

Cryptography overview

- Process of hiding information
- Can protect e.g. emails, files, and other sensitive data.

Cryptography terms

- **Cipher:** an algorithm performing encryption and decryption
- **Clear text / plaintext:** unencrypted data
- **Cipher text:** encrypted data
- **Key:** specifies the transformation of data for encryption / decryption

Cipher types

- **Cipher:** algorithm performing encryption and decryption.

Classical ciphers

- Used historically but no longer used for the most part.

Substitution cipher

- Every character is substituted with another one
- E.g. [Caesar Cipher](#) (100 BC)

Plaintext: THE QUICK BROWN FOX JUMPS OVER THE LAZY DOG
Ciphertext: QEB NRFZH YOLTK CLU GRJMP LSBO QEB IXWV ALD
Key: right shift of 3

Polyalphabetic cipher

- Based on substitution
- Uses multiple substitution alphabets
- E.g. [Vigenère cipher](#) from 1467
 - Has cipher has several Caesar ciphers in sequence with different shift values.

Transposition cipher

- Plain text is repositioned (shifted) to create a ciphertext
- Also called a **zigzag cipher**
- E.g. [Rail fence cipher](#) (invented by ancient Greeks)

Clear text: WE ARE DISCOVERED. FLEE AT ONCE

W . . . E . . . C . . . R . . . L . . . T . . . E
. E . R . D . S . O . E . E . F . E . A . O . C .
. . A . . . I . . . V . . . D . . . E . . . N . .

Ciphertext: WECRLTEERDSOEFEFAOCAIVDEN

Modern ciphers

Computational infeasibility

- Modern cryptographic systems are built on problems which are assumed to be **computationally infeasible**
- A computation which although computable would take far too many resources to actually compute.
- Cryptography tries to ensure an infeasible computation's cost is greater than the reward obtained by computing it
- "Secure" because it's "too slow" to achieve by computers.


Key-based ciphers

Symmetric encryption

- One (same) key to encrypt and decrypt
 - Known as **single key** or **shared key**.
- Either uses stream cipher or block cipher
- E.g. AES, DES
- Problems include key distribution and management
- Suitable for large amounts of data
- Harder for groups of people because more keys are needed as group increases
- Doing nothing for non-repudiation, only performs confidentiality

Asymmetric encryption

- Also known as **public key cryptography**
-  Different keys to encrypt and decrypt
 - **Public key** to encrypt
 - Known by everyone
 - Can be issued by Public Key Infrastructure (PKI)
 - **Private key** to decrypt
 - Only known by the owner, secret to the public.
- It's slower than symmetric encryption
 -  **Hybrid encryption**
 - Combining
 - public-key cryptography for ease of secure key exchange


- symmetric-key cryptography for speed
- E.g. SSL/TLS uses asymmetric encryption to create a key that's later used for encrypting/decrypting packets.
-  Used in • [digital certificates](#) • [PKI](#) • [SSH](#) • [PGP](#) • [IPSec](#)
- Algorithms include • [RSA](#) • [DSA](#) • [Diffie-Hellman](#)

Forward secrecy


- Also known as **perfect forward secrecy**
- Property of cryptographic systems ensuring future disclosure of encryption keys cannot be used to decrypt communications in past.
- Ensures that a session key derived from a set of public and private keys will not be compromised if one of the private keys is compromised in the future
- E.g. web traffic cannot be decrypted with only server's private key through a court order.

Input-based ciphers


Block cipher

-  Fixed-size blocks of data using a symmetric key
- Data bits are split up into blocks and fed into the cipher
- Each block of data (usually 64 bits) is encrypted with key and algorithm
- Simpler and slower than stream ciphers
- Key chosen for cipher must have a length larger than the data, if not, it is vulnerable to frequency attacks


Stream cipher

-  One bit at a time using a symmetric key
- Combines each bit with a pseudorandom cipher digit stream (keystream)
- Works at a high rate of speed
- Usually done by an combining XOR with random generated key.

XOR cipher


- Also known as • **exclusive OR** • **modulus 2** (addition/subtraction).
-  If the inputs match, the output is a 0; otherwise, it is a 1.

A	B	Output
0	0	0
0	1	1
1	0	1
1	1	0

-  If the key chosen is actually smaller than the data, the cipher will be vulnerable to frequency attacks
- Uses " \oplus " that denotes the exclusive disjunction (XOR) operation

- Applying XOR operator at every character encrypts, re-applying it decrypts.

Encryption algorithms and techniques

- **Algorithm:** step-by-step method of solving a problem
- **Encryption algorithms:** mathematical formulas used to encrypt and decrypt data
- Keys should still change on a regular basis even though they may be "unhackable"
-  In terms of speed
 - Symmetric encryption algorithms are faster than asymmetric algorithms
 - Stream ciphers (including AES in CTR) are [usually faster](#) than [hash functions](#)

Encryption types per OSI layer

OSI layer	Encryption type	Description
2	Link encryption	Everything including original headers is encrypted
3	Network encryption	Everything in the packet is encrypted
4	Protocol encryption	Specific protocols are entirely encrypted e.g. SSL
5	Service based encryption	Encryption for specific services on specific hosts
6	Data encryption	
7	Application based information encryption	

Symmetric algorithms

- Both ends of the transmission must use the same key.
 - Requires to find a secondary secure channel to send the symmetric key to the recipient to ensure security.

DES (Data Encryption Standard)

- Block cipher, 56-bit key, 64-bit block size
- Developed in the early 1970s at IBM
- Was a standard set by NSA but was withdrawn, quickly outdated
- Used in initial implementation of Kerberos but later deprecated
- Considered insecure, has been superseded by the [Advanced Encryption Standard \(AES\)](#)


3DES (Triple Data Encryption Standard)

- Block cipher, 168-bit key
- More effective than DES but much slower
- 3 keys are used:
 1. Used to encrypt the plain text
 2. Used to decrypt ciphertext resulting from the first round of encryption
 3. Used to encrypt the ciphertext that resulted from the decryption with the 2nd key

AES (Advanced Encryption Standard)

- Iterated block cipher, 128, 192 or 256 bit key, 128-bit block size
- Original name is **Rijndael**, named to AES after Advanced Encryption Standard contest.
- Symmetric-key algorithm
- Performs same operation multiple size
- Approved and used by NSA
- Much faster than DES and 3DES for bulk data encryption.

RC (Rivest Cipher)

-  Symmetric-key algorithm.
- Lost to AES in Advanced Encryption Standard contest.

RC4

- Variable length key algorithm
- Uses random permutations
- Stream cipher

RC5

- Parameterized algorithm
- Block cipher (slower)
- Variable
 - block size: can be 32, 64, or 128 bits
 - key size: between 0 and 2048 bits
 - number of rounds: between 0 and 255

RC6

- Extends RC5 with two features
 1. integer multiplication
 2. 4-bit registers (RC5 uses 2-bit registers)

Blowfish




- Fast symmetric block cipher, 64-bit block size, 32 to 448 bits key
- Considered public domain
- Insecure

Twofish



- Symmetric-key block cipher
- Uses 128-bit blocks
- Key size can range between 0 and 256 bits.
- Lost to AES as finalist in Advanced Encryption Standard contest.

Asymmetric algorithms

RSA (Rivest-Shamir-Adleman)

- One of the common used encryption standards for data transmission.
-  Achieving strong encryption through the use of two large prime numbers
- **Solves**
 - I want anybody to be able to encrypt a message, but I'm the only one who can decrypt it. I don't want to share decryption keys with anybody.
- There are no published methods to defeat the system if a large enough key is used.
- Wide usage, e.g. SSH for public key authentication
 - A lot safer than password authentication (should be disabled after copying keys)
 - Steps
 1. `ssh-gen -t rsa` creates private and public keys in `.ssh` folder
 2. Copy the key to the server to be connected
 - Can run e.g. `ssh-copy-id user@192.168.122.37` with ssh password
 - Or just copy `~/.ssh/id_rsa.pub` to the server manually.
-  In RSA crypto, when you generate a key pair, it's completely arbitrary which one you choose to be the public key, and which is the private key.
 - If you encrypt with one, you can decrypt with the other - it works in both directions.
 - When encrypting, private key is used to decrypt and public key to decrypt
 - "I want to be able to encrypt certain information and use it as a product key for my software."
 - When signing, private key is used to encrypt and public key to decrypt
 - "I want to use my private key to generate messages so only I can possibly be the sender."
- Recommended key size is 2048 bits or higher.
-  RSA Factoring Challenge
 - Held between 1991 and 2007 with prize awards
 - Goal was to factoring prime numbers used by RSA
 - Shows cryptanalytic strength of RSA


Diffie-Hellman

- Also known as **Diffie Hellman** or **DH**
- Used for generating a shared key between two entities over an insecure channel.
- One of the first public-key protocols
-  Used in • [SSL](#) • [IKE \(Internet Key Exchange\)](#) | [IPsec](#)
-  If digital signatures are waived, vulnerable to MITM attacks
- Enables two parties to encrypt their traffic with a shared key.
- **Solves:** We have a symmetric encryption scheme and want to communicate. We don't want anybody else to have the key, so we can't say it out loud (or over a wire).

Diffie-Hellman groups

- Determines the strength of the key used in the Diffie-Hellman key exchange process.
- Higher Diffie-Hellman Group numbers are more secure, but requires more computation power.
- • Group 1 (768-bit) • Group 2 (1024-bit) • Group 5 (1536-bit) • Group 14 (2048-bit) • Group 15 (3072-bit) • Group 19 (256-bit elliptic curve) • Group 20 (384-bit elliptic curve) • Group 21 (521-bit elliptic curve) • Group 24 (2048-bit, 256 bit subgroup)

Elliptic-curve Diffie-Hellman

- Also known as • **ECDH** • **Elliptic curve Diffie Hellman**
- Variant of the Diffie-Hellman protocol using [elliptic-curve cryptography](#).
-  Used in • [SSH](#) • [TLS 1.3](#)

ECC (Elliptic-curve cryptography)

- Also known as **elliptic curve cryptosystem**
- Using points on elliptical curve along with logarithmic problems
- Using less processing power, smaller keys, good for mobile devices


DSA (Digital Signature Algorithm)

- Asymmetric algorithm
 - Private key tells who signed the message
 - Public key verifies the digital signature
- In the message exchange, each entity creates a public and private key.
- Per [FIPS 186-5](#), it's not recommended for signature generation but approved for signature verification.


Symmetric vs asymmetric algorithms

Differentiator	Symmetric Key Encryption	Asymmetric Key Encryption
Key	Only one key (symmetric key) is used to encrypt and decrypt	Two different cryptographic keys (asymmetric keys), called the public and the private keys, are used to encrypt and decrypt
Complexity and Speed of Execution	Simple technique, fast encryption	More complicated process, slower
Length of key	Smaller, usually 128, 256 and 512 bits	Much larger, e.g. recommended RSA key size is 2048 bits or higher
Common usage	When large chunks of data need to be transferred.	Smaller transactions, primarily to authenticate and establish a secure communication channel prior to the actual data transfer.
Security	The secret key is shared. Consequently, the risk of compromise is higher.	The private key is not shared, and the overall process is more secure as compared to symmetric encryption.

Hashing algorithms


- Also known as **one-way functions**, **hash functions** or **message-digest-functions**
- Calculates unique fixed-size representation of a block of information.
- Cannot be reversed.
-  Used for
 - integrity e.g. when downloading a file in internet, you can compare downloaded files hash with hash given on the website to ensure the right file is downloaded
 - storing passwords in a database by e.g. operating systems
- **Checksum**
 - Number created by a message digest.

Collision resistance



- Property of a hash function
-  Computationally infeasible to find two colliding inputs
- See also Collision attack | Cryptanalysis

Hash functions

MD5

-  Most popular message digest algorithm.
- Takes any length of input and produces a 128-bit hash
- Considered very insecure as it is easy to reverse with current processing power
- Still used commonly in integrity checking e.g. file download verification

SHA

- Secure Hashing Algorithms
-  Generates a cryptographically secure message digest.
- Published by NIST (National Institute of Standards and Technology)
- Generations
 - **SHA-0**
 - Withdrawn shortly after publication due to a flaw and replaced with revised SHA-1
 -  **SHA-1**
 - Produces 160-bit digests
 - Designated by NSA
 - **SHA-2**
 - Primarily SHA-256 (32-bit block words), SHA-512 (64-bit block words)
 - Truncated versions: SHA-224, SHA-384, SHA-512/224 and SHA-512/256
 - Designated by NSA
 - **SHA-3**

- Chosen after competition by non-NSA designers
- Supports same hash lengths as SHA-2
- Internal structure differs significantly from the rest of the SHA family.

RIPEMD (RACE Integrity Primitives Evaluation Message Digest)

- 160-bit hash algorithm
- Working through 80 stages made up of 6 blocks that executes 16 times each
- Using modulo 32 addition

HMAC


- Expands either as
 - Keyed-hash message authentication code
 - Hash-based message authentication code
- Uses a combination of a cryptographic key and hash function such as SHA-1 or MD5.
- Used for authentication and integrity checks.
- E.g. `HMAC_SHA256("key", "The quick brown fox jumps over the lazy dog") = f7bc83f430538424b13298e6aa6fb143ef4d59a14946175997479dbc2d1a3cd8`
- Uses **keyed hashing** to generate hashed-based MACs (HMAC).
 - Involves hashing a message with a • hash function and • a secret key.
 - Message authentication codes (MACs)
 - Cryptographic checksums
 - Used to detect when an attacker has tampered with a message
 - Keyed hashing vs [salted hashes](#)
 - Keyed hashing is against tampering, hash salting is against brute-force attacks.
 - Salts are not assumed to be secret but keys are.
- HMAC can provide digital signatures without PKI infrastructure
 - Delegates the key exchange to the communicating parties
 - Parties are responsible for establishing and using a trusted channel to agree on the key prior to communication

Hash function attacks

- [Collision](#)
- [Birthday attack](#)
- [Rainbow tables](#)
- [Brute-force attack](#)

Hash function attack countermeasures

Salted hash

-  Hash is used with salt (collection of random bits) to obscure the hash.
- Goal is to increase protection against dictionary and brute-force attacks.
- Usually the salt is stored along with the hash of e.g. password.
- See also [Password cracking countermeasures](#)

Key stretching

- Converting a key (e.g. password) to a longer and more random key to e.g. use as encryption.
- Makes encryption stronger as it increases the time and resources for brute-force attacks.
- Usually done by re-hashing multiple (e.g. a few million) times
- E.g. using slow key derivation functions such as PBKDF2

Encrypting communication

- Can happen in different layers on Internet protocol suite e.g.
 - 1. Link layer: • [PPP]
 - 2. Internet layer: • [IPSec](#)
 - 3. Transport layer: • [TCP](#)
 - 4. Application layer: • [SSH](#) • [FTP](#), [SFTP](#), [FTPS](#) • [SSL / TLS](#)

SSL/TLS

SSL (Secure Sockets Layer)

- Protocol on the application layer
- Created to ensure the security of the message transmission over the network and Internet.
 - ⚠ Not secure and has known vulnerabilities, use > [TLS](#) 1.2 instead.
- 📝 Uses both asymmetric and symmetric authentication mechanisms
 - Asymmetric encryption is used to establish a secure session between a client and a server
 - Utilizing [RSA](#) and [digital certificates](#)
 - Symmetric encryption is used to exchange data within the secured session
- Encrypts data at Transport Layer of [TCP/IP stack](#) and above

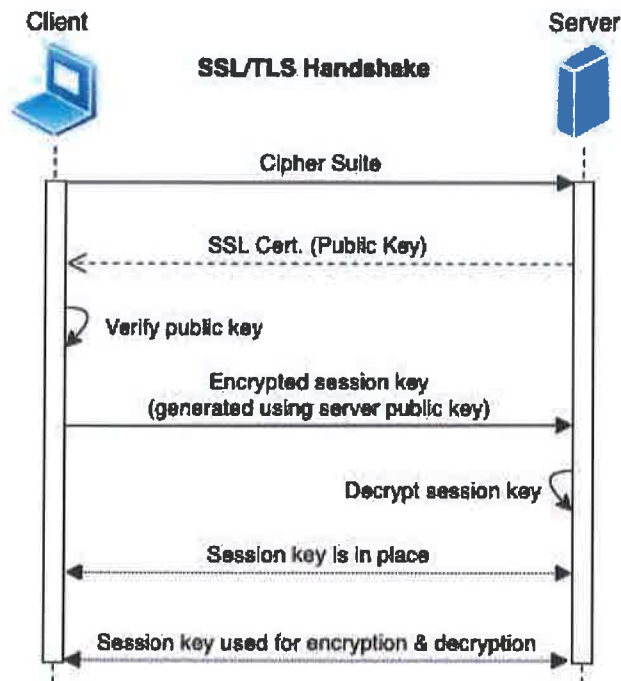
TLS (Transport Layer Security)

- 📝 More secure successor of [SSL](#)
- Protocol to establish secure client-server connection.
- Ensures the information integrity and privacy during transmission.
- Uses RSA 1024 and 2048 bits
- ⚠ Use at least TLS 1.2, lower version has vulnerabilities.
- Most commonly used implementation of SSL/TLS is OpenSSL library.
- Vulnerabilities include • [Heartbleed \(OpenSSL\)](#) • [POODLE](#) • [CRIME](#) and [BREACH](#) • [Renegotiation attack](#) • [FREAK](#) • [DROWN](#)

Two layers of TLS

- **TLS Record Protocol**
 - Provides secured communication channel
 - Private connection: using symmetric cryptography for data encryption
 - Reliable connection: providing message integrity check
- **TLS Handshake Protocol**
 - Providing connection security that has three basic properties:
 1. The peer's identity can be authenticated using asymmetric cryptography
 2. The negotiation of a shared secret is secure
 3. The negotiation is reliable
 - TLS Handshake Protocol operates on top of TLS record layer

SSL/TLS Handshake



Securing FTP

Insecure FTP options

FTP (File Transfer Protocol)

- Used for the transfer of computer files between a client and server on a computer network.
- Runs on Application layer (4) in internet protocol suite
- Insecure as it has minimal security through user ID and passwords.
- Runs on port 21

TFTP (Trivial File Transfer Protocol)

- Less secure than FTP using no authentication at all
- Runs on port 69



Secure options

- **Encrypting files before using FTP**
 - Does not protect user password, or prevent man in the middle from downloading encrypted files
- **SFTP (SSH File Transfer Program) | port: 22**
 - Uses [SSH](#) for authentication and data transport mechanism
- **FTP over SSH**
 - SSH connection must be established between the two systems before the FTP connection is established
- **FTP over [IPSec](#)**
 - IPSec runs on lower layer Internet layer (3) on [IP suite](#).
 - Can cause high amount of CPU cycles on interrupt or kernel level.

- **FTP over VPN**
- **FTP over TLS**
 - Also known as • **FTPS** • **FTP-SSL** (old name) • **FTP Secure**
- **Comparison**

Method	Password Privacy	Server Verification	Data Privacy	Data Integrity	Test and Binary Transfers	Automatic
Separate encryption	No	No	Yes	Maybe	Maybe	No, file must be encrypted and decrypted separated from the transfer
SFTP	Yes	Yes, private key	Yes	Yes	Maybe	Yes
FTP over SSH	Yes	Yes, private key	Yes	Yes	Yes	No, separate setup of the tunnel is needed
FTP over IPSec	Yes	Yes, private key or certificate authority	Yes	Yes	Yes	Yes, but configuration can be a challenge
FTP over VPN	Yes	Maybe	Yes	Yes	Yes	Maybe, but may have configuration challenges
FTPS	Yes	Yes, private key or certificate authority	Yes	Yes	Yes	No, but it's easy to use

Digital signatures

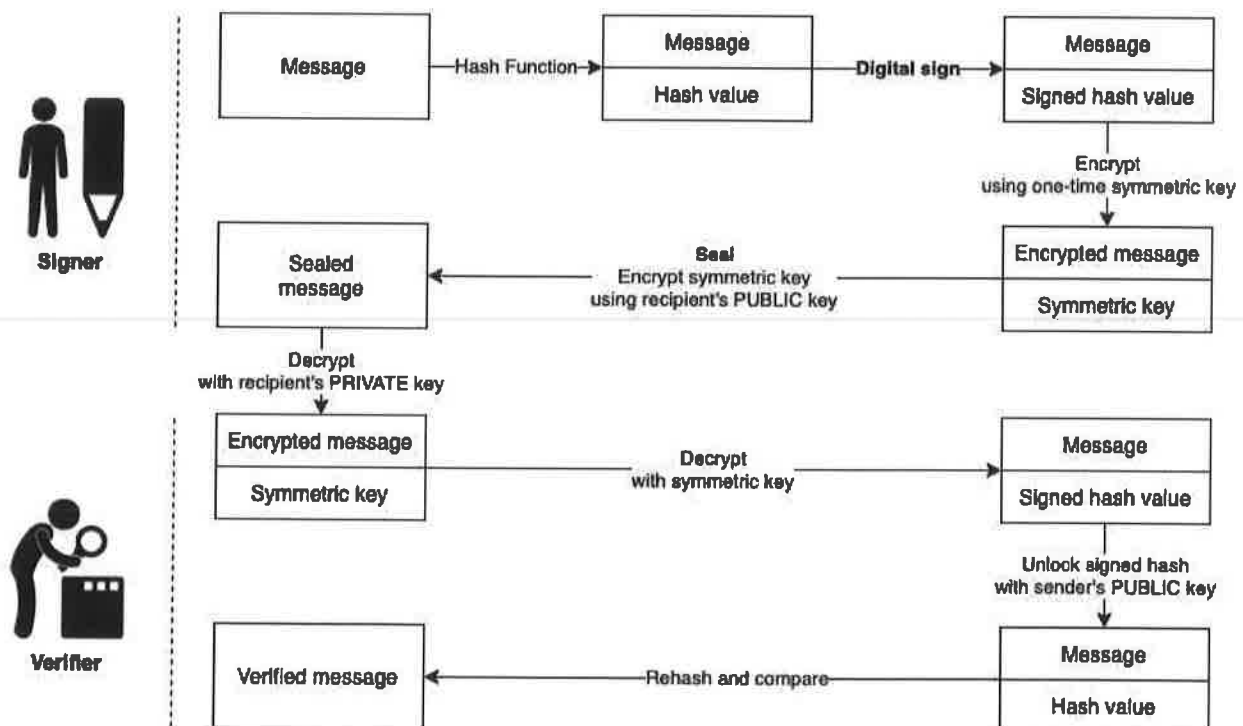
- Created using asymmetric cryptography
- Attached to the transmitted data
-  Usually used in cases where it is important to detect forgery or tampering
 - E.g. • software distribution • financial transactions • contract management software
-  Provides cryptographic way of
 - **Authentication**
 - Verifies identity of the source messages
 - Through a secret-key that's bound to a specific user
 - **Integrity**
 - Ensures the communication is not altered during transmission
 - Avoids forging as any change would invalidate the signature
 - **Non-repudiation** of origin
 - The sender cannot falsely claim that they have not signed the message

- ⚠ Risks
 - Revoked secret key (key pair) prior to its usage
 - Leaked secret keys that would continue to implicate the claimed owner of the key-pair
- 📝 Simplified flow
 1. Sender hashes the file and encrypts using **sender's private key**
 - Provides non-repudiation since only the sender has access to their private key.
 - However there's no confidentiality as it can be unencrypted by anyone with access to the sender's public key
 - The hashing of the email is what creates the integrity.
 2. Recipient decrypts the message with its **sender's public key**
- 🔒 Digital Signature Algorithm (DSA) is the standard used in generation and verification


Encryption vs Signing

- When encrypting:
 - You use **their public key** to write a message
 - They use **their private key** to read it
 - I want my public key to be used to read the messages and I do not care who reads them
- When signing:
 - You use **your private key** to write message's signature
 - They use **your public key** to check if it's really yours.
 - I only care that I am the only one who can generate these.

Digital signature flow






Public Key Infrastructure (PKI)

- Security architecture/structure for managing digital certificates
-  Goal is secure electronic transfer of information for a range of network activities
- Provides confidentiality, integrity, access control, authentication, and non-repudiation of the sender.
- Includes hardware, software, people, policies, and procedures
- Goal is to increase the confidentiality of information that is being exchanged.
- Used widely e.g. by browser, operating systems for system updates...
- **Cross-Certification**
 - Allowing a CA to trust another CS in a completely different PKI
 - Allowing both CAs to validate certificates from either side
- **Single-authority system**
 - CA at the top that creates and issues certificates
- **Hierarchical trust system**
 - CA at the top (root CA)
 - Making use of one or more RAs (subordinate CAs) underneath it to issue and manage certificates

Key escrow

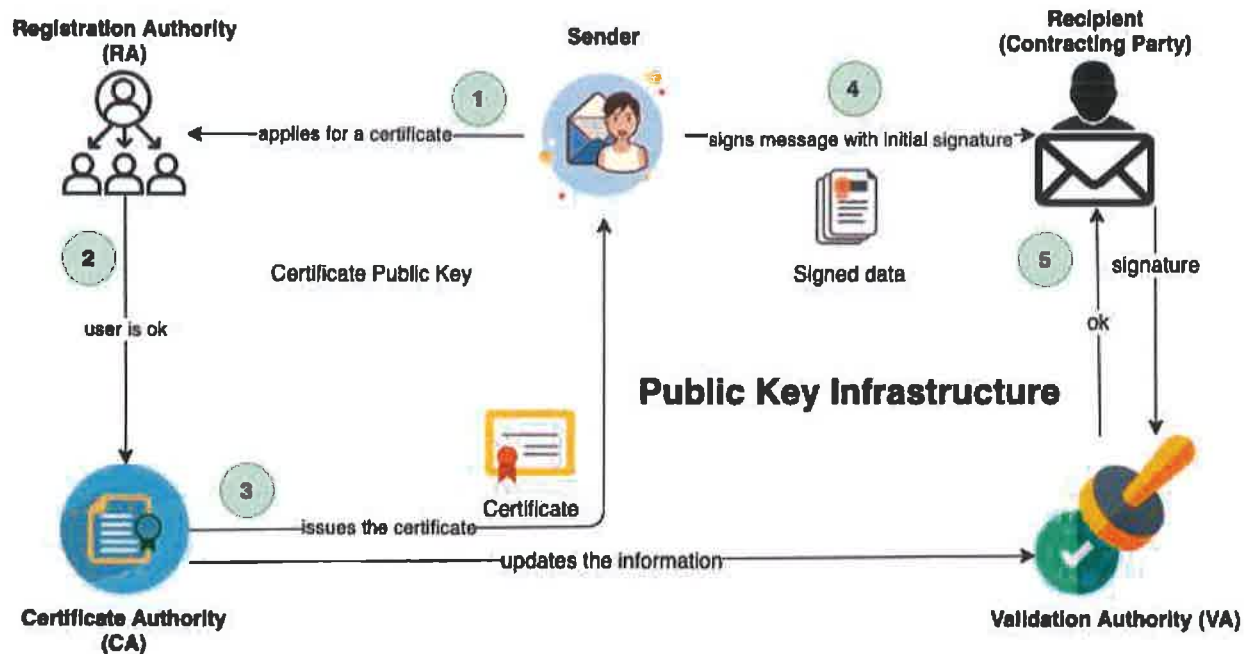
- Also known as **fair cryptosystem**
- Allows a third party to access sensitive data if the need arises.
- A copy of a private key is stored to provide third-party access and to e.g.
 - facilitate recovery operations
 - comply with a court order

Actors of PKI

- **Validation authority (VA)**
 -  Used to validate certificates, stores certificates with their public keys
- **Certificate authority (CA)**
 - Also known as **certification authority**
 -  Third party to issue and verify digital certificates
 - Digital certificates contain public key and the identity of the owner.
 - E.g. Comodo, IdentTrust, GoDaddy.
- **Registration authority (RA)**
 -  Acts as verifier for the certificate authority
- **Certificate Management System**
 - Generates, distributes, stores, and verifies certificates
- **End user**
 - Requests, manages, and uses certificates

PKI process flow

1. A **user** applies for a certificate with his public key at a **Registration Authority (RA)**
2. **Registration Authority (RA)** confirms the **user's** identity to **Certificate Authority (CA)** which in turn issues the certificate and informs **Validation Authority (VA)**
3. The **user** can then digitally sign a contract using his new certificate.
4. His identity is then checked by the contracting party with a **Validation Authority (VA)** which again receives information about issued certificates by the **Certificate Authority (CA)**.



Digital certificate

- Electronic file
- Used to verify a user's identity (= non-repudiation)

X.509

- Standard protocol used for digital certificates, public key encryption
- Defining all sorts of things regarding PKI including digital certificates
- Identifies components of a digital certificate: • version • algorithm ID • copy of the public key • key usage description

Digital certificate vs digital signature

- The certificate ties the digital signature to a data object.
 - Digital signature secures the data in transit.
- Digital certificate uses the public key to identify source of the data.
 - the digital signature uses the public key to verify integrity of the data.
- ⚠ Signatures do not help with revoked keys and stolen keys
 - CA maintains certificate revocation list
- Certificates helps to prove senders authenticity of public keys (e.g. who signed what)
 1. Sender obtains digital certificate by registering his public key to a certificate authority.

2. CA uses its private key to sign sender's certificate and attaches CA public key to his certificate.
3. Receiver decrypts and validates signature using certificate authority's public key.

Certificate types

- **Signed certificate**
 - Issued by Certification Authorities (CA)
 - Contains a public key and the owner's identity.
- **Self-signed certificate**
 - Issued and signed by oneself (not CA)
 - ⚠ Not to be trusted
 - Used for testing/development purposes
 - Otherwise better to use let's encrypt that offers free SSL/TLS certificates

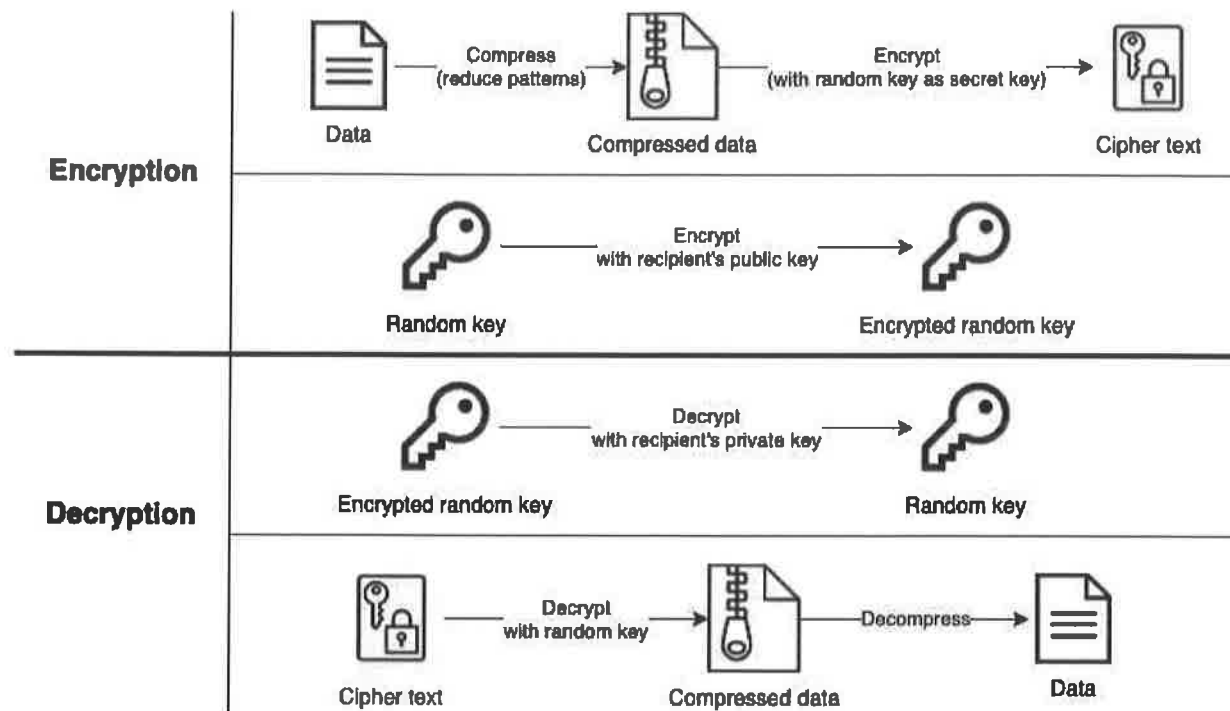
PGP (Pretty Good Privacy)

- Encryption program
- Protocol for
 - secure communication through encryption and decryption
 - authentication (ensures users are who they claim to be)
- Uses a serial combination of
 - hashing
 - data compression
 - symmetric-key cryptography
 - public-key cryptography
- Main uses include
 - Sending and receiving encrypted emails.
 - Verifying the identity of the person who has sent you this message
 - Encrypting files
- Available as both open-source and commercial software

Web of trust

- Used by PGP
- Decentralized model
- Individual users sign keys that belong to other people
- Validates that keys are who they say they are

PGP encryption and decryption workflow



PGP tools

GNU Privacy Guard

- Also known as **GnuPG** or **GPG**
- Open-source and free implementation of PGP
- Allows to encrypt and data and communications
- E.g. create key pair `gpg --gen-key`
- E.g. for symmetric encryption only
 - `gpg -c <file-name>` to encrypt
 - `gpg <file-name>` to decrypt

PGP Alternatives

PEM (Privacy-Enhanced Mail)

- Proposed IETF standard for secure email, obsoleted by PGP.
- Depended on a single root certificate for its public key infrastructure (PKI)
- Was impractical and had its own problematic implications for security.

S/MIME

- Secure/Multipurpose Internet Mail Extensions
- Standard for encrypting and authenticating MIME data
- Used primarily for Internet e-mail.
- PGP is an application, whereas S/MIME is a protocol.
- PGP can be used to encrypt not only e-mail messages but also files and entire drives.

Tunneling protocols

- Allows for the movement of data from one network to another
- Involves repackaging the traffic data into a different form, usually using encryption
 - Can hide the nature of the traffic that is run through a tunnel.



OpenVPN

- Open-source VPN system
- Create secure point-to-point or site-to-site connections in routed or bridged configurations and remote access facilities
- Allows peers to authenticate each other using
 - pre-shared secret keys
 - certificates
 - or username/password



SSH (Secure Shell)

- Cryptographic network protocol for operating network services securely over an unsecured network.
- Usually used for remote command-line, login, and remote command execution.
- Replaces insecure Telnet
- Introduces **SSH file transfer (SFTP)** or **secure copy (SCP)** protocols for secure file access, transfer and management.
- **SSH handshake**
 1. TCP three-way handshake
 2. Protocol/software version announcement
 3. Algorithm negotiation
 4. Key exchange using Elliptic-curve Diffie-Hellman



IPSec

- IPSec is an end-to-end security scheme
-  Part of IPv4 suite so it runs on layer 3 (internet layer) in TCP/IP model or layer 3 (transport) in OSI model
-  Provides security through
 - **Authentication** through authenticating both parts
 - **Integrity** through using a hash algorithm to ensure that data is not tampered with.
 - **Non-repudiation** through using public key digital signatures to prove message origin.
 - **Confidentiality** through encryption

IKE (Internet Key Exchange)

- Also known as **IKEv1** or **IKEv2** depending on the version.
-  Encrypts packets (payloads and headers) between parts
- Uses X.509 for public key and encryption
- Cryptographic settings are established through **Internet Key Exchange crypto profile** (IKE policies)
- Tool: ike-scan
 - Discover, fingerprint and test IPsec VPN servers and firewalls
 - Sends a specially crafted IKE packet to each host within a network
-  Deployed widely to implement e.g.
 - virtual private networks (VPNs)
 - remote user access through dial up connection to private networks


IPSec security architecture

- **Authentication Headers (AH)**
 - Provides connectionless integrity and authentication for the entire IP packet
 - Provides protection against replay attacks
- **Encapsulating Security Payloads (ESP)**
 -  In addition to AH it provides confidentiality through encryption
 - Unlike AH, it does not provide integrity and authentication for entire IP packet
 - The outer header (including any outer IPv4 options or IPv6 extension headers) remains unprotected
 - Supports encryption-only and ( insecure) authentication-only configurations
- **Security Associations (SA)**
 - Provides the bundle of algorithms and data to provide the parameters necessary for AH and/or ESP


IPSec modes of operation

- AH and ESP can be implemented in both modes

Transport mode

- Usually authenticated
- Only payload is optionally encrypted.
- Not compatible with NAT when authenticated.
-  Used within same network

Tunnel mode

- Entire packet is encrypted and authenticated.
- Compatible with NAT
-  Used to create virtual private networks between different networks.



PPP (Point-to-Point Protocol)

- Provide connection authentication, transmission encryption and compression.
- Provides router-to-router or host-to-network connections over asynchronous and synchronous circuits.
- Can be used to create e.g. SSH tunnels
- OSI Layer: 2 - Data link | Internet protocol suite layer: 1 - Link layer

Point-to-Point Tunneling Protocol (PPTP)

- Insecure/obsolete method for implementing virtual private networks
- Uses a TCP control channel and a Generic Routing Encapsulation tunnel to encapsulate PPP packets.

Disk encryption

- Encryption of all data stored on a disk.
- Data-at-rest protection
- Protect the data stored in the disk and ensure its confidentiality
-  Protects against someone who gains physical access to your device
 -  But does not protect from malware or from being attacked by hackers over the internet

Full Disk Encryption (FDE)

- Encrypting every bit of data stored on a disk or a disk volume
- Working similar to text-message encryption and protects data even OS is not active
- Protects system folders, files, and MBR until valid credentials are provided at pre-boot

Disk encryption tools

- [VeraCrypt](#)
- [Symantec Drive Encryption](#)
- [BitLocker Drive Encryption](#)
- [cryptsetup](#)
 - Open-source disk encryption utility tool
 - Supports [LUKS \(Linux Unified Key Setup\)](#), TrueCrypt, VeraCrypt, BitLocker, loop-AES, dm-crypt...
 - Encrypt
 1. `sudo yum install cryptsetup`
 2. Find mapped disk folder to encrypt using `sudo fdisk -l`
 3. `sudo cryptsetup -y -v luksFormat /dev/<mapped-folder>`
 - Decrypt
 - `sudo cryptsetup luksOpen /dev/<mapped-folder> <new-name>`
 - Will map unencrypted device to `/dev/mapper/<new-name>` (check `fdisk -l`)
 - More information about encryption method etc:
 - `sudo cryptsetup status <new-name>`
 - or `sudo cryptsetup luksDump /dev/<mapped-folder>`
 - Reformat device:
 - Clear: `sudo dd if=/dev/zero of=/dev/mapper/<mapped-folder> bs=128`
 - Create file system: `sudo mkfs.ext4 /dev/mapper/<mapped-folder>`
 - Mount: `sudo mount /dev/mapper/<mapped-folder> <mountad-name>/`

Cryptanalysis

- Process of decryption of ciphers and encrypted text
- Identifies vulnerabilities in cryptosystems

Cryptanalytic techniques

Linear cryptanalysis

- Known as **plaintext attack**
- Applicable to block ciphers and stream ciphers.
- Given enough pairs of plaintext and corresponding ciphertext, key can be obtained
- Discovered by By Matsui and Yamagishi in 1992
- Attacker identifies the linear relation between some bits of the plaintext, some bits of the ciphertext and some bits of the unknown key.

Differential cryptanalysis

- Discovered by Israeli researchers Eli Biham and Adi Shamir in the late 1980s.
- Applicable primarily to block ciphers, but also to stream ciphers and cryptographic hash functions.
- Applicable to symmetric key algorithms
- Comparing differences in the inputs to how each one affects the outcome
- Working with chosen plaintext originally, also works with known plaintext and ciphertext

Integral cryptanalysis

- Used on block ciphers
- Discovered by Lars Knudsen in 1997
- Input vs Output comparison same as differential, however, runs multiple computations of the same block size input
- Attacker analyzes outputs of encrypting sets of plaintexts where some of the content is held constant and some of the content is varied through all possibilities


Code-breaking methodologies

- **Frequency analysis**
 - Study of the frequency of letters or groups of letters in a ciphertext
 - E.g. checking cipher chunks against in languages some letters or combination of letters are used more often
 - Can be used to crack a substitution cipher, like rotation cipher ROT13
- **Trickery and deceit**
 - Requires a high level of mathematical and cryptographic skills
 - Using social engineering techniques to trick someone to encrypt and send a known message
- **One-time pad**


- A shared random key that has to be the same length or longer than the cipher text
- Each individual bit or character of plaintext is encrypted by combining it with the corresponding bit or character from the pad using modular addition
- Assuming to be unbreakable
- **Drawback**
 - Key distribution becomes impracticable for large messages as key length is same as as the messages

Cryptography attacks


Chosen-key Attack

-  Attacker knows keys that are used or can choose the secret key.
- May allow breaking the larger system which relies on that cipher


Rubber-hose attack

- Also known as **rubber hose** or **rubberhose** attack.
-  Extraction of cryptographic secrets (e.g. the password to an encrypted file) from a person by coercion or torture
- E.g. beating that person with a rubber hose until they give up the encryption key.


Ciphertext-only attack (COA)

- Also known as **known ciphertext attack**
-  Attacker has only access to cipher texts
- E.g. using frequency analysis to assume plain text
- Early ciphers (using pen-and-paper) were cracked this way
- Modern ciphers have strong protections against it
 - take years to separate statistical departure from random noise

Known-plaintext attack (KPA)



- Also known as **known-plain-text attack**
-  Attacker has access to parts of plaintext and corresponding ciphertext.
- Can be used to reveal secret keys, code books.
- Classical ciphers are typically vulnerable

Meet-in-the-middle attack

- Also known as **meet in the middle attack**.
- Attack over certain block ciphers by decomposing the problem in two halves and proceeds on each part separately
- Reduces the effort to perform a brute-force attack
-  Reason why re-encrypting an ciphertext reduces its security
 - The reason that Triple DES or Double DES is considered weak and are no longer used.

- E.g. transforming an attack that requires $2^{\text{exp}128}$ time into one that takes $2^{\text{exp}64}$ time and $2^{\text{exp}64}$ space
- Type of known-plaintext attack

Chosen-plaintext attack (CPA)

-  Attacker can choose random plaintexts to be encrypted and obtain the corresponding ciphertexts
-  Two forms
 - **Batch chosen-plaintext attack**
 - Cryptanalyst chooses all plaintexts before any of them are encrypted.
 - Not so effective
 - **Adaptive chosen-plaintext attack**
 - Cryptanalyst makes a series of interactive queries
 - Subsequent plaintexts are chosen based on the information from the previous encryptions.

Chosen-ciphertext attack (CCA)

- Also known as **chosen ciphertext attack** or **chosen-cipher-text attack**.
- Attacker gathers information by obtaining the decryptions of chosen ciphertexts.
- Early versions of RSA padding used in the SSL protocol were vulnerable.
- Types
 - **Adaptive chosen-ciphertext (CCA2)**
 - Attacker uses the results from prior decryptions to inform their choices of which ciphertexts to have decrypted
 - **Non-adaptive chosen-ciphertext**
 - Attacker chooses the ciphertexts to have decrypted without seeing any of the resulting plaintexts
 - **Lunchtime attack** or **midnight attack**
 - Attacker can have access to system for only a limited amount of time, can access only few plaintext-ciphertext pairs


Side-channel attacks

- Based on information gained from a computer, rather than weaknesses in the algorithm itself.
- Monitors environmental factors such as power consumption, sound, timing and delay.
- E.g. RSA keys can be revealed by listening to CPU work.

Timing attack

- Execution times are measured to learn more about the system
- Information to find can include e.g. key, CPU used, algorithms, input, implementation details etc.
- A type of side-channel attack


Brute-force attack

- Also known as **brute force**
- Trying every possible combination of characters to break the encryption
-  Requires a lot of time and processing power.
- See also [Brute-force attack](#) | [Cracking passwords](#)

Birthday Attack

- Type of [brute-force](#) attack but faster that focuses on collisions
- Based on collisions where attacker uses own plain texts to match hashes (find collisions)
- Depends on the higher likelihood of collisions found between random attack attempts and a fixed degree of permutations
- Exploits [birthday problem](#) in probability theory
 - E.g. 23 people in room, chance of two having same birthday is not $23 / 365 = \approx 6\%$ but it's 50%. For 75 people it's 99% chance.

Rainbow table attack

-  Rainbow table contains precomputed hashes to try and find out passwords
- Faster than [brute-force](#) however the trade-off is that it takes a lot of storage (sometimes Terabytes) to hold the Rainbow Tables themselves.
- Tools
 - [HashCalc](#)
 - [MD5 Calculator](#)

Dictionary attack

- Attacker creates and uses a dictionary of plaintext and its ciphertext.
- E.g. words in a dictionary
- E.g. previously used passwords, often from lists obtained from past security breaches
 - See also [Dictionary attacks](#) | [Cracking passwords](#)



Related-key attack

- Attacker observes the operation of a cipher under several different keys
- Some relationship connecting the keys is known to attacker while key values are unknown.
- E.g. attacker knows that last 80 bits of the keys are the same


DUHK Attack (Don't Use Hard-Coded Keys)

- Allowing attackers to access keys and read communications in certain VPN implementations
- Based on vulnerability affecting devices using ANSI X9.31 Random Number Generator (RNG) with a hard-coded seed key

Collision attack

- Also known as **hash collision attack**
-  Tries to find two inputs resulting in same hash value, i.e. a hash collision.
 - Find two different messages `m1` and `m2` such that `hash(m1) = hash(m2)`.
- Extended by **chosen-prefix collision attack**
 - Given two different prefixes `p1`, `p2`
 - The attack finds two appendages `m1` and `m2` such that `hash(p1 || m1) = hash(p2 || m2)`
 - where `||` is the concatenation operation.
 - More powerful
-  The larger the hash value size, the less likely there are for collisions to occur and therefore the more [collision resistant](#) the hash algorithm

Cryptography attack tools

- [L0phtcrack](#)
 - Password cracking tool
 - Used mainly against Windows [SAM files](#)
-  [John the Ripper](#)
 - Password cracking tool
 - Can run against hashed/encrypted passwords of OSes, databases etc.
 - See also [John the Ripper | Password cracking tools](#)
- [CrypTool](#)
 - Open-source program for for cryptography and cryptanalysis
 - GUI to experiment with cryptographic procedures and to animate their cascades
- [Cryptobench](#)
 - Encrypt, decrypt, hash using many algorithms
 - Helps in the cryptanalysis process of common cryptographic schemes