IN2120 Information Security
Autumn 2019

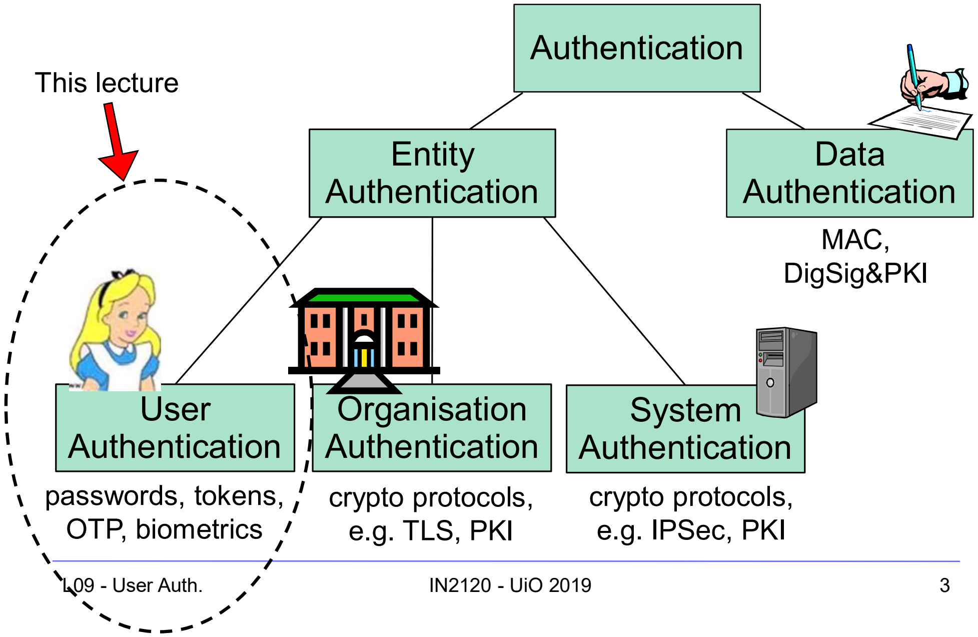
Lecture 9: User Authentication

Nils Gruschka
University of Oslo

Outline

- Context of user authentication
  - Component of IAM (Identity and Access Management)
- User Authentication
  - Knowledge-based authentication
  - Ownership-based authentication
  - Inherence-based authentication
  - Authentication based on secondary channel
- Authentication frameworks for e-Government

Taxonomy of Authentication



Authentication

Entity Authentication

Data Authentication

MAC, DigSig&PKI

This lecture

User Authentication

passwords, tokens, OTP, biometrics

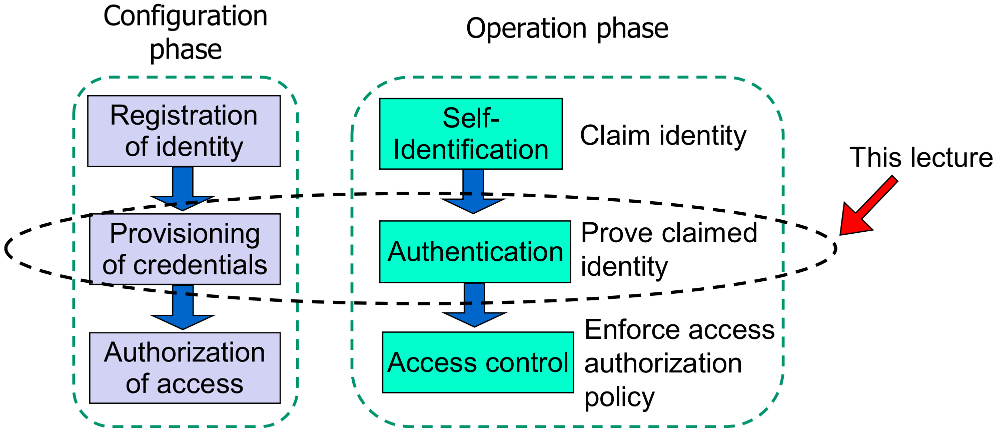
Organisation Authentication

crypto protocols, e.g. TLS, PKI

System Authentication

crypto protocols, e.g. IPSec, PKI

Identity and Access Management (IAM) Phases



Configuration phase

Operation phase

Registration of identity

Provisioning of credentials

Authorization of access

Self-Identification — Claim identity

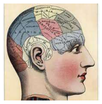
Authentication — Prove claimed identity

Access control — Enforce access authorization policy

This lecture

User authentication credentials

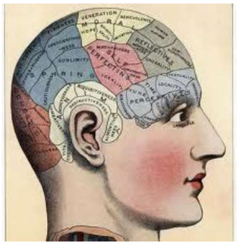
- A credential is the ‘thing’ used for authentication.
- Credential categories (“factors”) and typical examples:
  1. Knowledge-based (“something you know”): Passwords
  2. Ownership-based (“something you have”): Tokens
  3. Inherence-based (“something you are/do”): Biometrics
     - physiological biometric characteristics
     - behavioural biometric characteristics
  4. Secondary channel (a channel you control): SMS, email, etc.



- Combinations, called multi-factor authentication

Knowledge-Based Authentication
“Something you know”

Example: Passwords



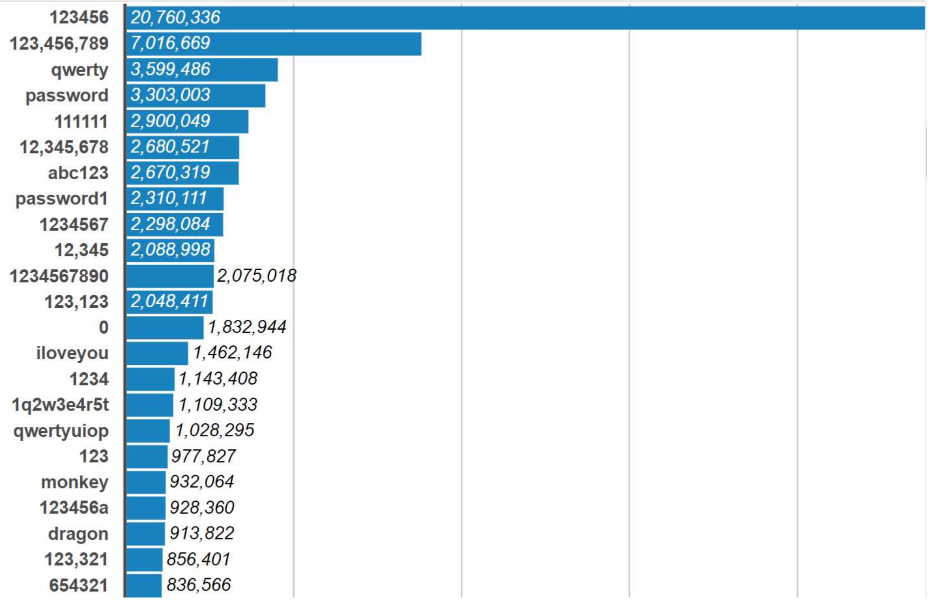
Authentication: Static passwords

123456

- Passwords are a simple and the most common authentication credential.
  - Something the user knows
- Problems:
  - Easy to share (intentionally or not)
  - Easy to forget
  - Often easy to guess (weak passwords)
  - Can be written down (both good and bad)
    - If written down, then “what you know” is “where to find it”
  - Often remains in computer memory and cache

https://haveibeenpwned.com/Passwords

500,000,000 passwords (2018)



Secure password strategies

- Passwords length ≥ 13 characters
- Use ≥ 3 categories of characters
  - L-case, U-case, numbers, special characters
- Do not use ordinary words (names, dictionary wds.)
- Change typically once per year
- OK to reuse between low-sensitivity accounts
- Do **not** reuse between high-sensitivity accounts
- Store passwords securely
  - In brain memory
  - On paper, adequately protected
  - In cleartext on offline digital device, adequately protected
  - Encrypted on online digital device

Strategies for strong passwords

- User education and policies
  - Not necessarily with strict enforcement
- Proactive password checking
  - User selects a potential password which is tested
  - Weak passwords are not accepted
- Reactive password checking
  - SysAdmin periodically runs password cracking tool (also used by attackers) to detect weak passwords that must be replaced.
- Computer-generated passwords
  - Random passwords are strong but difficult to remember
  - FIPS PUB 181 http://www.itl.nist.gov/fipspubs/fip181.htm specifies automated pronounceable password generator

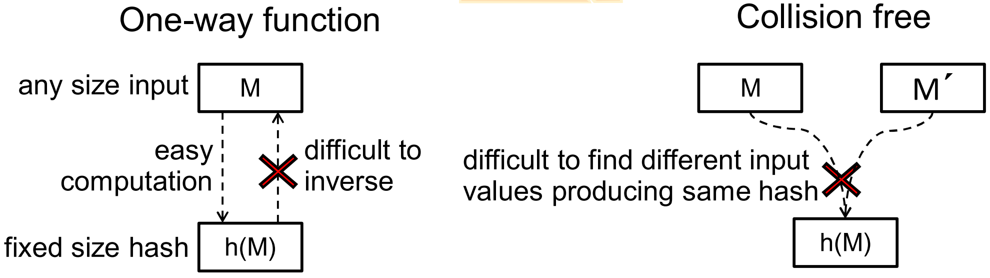
Password storage in OS

- /etc/shadow is the file where modern Linux/Unix stores it passwords
  - Earlier version stored it in /etc/passwd
  - Need root access to modify it
- \windows\system32\config\sam is the file Windows system normally stores its passwords
  - Undocumented binary format
  - Need to be Administrator to access it
- Network environments store passwords centrally
  - AD (Active Directory) on Windows servers
  - LDAP (Lightweight Directory Access Protocol) on Linux

Protection of password file

- Systems need to verify user passwords against stored values in the password file
  - Hence, the password file must be available to the OS
  - But this file needs protection from users and applications
- Protection measures for password file
  - Access control (only accessible by Root/Admin)
  - Hashing or encryption (passwords not stored in cleartext)
- In case a password file gets stolen, then hashing/encryption provides a level of protection
  - It happens quite frequently that password files get stolen and also leaked to the Internet

Hash functions


One-way function

any size input
M
easy computation
difficult to inverse
fixed size hash
h(M)

Collision free

M
M´
difficult to find different input values producing same hash
h(M)

- *A hash function is easy to compute but hard to invert.*

- Passwords are typically stored as hash values.
- Authentication function first computes hash of received password, then compares against the stored hash value

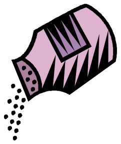
Cracking hashed passwords

- The attacker hashes a possible password and checks if the hash value is found in the password file.
  - The password has been cracked if the hash value is found
- Brute-force search
  - Hash and check all possible passwords (a powerful GPU computer can test passwords up to 8 characters in 1 day)
- Intelligent search
  - User names
  - Names of friends/relatives
  - Phone numbers
  - Birth dates
  - Dictionary attack
    - Try all words from a dictionary

Cracking with hash and rainbow tables

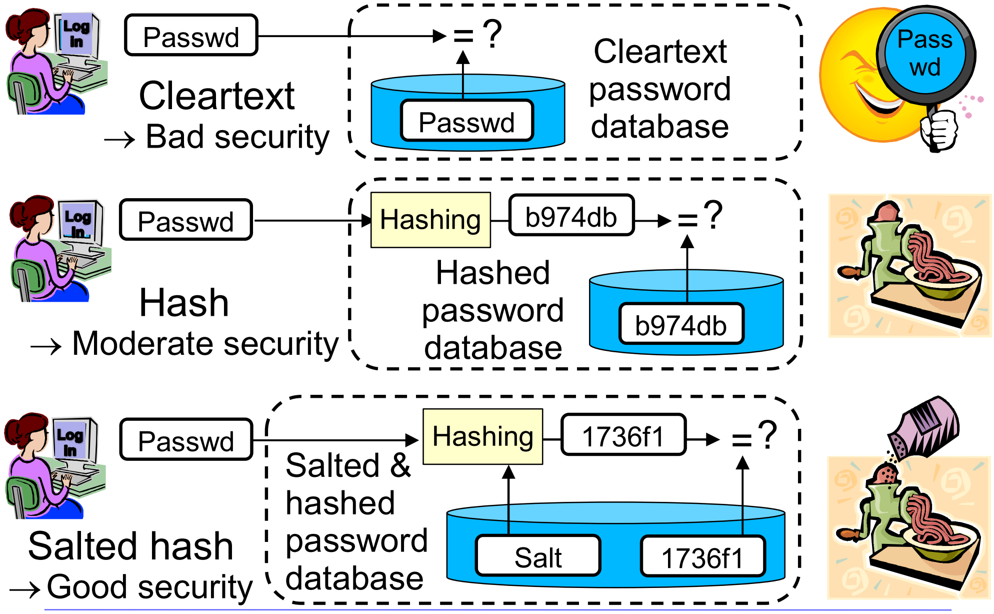
- Attackers can compute and store hash values for all possible passwords up to a certain length
- A list of password hashes is a **hash table**
- A compressed hash table is a **rainbow table**
- Comparing and finding matches between hashed passwords and hash/rainbow table is the method to determine cleartext passwords.

Password salting: Prevents cracking with hash-tables



- Prepend or append random data (salt) to a user's password before hashing
  - In Unix: a randomly chosen integer from 0 to 4095.
  - Different salt for each user
  - Produces different hashes for equal passwords
  - Prevents that users with identical passwords get the same password hash-value
  - Increases the amount of work for hash precomputation
  - Makes it necessary to compute new table for each user
  - Makes hash tables and rainbow tables impractical for password cracking

Storing and checking passwords



Brute Force Attacks

- Effort of brute force attacks depends on:
  - length + complexity of passwords
    - Example: duration of brute force search for NTLM hashes

length	letters (52 keys)	letters + symbols (84 keys)
4	not measurable	0.3 ms
5	3.7 ms	47 ms
6	0.2 s	3.4 s
7	10 s	4.8 min
8	8.75 min	6.7 h
9	7.6 h	23.2 d
10	16.4 d	5.4 y
11	2.4 y	454 y
12	122 y	38,147y

Source: Dirk Fox: Mindestlängen von Passwörtern und kryptographischen Schlüsseln, DuD, 2009

Brute Force Attacks

- Effort of brute force attacks depends on:
  - length + complexity of passwords
  - complexity of hash algorithm
- Hash algorithms are optimized for runtime and memory consumption
- Simple key stretching schemes:

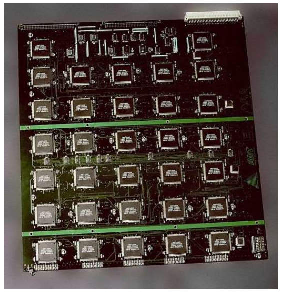
```
key = hash(password)
for 1 to 65536 do
    key = hash(key)
```

```
key = ""
for 1 to 65536 do
    key = hash(key + password)
```

```
key = ""
for 1 to 65536 do
    key = hash(key + password + salt)
```

Source: http://en.wikipedia.org/wiki/Key_stretching

Brute Force Attacks



- Special hashing algorithms:
  - PBKDF2
    - large runtime
    - Applications (Examples): WPA, WPA2, TrueCrypt
    - Problem: can be „reversed“ using special crypto hardware
  - bcrypt
    - additionally: high memory consumption
  - scrypt
    - additionally: very high memory consumption
  - Argon2
    - currently best password hashing function

Image-Source: https://w2.eff.org/Privacy/Crypto/Crypto_misc/DESCracker/

Brute Force Attacks

- Comparision of hashing/key derivation functions:

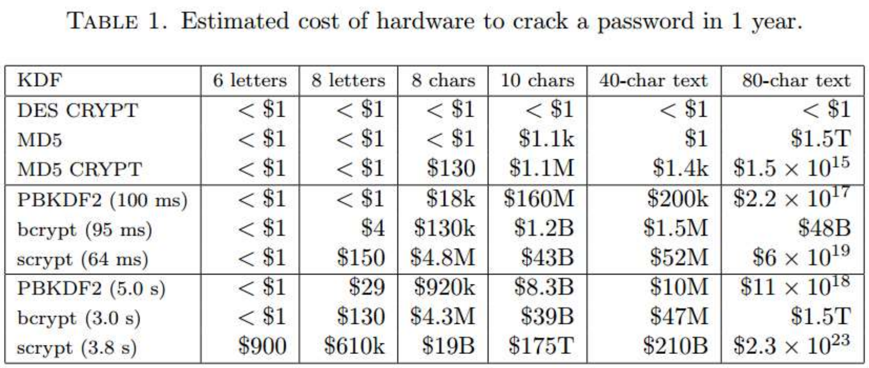


TABLE 1. Estimated cost of hardware to crack a password in 1 year.

KDF	6 letters	8 letters	8 chars	10 chars	40-char text	80-char text
DES CRYPT	$< \$1$	$< \$1$	$< \$1$	$< \$1$	$< \$1$	$< \$1$
MD5	$< \$1$	$< \$1$	$< \$1$	\$1.1k	\$1	\$1.5T
MD5 CRYPT	$< \$1$	$< \$1$	\$130	\$1.1M	\$1.4k	$\$1.5 \times 10^{15}$
PBKDF2 (100 ms)	$< \$1$	$< \$1$	\$18k	\$160M	\$200k	$\$2.2 \times 10^{17}$
bcrypt (95 ms)	$< \$1$	\$4	\$130k	\$1.2B	\$1.5M	\$48B
scrypt (64 ms)	$< \$1$	\$150	\$4.8M	\$43B	\$52M	$\$6 \times 10^{19}$
PBKDF2 (5.0 s)	$< \$1$	\$29	\$920k	\$8.3B	\$10M	$\$11 \times 10^{18}$
bcrypt (3.0 s)	$< \$1$	\$130	\$4.3M	\$39B	\$47M	\$1.5T
scrypt (3.8 s)	\$900	\$610k	\$19B	\$175T	\$210B	$\$2.3 \times 10^{23}$

Source: Colin Percival, Stronger Key Derivation via Sequential Memory-Hard Functions, BSDCan'09, May 2009

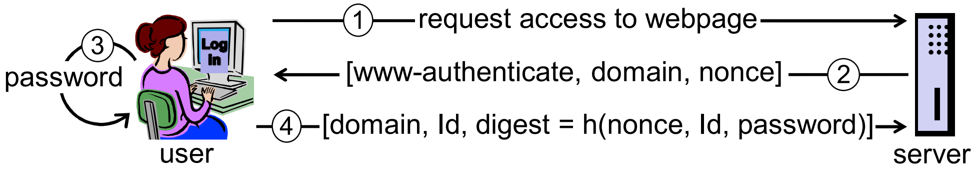
Never send unprotected passwords in clear

- A password sent “in clear” can be captured during transmission, so an attacker may reuse it.
- An attacker setting up a fake server can get the password from the user
  - E.g. phishing attack.
- Solutions to these problems include:
  - Encrypted communication channel
  - One-time passwords (token-based authentication)
  - Challenge-response protocols

HTTP Digest Authentication

A simple challenge-response protocol (rarely used)

- A simple challenge response protocol specified in RFC 2069
- Server sends:
  - WWW-Authenticate = Digest
  - realm=“service domain”
  - nonce=“some random number”
- User types Id and password in browser window
- Browser produces a password digest from nonce, Id and password using a 1-way hash function
- Browser sends Id and digest to server that validates digest



1 request access to webpage

2 [www-authenticate, domain, nonce]

3 password

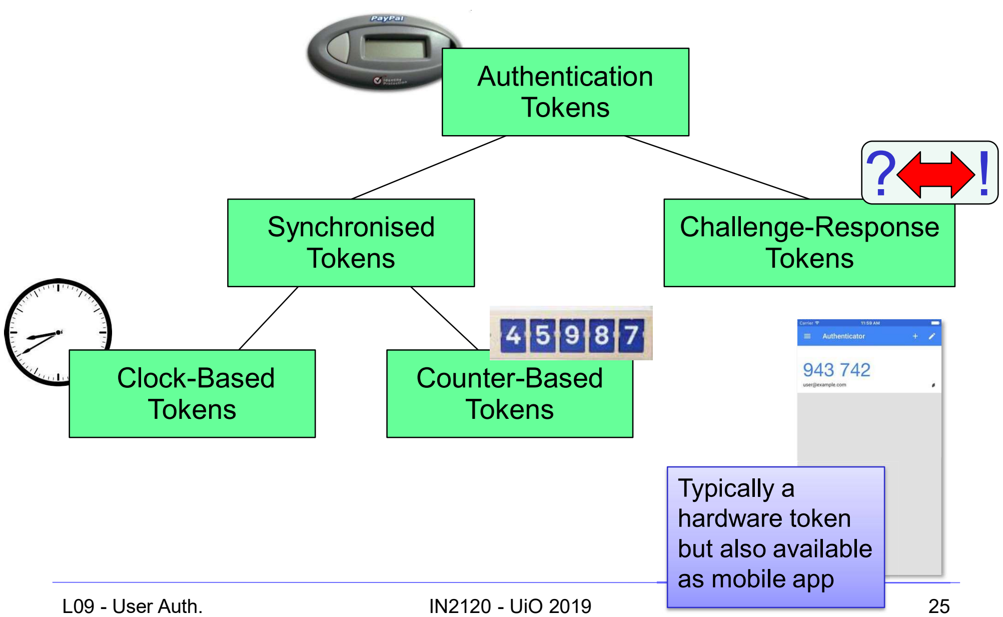
4 [domain, Id, digest = h(nonce, Id, password)]

Ownership-Based Authentication

“Something you have”

Example: Authentication Tokens (OTP)

Taxonomy of Authentication Tokens

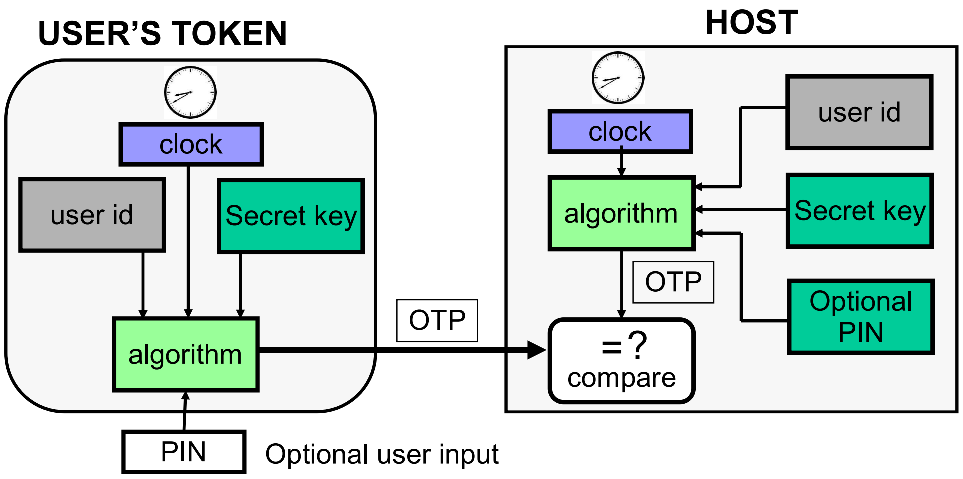


Typically a hardware token but also available as mobile app

Clock-based OTP Tokens: Operation

- Token displays time-dependent code on display
  - User copies code from token to terminal to log in
- Possession of the token is necessary to know the correct value for the current time
- Each code computed for specific time window
- Codes from adjacent time windows are accepted
- Clocks must be synchronised
- Example: BankID and SecurID

Clock-based OTP Token Operation with (optional) input PIN



Clock-based OTP Tokens:

SafeID OTP token with PIN

ActiveID OTP token with PIN

BankID OTP token with PIN

Feitan OTP token witout PIN

RSA SecurID without PIN

BankID OTP token without PIN

Compromised OTP Tokens

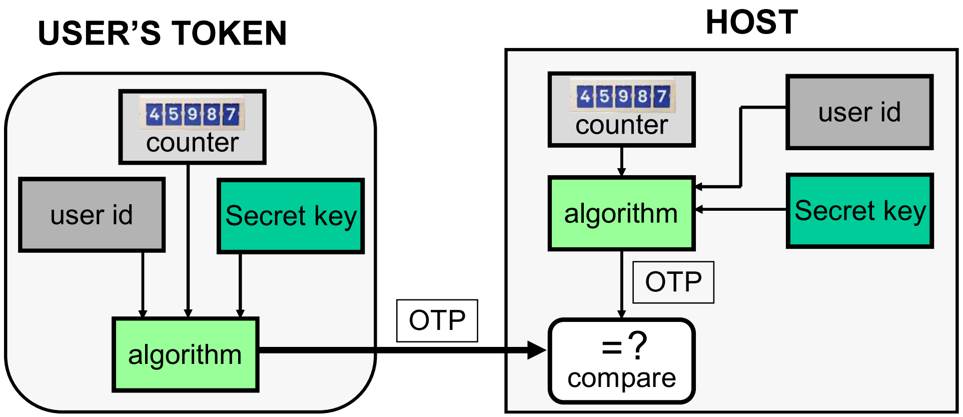
- RSA was hacked in 2007.
- Secret key for OTP tokens stolen
- Hackers could generate OTP and spoof users
- Companies using RSA SecureID were vulnerable
- Lockheed Martin used RSA SecureID
- Chinese attackers spoofed Lockheed Martin staff
  - Stole plans for F-35 fighter jet

Counter-based OTP Tokens: Overview

- Counter-based tokens generate a 'password' result value as a function of an internal counter and other internal data, without external inputs.
- HOTP is a HMAC-Based One-Time Password Algorithm described in RFC 4226 (Dec 2005)
  http://www.rfc-archive.org/getrfc.php?rfc=4226
  - Tokens that do not support any numeric input
  - The value displayed on the token is designed to be easily read and entered by the user.

Diagram

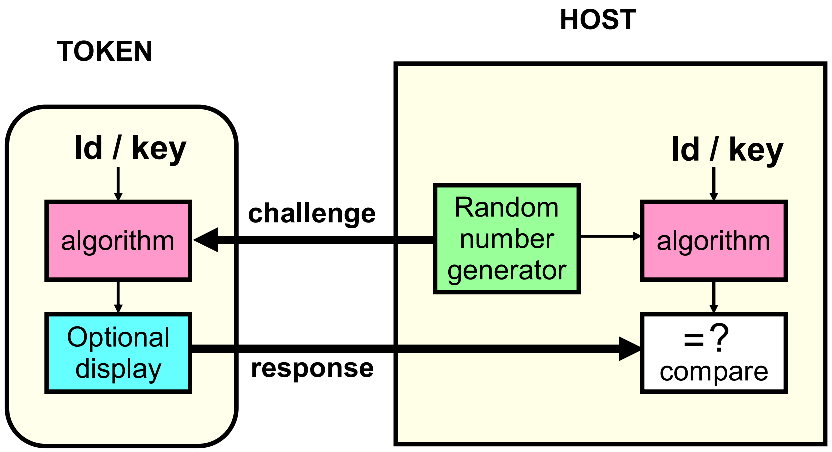
Counter-based OTP Token Operation



Challenge Response Based Tokens for User Authentication:

- A challenge is sent in response to access request
  - A legitimate user can respond to the challenge by performing a task which requires use of information only available to the user (and possibly the host)
- User sends the response to the host
  - Access is approved if response is as expected by host.
- Advantage: Since the challenge will be different each time, the response will be too – the dialogue can not be captured and used at a later time
- Could use symmetric or asymmetric crypto

Token-based User authentication Challenge Response Systems



Inherence-Based Authentication

Biometrics

“Something you are”

“Something you do”

Biometrics: Overview

- What is it?
  - Automated methods of verifying or recognizing a person based upon a physiological characteristics.
- Biometric modalities, examples:
  - fingerprint
  - facial recognition
  - eye retina/iris scanning
  - hand geometry
  - written signature
  - voice print
  - keystroke dynamics

Biometrics: Requirements

- **Universality**:
  Each person should have the characteristic;
- **Distinctiveness**:
  Any two persons should be sufficiently different in terms of the characteristic;
- **Permanence**:
  The characteristic should be sufficiently invariant (with respect to the matching criterion) over a period of time;
- **Collectability**:
  The characteristic should be measurable quantitatively.

Example:
Does “face recognition” fulfill these requirements?

Biometrics: Practical considerations

- Accuracy:
  - The correctness of a biometric system, expressed as ERR (Equal Error Rate), where a low ERR is desirable.
- Performance:
  - the achievable speed of analysis,
  - the resources required to achieve the desired speed,
- Acceptability:
  - the extent to which people are willing to accept the use of a biometric identifier (characteristic)
- Circumvention resistance:
  - The difficulty of fooling the biometric system
- Safety:
  - Whether the biometric system is safe to use

Biometrics Safety

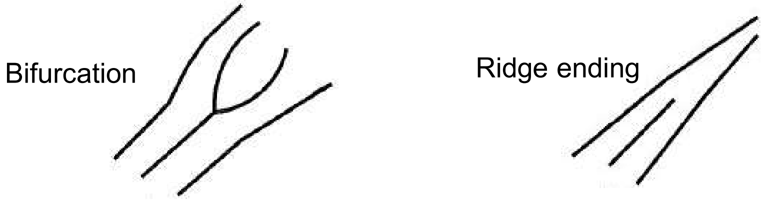
- Biometric authentication can be safety risk
  - Attackers might want to “steal” body parts
  - Subjects can be put under duress to produce biometric authenticator
- Necessary to consider the physical environment where biometric authentication takes place.

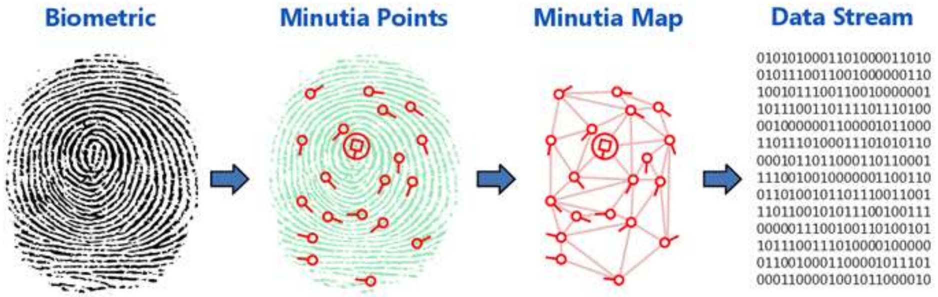
Car thieves chopped off part of the driver’s left index finger to start S-Class Mercedes Benz equipped with fingerprint key. Malaysia, March 2005
(NST picture by Mohd Said Samad)

Biometrics: Modes of operation

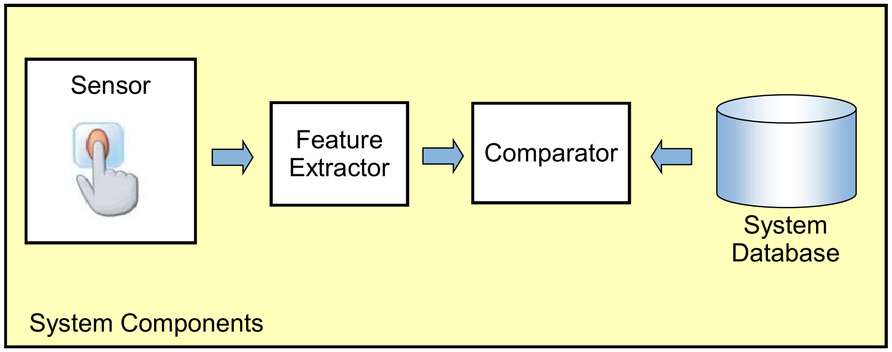
- Enrolment:
  - analog capture of the user’s biometric attribute.
  - processing of this captured data to develop a template of the user’s attribute which is stored for later use.
- Verification of claimed identity (1:1, one-to-one):
  - capture of a new biometric sample.
  - comparison of the new sample with that of the user’s stored template.
- Identification (1:N, one-to-many)
  - capture of a new biometric sample.
  - search the database of stored templates for a match based solely on the biometric.

Extracting biometric features
Example fingerprints: Extracting minutia

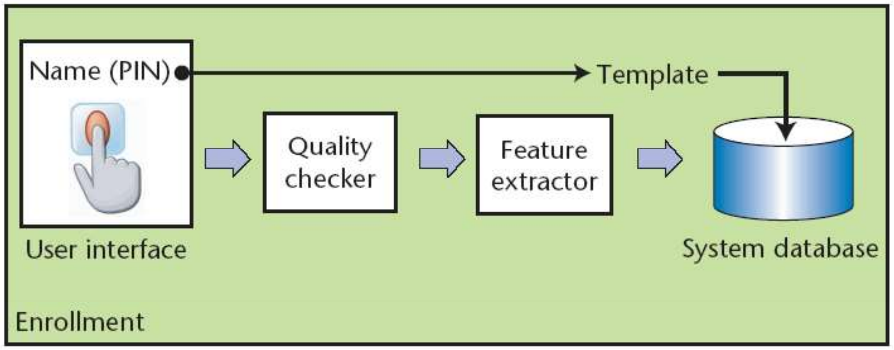




Biometrics: System components

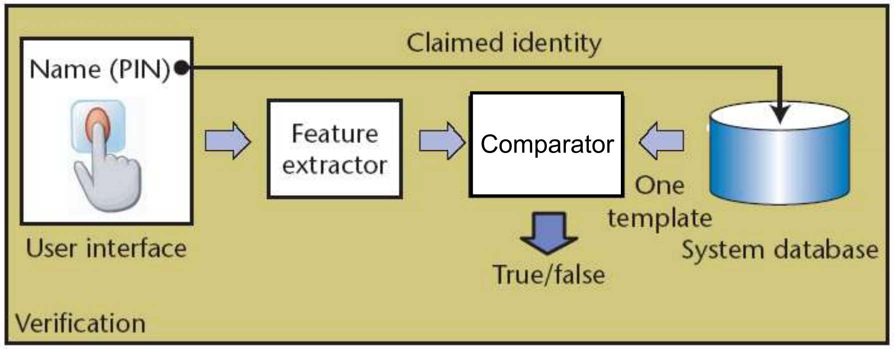


Biometrics Enrolment Phase



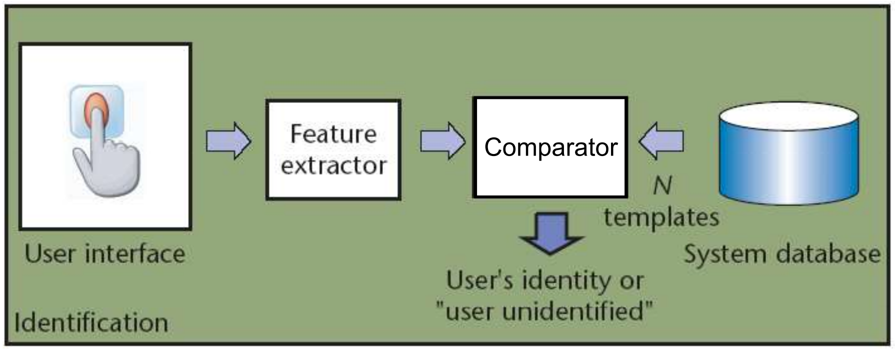
Biometric Recognition: Security and Privacy Concerns

Biometric Verification / Authentication



Biometric Recognition: Security and Privacy Concerns

Biometric Identification



Biometric Recognition: Security and Privacy Concerns

Evaluating Biometrics:

- Features from captured sample are compared against those of the stored template sample
- Score ***s*** is derived from the comparison.
  - Better match leads to higher score.
- The system decision is tuned by threshold ***T***:
  - System gives a match (same person) when the sample comparison generates a score ***s*** where **s ≥*****T***
  - System gives non-match (different person) when the sample comparison generates a score ***s*** where ***s*** **<** ***T***

Comparison characteristics

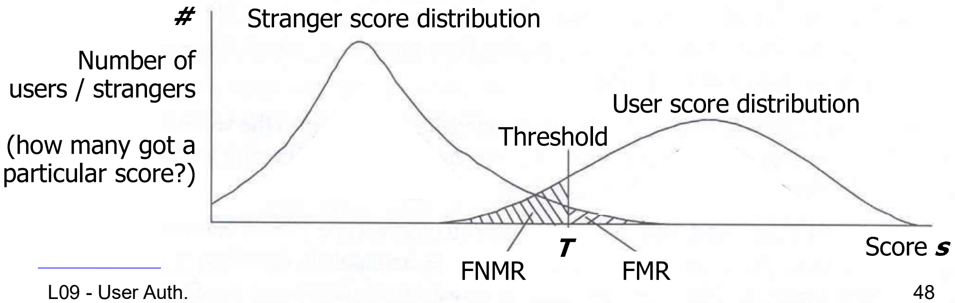
- True positive
  - User's sample matches → User is accepted
- True negative
  - Stranger's sample does not match → Stranger is rejected
- False positives
  - Stranger's sample matches → Stranger is falsely accepted
- False negatives
  - User's sample does not match → User is falsely rejected
- False Match Rate vs. False Non-Match Rate

  FMR = (# matching strangers) / (# strangers in total)

  FNMR = (# non-matching users) / (# users in total)
- *T* determines tradeoff between FMR and FNMR

Evaluating Biometrics: System Errors

- Comparing biometric samples produces score ***s***
- Acceptance threshold ***T*** determines FMR and FNMR
  - If ***T*** is set low to make the system more tolerant to input variations and noise, then FMR increases.
  - On the other hand, if ***T*** is set high to make the system more secure, then FNMR increases accordingly.
- EER (Equal Error Rate) is the rate when FMR = FNMR.
- Low EER is good, it means good separation of curves.



Spoofed Biometrics: Presentation Attacks

- It is relatively simple to trick a biometric system
  - Terminology: *Presentation Attacks*

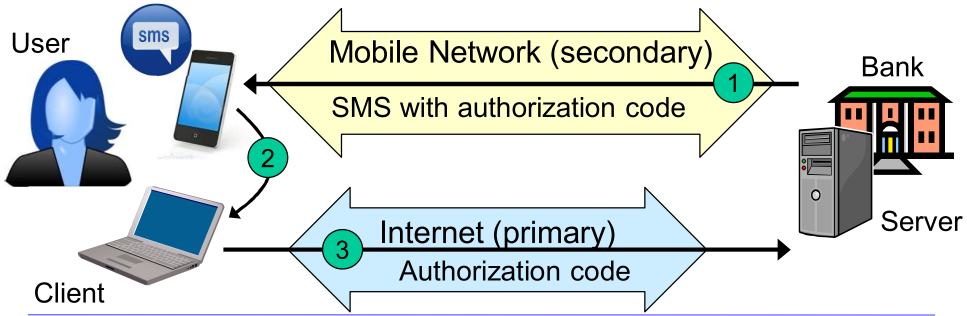
False finger

False face

- Biometric authentication on smartphones is insecure
- PAD (Presentation Attack Detection) is the subject of intensive research, to make biometrics more secure
- Alternative solution is to capture biometrics in controlled environments

Secondary Channel

- Independent from the primary channel !
- Controlled by user, not necessarily very secure
- Increased authentication assurance through Increased complexity for attackers
- Typically used as second authentication factor

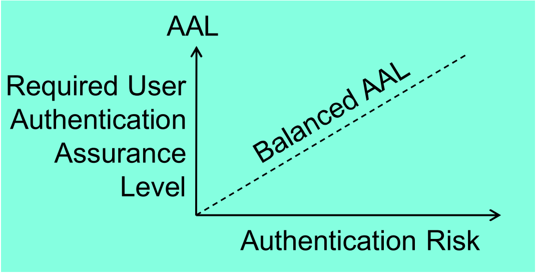


Authentication: Multi-factor

- Multi-factor authentication aims to combine two or more authentication techniques in order to provide stronger authentication assurance.
- Two-factor authentication is typically based on something a user knows (factor one) plus something the user has (factor two).
  - Usually this involves combining the use of a password and a token
  - Example: BankID OTP token with PIN + static password

Authentication Assurance

- Authentication assurance = robustness of authentication
- Resources have different sensitivity levels
  - High sensitivity gives high risk in case of authentication failure
- Authentication has a cost
  - Unnecessary authentication assurance is a waste of money
- Authentication assurance should balance authentication risk



e-Authentication Frameworks for e-Gov.

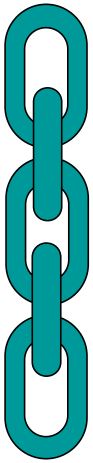
- Trust in identity is a requirement for e-Government
- Authentication assurance produces identity trust.
- Authentication depends on technology, policy, standards, practice, awareness and regulation.
- Common e-authentication frameworks allow cross-national and cross-organisational solutions that give convenience, cost savings and security.

Alignment of e-Authentication Frameworks

Authentication Framework	*User Authentication Assurance Levels*				
NIST SP800-63-3 USA 2017			Some (1)	High (2)	Very High (3)
eIDAS EU 2014			Low (1)	Substantial (2)	High (3)
ISO 29115 ISO/IEC 2013	Low (Little or no) (1)		Medium (2)	High (3)	Very High (4)
e-Pramaan India 2012	None (0)	Minimal (1)	Minor (2)	Significant (3)	Substantial (4)
NeAF Australia 2009	None (0)	Minimal (1)	Low (2)	Moderate (3)	High (4)
RAU / FAD Norway 2008	Little or no assurance (1)		Low (2)	Moderate (3)	High (4)

AAL: Authentication Assurance Level

- AAL is determined by the weakest of three links:



User Identity Registration Assurance (UIRA) requirements

Requirements for correct registration:

- Pre-authentication credentials, e.g.
  - birth certificate
  - biometrics

User Credential Management Assurance (UCMA) requirements

Requirements for secure handling of credentials:

- Creation
- Distribution
- Storage

User Authentication Method Strength (UAMS) requirements

Requirements for mechanism strength:

- Password length and quality
- Cryptographic algorithm strength
- Tamper resistance of token
- Multiple-factor methods

eIDAS

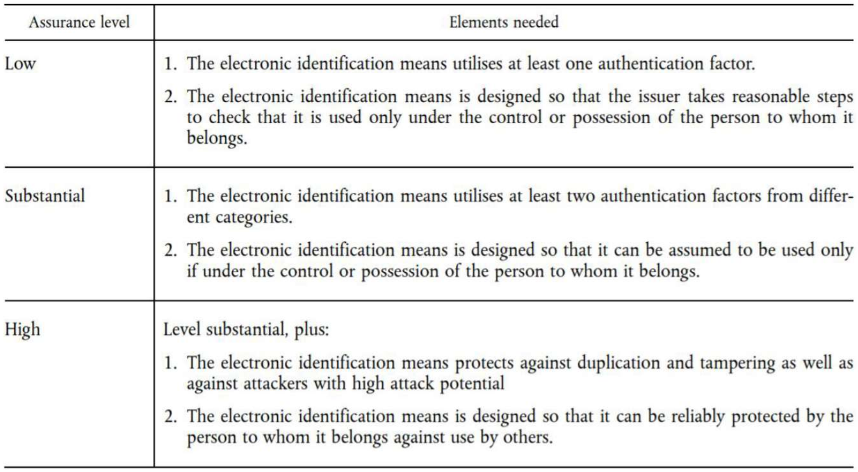
electronic IDentification, Authentication and trust Services

- eIDAS is EU's regulation on e-Authentication and trust services for e-transactions.
- "Trust service" is EU jargon for PKI certification services.
- eIDAS specifies three authentication assurance levels (AALs).

The EU trust mark for qualified trust services

Low Assurance eDAS AAL-1	Substantial Assurance eIDAS AAL-2	High Assurance eIDAS AAL-3
Limited degree of confidence in the claimed or asserted identity of a person	substantial degree of confidence in the claimed or asserted identity of a person	higher degree of confidence in the claimed or asserted identity of a person

eIDAS: Authentication



Assurance level	Elements needed
Low	1. The electronic identification means utilises at least one authentication factor. 2. The electronic identification means is designed so that the issuer takes reasonable steps to check that it is used only under the control or possession of the person to whom it belongs.
Substantial	1. The electronic identification means utilises at least two authentication factors from different categories. 2. The electronic identification means is designed so that it can be assumed to be used only if under the control or possession of the person to whom it belongs.
High	Level substantial, plus: 1. The electronic identification means protects against duplication and tampering as well as against attackers with high attack potential 2. The electronic identification means is designed so that it can be reliably protected by the person to whom it belongs against use by others.

Source: Commission implementing regulation (EU) 2015/1502

RAU Norway 2008
Rammeverk for Autentisering og Uavviselighet (Framework for Authentication and Non-Repudiation)

RAU AAL-4: High authentication assurance

- E.g. two-factor, where at least one must be dynamic, and at least one is provisioned in person

RAU AAL-3: Moderate authentication assurance

- E.g. OTP calculator with PIN provisioned by mail to user's official address

RAU AAL-2: Low authentication assurance

- E.g. fixed password provisioned in person or by mail to user's official address

RAU AAL-1: Little or no authentication assurance :

- E.g. Online self-registration and self-chosen password

Norway has adopted eIDAS in 2018 (RAU will no longer be used)

End of lecture