Computer and Network Security

Computer & Information Sciences & Engineering
University Of Florida
Gainesville, Florida 32611-6120
nemo@cise.ufl.edu

Advanced Cryptographic Protocols (Pfleeger Ch. 4, Schneier Ch. 4, 23)

1 Types of Protocol Considered

1.1 Arbitrated

Trusted third party involved vs. Non-arbitrated - only the principals, mutually suspicious

1.2 Adjudicated

Third party can verify what has happened and determine if one of the parties cheated

1.3 Self-enforcing

Either one of the parties can determine and prove that cheating has occurred if it did, as the protocol proceeds

2 What to look for

1. Initial assumptions
2. Trust relationships - who trusts whom, and for what
3. Goals of the protocol
4. Hidden assumptions (trust, keys, etc.)
5. Weaknesses to various forms of attack
6. Requirements on underlying mechanisms (clock, PRNG, crypto)

3 Tools

3.1 Digital signatures

1. source
2. association
3. authenticity
4. integrity

3.2 Encryption

1. secrecy
2. association
3. authenticity
4. integrity
5. binding encrypted message elements

3.3 Nonces

1. prevent replay

2. allow association of messages in same run

3. must be random

4. must be used only once

5. may also act as confounder

6. may be altered in reply if symmetric key used

3.4 Timestamps

1. prevent replay
2. must protect time service
3. clock skew issues - acceptable bounds on error
4. must remember recent past

4 Advanced Protocols

4.1 Notation

1. $\{x|y\}$ is x concatenated with y (often used to randomize an otherwise small set of possible x's)

2. $\{M\}K$ is message M encrypted with key K

3. $\langle M \rangle K$ is message M signed with key K

4. K_{ab} is a symmetric key used by A and B

5. K_a is A's public key

6. K_a^{-1} is A's private key

The following two forms are used when we need to be explicit about encrypting and decrypting

1. $C = E(M, K)$ is also message M encrypted with key K

2. $M' = D(C, K)$ is ciphertext C decrypted using key K (Note that if C is not a message encrypted using key K, then M' is garbage.)

4.2 Secure Voting

4.2.1 Statement

N voters must be able to cast a ballot such that every voter knows

1. their vote counted,
2. every other voter voted just once,
3. nobody else knows how they voted, and
4. the final results (all the vote contents, but without IDs)

Let E_A and D_A be encryption and decryption (using public key system) for user A. Let $R_i(m, r)$ be a randomizing encryption, in which user U_i embeds random string r in message m and encrypts (so that two identical messages will look different). Only U_i knows R_i or R_i^{-1} (how to decrypt and extract m from $R_i(m, r)$.

4.2.2 Protocol (Original 3 voter version)

1. Each user U_i chooses a vote v_i,
2. encrypts it using the public keys (in order),
3. and then applies randomizing encryptions (again in order), producing

$$R_1(R_2(R_3(E_1(E_2(E_3(v_i))))))$$

4. and sends this secretly to U_1.

Note: Each voter can recognize any of the partial results in this chain for their own vote.

- Phase I - Shuffling the votes U_1 can tell who sent what, but can't tell what each is (due to the randomizing encryptions).

  1. U_1 verifies that U_1's vote is there, then produces for each i

  $$R_2(R_3(E_1(E_2(E_3(v_i)))))$$

  and sends these secretly to U_2.

  2. U_2 verifies that U_2's vote is there, then produces for each i

  $$R_3(E_1(E_2(E_3((v_i)))))$$

  and sends these secretly to U_3.

  3. U_3 verifies that U_3's vote is there, then produces for each i

  $$E_1(E_2(E_3(v_i)))$$

  and sends these secretly to U_1.

- Phase II - Revealing the results

  1. U_1 then decrypts, sends

  $$E_2(E_3(v_i))$$

  to U_2 and signatures to U_2 and U_3.

  2. U_1 then decrypts, sends

  $$E_3(v_i)$$

  to U_3 and signatures to U_1 and U_3.

  3. U_3 then decrypts, sends

  $$v_i$$

  and signatures to all.

4.3 Timestamping Services

4.3.1 Statement:

A wants to be able to prove to B that some message M was created by a certain time. It may be important that it was A who held M, and it may also be important not to reveal M to the timestamping service (TS), or to prevent collusion by A and TS.

4.3.2 TS Protocol 0:

TS just keeps each M that A (and others) send, along with the timestamp of when each M was received. Not only does TS see each M, but the database could be huge!

4.3.3 TS Protocol 1:

$$M1: \quad A \rightarrow TS: \quad M$$

$$M2: \quad TS \rightarrow A: \quad X = \langle M, ts_M \rangle_{K_{TS}}$$

$$M3: \quad A \rightarrow B: \quad TS, X$$

B then uses K_{TS} to decrypt and verify the message M with its timestamp, ts_M, signed by TS. TS does not have to keep the DB now.

However, TS gets to see the message, M, even if A encrypts it so that eavesdroppers can't see it.

4.3.4 TS Protocol 2 (M kept secret from TS):

$$M1: \quad A \to TS: \quad Y = H(M)$$

$$M2: \quad TS \to A: \quad X = \langle Y, ts_M \rangle_{K_{TS}}$$

$$M3: \quad A \to B: \quad TS, M, X$$

B uses M and H to compute Y, then uses K_{TS} obtain and verify the timestamp, ts_M, signed by TS. TS does not get to see M now.

However, TS and A can collude (TS can backdate ts_M).

4.3.5 TS Protocol 3 (Linking Protocol):

$$M-2: \quad I_{i-1} \rightarrow TS: \quad Y_{i-1} = H(M_{i-1})$$

$$M-1: \quad TS \rightarrow I_{i-1}: \quad T_{i-1}$$

$$M1: \quad A \rightarrow TS: \quad Y_i = H(M_i)$$

$$M2: \quad TS \rightarrow A: \quad T_i = \langle i, A, Y_i, ts_{M_i}, I_{i-1}, Y_{i-1}, ts_{M_{i-1}}, L_i \rangle_{K_{TS}}$$

where $L_i = H(I_{i-1}, Y_{i-1}, ts_{M_{i-1}}, L_{i-i})$.

This allows validation forward and backward as far as you like (as long as you can get ahold of each I_j in the chain). It may be hard to get each I_j, so force A to use multiple signers, as determined by $H(M)$, which makes it hard for A to select them.

4.4 All-or-Nothing Disclosure of Secrets (ANDOS)

4.4.1 Statement:

Here A wants to obtain one secret from B out of several that B holds, but does not want B to know which one she wants. Nor does B want to reveal more than one secret.

4.5 Blinded Signatures

4.5.1 Statement:

A wants B to sign a message M without revealing M to B. (Note that B had better make it clear that it is just providing a blind notary service, and not signing binding contracts with that particular key!)

4.5.2 Blind Signature Protocol:

A selects a random number N_a.

$$M1: \quad A \rightarrow B: \quad MN_a$$

$$M2: \quad B \rightarrow A: \quad \langle MN_a \rangle_{K_b}$$

A then unblinds the signature by dividing by N_a, to obtain

$$\langle M \rangle_{K_b}$$

Note that his requires multiplication (and division) and signing to commute, which it will for RSA and multilplication by nonces modulo p, for the same modulus.