Computer and Network Security

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

Basic Cryptographic Protocols (Pfleeger Ch. 4, Schneier Ch. 3, 4)

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 Basic 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 Mental Poker

4.2.1 Statement

Fairly and secretly distribute to each of N parties, K items chosen from a total of M, such that no two parties has the same item in their set.

Two person version ($N = 2, K = 52$):

In this protocol, Alice and Bob "deal" a five-card poker hand to each other in a distributed manner. K_x is a key only known to X, and $C_1, C_2, ..., C_{52}$ are the digital versions of the 52 cards.

$$\begin{aligned}
M1: \quad & A \rightarrow B: \quad \{C_{i1}\}_{Ka}, \{C_{i2}\}_{Ka}, ...\{C_{i52}\}_{Ka} \\
M2: \quad & B \rightarrow A: \quad \{\{C_{j1}\}_{Ka}\}_{Kb}, \{\{C_{j2}\}_{Ka}\}_{Kb}, ...\{\{C_{j47}\}_{Ka}\}_{Kb}, \\
& \qquad\qquad\quad \{C_{j'1}\}_{Ka}, \{C_{j'2}\}_{Ka}, ...\{C_{j'5}\}_{Ka} \\
M3: \quad & A \rightarrow B: \quad \{C_{i'1}\}_{Kb}, \{C_{i'2}\}_{Kb}, ...\{C_{i'5}\}_{Kb}
\end{aligned}$$

where the i indices are a permutation of [1..52], the j and j' indices partition [1..52], and the i' indices are a subset of the j indices.

In essence,

1. Alice masks the "cards" for Bob
2. Bob chooses Alice's hand by not encrypting those five, randomly chosen quantities.
3. Alice can decrypt and see only these five, while the other 47 are hidden by Bob's encryption.
4. Alice then chooses Bob's hand blindly by removing her encryption from five of the doubly encrypted quantities and sending these to Bob
5. Bob can then remove his encryption and see his hand.

Nota bene: This assumes that encryption with the two keys commutes! i.e., $D(E(E(M, K_A), K_B), K_A) = E(M, K_B)$, which may not be so.

4.3 A Variant - Secret Item Distribution

1. A sends a sequence of items encrypted with K_A
2. B selects one (can't see what it is yet) and encrypts with K_B, sends back to A
3. A decrypts the returned doubly encrypted item using K_A, so that it is now only encrypted using K_B, and returns to B
4. B decrypts using K_B and gets item.

Possible Problem - A knows that whatever B got it had to be one of the items that A had sent out before receiving B's selection....

4.4 Probabilistic Transfer (a.k.a.Oblivious Transfer)

4.4.1 Statement:

A wants to transfer some preset message M to B with probability one half. A must be able to verify that B did get it if B claims to have gotten it, and B must be able to verify that A did not cheat if B does not get it.

4.4.2 Original Version

Here, Nancy and Pete flip a fair coin in a distributed manner.

- K is a symmetric key generated by Nancy,
- K_i and K_j are public keys generated by Pete, and
- K_i^{-1} and K_j^{-1} are their inverses, respectively.
- M is a message stating that the coin has value *heads*.
- $\{X\}_K$ is a message X that has been processed using key K, which will either encrypt X or decrypt it, depending on the application (i.e., $\{X\}_K$ encrypts X with symmetric key K, but if $X = \{Y\}_K$, then $\{X\}_K = \{\{Y\}_K\}_K = Y$).
- x is either i or j (Nancy's choice),
- and y is either i or j (Pete's choice).

$$
\begin{array}{lll}
M1: & P \rightarrow N: & K_i, K_j \\
M2: & N \rightarrow P: & \{K\}_{Kx} \\
M3: & P \rightarrow N: & \{M\}_{\{\{K\}_{Kx}\}_{K_y^{-1}}} \\
M4: & N \rightarrow P: & \{\{M\}_{\{\{K\}_{Kx}\}_{K_y^{-1}}}\}_K, K \\
M5: & P \rightarrow N: & M, K_i^{-1}, K_j^{-1}
\end{array}
$$

Discussion

If $x = y$, then Pete uses K to encrypt M, and Nancy sees M when she decrypts with K. Otherwise, Nancy sees garbage, since a different "key" was used to encrypt than was used to decrypt M. Nancy "wins" if she sees M, otherwise Pete "wins." In the end, Pete can use Nancy's K (sent in message $M4$) to verify that Nancy did not cheat at $M2$ or $M4$ (she used the public key corresponding to the private key Pete used to decrypt K to obtain the encryption key for M used in $M3$), and satisfy himself that Nancy won. Nancy can verify that Pete won by using the private key corresponding to the public key that she did not use in $M2$ to generate the same key that Pete used to encrypt M in $M3$, then decrypting that message and seeing that M really was encrypted using the "wrong" key.

4.5 One-way Accumulators

4.5.1 Statement:

Want a function that takes multiple inputs and returns the same value, regardless of the order of the inputs. Also, want it to be one-way, so that the inputs can remain secret.

4.5.2 Example of use:

1. A and B are members of secret group $G = \{A, B, C, ...\}$
2. A has her ID, I_a, and the one-way accumulation of the other member IDs, $X_a = ACC_{j \in G, j \neq A}(ID_j)$
3. B has his ID, I_b, and the one-way accumulation of the other member IDs, $X_b = ACC_{j \in G, j \neq B}(ID_j)$
4. A and B want to verify each other - exchange IDs and Xs.
5. Each checks that the computed value is the same as their own.

4.5.3 Example Implementation:

$$ACC(x_i, y) = x_{i-1}^{y} mod n$$

where $n = pq$ for two primes p and q and x_0 is agreed upon in advance. The value x_i is the accumulation so far, and y is the next value to include in the accumulation. So,

$$\begin{aligned} ACC(ACC(x_0, y_1), y_2) &= (x_0^{y_1})^{y_2} mod n \\ &= x_0^{y_1 y_2} mod n \\ &= x_0^{y_2 y_1} mod n \\ &= (x_0^{y_2})^{y_1} mod n \\ &= ACC(ACC(x_0, y_2), y_1) \end{aligned}$$