1 Security Goals

Confidentiality Third parties cannot read what A and B are saying.

Integrity Third parties cannot modify the contents of the communication.

Authenticity A and B are certain about each other's identities.

Availability Third parties cannot prevent A and B from communicating.

Non-repudiation Someone cannot deny what they have communicated.

Accountability Reliale log of communication history.

2 Crypto primitives

# Keys	Name	Key names	Notation
0	Hash functions	—	$h(m)$
1	Symmetric Crypto	Shared, secret	$K\{m\}$
2	Asymmetric Crypto (public key crypto)	Public & private keypair	$\{m\}_K$

3 Active Attacks

3.1 Replay attack

Intercepted data is sent again. Countermeasures: nonces, timestamps.

3.2 Reflection attack

Data from one session is reused in another session, to for example make the target do encryption/decryption work.

3.3 Man-in-the-middle attacks

Also passive: relay attack. Active involves reencryption. countermeasure: strong authentication.

4 Cypher modes

4.1 Electronic Code Book (ECB)

$m = m_0 m_1 m_2 \cdots \rightarrow C = K\{m_0\}, K\{m_1\}, K\{m_2\} \cdots$.

Attack vectors: occurrence frequency, swappping can go unnoticed easily.

4.2 Cypher Block Chaining (CBC)

$$c_0 = K\{m_0 \oplus IV\}$$
$$c_{n+1} = K\{m_{n+1} \oplus c_n\}$$

IV may be sent openly or be constant.

One garbled block means two blocks are lost in decyphering. Last block can be used to verify integrity.

4.3 Output Feedback Mode (OFB)

First, pick a random number (Initialisation Vector, IV) and use it to create a keystream:

$$K\{IV\}, K\{K\{IV\}\}, K\{K\{K\{IV\}\}\} \cdots$$

Then XOR with incoming bitstream. Garbled bits are lost, but only those bits. If sender/receiver are out of sync, everything is lost.

Variation: $c_n = m_n \oplus K\{IV + n\}$ (Counter mode).

4.4 Cipher Feedback (CFB)

$$c_0 = IV$$
$$c_{n+1} = K\{c_n\} \oplus m_{n+1}$$

5 Symmetric Crypto

5.1 Basic Techniques

1. **Substitution**: Swapping characters from the alphabet. Key K is the subsitution function.
2. **Transposition**: Changing positions of characters (by block). K is the position exchange function.
3. **One-time pad**: Take bitwise XOR of message with keystream. K is keystream of at least the same length as the message.

One-time pads are sometimes generated using linear feedback shift registers. See slide 25.

A downside of symmetric crypto is that one needs $\binom{N}{2} = \frac{N(N-1)}{2}$ keys if N people want to communicate pairwise securely.

Also, if key K is lost by A, B is also affected.

5.2 Basic Protocols

5.2.1 Integrity

$$A \longrightarrow B : m, K_{AB}\{h(m)\}$$

Hash function for efficiency.

B can verify integity by decrypting and comparing $h(m)$ with the hash he creates from m.

5.2.2 Confidentiality

$$A \longrightarrow B : K_{AB}\,\{m\}$$

Only those with K_{AB} can read this, obviously.

When combined with integrity ($A \longrightarrow B : K\{m, K\{h(m)\}\}$) it is important to use different keys, and thus if one is compromised, both integrity and confidentiality are broken.

5.2.3 Authenticity

"Shared Secret": You can be authenticated by something only you and the other party know. Problem: secret used in the clear.

It is better to send riddles that can only be solved efficiently using the secret key. The riddle needs to be fresh every time (against replay attacks). Often achieved by using *nonces*.

$$\begin{aligned} A &\longrightarrow B : A, N_A \\ B &\longrightarrow B : K_{AB}\{N_A, N_B\} \\ A &\longrightarrow B : N_B \end{aligned}$$

6 Hashing

6.1 Properties

Preimage Resistant (one-way) Given hash value x, it should be hard to find m with $h(m) = x$.

Second preimage resistant Given an m, it should be hard to find $m' \neq m$ with $h(m) = h(m')$.

Collision resistant It should be hard to find *any* pair $m \neq m'$ with $h(m) = h(m')$.

6.2 Non-revealing commitment

e.g. for flipping coins one can use:

$$\begin{aligned} A &\longrightarrow B : h(C_A, N_A) \\ B &\longrightarrow A : h(C_B, N_B) \\ A &\longrightarrow B : C_A, N_A \\ B &\longrightarrow A : C_B, N_B \end{aligned}$$

(nonces are used to prevent cheating by lookup table - coin outcomes are very limited)

6.2.1 Lamport's hash

C has for each user A a pair $[n \in \mathbb{N}, h^n(\text{passwd}_A)]$.

$$\begin{aligned} A &\rightarrow C : A \\ C &\rightarrow A : n \\ A &\rightarrow C : h^{n-1}(\text{passwd}_A) = x \end{aligned}$$

C can then verify the authenticity of A by checking $h(x) = h^n(\text{passwd}_A)$. C can also then set a new pair $[n-1, x]$ or $[n+1, h(h(x))]$.

7 Asymmetric Crypto

Using two keys: one for decryption (private key, K_d) and one for encryption (public key, K_e).

- Encryption: $\{m\}_{K_e}$

- Decryption: $[c]_{K_d}$.

Identity function:

$$[\{m\}_{K_e}]_{K_d} = m$$

Though for some systems $\{[m]_{K_d}\}_{K_e} = m$ is also valid, this is not a requirement.

8 Number theory

8.1 Equivalence classes

$$\forall N \in \mathbb{N}, n \in \mathbb{Z}, m \in \mathbb{Z}\, [n \equiv m \pmod N \Leftrightarrow \text{ there is a } k \in \mathbb{Z} \text{ with } n = m + k \cdot N]$$

$\mathbb{Z}_N$ is the set of numbers modulo N. Thus

$$\mathbb{Z}_N = \{0, 1, \ldots N-1\}$$

For every $m \in \mathbb{Z}$ we have $m \mod N \in \mathbb{Z}_N$.

If a product modulo N is 1, for instance, $4 \cdot 4 \equiv 1 \pmod{15}$, you can say $\frac{1}{4} = 4 \pmod{15}$. For $\mathbb{Z}_p$ where p is prime, every non-zero number $n \in \mathbb{Z}_p$ has an inverse $\frac{1}{n} \in \mathbb{Z}_p$.

A finite cyclic group often has a generator g that $g^n = k$ so that k can become any number in the group.

8.2 Greatest Common Divisor

$\gcd(n, m)$ = greatest k which divides both n and m. If $\gcd(n, m) = 1$, one calls n, m relative prime.

$$\gcd(n, m) = \begin{cases} n & m = 0 \\ \gcd(m, n \bmod m) & \text{otherwise} \end{cases}$$

8.3 Extended GCD

$$\text{egcd}(m, n) = \begin{cases} \langle 1, 0 \rangle & n = 0 \\ \langle y, x - (y \cdot (m \operatorname{div} n)) \rangle & \langle x, y \rangle = \text{egcd}(m, n \mod n) \end{cases}$$

Results in the solutions a, b for $a \cdot m + b \cdot n = \gcd(m, n)$.

9 RSA

RSA has a public key e, a private key d and a modulo $n = p \cdot q$.
Public key: (n, e)
Private key: (n, d)

9.1 RSA key creation

1. Pick two primes p and q. $n = pq$
2. Calculate $\phi = (p-1)(q-1)$.
3. Pick e from $\mathbb{Z}^*_\phi$.
4. Calculate d from $\frac{1}{e} \in \mathbb{Z}^*_\phi$ using $\text{egcd}(\phi, e)$.

9.2 Encryption / Decryption

Encrypt using RSA by $c = m^e \mod n$
Decrypt using $m = c^d \mod n$.

9.3 Signatures with RSA

$$A \longrightarrow B : m, [h(m)]_{d_A}$$

Works because RSA is symmetric.

9.3.1 Blind signatures

1. A computes $m' = r^e \cdot m \mod n$, where r is a random number, and gives this to B.
2. B signs m', giving $k = [m']^d \mod n$ to A.
3. A computes $\frac{k}{r} = \frac{r^{ed} \cdot m^d}{r} = \frac{r \cdot m^d}{r} = m^d \mod n$

Can be used for anonymous signatures on e-cash, e-voting…

9.4 Certificates

A certificate is a public key that has been signed by a thrusted third party.

10 Diffie-Hellman Key Exchange

Using a generator g in a finite cyclic group:

$$A \longrightarrow B : A, g^{s_A}$$
$$B \longrightarrow A : B, g^{s_B}$$

Then $K_{AB} = g^{s_A s_B} = (g^{s_A})^{s_B} = (g^{s_A})^{s_B}$.
This is not authentication!

11 El-Gamal

Fix a generator $g \in G$ of size N.
Private key: $n \in \mathbb{N}$ with $n < N$.
Public key: $h = g^n \in G$.

11.1 Encryption / Decryption

11.1.1 Encryption

1. Represent message as $m \in G$.
2. Choose a random $r < N$.
3. $\{m\}_h = (g^r, m \cdot h^r)$.

11.1.2 Decryption

1. Ciphertext $c = (c_1, c_2)$ with $c_i \in G$.
2. $$[(c_1, c_2)]_n = \frac{c_2}{(c_1)^n} = \frac{m \cdot h^r}{(g^r)^n} = \frac{m \cdot (g^n)^r}{(g^r)^n} = m$$

11.1.3 Signatures

Choose random $r < p - 1$ in $\mathbb{Z}_p^*$ with $\gcd(r, p-1) = 1$ and then:

$$\operatorname{sign}_n(m) = \left(g^r, \frac{H(m) - n \cdot g^r}{r} \mod p - 1 \right)$$

Verification of signature (s_1, s_2):

$$g^{H(m)} \overset{??}{=} s_1^{s_2}, h^{s_1}$$