Authentication & Access Control

- In the security pyramid, auth & access are on the second layer
- Def of auth:

  Verify a claim of identity!

  i.e auth of a server → TLS/certificates
  i.e auth of identity → e-passport

- Factors:

  Knowledge - Something you know - Password, PIN, Key, etc...

  Possesion - Something you have - smartcard, smart token.

  Inherence - Something you are - fingerprint, iris, typing dynamics.

Knowledge Based Auth

- Problems:
  - Guessing, Sniffing, Server Breach
- Classic Attack:

  Short, predictable, user-related password i.e Mony1234

  Dictionary Attack → Try all words in a list

  Brute Force Attack → Try all chars from charset i.e. [a-z]{3}[0-9]{3}[a-z]{3}

  Mask attack → take advantage of how human design passwords.

- Choosing the password:

  ✗ Psw obtained from previous breach corpuses

  ✗ Dictionary words

  ✗ Repetitive or sequential chars 'aaaaa'

  ✗ Context-Specific words 'username'

  ✗ Psw < 8

  ✓ Balance mnemonic (don't write it)

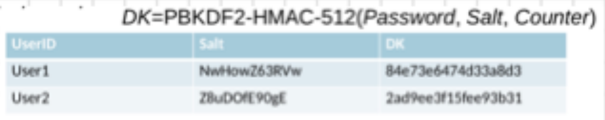
  ✓ Use pass manager.

Bruce Schneier:

1. Choose Personal sentence
   - Easy to mem for you
2. Take initial letters
3. Combine some personal tricks
4. Use it as master pass manage

Server Breach:

- Store Passwords:
  - Not in plaintext!!!
    - what if DB is violated?
  - Not using encryption!!!
    - use salting + special HMAC f
    - salting minimizes dictionary attacks
    - Special f minimizes brute force


DK=PBKDF2-HMAC-512(*Password*, *Salt*, *Counter*)

UserID	Salt	DK
User1	NwHowZ63RVw	84e73e6474d33a8d3
User2	ZBuDOfE90gE	2ad9ee3f15fee93b31

User Computer Auth

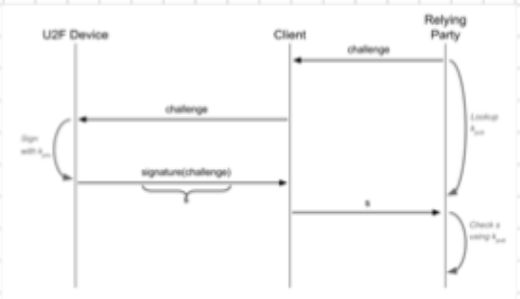
- Often multi-factor (i.e credit card)
- User identity proved by possessing an object (smartcard or smart token)
- May store sensible infos.
- Different level of Security
- Limits:
    - Auth process identifies the object rather than the user carrying it
    - Losing obj easier than forgetting a secret?
- Solution?
    - Multi-factor auth! ⟵ ≠ Multi-step
- Smart token:
    - Software or Powerful device
    - User-smart token auth often based on knowledge
    - Generates a one-time password that is accepted by authenticator.

1. Password token (OTP)

    Offline - one-time password based on secret key/seed stored in the token

2. Challenge-response token (U2F)

    Online - Password based on a challenge sent by authenticator



Private key K_{priv} stored in device by manufacturer

Public key stored in relying party.

Inherence-Based Auth

- Unique biometric of the user
- Physiological (Fingerprint, iris, blood pressure...)
- Behavioural (Signature, voice, keystroke)
- Highest level of security!
- There could be false positive & false negative.
- Requires sampling of key feature
    - Sampling generates template that best-approximates key features.
- Auth based upon the comparison of current vs template
    - There is a margin tolerance

Discussion:

Privacy -

- How does the server store the template?
- Can the scanner reproduce the features?

Safety -

- Are iris scanners dangerous for the eye?
- Finger-chopping fear.

Security -

- How hard is to fool iPhone fingerprint sensor?
- And Samsung iris scanner?

Fingerprint:

- Pattern Ridges on hands & feets, determined before birth.
  ↳ unique & don't change over time
- ∃ three basic schemes

  Loop (60%), curve pattern, ridges enter & exit same side

  Arch (5%), not very curvy, ridges enter & exit opposite side

  Whorl (35%), circular pattern, enter & don't exit.

- Detection:
  - Advance algorithm to identify minutiae
  - Ridge ending.
  - Ridge bifurcation.

Access Control

- Access Control is the core goal of computer security!
- Auth - Granting of right or permission to a system entity to access a resource
- Audit - Review or examination of system records
- AC mediates between User & System.

  ① Auth
  ② Permission check
  ③ Maintainance of permission DB
  ④ Audit & monitoring

- Important aspects are also:

  Discretionary access Control (DAC): controls on identity of requestor

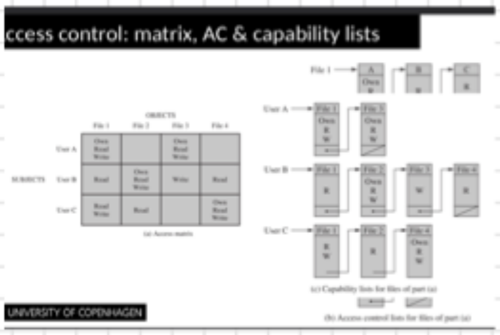
  Mandatory access Control (MAC): controls access by comparing security labels

  Role-based access Control (RBAC): controls access based on roles

  Attribute-based access Control (ABAC): controls based on user attributes

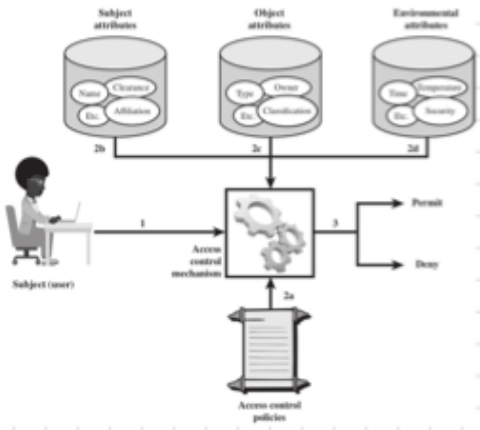
- Subject: Entity capable of accessing objects, typically a process
- Object: Resources with restricted access (files, DB records, ...)

→ Typically implemented by DB, OS
  - └→ Access control matrix.
  - └→ Column View: Access control lists.
  - └→ Row View: Capability tickets.



ccess control: matrix, AC & capability lists
OBJECTS
SUBJECTS
User A
User B
User C
File 1
File 2
File 3
File 4
(a) Access matrix
(b) Access control lists for files of part (a)
(c) Capability lists for files of part (a)
UNIVERSITY OF COPENHAGEN

→ ABAC model can define authorizations that express conditions on properties of:
  - Both resourse & subject
  - └→ Used on the web with XACML



2 principle Rules:

All that is not permitted is denied

Subjects, permissions & obs are grouped