Protocols

Adapted from [Pfl96, Chapter 4].

A protocol is an orderly sequence of steps two or more parties take to accomplish some task. A good protocol should be

- Established in advance.
- Mutually subscribed to.
- Unambiguous.
- Complete.

For example, the "hello" protocol on phone connections.

We are interested in protocols by which mutually suspicious parties can interact with each other and be convinced of fairness.

Protocols help verify correctness of a process at a high level (modeling tool).

Types of Protocols

- Arbitrated—trusted third party involved in the interaction.
  - Finding a mutually trustworthy third party?
  - Availability of the third party (may become a bottleneck).
  - Shares secrets with involved parties.
- Adjudicated—disinterested third party can judge fairness based on evidence.
  - Detect failure after the fact.
- Self-Enforcing—guarantees fairness. If either party cheats, it becomes evident to the other party.

Key Exchange Protocols

It's not a good idea to exchange too much information encrypted with a single key.

1. Symmetric Key Exchange Without Server

   Send $E_{K_{old}}(K_{new})$ to the other party! O my God, no!

2. Symmetric Key Exchange Using a Trusted Server

   Say Pablo wants to communicate with Renee. S is the trusted server [NS78]. Authentication with key exchange as side-effect.

   - Pablo and Renee each share a key with the server, say K_P and K_R.
   - $P \rightarrow S$: (P, R, I_P). P requests appropriate *credentials* to authenticate himself to R.
   - $S \rightarrow P$: $E_{K_P}(I_P, R, \overbrace{K_{PR}}^{\text{sess key}}, \overbrace{E_{K_R}(K_{PR}, P)}^{\text{ticket}})$.

     S returns a session key encrypted for P and a *ticket* encrypted for R.
   - $P \rightarrow R$: $E_{K_R}(K_{PR}, P)$.

   Compromise of the session key results in spoofing [DS81]. The protocol fails to provide key freshness from the viewpoint of R [Seb].

Key Exchange Protocols - II

1. Asymmetric Key Exchange Without Server (P knows R's public key)
   - Reduces the need for individual keys.
   - Reduces the vulnerability of a central repository.
   - P could send $E_R(K_{PR})$ directly to R.
     ▷ No authentication.
     ▷ No replay prevention.
   - P could send $E_R(D_P(K_{PR}))$ to R. One message passes an authenticated, confidential key.
     ▷ No replay prevention. Have P decrypt a nonce with K_{PR} to avoid that.
2. Asymmetric Key Exchange With Server (P doesn't know R's pubkey)
   - The server provides public keys for everyone.
   - Exchange is as in the previous case.
   - How do you ensure that the server has the right public key for everyone?

   In practice, the server issues certificates encoded in DER.

   Show "openssl x509 -inform der -in pub.der -text".

Digital Signatures

Suppose you send e-mail to your bank to transfer $100 to Tim's account.

- Why should the bank believe the e-mail came from you [unaltered]?
  - Authentication + integrity.
- If the bank transferred the money, maybe you can disavow the e-mail.
  - Non-repudiation.
- In case of dispute, can it be settled by a neutral third party?

Signatures basically provide non-repudiation that shared key system do not.

Digital Signatures with Symmetric Encryption

With the aid of a trusted third party! A signature is the encryption of the message.

- $S \rightarrow A: \ E_{K_S}(M)$.
- $A \rightarrow R: \ \underbrace{E_{K_R}(M, S, E_{K_S}(M))}_{\text{A says that 'S said M'}}$.

Digital Signatures With Public Key Encryption

- $S \rightarrow R: \ D_S(M)$.
  - Authentic but not private.
- $S \rightarrow R: \ E_R(D_S(M))$.
  - But what if R decrypts the outer layer and reencrypts the inner to create a message $E_U(D_S(M))$!

El Gamal Signature Scheme

p prime, g generator of Z_p^*, x private key, $y = g^x$ public key, k pseudo random integer relatively prime to $(p-1) = \text{ord}_g$, $r = g^k$.

Signature on m: $s = (m - xr)k^{-1} \bmod (p-1)$. Computing s seems to require knowledge of x (private key) & k, which requires the ability to compute discrete logs. Signature is the pair (r, s). Precomputing of (k, r) pairs is possible and signature generation is then easy to compute!

Verification: $g^m = y^r \cdot r^s$. Verification only requires (y, r) both of which are public. Verification requires exponentiation! To derive the verification equation from first principles, consider that

$$\begin{aligned}
s &= (m - xr)k^{-1} \bmod (p-1) \\
g^s &= g^{(m-xr)k^{-1}} \bmod p \\
(g^s)^k &= \left(g^{(m-xr)k^{-1}}\right)^k \bmod p \\
\left(g^k\right)^s &= g^{(m-xr)} \bmod p \\
r^s &= g^m \cdot y^{-r} \bmod p \\
r^s \cdot y^r &= g^m \bmod p
\end{aligned}$$

Look at [Sta99, pg. 229] for why we go from the first equality which is mod $(p-1)$ to the second equality which is mod p. It's because $p - 1 = \phi(p)$.

Digital Signatures without Encryption

Use a strong hash function and a trusted third party. S uses a hash function f_s and R uses f_r. Both share these functions with the arbiter A.

- $S \rightarrow A: \overbrace{(M, f_s(M))}^{\text{S said M}}$.
- $A \rightarrow R: (\underbrace{M}_{e}, S, \underbrace{f_s(M)}_{e}, \underbrace{f_r(M, S)}_{\text{A says that 'S said M'}})$.

  e: Evidence in case of future dispute.

Key Escrow

Provide strong security for communications while simultaneously allowing authorized government access to particular communications for law enforcement and national security purposes [DS94].

- Encryption could be used to conceal criminal and terrorist activities.
- By rendering communications immune from lawful interception, encryption threatens law enforcement and public safety.
- Special tamper-resistant hardware encryption device (Clipper and a Key Escrow System (KES).
- The EES uses SKIPJACK (64-bit block, 80-bit key) and a Law Enforcement Access Field (128-bit LEAF).
- Each Clipper chip has an 80-bit Device Unique key (KU) and an 80-bit common Family Key (KF).
- Key Exchange method unspecified. A session key KS is somehow generated.
- The LEAF and IV are transmitted for synchronization and LEAF validation.
- Infeasible to deploy the system without transmitting a valid LEAF [Bla94].

LEAF = | session key $(80b)$ | unit id $(32b)$ | checksum $(16b)$ |

Mental Poker

- $A \rightarrow B$: $E_{K_A}(C_1) \ldots E_{K_A}(C_{52})$. $C_i =$ "Jack of Spades".
- B chooses five and sends to A: $E_{K_B}(E_{K_A}(C_i)), \ldots E_{K_B}(E_{K_A}(C_m))$.
- A unlocks the five that B has chosen

  $$\left[D_{K_A}(E_{K_B}(E_{K_A}(C_i))), \ldots, D_{K_A}(E_{K_B}(E_{K_A}(C_m)))\right]$$

  to yield

  $$\left[E_{K_B}(C_i), \ldots, E_{K_B}(C_m)\right]$$

  and sends them back to B.
- B can now get $C_i \ldots C_m$.

Note: To realize this scheme, one can use $\left((C_i^{\alpha})^{\beta}\right)^{\alpha^{-1}} = C_i^{\beta}$

Who will pay for dinner?

If it's heads, Pete will pay, if it's tails, Nancy will. So Pete flips a coin in his office and tells Nancy the result over the phone!

- Pete selects two public key pairs: $(E_i, D_i), (E_j, D_j)$.
- Nancy chooses K_N to a symmetric algorithm S known to both.
- $P \rightarrow N$: (E_i, E_j).
- $N \rightarrow P$: $E_h(K_N)$ picked at random, $h = i \mid j$.
- P "guesses" h and retrieves $K_P = D_{h'}(E_h(K_N))$.
- $P \rightarrow N$: sends $M = E_{K_P}(\text{"Pete will pay"})$.
- If Nancy can read $D_{K_N}(M)$, Pete pays, otherwise Nancy pays.

References

[Bla94] Matt Blaze. Protocol failure in the escrowed encryption standard. *Proceedings of Second ACM Conference on Computer and Communications Security*, November 1994.

[DS81] Dorothy Denning and G. Sacco. Timestamps in key distribution protocols. *Communications of the ACM*, 24(8):533–536, 1981.

[DS94] Dorothy E. Denning and Miles Smid. Key escrowing today. *IEEE Communications Magazine*, pages 58–68, September 1994.

[NS78] R. M. Needham and M. D. Schroeder. Using Encryption for Authentication in Large Networks of Computers. *Communications of the ACM*, 21(12):993–999, December 1978.

[Pfl96] Charles P. Pfleeger. *Security in Computing.* Prentice Hall, iind edition, 1996.

[Seb] Jennifer Seberry. *?* ?, ? Available online at http://www.itacs.uow.edu.au/people/jennie/.

[Sta99] William Stallings. *Cryptography and Network Security.* Prentice-Hall Inc., 2nd edition, 1999.