Message 1  $A \rightarrow B: \{ K_{ab} \}_{K_b}$

Message 2  $B \rightarrow A: \{ N_b \}_{K_{ab}}$

Message 3  $A \rightarrow B: \{ CA, \{ N_b \}_{K_a^{-1}} \}_{K_{ab}}$

# SSH attack

---

Message 1  $A \rightarrow C: N_a$

Message 1'  $C \rightarrow B: N_a$

Message 2  $B \rightarrow C: N_b$

Message 2'  $C \rightarrow A: N_b$

Message 3  $B \rightarrow C: K_{bh}, K_{bs}$

Message 3'  $C \rightarrow A: K_{ch}, K_{cs}$

Message 4  $A \rightarrow C: \{ \{ H(\text{prev. msgs}), K \}_{K_{cs}} \}_{K_{ch}}$

Message 4'  $C \rightarrow B: \{ \{ H(\text{prev. msgs}), K \}_{K_{bs}} \}_{K_{bh}}$

Message 5  $A \rightarrow C: \{ A, K_a, \{ H(A, N_a, N_b) \}_{K_a^{-1}} \}_{K'}$

Message 5'  $C \rightarrow B: \{ A, K_a, \{ H(A, N_a, N_b) \}_{K_a^{-1}} \}_{K'}$

# SSL attack

---

Message 1  $A \rightarrow C: \{ K_{ac} \}_{K_c}$

Message 1'  $C \rightarrow B: \{ K_{ac} \}_{K_b}$

Message 2  $B \rightarrow C: \{ N_b \}_{K_{ac}}$

Message 2'  $C \rightarrow A: \{ N_b \}_{K_{ac}}$

Message 3  $A \rightarrow C: \{ CA, \{ N_b \}_{K_a^{-1}} \}_{K_{ac}}$

Message 3'  $C \rightarrow B: \{ CA, \{ N_b \}_{K_a^{-1}} \}_{K_{ac}}$

# (4) Uses of Encryption

---

- Be clear about why encryption is being done. It is not synonymous with security, improper use can lead to errors or redundancy.
- Used for:
  - preservation of confidentiality
  - to guarantee authentication
  - to bind together parts of a message

# (5) Sign before Encrypting

---

- When a principal signs material that has already been encrypted, it should not be inferred that the principal knows the content of the message
- Example:  
Message 1  $A \rightarrow B: \{X\}_{K_b}, \{H(X)\}_{K_a}^{-1}$

# Sign before Encrypting (cont)

---

- Exception: Denning-Sacco protocol

Message 1  $A \rightarrow S: A, B$

Message 2  $S \rightarrow A: CA, CB$

Message 3  $A \rightarrow B: CA, CB, \{ \{ K_{ab}, T_a \}_{K_a^{-1}} \}_{K_b}$

Message 3'  $B \rightarrow C: CA, CC, \{ \{ K_{ab}, T_a \}_{K_a^{-1}} \}_{K_c}$

- Better encrypt before sign when anonymity is more important than nonrepudiaton

# (6) Handling of Errors

---

- Proper handling of errors can be crucial to system security: the protocol have to explain how principals react when they perceive errors

## (7) Uses of Timestamps

---

- If timestamps are used as *freshness guarantees* by reference to absolute time, then the difference between local clocks at various machines must be much less than the allowable age of a message deemed to be valid

# (8) Trust

---

- The protocol designer should know which trust relations his protocol depends on, and why the dependence is necessary
  - KDC Key Distribution Center
  - Certificate Authority
  - Server issuing timestamps

## (9) Recognize messages

---

- It should be possible to deduce that the message belongs to a protocol, to a particular run, and to know its number in the protocol

# Conclusions

---

- Design principles are neither necessary nor sufficient conditions to build robust protocols
- They are guidelines to help prevent errors
- Following a design principle will sometimes lead to violating another
- There are exceptions to some principles
- To be used with formal methods

# Encryption Primitives

---

- Symmetric-key

- $E_k(m) = c$

m: plaintext

- $D_k(c) = m$

c: ciphertext

K: shared key

- Public-key

- $E_e(m) = c$

- $D_d(c) = m$

- e: public key

d: private key