

Key Establishment

BMEVITMAV52

Information and Network Security

feher.gabor@tmit.bme.hu

Key establishment

- *Key establishment* definition: a shared secret becomes available to two or more parties, for subsequent cryptographic use
 - The established keys vary on subsequent executions of the protocol (dynamicity)
 - The shared secret is often used as a *session key* protecting the communication
 - Limit the available ciphertext
 - Limit the exposure caused by compromised keys
 - Keys are created on-demand (No storing required)
 - Independent communication sessions
 - Key transport and key agreement
- *Authenticated key establishment protocol*: Establish a shared secret with an authenticated party

Key transport and agreement

- *Key transport* definition: one party creates or otherwise obtains a secret value, and securely transfers it to the other(s).
- *Key agreement* definition: a shared secret is derived by two (or more) parties as a function of information contributed by, or associated with, each of these, (ideally) such that no party can predetermine the resulting value.

Key authentication and confirmation

- (Implicit) *Key authentication* definition: one party is assured that no untrusted third party may gain access to a particular secret key
 - Key authentication is independent of the actual possession of such key by the second party, or knowledge of such actual possession by the first party; in fact, it need not involve any action whatsoever by the second party
- *Key confirmation* definition: one party is assured that a second party (possibly unidentified) actually has possession of a particular secret key
 - Identify the key
 - Can be easily added (keyed hash, hash on key)
- *Explicit key authentication* definition: is the property obtained when both (implicit) key authentication and key confirmation hold

Characteristics

- Nature of authentication
 - Entity authentication
 - Key authentication
 - Key confirmation
- Reciprocity of authentication
 - Unilateral or mutual
- Key control
 - A party control the value of a key or no party can predict the value of the key
- Key freshness
 - The key is never used before

Characteristics (cntd.)

- Efficiency
 - Number of messages required
 - Number of bits transferred (bandwidth)
 - Complexity of computations
 - Precomputation possibility
- Third party requirements
 - On-line, off-line or no third party
 - Degree of trust required in the third party
- Type of certificate is used (if any)
- Non repudiation
- System setup
 - Initial key setup

Adversaries in key establishment

- The underlying cryptographic mechanisms used (encryption, hash, digital signatures, ...) are assumed to be secure
 - Attacking the protocol itself (The adversary is not a cryptanalyst)
- Passive attack
 - Record and analyze protocol messages
- Active attack
 - Modifies, remove or inject messages

Adversaries in key establishment (cntd.)

- Messages are transported over unprotected channel
 - record, alter, delete, insert, redirect, reorder, and reuse past or current messages, and inject new messages
- Model: parties receiving messages exclusively via intervening adversaries
 - relaying messages unaltered to the intended recipients, or carrying out (with no noticeable delay) any of the above actions

Attack types

- Deduce a session key using information gained by eavesdropping
- Participate covertly in a protocol initiated by one party with another, and influence it,
- Initiate one or more protocol executions (possibly simultaneously), and combine (*interleave*) messages from one with another
- Deceive a legitimate party regarding the identity of the party with which it shares a key, without being able to deduce the session key itself

Perfect forward secrecy

- *Perfect forward secrecy* definition: compromise of long-term keys does not compromise past session keys
 - Also known as break-backward protection
 - Previous traffic is locked securely in the past
- *Known-key attack* definition: compromise of past session keys allows
 - either to compromise future session keys (passive adversary) or impersonation in the future (active adversary)
 - compromise of session keys may be easier than that of long-term keys
 - time extensive cryptanalytic effort may uncover past session keys

Key transport protocols

- Based on symmetric encryption
 - Serverless
 - With server
- Based on asymmetric encryption
 - With encryption
 - Encryption + signing

Point-to-point key update

- Based on a previously shared long-term, symmetric key
 - Participant: A,B
 - r_A : random number, t_A : timestamp, n_A : sequence number
 - Key: K
 - Session key: S
- Key transport with one pass
 - (1) $A \rightarrow B : \{r_A\}K$
 - Implicit key authentication. The new session key is r_A
 - Additional fields
 - (1') $A \rightarrow B : \{r_A, t_A^*, B^*\}K$
 - Timestamp provides freshness
 - B^* prevent undetectable immediate message replay back to A
 - Redundancy to provide explicit key authentication (B^*)

Point-to-point key update (cntd.)

- Key transport (cntd.)
 - If both party wants to contribute to the session key
 - (1) $A \rightarrow B : \{r_A\}K$
 - (2) $A \leftarrow B : \{r_B\}K$
 - The session key is $f(r_A, r_B)$
- Key transport with challenge-response
 - (1) $A \leftarrow B : n_B$
 - (2) $A \rightarrow B : \{r_A, n_B, B^*\}K$
 - n_B replace the timestamp
 - If both party wants to contribute to the session key
 - (1) $A \leftarrow B : n_B$
 - (2) $A \rightarrow B : \{r_A, n_A, n_B, B^*\}K$
 - (3) $A \leftarrow B : \{r_B, n_B, n_A, A^*\}K$
 - The session key is $f(r_A, r_B)$
- Properties of point-to-point key update
 - Fail completely if long-term key K is compromised
 - Subject to replay attacks
 - Message modification can be detected with a built-in data integrity mechanism

Authenticated Key Exchange Protocol 2 (AKEP2)

- Based on a previously shared longterm, symmetric keys K and K' .
 - h_K is a MAC for entity authentication
 - $h_{K'}$ is a hash to generate the session key

(1) $A \rightarrow B : r_A$

(2) $A \leftarrow B : (B, A, r_A, r_B), h_K(B, A, r_A, r_B)$

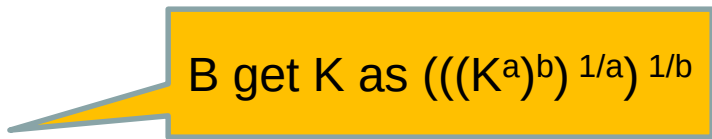
(3) $A \rightarrow B : (A, r_B), h_K(A, r_B)$

The session key is $h_{K'}(r_B)$

- There is no need to encrypt the base parameters of the session key

Shamir's no-key protocol

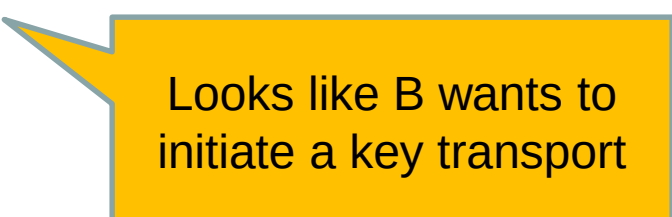
- Key transport without a priori shared keys
 - Using symmetric techniques (but involves modular exponentiation)
 - p prime; a, b random numbers
 - $1 \leq a, b \leq p-2$, each coprime to $p-1$
 - K is random, $1 \leq K \leq p-1$
 - (1) $A \rightarrow B : K^a \bmod p$
 - (2) $A \leftarrow B : (K^a)^b \bmod p$
 - (3) $A \rightarrow B : ((K^a)^b)^{1/a} \bmod p$
 - K is the session key
 - The Shamir's no-key protocol can use ciphers instead of modular exponentiation, where the cipher's encryption and decryption order is interchangeable. But Vernam cipher (XOR) can not be used!



B get K as $((K^a)^b)^{1/a} \bmod p$

Wide Mouth Frog protocol

- Key transport through a trusted third party
 - The server stores all the keys of the clients
 - (1) $A \rightarrow S : A, \{t_A, K_{AB}, B\}K_{AS}$
 - (2) $S \rightarrow B : \{t_S, K_{AB}, A\}K_{BS}$
 - Previously shared long-term keys
 - Timestamps required
 - Party A controls the key
 - Security flaw in wide mouth frog:
 - Adversary M performs a man-in-the-middle attack on the run of the protocol: $A \rightarrow M \rightarrow S \rightarrow M \rightarrow B$
 - (1a) $A \rightarrow M : A, \{t_A, K_{AB}, B\}K_{AS}$
 - (1b) $M \rightarrow S : A, \{t_A, K_{AB}, B\}K_{AS}$
 - (2a) $S \rightarrow M : \{t_S, K_{AB}, A\}K_{BS}$
 - Now the adversary can repeat the key transport several times
 - (1b') $M \rightarrow S : B, \{t_{Si-1}, K_{AB}, A\}K_{BS}$
 - (2a') $S \rightarrow M : \{t_{Si}, K_{AB}, B\}K_{AS}$
 - And can reinit K_{AB}
 - (2b) $M \rightarrow A : \{t_{Si}, K_{AB}, B\}K_{AS}$



Looks like B wants to initiate a key transport

Needham-Schroeder protocol

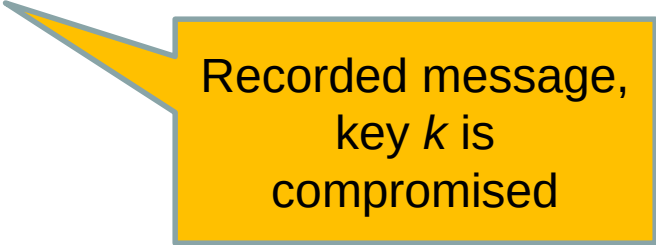
- Key transport using a trusted third party, with entity authentication and key confirmation
 - Independent of timestamps
 - (1) $A \rightarrow S : A, B, n_A$
 - (2) $A \leftarrow S : \{n_A, K_{AB}, B, \{K_{AB}, A\}K_{BS}\}K_{AS}$
 - (3) $A \rightarrow B : \{K_{AB}, A\}K_{BS}$
 - (4) $A \leftarrow B : \{n_B\}K_{AB}$
 - (5) $A \rightarrow B : \{n_B-1\}K_{AB}$
 - The server generates the session key: K_{AB}

Key
confirmation

This part comes from
the previous
message

Flaw in Needham-Schroeder protocol

- The server generates fresh keys, but party B is unable to verify it
 - If one session key is compromised, B can be tricked to use that key (from step 3):
 - (3) $M \rightarrow B : \{k, A\}_{K_{BS}}$
 - (4) $M \leftarrow B : \{n_B\}_k$
 - (5) $M \rightarrow B : \{n_B-1\}_k$



Recorded message,
key k is
compromised

Otway-Rees protocol

- Authenticated key transport using a trusted third party. Key authentication and key freshness
 - Using a transaction authentication ID: ID
 - (1) $A \rightarrow B : ID, A, B, \{n_A, ID, A, B\}_{K_{AS}}$
 - (2) $B \rightarrow S : ID, A, B, \{n_A, ID, A, B\}_{K_{AS}}, \{n_B, ID, A, B\}_{K_{BS}}$
 - (3) $B \leftarrow S : ID, \{n_A, K_{AB}\}_{K_{AS}}, \{n_B, K_{AB}\}_{K_{BS}}$
 - (4) $A \leftarrow B : ID, \{n_A, K_{AB}\}_{K_{AS}}$
 - Can be extended with key confirmation and entity authentication. Modified 4th message + a new one
 - (4) $A \leftarrow B : ID, \{n_A, K_{AB}\}_{K_{AS}}, \{B, n_B\}_{K_{AB}}$
 - (5) $A \rightarrow B : \{n_B-1, A\}_{K_{AB}}$

Key transport using PK encryption

- One-pass key transport by public-key encryption
 - The session key is sent encrypted by the other party's public key
 - (1) $A \rightarrow B : \{k\}P_B$
- Reply attacks in the case of compromised keys can be avoided using a timestamp
 - (1') $A \rightarrow B : \{k, t_A\}P_B$

Needham-Schroeder PK protocol

- Mutual entity authentication and key transport
 - (1) $A \rightarrow B : \{k_1, A\}P_B$
 - (2) $A \leftarrow B : \{k_1, k_2\}P_A$
 - (3) $A \rightarrow B : \{k_2\}P_B$
 - Encryption with the public key of the other party: $\{ \}P_X$
 - The session key is a function of k_1 and k_2
- Encryption in step 3 can be eliminated
 - (1) $A \rightarrow B : \{k_1, A, n_A\}P_B$
 - (2) $A \leftarrow B : \{k_2, n_A, n_B\}P_A$
 - (3) $A \rightarrow B : n_B$

Protocols with encryption + signing

- Provides source authentication
 - Encrypting signed keys
 - (1) $A \rightarrow B : \{k, t_A^*, \{B, k, t_A^*\}S_A\}P_B$
 - Timestamp is optional
 - B in the signature prevents B to send the key to other parties
 - Disadvantage: large information to protect
 - Encrypting and signing separately
 - (1') $A \rightarrow B : \{k, t_A^*\}P_B, \{B, k, t_A^*\}S_A$
 - Signing encrypted keys
 - (1'') $A \rightarrow B : t_A^*, \{A, k\}P_B, \{B, t_A^*, \{A, k\}P_B\}S_A$

Key arrangement

- Based on asymmetric techniques
 - Diffie-Hellman (-Merkle) key agreement (basic setup) - 1976
 - Prime p and generator g , $2 \leq g \leq p-2$
 - (1) $A \rightarrow B : g^x \bmod p$
 - (2) $A \leftarrow B : g^y \bmod p$
 - x and y are random, $1 \leq x, y \leq p-2$
 - The session key is $K = (g^x)^y \bmod p = (g^y)^x \bmod p$
 - Protect only from eavesdropping, but not from active attacks
 - No entity or key authentication

Diffie-Hellman key exchange example

- Alice and Bob agree on $p=23$ and $g=5$
 - Alice select $x=6$
 - Alice sends $5^6 \bmod 23 = 8$ ($g^x \bmod p$)
 - Bob select $y = 15$
 - Bob sends $5^{15} \bmod 23 = 19$ ($g^y \bmod p$)
 - Bob computes the session key ($g^{xy} \bmod p$)
 - $8^{15} \bmod 23 = 2$
 - Alice computes the session key ($g^{yx} \bmod p$)
 - $19^6 \bmod 23 = 2$

Station-to-station protocol (STS)

- The three pass variation of the Diffie-Hellman protocol. With mutual entity authentication and mutual explicit key authentication.
 - (1) $A \rightarrow B : g^x \bmod p$
 - (2) $A \leftarrow B : g^y \bmod p, \{\{g^x, g^y\}S_B\}E_K$
 - (3) $A \rightarrow B : \{\{g^x, g^y\}S_A\}E_K$
- There is digital signature + using the session key
- Moreover identities of A and B are protected
- Encryption can be avoided using MAC or alternatively signing the hash of the key

Secret sharing

- Multi-party key establishment protocols
 - Originally: enhanced reliability without increased risk
 - Gating the critical action on cooperation of t of n users
 - Secret is divided into shares
 - Specific subset of the shares enable to reconstruct the key
 - Usually a trusted device is necessary to combine the shares

Shamir's threshold scheme

- Based on polynomial interpolation
- $y=f(x)$ of degree $t-1$ is uniquely defined by t point (x_i, y_i)
 - S is the secret that should be distributed among n users
 - p is a prime, $p > \max(S, n)$
 - $a_0 = S, a_1, \dots, a_{t-1}$ random coefficients, $0 \leq a_i \leq p-1$
 - $f(x) = a_0 + a_1x + a_2x^2 + a_3x^3 + \dots + a_{t-1}x^{t-1}$
 - $S_i = f(i)$ (or any n distinct point)
- Any t shares reveal the secret using the Lagrange interpolation ($S = f(0)$)

$$f(x) = \sum_{i=1}^t y_i \prod_{1 \leq j \leq t, j \neq i} \frac{x - x_j}{x_i - x_j}.$$

Shamir's threshold scheme (cntd.)

- Properties
 - Perfect: Given knowledge of any $t - 1$ or fewer shares the shared secret remain equally probable
 - Ideal: The size of one share is the size of the secret
 - New shares (for new users) may be computed and distributed without affecting shares of existing users
 - Unlike many cryptographic schemes, its security does not rely on any unproven assumptions

References

- Alfred J. Menezes, Paul C. van Oorschot and Scott A. Vanstone, “Handbook of Applied Cryptography”, CRC Press, ISBN: 0-8493-8523-7
 - <http://www.cacr.math.uwaterloo.ca/hac/>
- Wikipedia - The free encyclopedia
 - <http://www.wikipedia.org/>