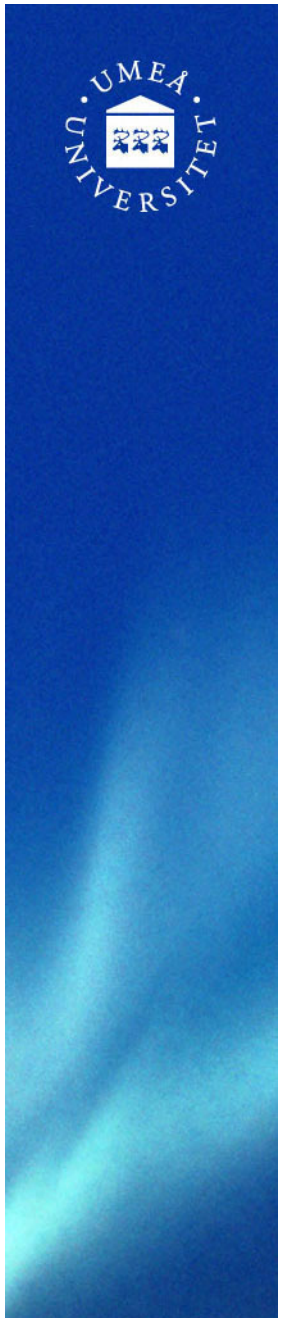
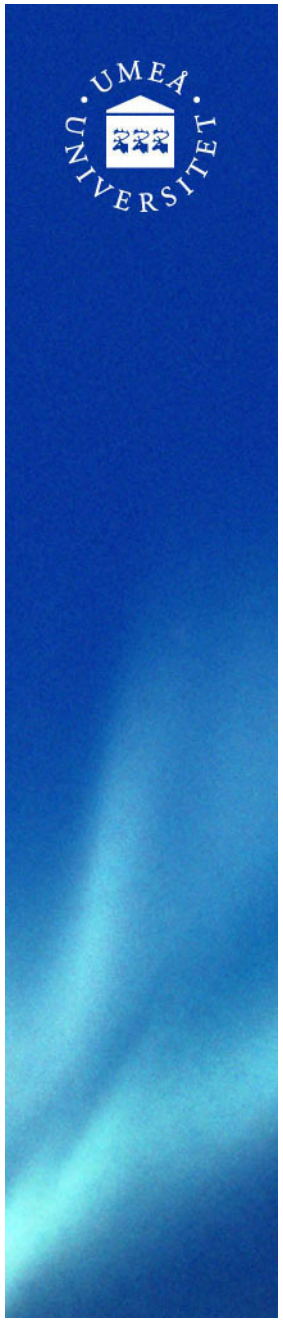


Distributed Systems
Security

2013-11-14

Cristian Klein
Department of Computing Science
Umeå University



Outline

- Why security?
- What is security?
- Cryptographic algorithms
- Security protocols
- Best practices

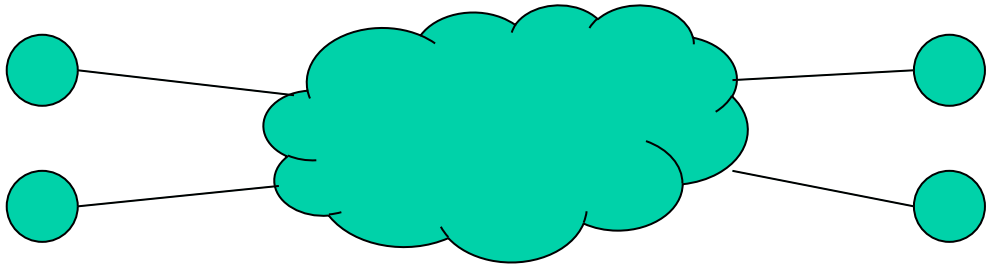
Instead of introduction

- 2012 losses due to hacking
  - Sony: 171 M$
  - Citigroup: 2.7 M$
  - Stratfor:2 M$
  - AT&T: 2 M$
  - Scottrade: 1 M$

http://www.hotforsecurity.com/blog/top-5-corporate-losses-due-to-hacking-1820.html

Why security?

- Distributed systems
  - Process provide access to resources
  - Exchange information through a shared network
- One needs to control
  - Who is accessing the exposed resource
  - What operations are allowed



What is security? (1/2)

- Three core values
  - Privacy
    - Only authorized principals are allowed to read certain information
  - Integrity
    - Only authorized principals are allows to modify certain information
  - Availability
    - Authorized principals can access information at all times

What is security? (2/2)

- Security policies
  - E.g., user A cannot see user B's bank statement
  - Technology independent
- Security mechanism
  - E.g., require an ID
  - Technology dependent

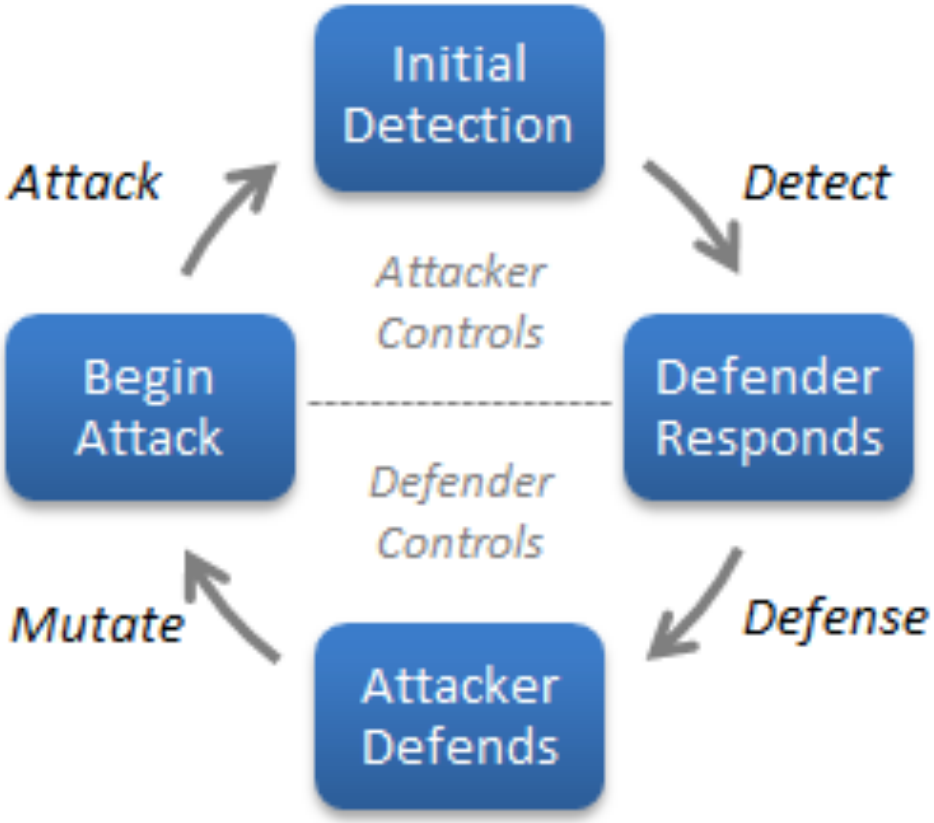
Security goals (1/2)

- Authentication
  - Who are you? Can you prove it?
  - E.g., PIN, password, Eduroam certificate
- Authorization
  - Access Control Lists
    - User A may read/write file F
  - Capabilities
    - Capability C: may read/write file F
    - User A has capability C
- Confidentiality
  - User A and B communicate with one another
  - User E cannot intercept their communication

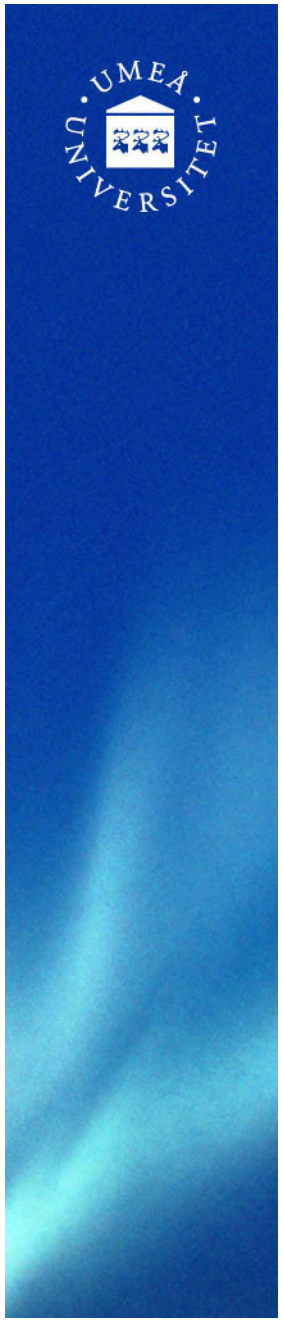
Security goals (2/2)

- Data integrity
  - Data has only been altered as intended
  - Was file F tampered with?
- Delegation
  - User A is allow to do operation O
  - User A passes this privilege to user B
- Non-repudiation
  - User A signs contract
  - User A cannot deny signature

Computer Attacks and Defenses



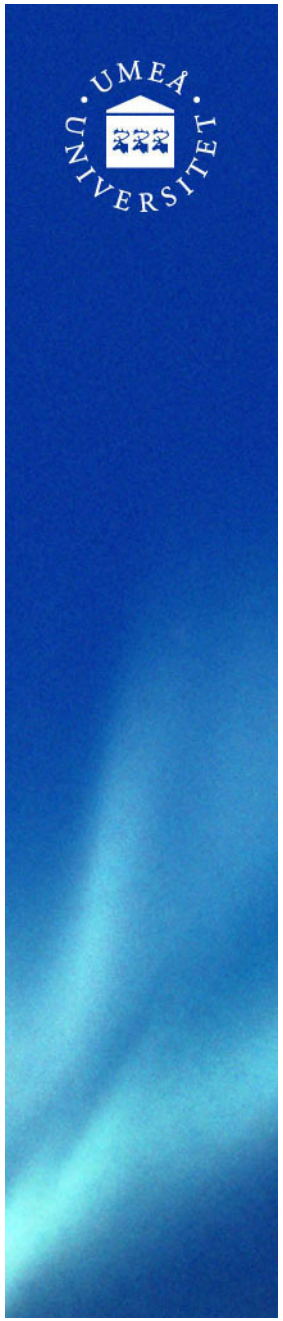
Stein et al, Facebook immune system, SNS '11



Cryptographic Algorithms

Cryptography

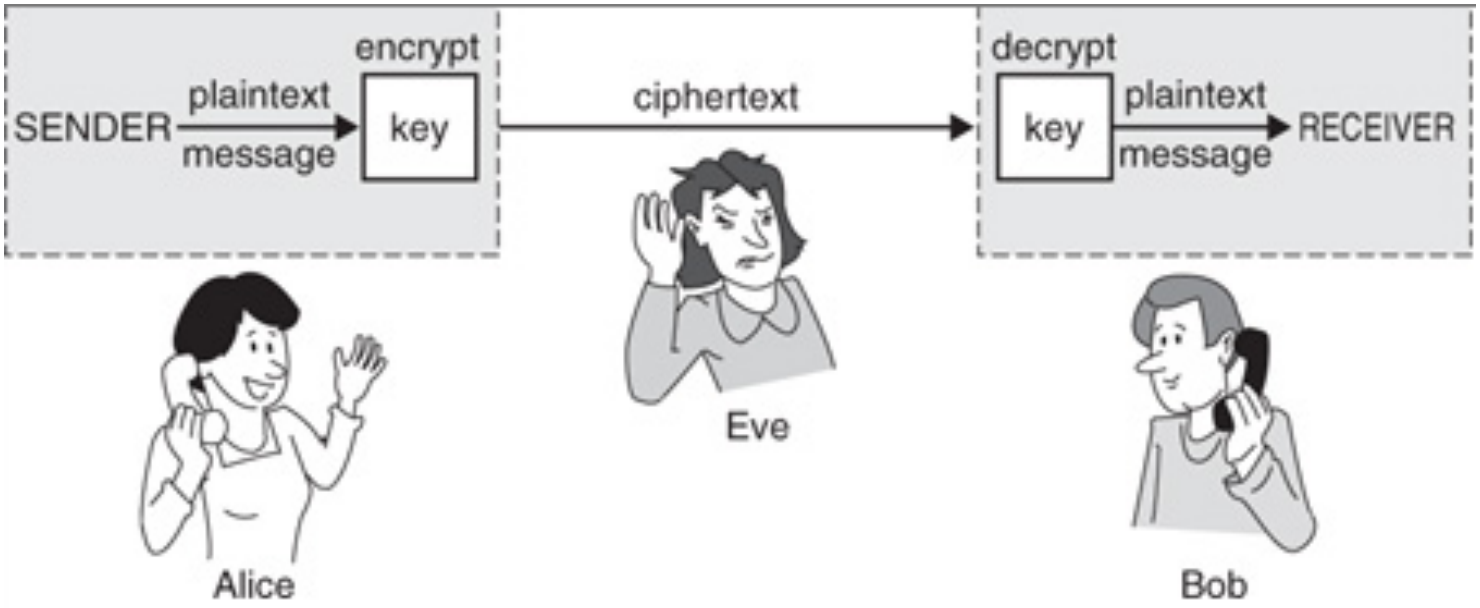
- "the practice and study of techniques for secure communication in the presence of adversaries" (Wikipedia)
- Highly mathematical field
- Cryptology
  - Creating secure algorithms
- Cryptanalysis
  - Breaking secure algorithms



Cryptographic Primitives

- Encryption algorithms
- Hashing algorithms
- Homomorphic encryption

Encryption



http://rosinstrument.com/pb/m/12317.htm

One-time padding

```
Plaintext : hello world
          XOR
Key       : random key
          =
Cyphertext: %^&*#A%323@
```

(not actual result)

- **Impossible** to crack

- Problem: key length = plaintext length
  - Key distribution?

Encryption: idea

- Invent a secret algorithm (bad)
- Use an open, proven algorithm
  - Keep key(s) secret
- Two families
  - Shared key
    - Block
    - Stream
  - Public key

Block cyphers

- Operate on a fixed block size
  - E.g., 128 bits
  - E(M, K) = M’
  - D(M’, K) = M
- Combine key with plaintext
  - Add, subtract, rotate, shift
  - Obtain confusion and diffusion
- Ideally
  - Key size = cryptographic strength
  - Brute-force attack only
    - 80 bits (okey)
    - 128 bits (strong)
    - 256 bits (very strong)

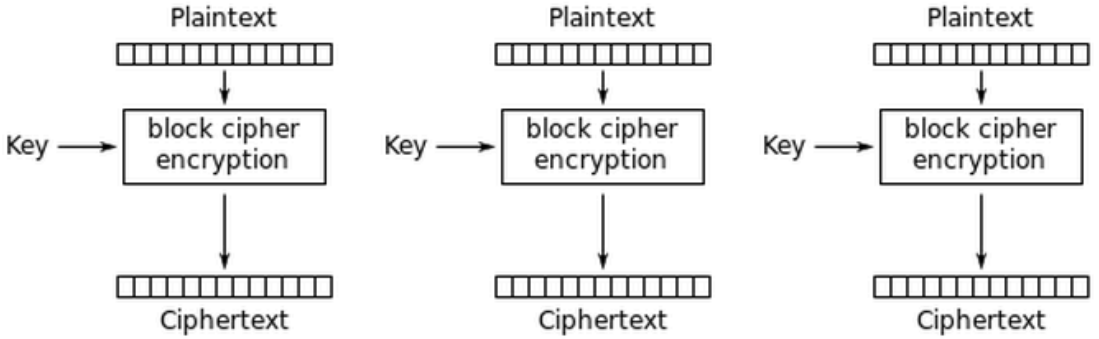
Block cyphers: examples

Name	Key size	Block size
3DES	168	64
Blowfish	32–448	64
IDEA	128	64
AES	128, 192, 256	128

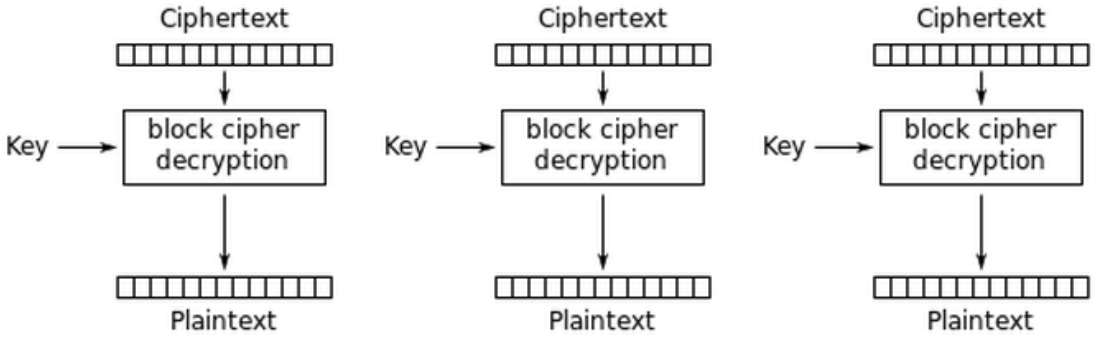
Stream cyphers

- Operates on the whole data
  - E(M, K) = M’
  - D(M’, K) = M
- Can be derived from block cyphers

Stream cyphers: ECB

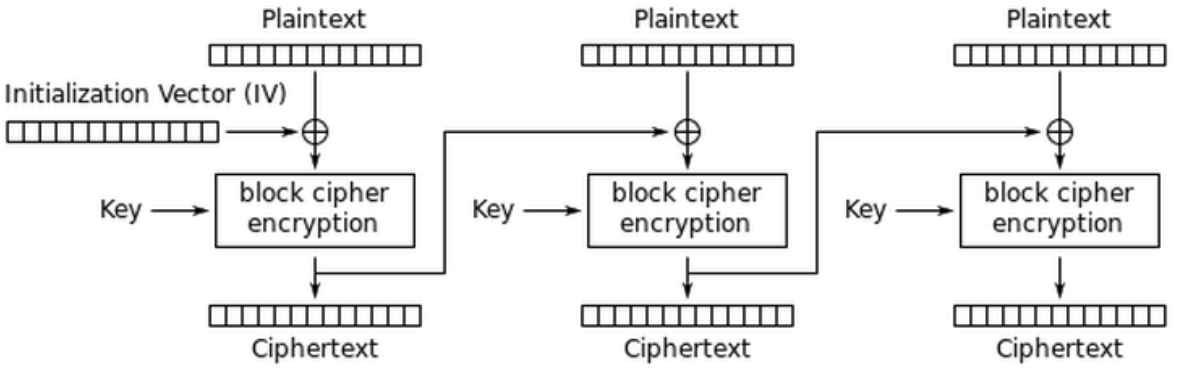


Electronic Codebook (ECB) mode encryption

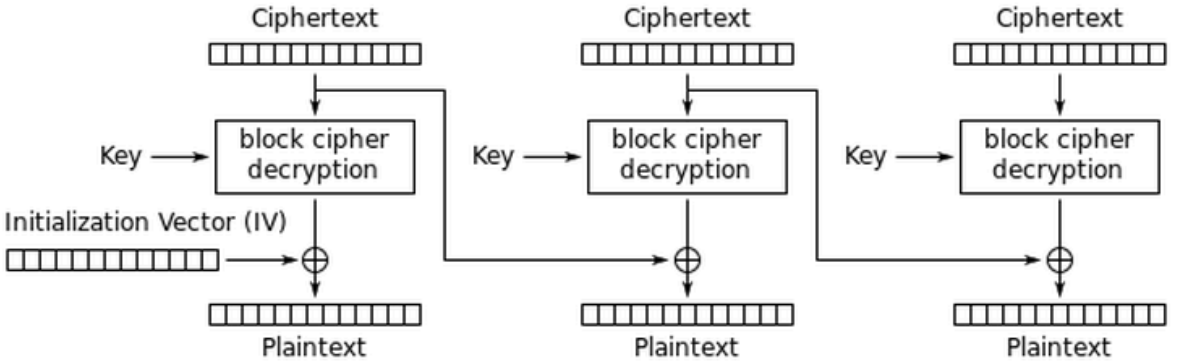


Electronic Codebook (ECB) mode decryption

Stream cyphers: CBC



Cipher Block Chaining (CBC) mode encryption



Cipher Block Chaining (CBC) mode decryption

Other stream cyphers

- Principle
  - "Strong" pseudo-random number generator => keystream
  - XOR keystream with message
- Example
  - RC4 (40–2048 bits security)

- What about key distribution?

Public-key cyphers

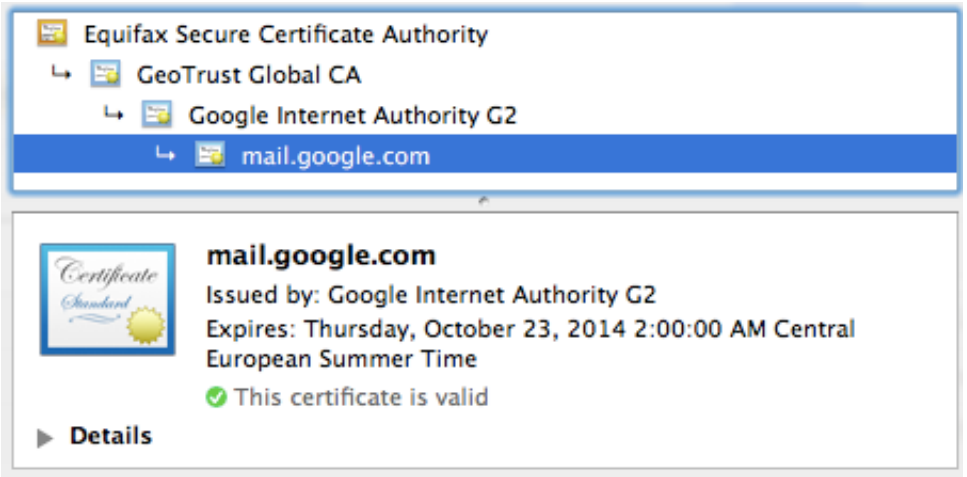
- Idea: two keys
  - Public key used for encryption (Ke)
  - Private key used for decryption (Kd)
  - Algorithm to derive Ke, Kd
  - E(M, Ke) = M'
  - D(M', Kd) = M D(M', **Ke**) **!=** M
  - Cannot compute Kd from Ke
- Based on "hard" problems
  - Integer factorization
  - Discrete logarithm
- Example: DSA, RSA, ECDSA

Signing with a public key

- Goal: ensure a document is authentic
  - Private key (Ks) used for signing
  - Public key (Kv) used for verification
- Example protocol
  - Signer computes S = E(M, Ks)
  - Signer publishes S, M
  - Verifier computes M’ = D(S, Kv)
  - Verifier checks that M = M’

Public key distribution (1/2)

- Out-of-band
  - Face-to-face meeting
  - Sealed envelope etc.
- Public-key infrastructure (PKI)



  - Certificate chain up to a trusted root CA
  - Revocation: expiry (slow), revocation list (fast)
  - E.g., Internet

Public key distribution (2/2)

- Web of trust
  - Certificate: public key, owner (email)…
  - Reciprocal signing of certificates
  - Think social networks
  - Revocation list
  - E.g., Pretty Good Privacy (PGP)

Cryptographic hash functions

- Problem
  - Public key cyphers only work on small messages
  - Need a method to securely transform large messages into small "summaries"
- Hashing algorithms
  - H(M) = h
  - M = message, arbitrary size
  - H = hash, small (e.g., 128 bits)

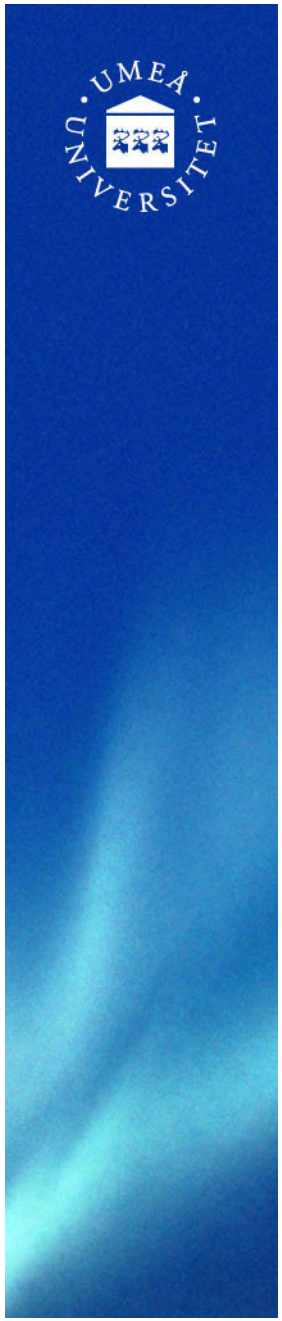
Cryptographic hash functions: properties

- H(M) is fast to compute
- If M=M’ then H(M)=H(M’)
- If M≠M’ then H(M)=H(M’) is unlikely
- Given h, infeasible to find M, s.t., H(M)=h
- Cryptographic strength = hash size / 2
  - Due to birthday paradox

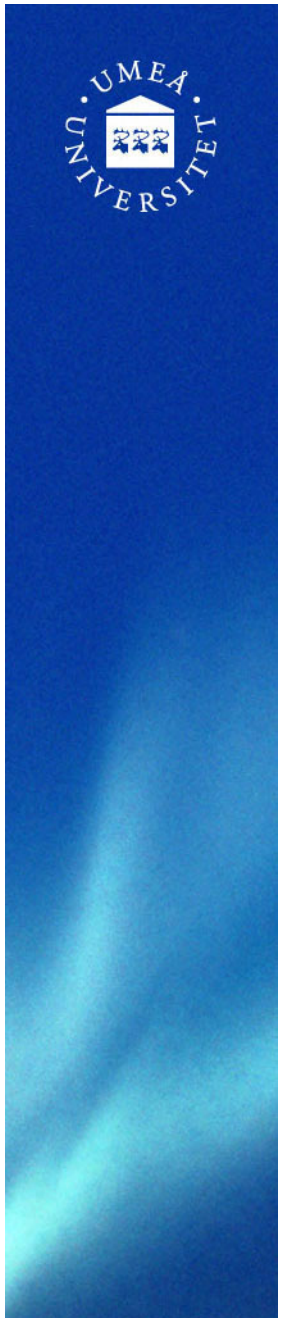
MD5	128 bits	Broken
SHA-1	160 bits	Weak
SHA-2 family	256 or 512 bits	
SHA-3 family	256 or 512 bits	

Homomorphic encryption

- Allow receiver to do certain operations on cyphertext without knowing the result
- E.g., process user query over a database
- Fundamentally very slow



Security Protocols



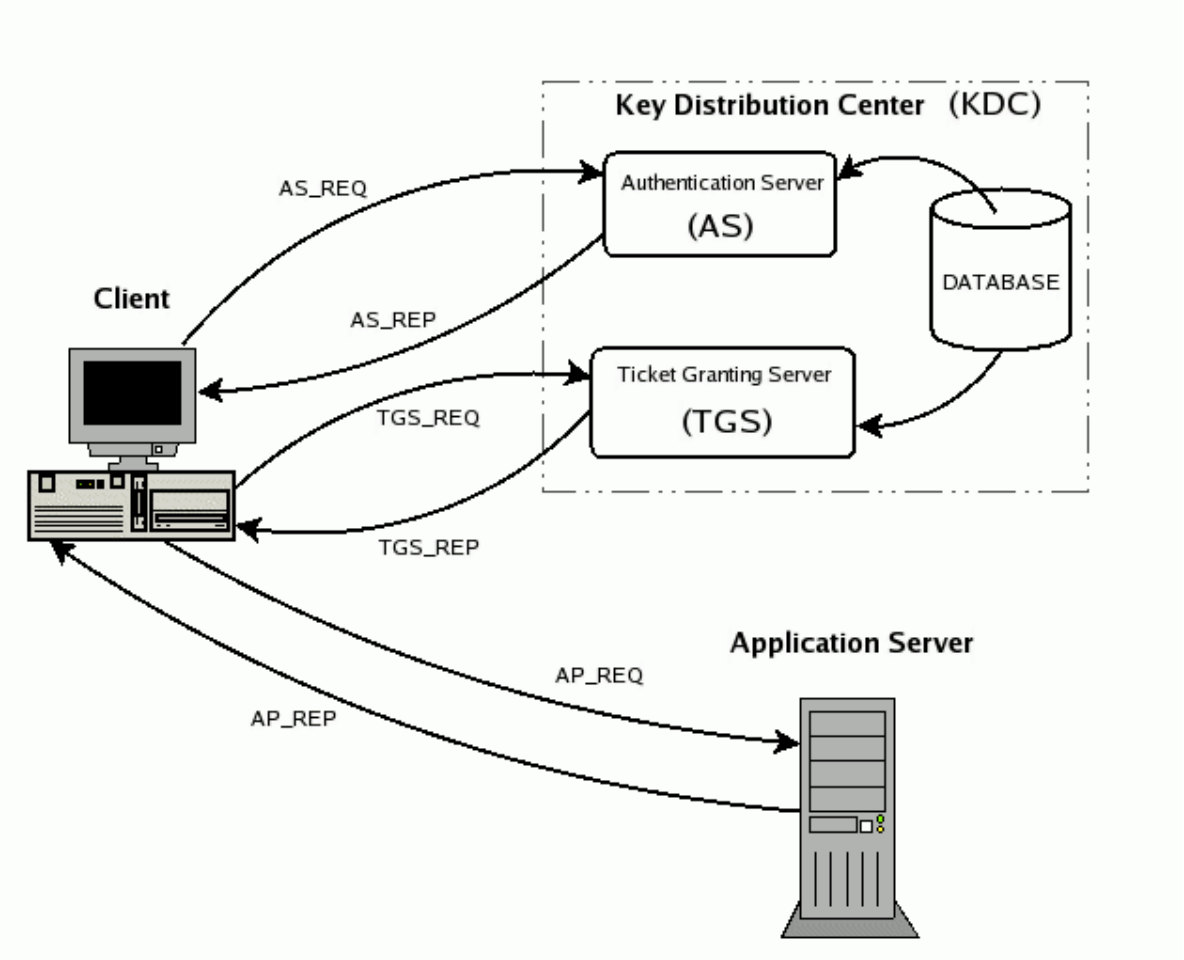
Security protocols

- Kerberos
- TLS

Kerberos

- Client-server model
- Authenticates both client and server
- Uses shared key cryptography
- Requires a trusted **Authentication Server**
- Issues **tickets**

Kerberos: architecture



http://www.zeroshell.org/kerberos/Kerberos-operation/

Kerberos: implementation

- The devil is in the details
- How to convert a password to a shared key?
  - Use a cryptographic hash repeatedly
- How to avoid replay attacks?
  - Use timestamps

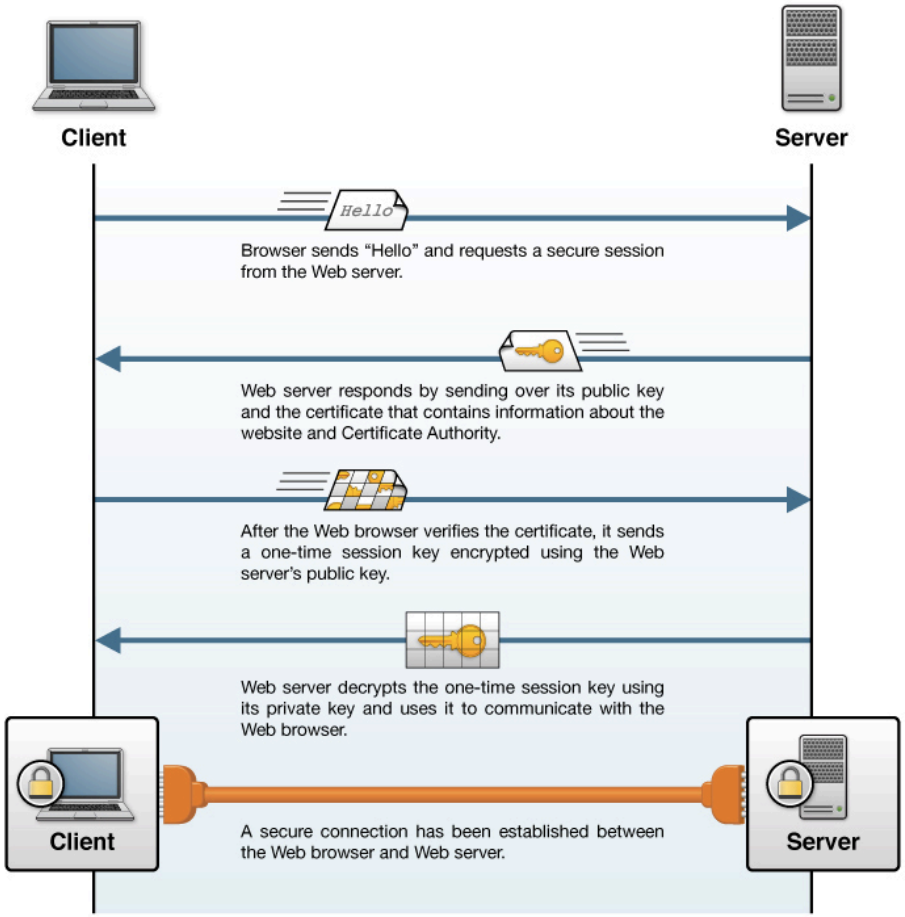
Kerberos: disadvantages

- Single point of failure
- Single point of attack
- Requires synchronized clocks
- Database is centralized

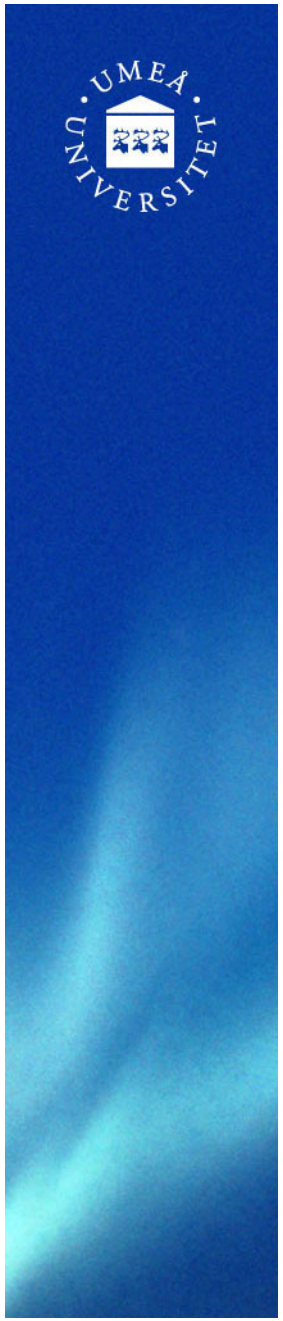
Transport Layer Security (TLS)

- Client-server model
- Provides mutual authentication
  - Mostly only server is authenticated
- Hybrid encryption
  - Public-key to initialize session
  - Shared key for transmission
- Widely used
  - Internet: HTTPS, email
  - Infrastructure: WiFi, Ethernet

TLS: implementation

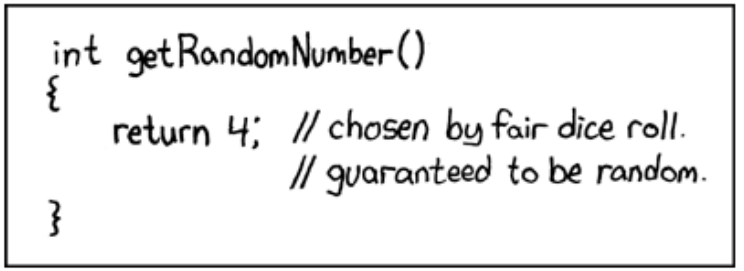


http://tech.kaazing.com/documentation/xmpp/3.5/security/c_tls.html



Best Practices

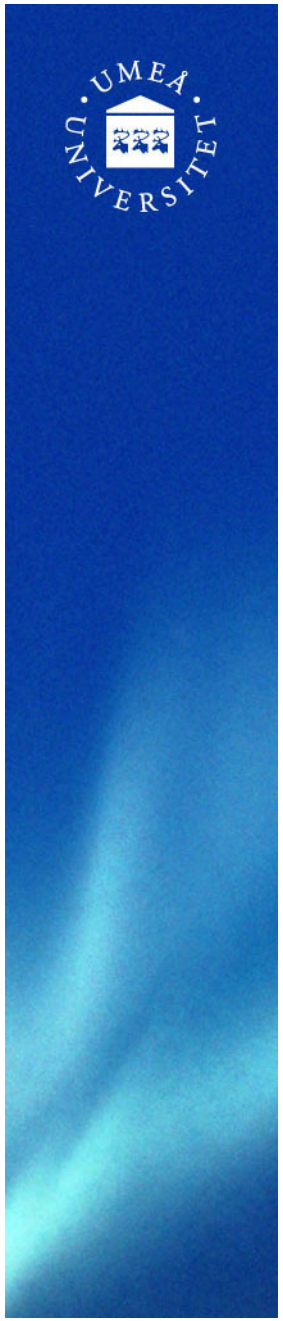
Security bugs (vulnerabilities)



- Netscape predictable PRNG in 1994
  - Used time-of-day, process ID and parent process ID
  - Session key was predictable
- Debian vulnerability in 2006
  - Uninitialized PRNG
  - Reduced key space to 32768
- Buffer overflows, dangling pointers, SQL injection, …
- Side-channel attacks

Security is difficult

- **Design with security from day 0**
  - Security in depth
- Use known algorithms, techniques, libraries
- Follow vulnerability announcements
  - CVE, Bugtraq, CERT, software-specific
- Do audits
- Review often!
- Do penetration testing



Conclusions

- Distributed systems need to be secure
  - Control how resources are shared
  - Allow them to run over public networks
- Cryptography
  - Encryption
  - Hashing
  - Homomorphic encryption
- Secure protocols
  - Kerberos, TLS, …
- Security is hard
  - Keep up-to-date with best practices