Course information

Week 1: Introduction to basic concepts, and Cryptography and Confidentiality
Week 2: Cryptography and Authentication
Week 3: Key Management and Infrastrucures
Week 4: Network Security, Protocols etc.
Week 5: System security and Security Policies
Week 6: Security Threats and Attacks
Week 7: A Case from Real Life: Security at the Science Faculty

Exam Questions

1. Cryptography, confidentiality
2. Cryptography, authentication
3. Key management and Infrastructures
4. Network Security
5. System Security and Models for Security Policies
6. Threats and Pitfalls

Note 1 - Confidentiality

Security objectives

Confidentiality	Information does not leak to other people than the ones intended
Authenticity	There has not been tampered with the data by third party
Availability	The system must be working when we need it

Security policy	A precise statement about the security goals of a specific system
Threat model	Defines which threats to worry about
Security mechanism	How the system technically is going to enforce the *security policy*

Two types of cryptographers

Concerned with *condentiality or authenticity*
Concerned with *kind of security provided*
    Can be either **unconditional** or **computational**

Unconditional security	Impossible to break encryption
Computational security	Encryption it takes very long time to break (*secret-key / public-key*)

Secret key	Encryption and decryption with *the same key*
Public key	Encryption and decryption with *different keys*

Secret key

Pros Fast to encrypt and decrypt
Cons both ends has to agree on a shared key (e.g. physically meet)

One-time pad (*Impossible to break - unconditional security*)
Messages are encrypted with a key of the same length as the message. The message and key is XOR'ed which outputs the ciphertext. The message is deciphered by XOR'ing the ciphertext with the key.
Each key must be used only once. This constraint renders it useless in practice.

Nonce Number used once. Used when more messages are sent using the same key so two messages with the same plaintext looks unrelated. It must hold that $m = D_k(E_k(m, nonce))$.

Exhaustive key search Try decrypting until the key is found (*requires a known message*)

Stream ciphers

A function that takes a key and a nonce and produce a longer bitstring. This bitstring is used as with the one-time pad (XOR).

Advantages Stream ciphers can handle messages of any length.
Problems Hard to make efficient, and block ciphers can also be used as stream ciphers
Examples RC4 (outdated), SALSA20, SNOW

Block ciphers

Encrypts a fixed size message.

DES (56 bit keys, 64 bit blocks)
Triple-DES (112 bit keys, 64 bit blocks)
AES (128 bit keys, 128 bit blocks)

Modes of opeartion

Cipher Block Chaining (CBC)
- A message of *t* blocks makes a cipher of *t+1* blocks
- First block is the *nonce (IV)*
- Blocks are encrypted this way $C_i = AES_K(M_i \text{ XOR } C_{i-1})$

Counter (CTR)
- A message of *t* blocks makes a cipher of *t+1* blocks
- First block (C_0) is the *nonce (IV)*
- Blocks are encrypted this way $C_i = AES_K(IV + i) \text{ XOR } M_i$
- Advantage over CBC, can encrypt blocks in parallel
- Only need the encryption function (*even to decrypt*)

Output Feedback (OFB)
- Makes the block cipher work like a stream cipher
- Repeadetly feed the output block into the cipher function

In *modes-of-operation lingo*, the nonce is called an **initialization vector** (*IV*)

Public key

Two algorithms, one for encryption and one for decryption. Also 2 keys. Everyone knows public key *pk*, but only one knows the secret key *sk*.

RSA Most widely used, based on *factoring problem*
El-hamal Based on elliptic curves. Will take over in a few years because of shorter keys

Public key in practise

Public key encryption is orders of magnitude slower than secret key encryption
Key enveloping Use *public key* to exchange a *secret key*. Public key is only used once.
Problem: secret keys are short and does not fill a complete block

Optimal Asymmetric Encryption Padding (OAEP)
Padding function to use with short messages (i.e. *key enveloping*)

Public key encryption should never be used without a proper padding function like **OAEP**

Note 2 - Authenticity

Authenticity Used to be sure who the sender is
MAC Message Authentication Code

Secret key systems

Two algorithms S and V
S - authenticates a message by producing a MAC
V - verifies a recieved message (*accepts or rejects message*)
$V_k(m, S_k(m)) = accept$

CBC-MAC Works with all block ciphers. The last block cipher is also the MAC. This is smart as no more computing is needed. The MAC is checked by recomputing the last block cipher from the message. As the last block depends on every other block, nothing in the message can be altered without affecting the last block. That is why it makes a good MAC.

HMAC TROLOLOLOL

Public key systems

Two algorithms S and V (*like secret key, but here they use different keys*)
S - authenticates a message by producing a MAC
V - verifies a recieved message (*accepts or rejects message*)

$V_{pk}(m,S_{sk}(m)) = accept$

Public key crypto-systems can easily be used for authentication. E.g. RSA is just the other way around than encryption. So $s^e \bmod n = (m^d \bmod n)^e \bmod n = m$.

Can be used as digital signatures

Problems An adversary can take a random string and lift it to the power of *e*, and thereafter claim than the result is a signed message. The adversary can't control the contents of m, so it will most likely not make sense. In some cases this can be used as an attack.

Important difference between public and secret key systems

In secret key systems B can be sure that the message came from A, but he can't convince anyone else about it because he could have made the message himself.
This does not apply to public key, so they are often known as *digital signature schemes*. Only A has his own secret key, but everyone can verify messages with the public key.

Hash functions

Solution to speed and security problems from the public key systems (*digital signatures*)
Properties that must be met

- It should be able to take a message of any length as input
- The output should always be a fixed length
- It should be fast to compute (similar to the fastest secret key encryptions)
- It should be a hard computational problem to produce a collision ($x \neq y \;\&\&\; h(x) = h(y)$)

Birthday attack Hash random strings to find collisions. The attack takes about $2^{k/2}$ evaluations of the hashfunction. Therefore the key must be long enough. 160 bit is said to be enough.

Digital signatures in real life

You must be able to report your key stolen. This poses a problem as no documents are valid anymore because the theif can forge old documents.
There must be some accompaning paperwork. The user must physically sign that he takes responsibility for his signature, and reports it stolen right away.
A digital signature commits the signer to exactly what was signed and nothing else. Example to sign an email with or without the recipient. With the recipient means that the intended recipient is the one signed. If only the content is signed, the message could be to anyone.
Replay attacks are possible. Transfer of money. The same message can be sent more times. Several ways to prevent this can be seen i chapter 7.

	Confidentiality	Authenticity
Secret-Key	AES, DES, RC4, IDEA	CBC-MAC, HMAC
Public-Key Hash Functions	RSA, El-Gamal, Ell. curves	RSA, El-Gamal, Ell. Curves RIPEMD-160, SHA-1, MD5

Week 3 - Key management and infrastructures

Any secret system parameter runs a greater risk of being revealed, the longer time you keep it constant, and the more you use it.

Solution is to use two layers of keys. One key K_{AB} that is only used to exchange ***session keys***. The session keys is used for a limited time and a new session key is chosen.

Does not work for multiuser systems. Instead ***Key Distribution Centers*** or ***Certification Authorities*** are used.

Key Distribution Centers (KDC)

A server (the KDC) holds a secret key for all members of the network. When A wants to talk to B, the KDC generates a session key K, encrypts it with the respective keys and send it to A and B. They can now communicate using the session key.

Kerberos is a KDC system.

Problems Everyone must trust the KDC completely
The KDC can forge traffic and pretend it is some user
If the KDC goes down, the system goes down

Certification Authorities (CA)

- Instead of a secret key scheme, a public key scheme could be used to echange the session key. So A encrypts a key K with B's public key pk_B and send it.
- But we can not assume that A has B's public key.
- A Certification Authority (CA) has its own keypair (pk_{CA}, sk_{CA}) which everyone is assumed to have. It is often coded directly into the browser.
- A must then verify himself in some way (*this must be done "physically"*) and send his public key to the CA.
- When A is authenticated, the CA sends a certificate containing ID_A, pk_A and $S_{skCA}(ID_A, pk_A)$. Everyone can check A's certificate because everyone has pk_{CA}.
- A and B can now communicate securely by first exchanging certificates.

Difference between CA and KDC

We only need to trust the CA not to issue certificates on false grounds. It does not see any secret keys, so A and B can communicate without involving the CA.

Certification in real life
A certificate may be reported stolen. Therefore the CA must publish blacklists of invalid certificates.
Therefore a certificate must have an expiry date, or the blacklists will grow indefinitely.

Other stuff a certificate could contain

- Name of the certificate owner
- Name of the CA who signed the certificate
- Method used by CA to check identity of certificate owner
- Date issued
- Validity period
- Rights and privileges of certificate owner.
- Crypto algorithm to be used for checking this certificate
- Crypto algorithm used by certificate owner
- Public key of certificate owner
- Signature of CA

Problem: A and B has certificates from different Certificate Authorities
A might not have the public key from CA_B on his computer, so he needs to verify that before verifying B's certificate. The solution to this is to make the CA's sign eachothers certificates. A then asks for CA_B's certificate, and can thereafter veryif B's certificate.

This can be implemented as certificate chains, where A will be lead through several CA's before authenticating CA_B's public key. It is important to note here that A can only trust B's public key as must as he trusts every CA in the chain.

X.509 Most common standard for certificates

Limitations on key management

Any secure system using cryptography must make use of one or more keys that are protected only by physical, non-cryptographic means.

Password security

Four important aspects of password security

- How is the password chosen?
- How is the password transmitted between password user and verifier?
- How is the password stored by the password user?
- How is the password stored by the password verifier?

Attack: Guessing the password

Client side

- The length of passwords makes a difference because there are more possibilities.
- Use of passphrases can lead to better passwords

Server side

- Account freezes after *x* failed login attempts. But then adversary can freeze a lot of accounts and do harm that way
- Take longer time to respond for every failed login attempt (*Good solution*)

Attack: Stealing the password from the user

- Watch over the shoulder physically (*in person & cameras*) or digitally (*spyware & keyloggers & eavesdropping*)
- Social engineering (*fooling the user to give his password*)
- Phishing attacks (*a kind of social engineering*)

Attack: Stealing the password from the verifier

- If the password database is stored in cleartext this can be stolen. Many passwords are stored in hashed form (*like UNIX*). A ***dictionary attack*** can be very efficient against this.

Hardware security

A code or signature can be copied without the user knowing. Therefore it might improve security by having some hadware that is nessesarry to break/use the code. This will cost more to steal, and the user can discover that the device is stolen and report it.

Tamper-evident Leaves traces showing it has been attacked
Tamper-resistant Devices very difficult to break into (*IBM 4758*)

The chain is only as strong as the weakest link.

Biometrics

- Can provide accescontrol to keys, but cannot replace cryptography.
- Some fingerprint and iris scanners perform acceptably.
- Again, the chain is only as strong as the weakest link

Preventing bypass of the system

If we want to protect our secret key *sk*, the only way to do it is to use *secure hardware* or *encrypt it with a password*. But passwords are often shorter than 128 bits, so the hash value is used for the encryption instead, $E_{h(pw)}(sk)$. But the adversary can just try all possible passwords so the

encryption is too weak. The solution is to chain the hashing functions so it will be hashed several thousand times. It will now take a noticable amount of time for each password check, and it becomes infeasable for the adversary. The user will not notice much because he only needs to do it once.

$$h(pw) = SHA\text{-}1(SHA\text{-}1(\ldots SHA\text{-}1(pw)\ldots))$$

Week 4 - Network Security

Needham and Schroeder

1. *A* chooses n_A and sends $E_{pkB}(ID_A, n_A)$ to *B*
2. *B* decrypts, checks ID_A, chooses n_B and sends $E_{pkA}(n_A,n_B)$ to *A*
3. *A* decrypts, checks that the correct value of n_A appears in the result, and sends $E_{pkB}(n_B)$ to *B*
4. *B* decrypts and checks that the correct value of n_B appears in the result

Attack

1. Suppose A starts a session with E, thus A will send $E_{pkE}(ID_A, n_A)$ to E.
2. E decrypts this and starts up a session with B, pretending to be A. So E will send $E_{pkB}(ID_A, n_A)$ to B.
3. B decrypts this, finds the right ID in the result, and sends $E_{pkA}(n_A,n_B)$ to E.
4. E is of course not able to decrypt this, but can instead simply forward the message $E_{pkA}(n_A,n_B)$ to A.
5. A will decrypt this, find a result that has exactly the form he expected, and will therefore return $E_{pkE}(n_B)$ to E.
6. E can decrypt, find nB, and is now able to send $E_{pkB}(n_B)$ to B.
7. When B decrypts, he will accept since he finds the right value of n_B in the result.

Secure Socket Layer (SSL)

- It takes a different approach than Needham-Schroeder by using digital signatures in addition to the encryption
- In its basic form, it requires that both Server and Client have public-key certificates and access to the corresponding private keys
- It is also possible, however, to do a “one-sided” SSL, where only the server has a certificate. Then login is used to verify the client

IPSec

IPSec is a set of protocols that do a job very similar to what SSL can do, but on a different layer in the standard protocol stack, namely inside the transport layer
IKE uses Diffie-Hellman key exchange

Week 5

Week 6 - System Security and Models for Security Policies

Goals for attacks (STRIDE model)

Spoofing (if identity) - pretend that you are someone else

Tampering (with data) - modify data

Repudiation - noone can prove that you did the action (opposed to nonrepudiation with digital signatures)

Information disclosure - get data that you are confidential

Denial of service - bring down the server so noone can use it

Elevation of privelege - become more powerful than supposed to be

Means and tools used by adversary

X.800

Passive attacks

Eavesdropping (*listen on the network connection*) (*fixed with encryption*)

Traffic analysis (*who is sending to who, how much and when*) (*fix: always send, or steganography*)

Active attacks

Replay (*fix: nonce*)

Modification (*fix: signatures*)

steganography Hiding information in other messages. Example, secrectly encode messages in images.