# Authentication: Humans

Applied Information Security
Autumn 2020, Lecture 6

# PREVIOUS IN AIS...

- Hacking
- Security administration
- Security Engineering
  - Security Principles
  - Security Mechanisms
  - Security Requirements
  - Security Evaluation

Interesting learning resource:
https://www.hacksplaining.com/lessons

HACKSPLAINING

# PREVIOUS IN AIS...

- Hacking
- Security administration
- Security Engineering
    - Security Principles
    - **Security Mechanisms**
    - Security Requirements
    - Security Evaluation

# Gold Standard

## Butler W. Lampson

**authenticate** principals

today

- "Who said that?"
- "Who is getting that information?"

**authorize** access

future

- "Who can do which operation on which object?"

**audit** decision of guard

done

- "What happened? Why?"

# TODAY'S TOPICS

- Identities
- Authentication Methods
    - Something you know
        - **Protocol Design** ← Important for assignments and exercises
    - Something you have
    - Something you are
- Privacy Pitfalls

# IDENTITIES

# WHAT IS AN IDENTITY?

- We define an **identity** as a **set of attributes**
  - $I = \{a_1, a_2, a_3, \ldots\}$
- An attribute is a **statement** or **property** about an individual
  - Name
  - Email
  - CPR Nr
  - IP Address
  - Citizenship
  - Age
- When an attribute belongs to exactly one person it is called an **identifier**
  - CPR Nr

# WHAT IS AN IDENTITY? EXAMPLE

- Attributes
  - Name: Homer Simpson
  - Middle name: Joe
  - Birthdate: May 12, 1956
  - Married to Marge Simpson
  - Social Security Number: 568-47-0008 (*identifier*)
  - ...
- Identities
  - Security technician at Nuclear plant
  - Cashier at  Kwik-E-Mart
  - Student at Springfield University, Degree in Nuclear Physics
  - ...

# IDENTIFICATION VS AUTHENTICATION

- Identification
  - Determine the identity of an individual from a set of attributes
  - Example: Surveillance cameras looking for an individual in a crowd

- Authentication
  - Determining whether an identity matches a set of attributes
  - Example: A security officer at border control verifying that a passport belongs to its bearer

# ENROLLMENT

- **Enrollment** is the process of establishing **validity** of a set of attributes, with an **identity**, in a system
  - Create an account
  - Get an ID card, a visa, …
- Claimed attributes are not always checked
  - Airport Hotspots do not verify emails
  - Websites do not verify name or age
  - Governments do verify attributes by requiring physical presence, digital certificates, ...

# SOMETHING YOU KNOW

# SOMETHING YOU KNOW

- Knowledge based authentication
    - Mother’s maiden name
    - Favourite book
    - Best friend

- Secret based authentication
    - Personal Identification Numbers (PINs)
    - **Passwords**
    - Paraphrases

# KNOWLEDGE BASED AUTHENTICATION

- During enrollment, individuals provide answers to a set of queries
- The system uses a subset of these queries to authenticate the individual in the future
- The best queries have answers that are not widely known
  - Ideally only by the person to be authenticated
  - Bad Example: What is Raúl Pardo Jiménez mother's first surname?
    - In Spain, typically, newborns' surnames are constructed using the first surname of the father followed by the first surname of the mother.

# KNOWLEDGE BASED AUTHENTICATION

- Vulnerability: More than one system may use the same questions
  - Therefore, they know the answer of the individuals and can impersonate them.
- Identification is also vulnerable, e.g., right to data deletion in GDPR
  - Many companies ask for your passport or valid ID to prove your identity

# KNOWLEDGE BASED AUTHENTICATION

- Pros: Convenient
  - Doesn't place much burden on people to remember things

- Cons: Relies on how *secret* the information is
  - That is, how easy is for an attacker to access the answers to the questions

# SECRET BASED AUTHENTICATION

- Authentication can be based on a secret a person knows

- Given that the secret is
  - Unknown to attackers
  - Difficult to guess
  - Difficult to steal

- Examples
  - Personal Identification Numbers (PINs)
  - Passwords
  - Paraphrases

# SECRET BASED AUTHENTICATION

- Authentication can be based on a secret a person knows

- Given that the secret is
  - Unknown to attackers
  - Difficult to guess
  - Difficult to steal

- Examples
  - Personal Identification Numbers (PINs)
  - **Passwords**
  - Paraphrases

We focus on passwords, but the content applies to PINs, paraphrases or other types of secrets for human authentication

# PASSWORD LIFE CYCLE

- Create: User chooses a password
- Store: Human/System stores a password
- Use: User request system to supply a password for authentication
- Change/recover/reset: User changes password

# CREATE A PASSWORD

- How to choose a password?

# CREATE A PASSWORD

- Invented by a human
  - Easy to remember
    - Word in dictionary
    - Loved-one's name
    - "asdf", 12345, "password", ...
  - Weak-passwords
    - Easy to guess

- Generated by a computer
  - Pseudorandom string
  - Difficult to remember
  - Strong passwords

- Generated by a sysadmin
  - Any of the two previous cases

- Top ten passwords [cnn.com, 2019]
  1. 123456
  2. 123456789
  3. qwerty
  4. password
  5. 111111
  6. 12345678
  7. abc123
  8. 1234567
  9. password1
  10. 12345

# CREATE A PASSWORD

- Invented by a human
  - Easy to remember
    - Word in dictionary
    - Loved-one’s name
    - “asdf”, 12345, “password”, ...
  - Weak-passwords
    - Easy to guess

- Generated by a computer
  - Pseudorandom string
  - Difficult to remember
  - Strong passwords

- Generated by a sysadmin
  - Any of the two previous cases

# STRONG PASSWORDS

- Strong passwords are passwords that are difficult to guess
  - Difficult to brute force, if $2^x$ guesses required then the password has strength “x”
- Consider passwords that are *l* characters long from an alphabet of *n* characters
  - There are $n^l$ different passwords
  - Solve $x$ in $2^x = n^l$
  - Then $x = l \log_2 n$
  - $x$ is also known as the entropy of the password

# STRONG PASSWORDS

- Strong passwords are passwords that are difficult to guess
  - Difficult to brute force, if $2^x$ guesses required then the password has strength "x"
- Consider passwords that are *l* characters long from an alphabet of *n* characters
  - There are $n^l$ different passwords
  - Solve $x$ in $2^x = n^l$
  - Then $x = l \log_2 n$
  - $x$ is also known as the entropy of the password
- Assumes all elements equally likely (uniformly distributed)
  - "12345" as strong as "@`+2F"? 🤔

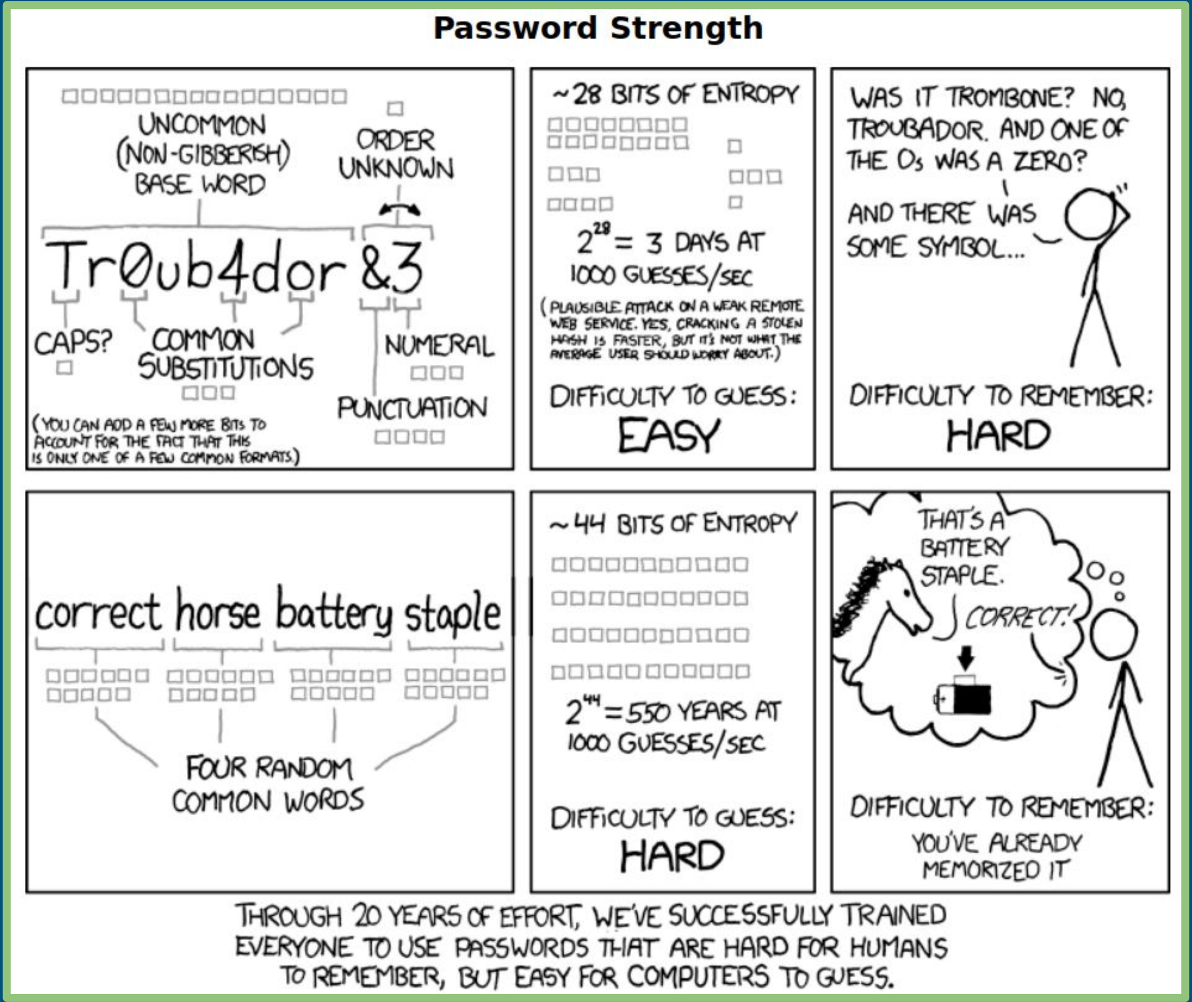
# PASSWORD RECIPES

- Rules for composing passwords
  - For instance:
    - At least one upper- and lower-case,
    - At least one special symbol
    - At least one digit
    - Minimum length 20-30 characters
    - (...)

In the exercise session you will implement your own password recipe validator with

Passay
password policy for Java

- Recipes tend to be burdensome for users
  - Users try to pick the easiest possible password that complies with requirements
    - Attackers know this, therefore the recipe loses effectiveness
    - [Mentimeter]




Source: https://www.explainxkcd.com/wiki/index.php/936:_Password_Strength

# PASSWORD GUESSING: ONLINE ATTACKS

- The system is used by the attacker to determine whether a guessed password is correct for an individual
  - Brute force
  - Using a dictionary (collection of possible passwords)

- Defences
  - Make authentication time consuming
  - Impose a limit on unsuccessful attempts
  - Restrict amount of information from unsuccessful attempts
    - Do not mention whether attributes are in the system (e.g, email address or username)

- Covert channels
  - Time

# PASSWORD GUESSING: ONLINE ATTACKS

- The system is used by the attacker to determine whether a guessed password is correct for an individual
  - Brute force
  - Using a dictionary (collection of possible passwords)

We discuss in detail offline attacks in Lecture 8.

- Defences
  - Make authentication time consuming
  - Impose a limit on unsuccessful attempts
  - Restrict amount of information from unsuccessful attempts
    - Do not mention whether attributes are in the system (e.g, email address or username)
- Covert channels
  - Time

# STORING PASSWORDS

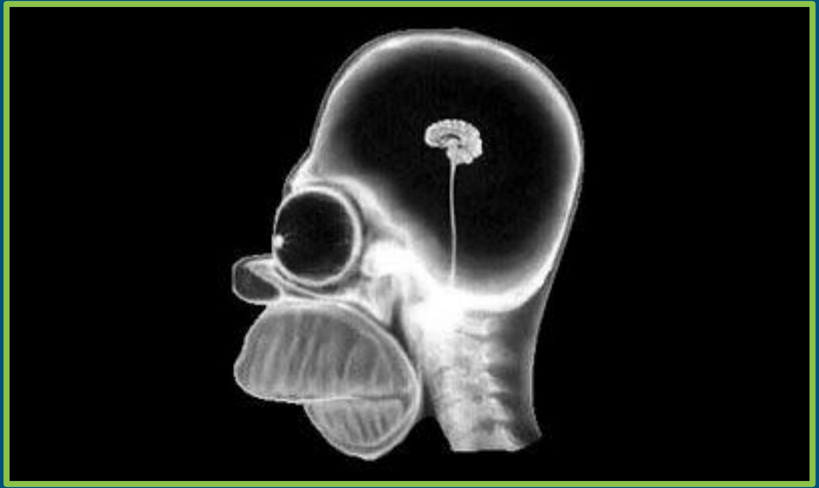
- Ideally there should be a password for each identity

- Two main storage options
  1. Storage by humans
  2. Storage by machines ← Explained in detail in Lecture 8

# STORAGE BY HUMANS

- Little memory capacity

- Consequently:
  - Reuse passwords
  - Record them physically



# VULNERABILITIES OF HUMANS STORAGE

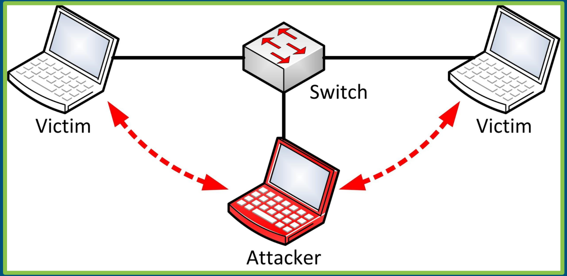
- Reuse passwords
  - Attacker needs to comprise one password and he will be able to authenticate in all other systems
    - Undermines **Principle of Least Privilege**

- Record physically
  - Can be seen by anyone
  - Typically, the storage place is not very secure and is nearby the authentication interface

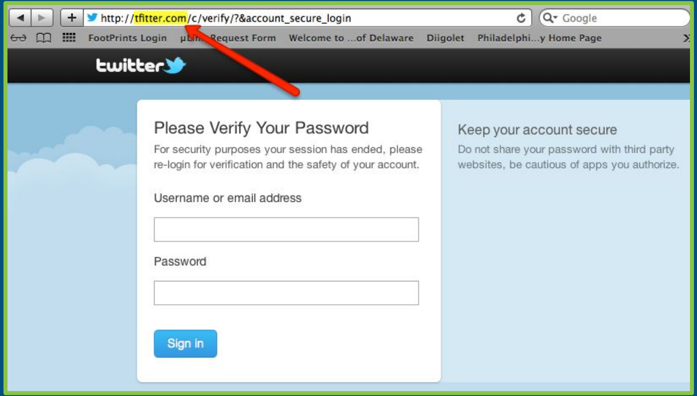
Wifi password leak during the Estonian 2013 elections (source: https://estoniaevoting.org/photos/opsec-wifi/ )

# SOME ATTACKS TO STEAL PASSWORDS

- Compromised I/O
- Man-in-the-middle (network)


- Fake login forms
- Social Engineering

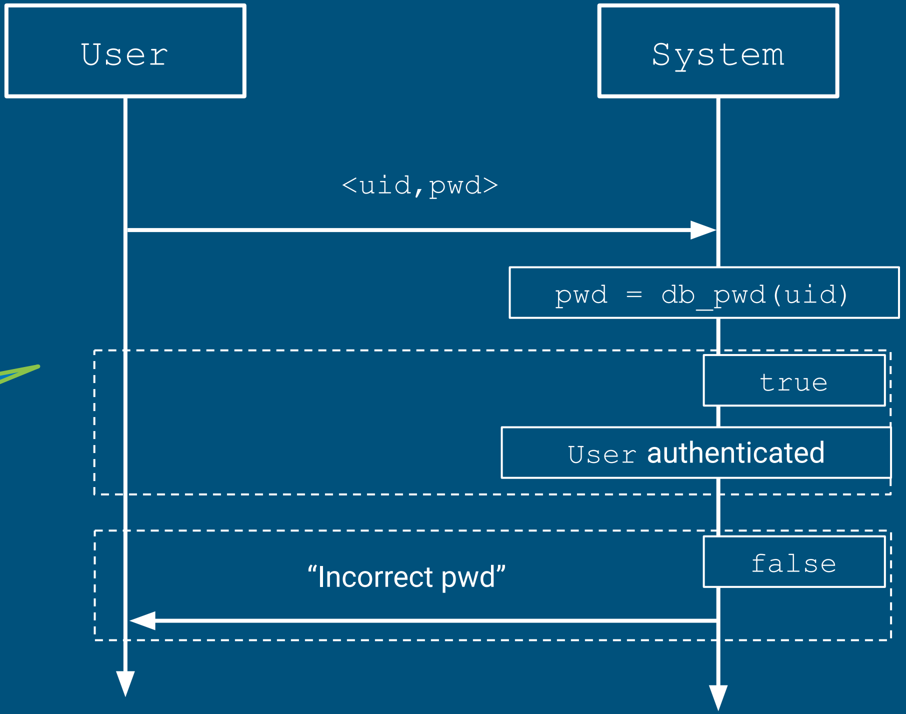


# INTERMEZZO

## *Protocol Design*

# PROTOCOL DESIGN: PWD AUTHENTICATOR

Consider a user (`User`) and a system (`System`). At enrollment, `User` provides a password to the system, then `System` stores it in the database. `System` can retrieve the password by using the function `db_pwd(uid)` where `uid` is the user identifier.




<u>Possible option:</u> using sequence diagrams to define the protocol **(we will not use them in this course)**

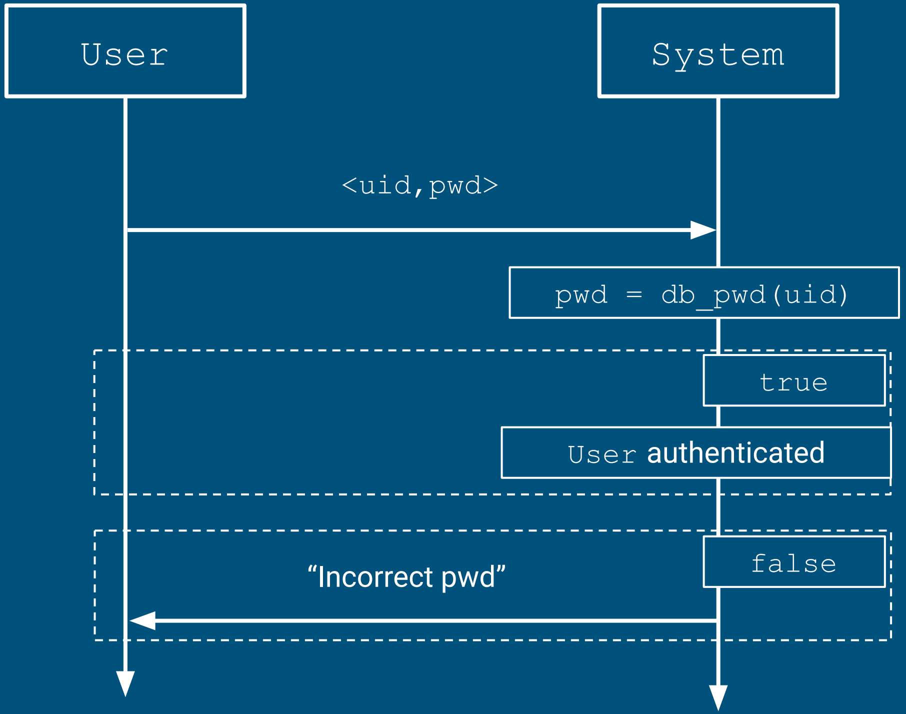
# PROTOCOL DESIGN: PWD AUTHENTICATOR

Consider a user (`User`) and a system (`System`). At enrollment, `User` provides a password to the system, then `System` stores it in the database. `System` can retrieve the password by using the function `db_pwd(uid)` where `uid` is the user identifier.

The password authentication protocol is defined as follows:

```
1.   User -> System: <uid,pwd>
2.   System: if pwd == db_pwd(uid)
             then Deem User authenticated
             else System -> User:  "Incorrect pwd"
```

Protocol design language introduced in [NS78] specifically to **design and analyze cryptographic protocols.**



# ONLINE LOGIN PWD EXAMPLE

```
1.  User -> PC: I want to login at http://server.com/
2.  PC -> User: “Enter user id and password”
3.  User -> PC: <uid,pwd>
4.  PC -> Server: <uid,pwd>
5.  Server: if pwd = db_pwd(uid)
              then Deem uid authenticated
                   Server -> PC: res with res = OK
              else Server -> PC: res with res = INCORRECT_PWD
6.  PC: if res = OK
          then PC -> User: “logged-in correctly”
          else PC -> User: “Incorrect username/password”
```

# ONLINE LOGIN PWD EXAMPLE

1. `User -> PC:` I want to login at http://server.com/
2. `PC -> User:` “Enter user id and password”
3. `User -> PC: <uid,pwd>`
4. `PC -> Server: <uid,pwd>`
5. `Server:` **`if`** `pwd = db_pwd(uid)`
   **`then`** Deem `uid` authenticated
   `Server -> PC: res` with `res = OK`
   **`else`** `Server -> PC: res` with `res = INCORRECT_PWD`
6. `PC:` **`if`** `res = OK`
   **`then`** `PC -> User:` “logged-in correctly”
   **`else`** `PC -> User:` “Incorrect username/password”

Steps 1-4 model the usual interaction of a user accessing a login form (e.g. via a web browser)

Step 5 models the check performed on the server side. Also, the effect of result.

Step 6 is optional, it models notification of the result to the user

# PROTOCOL DESIGN LANGUAGE

- Each step of the protocol must be enumerated

  1. $s_1$
  2. $s_2$
  3. $s_3$

  …

See the **protocol design notes** on Learnit

Each step $s_n$ must be of the form:

i) `Sender -> Receiver: message`

- Meaning that `Sender` sends `message` to `Receiver`

ii) `Actor: program`

- Meaning that `Actor` executes `program`
- A `program` is defined using pseudo-code, e.g.
  - **`if-then-else`** (conditional statements)
  - `z := v` with `condition` ([conditional] assignments)
  - `User` is authenticated (English statements)
  - `Sender -> Receiver: message` (sending message)
  - “message for user” (string)
  - …

# ONLINE LOGIN PWD EXAMPLE (ELEMENTS)

We have 3 actors in this protocol: User, PC and Server.

English statement

Sending a string

```
1.  User -> PC: I want to login at http://server.com/
2.  PC -> User: “Enter user id and password”
3.  User -> PC: <uid,pwd>
4.  PC -> Server: <uid,pwd>
5.  Server: if pwd = db_pwd(uid)
            then Deem uid authenticated
                 Server -> PC: res with res = OK
            else Server -> PC: res with res = INCORRECT_PWD
6.  PC: if res = OK
        then PC
        else PC                wo
```

Sending pairs

English statement

**if-then-else**

Conditional assignment

It is important to **always define the elements of the protocol** before writing the protocol: **Actors, variables, functions, datasets, etc, ...**

Don’t panic about details in defining `programs`. Simply be **rigorous in the definition**.

# MAN-IN-THE-MIDDLE ATTACKS

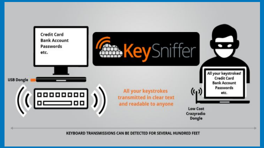
1. User -> PC: I want to login at http://server.c
2. PC -> User: “Enter user id and password”
3. User -> PC: <uid,pwd>
4. PC -> Server: <uid,pwd>
5. Server: **if** pwd = db_pwd(uid)
   **then** Deem uid authenticated
   Server -> PC: res with res = OK
   **else** Server -> PC: res with res = INCORRECT_PWD
6. PC: **if** res = OK
   **then** PC -> User: “logged-in correctly”
   **else** PC -> User: “Incorrect username/password”

By default we impose no assumptions on communication channels.
**Attackers can read the information being transmitted.**

# MAN-IN-THE-MIDDLE ATTACKS *EXAMPLES*

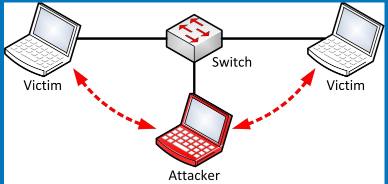
1. User -> PC ... er.co
2. PC -> User: ... d”
3. User -> PC: <uid,pwd>
4. PC -> Server: <uid,pwd>
5. Server: **if** pwd ... d(uid)
   **then** Deem uid authenticate...
   Server -> PC: res with res...
   **else** Server -> PC: res with res...
6. PC: **if** res = OK
   **then** PC -> User: “logged-in correctly”
   **else** PC -> User: “Incorrect username/password”

Example 1: Insecure keyboard




By default we impose no assumptions on communication channels. <u>**Attackers can read the information being transmitted.**</u>

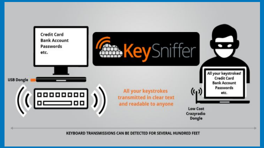
Example 2: Network communication



# MAN-IN-THE-MIDDLE ATTACKS *SOLUTIONS*

```
1.  User -> PC                         er.c
2.  PC -> Us                          d"
3.  User -> PC:
4.  PC -> Serve
5.  Server: if p
           then
                Server -> PC: res with res
           else Server -> PC: res with res
6.  PC: if res = OK
        then PC -> User: "logged-in correctly"
        else PC -> User: "Incorrect username/pas
```

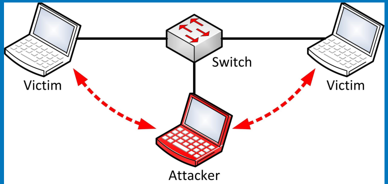
Example 1: Insecure keyboard




<u>What to do?</u> We can place trust assumptions, e.g.,
**Assumption 1: Users only use wired keyboards and the keyboard driver is trusted.**

By default we impose no assumptions on communication channels.
**<u>Attackers can read the information being transmitted.</u>**

Example 2: Network communication



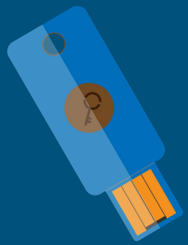

<u>What to do?</u> We can **specify secure communication via cryptosystems.**
*(Topic of the next two lectures.)*

# SOMETHING YOU HAVE

# TOKENS


FORTINET
ONE TIME PASSWORD
FortiToken 200
FORTINET
FortiToken
PRESS


BANK
1234 5678 9123




TAN, F oder Menü
1
2
3
TAN
4
5
6
F
7
8
9
MENÜ
C
0
,
OK

# TOKEN TYPE SELECTION FACTORS

- Form Factor
    - Convenience for end users
- Computational Capabilities
    - Computationally capable devices may perform complex tasks
        - Cryptographic operations
- Economics
    - Typically cheaper devices are easier to attack

# ONE-TIME PASSWORDS

- Many tokens simply produce *one-time passwords*
- Password may be used only once
- Attackers cannot predict future passwords from old ones

# ONE-TIME PASSWORDS

- Consider a User (U), Token (T) and a system (S)
  - At enrollment T is given a secret ($s_T$) and S keeps a local copy secret ($s_S$)
  - S contains a set of enrolled users and their corresponding secrets (denoted as Users)
  - db_secrets($id_U$) returns the secret stored by the server for $id_U$
  - Both the server and the token can compute a *hash function* h(r || s) where r is a random nonce and s a secret

1. U -> T: I want to authenticate
2. T -> S: $id_U$
3. S: **if** $\langle id_U, s_T \rangle \in$ Users
   **then** S -> T: r where r is unpredictable, e.g., random nonce
4. T -> S: t with t = h(r || $s_T$)
5. S: **if** t = h(r || $s_S$)
   **then** T is authenticated

We explain in detail hash functions in Lecture 8

The operation || means concatenate

# TOKEN AUTH: DIGITAL SIGNATURES

- Consider a User (`U`), Token (`T`) and a system (`S`)
  - At enrollment `T` generates a secret key ($s_T$) and a public key ($p_T$) accessible by `S`
  - `S` contains a set of enrolled users and their corresponding secrets (denoted as `Users`)
  - $\texttt{db\_secrets}(id_U)$ returns the secret stored by the server for $id_U$
  - Both the server and the token can compute a *hash function* `h(r || s)` where `r` is a random nonce and `s` a secret

1. `U -> T:` I want to authenticate
2. `T -> S:` $id_U$
3. `S:` **if** $\langle id_U, s_T \rangle \in$ `Users`
   **then** `S -> T: r` where `r` ~~is unpredictable, e.g., random nonce~~
4. `T -> S: t` with $t = \mathrm{sign}(r, s_T)$
5. `S:` **if** $\mathrm{verify}(t, r, p_T)$
   **then** `T` is authenticated

Computationally expensive, not implementable in all devices, e.g., plastic cards

# THEFT

- What if the user's token is stolen?

# MULTI-FACTOR AUTHENTICATION

- In order to avoid attacks it is advisable to combine more than one authentication method
  - Principle of **Defense in Depth**
- Require users to enter a PIN
- Require user to enter a code sent to her email
  - Email must have been registered during enrollment

# ONE-TIME PASSWORD WITH MULTIFACTOR

- Consider a User (`U`), Token (`T`) and a system (`S`)

- At enrollment:
  - `T` is given a secret ($s_T$) and `S` keeps a corresponding secret ($s_S$)
  - `U` chooses a PIN that is hashed and stored in `T` ($hpin_T$)
    - Assumption: the PIN is stored in a tamper proof manner

- `S` contains a set of enrolled Users (`Users`)

- Both the server and the token can compute a *hash function* `h(r || s)` where `r` is a random nonce and `s` a secret

# ONE-TIME PASSWORD WITH MULTI-FACTOR AUTHENTICATION

1. U -> T: I want to authenticate
2. T -> U: "Enter PIN"
3. U -> T: $pin_U$
4. T: **if** $h(pin_U) = hpin_T$
   **then** T -> S: $id_U$
   **else** T -> U: "Incorrect PIN"
5. S: **if** $\langle id_U, s_T \rangle \in Users$
   **then** S -> T: r where r is unpredictable, e.g., random nonce
6. T -> S: $t = h(r \,||\, s_T)$
7. S: **if** $t = h(r \,||\, s_S)$
   **then** T is authenticated

# EXAMPLE: ITU'S ACCESS CARD — GYM

Complete in
[protocol design notes]

1. U -> R: $id_U$ // by showing the card nearby the card reader
2. R -> S: $id_U$, room
3. S: **if** $\langle id_U, room \rangle \in$ RoomAccess and room = GYM
   **then** S -> R: res with res = pin_required
   **else** S -> R: res with res = not_registered
4. R: **if** res = pin_required
   **then** R -> U: Show blinking orange light // meaning "enter pin"
   **else** R -> U: Show red light
5. U -> R: $pin_U$
6. R -> S: $pin_U$
7. S: **if** $pin_U$ = db_pins($id_U$)
   **then** S -> R: Show green light and open door
   **else** S -> R: Show red light

# SOMETHING YOU ARE

# BIOMETRICS

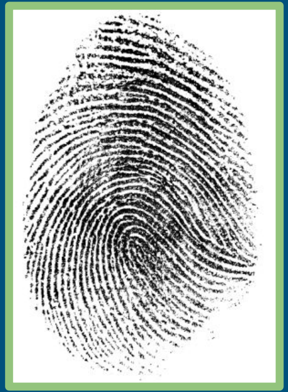
- Humans identify each other by means of biometrics
  - Physical traits
  - Behaviour
  - Voice
  - …
- Police often uses fingerprints to locate suspects on a crime scene
- Some of them useful for authentication

# REQUIREMENTS

- In order for biometrics to be useful for authentication they must comply with the following requirements.
  - Uniqueness
  - Small variation over time
  - Easy to measure
  - Difficult to spoof
  - Acceptable for users
    - Biometrics are personal data, in some cases very sensitive

# FINGERPRINTS

- Characterised by *minutiae*
    - Features of the raised ridges that appear in the skin on human fingertips
- Fingerprint readers are cheap
    - Included in phones
    - Laptops
        - Can be spoofed, since do not implement liveness tests
- Finger must be placed on the reader
    - Short distance



# FACES

- Based on absolute proportions and specific features of faces

- Different approaches
  - Image processing looks for specific facial features
  - Statistical learning (e.g., neural networks) that have been trained to match faces

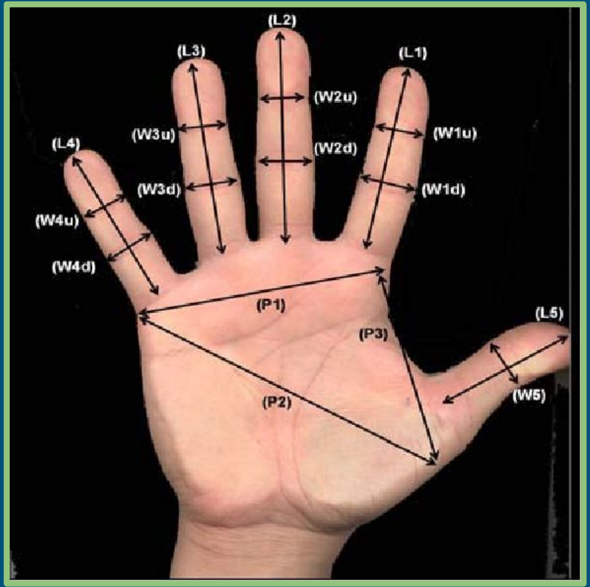
- Measurement can be done in distance

# EYES

- Iris
  - It is based on the pattern of pigments in the ring of coloured tissue that surrounds the pupil
  - It stabilizes after a person has reached adolescence
  - Measurement can be performed around half a meter away
- Retina
  - Unique pattern of veins can be found
  - Requires individual to focus on a point for some seconds
  - Typically consider uncomfortable by individuals

# HANDS

- A sensor measures
  - Palm, length, width, thickness.
- Images reduces from (e.g.) 31000 points to 90 measurements then to 9 bytes of data
- High resilience to scars, ridges or tattoos
  - But rings, bandages or gloves lead to errors



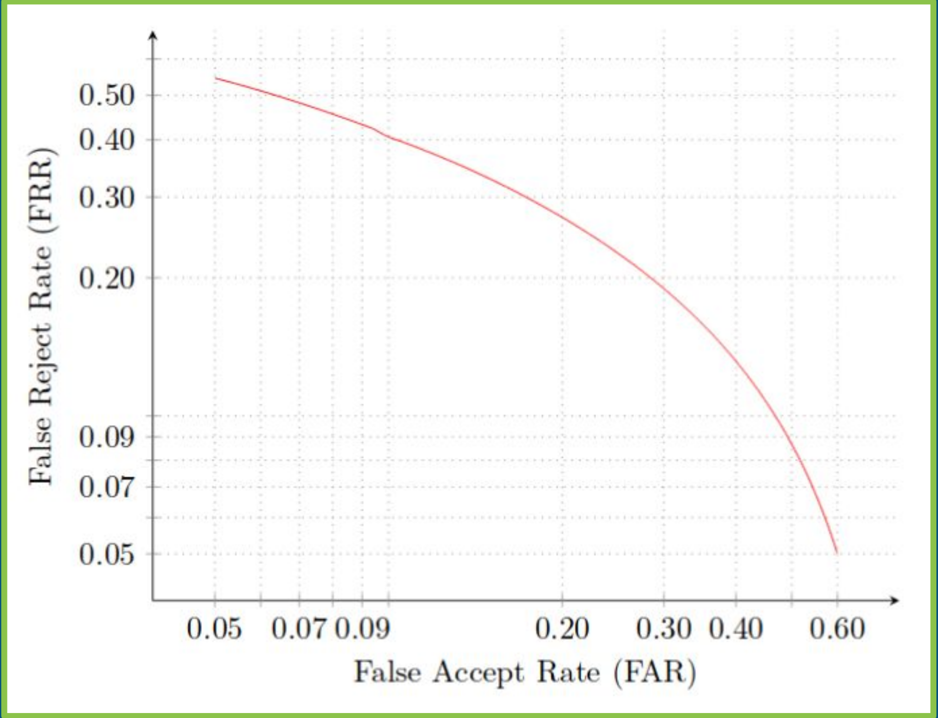
# BIOMETRICS LESS LIKELY TO BE USED

- Handwritten signatures
  - Too much variation
- Voice
  - Voice changes very often due to, e.g., colds or sore throats
  - Background noise may affect as well
- Body odor
  - We haven’t reached dog level smelling :)
  - Good sensors do not exist
- Brain waves
  - Attacker may spoof targets by becoming familiar with similar images

# ACCURACY

- False accept
  - Authenticate individual with wrong identity
- False reject
  - Fail to authenticate individual with right identity
- Detection Error Trade off (DET)
  - In a military base it is better to increase accuracy even if it increases false rejects
    - A false accept can be catastrophic
  - In a golf club it might be better to minimize false rejects
    - The reputation of the club might be affected by rejecting a member

# Detection Error Trade-off (DET)




Source: https://www.cs.cornell.edu/fbs/publications/chptr.AuthPeople.pdf

# BIOMETRICS: ENROLLMENT

- In order for biometrics to be used for authentication, the authentication system stores a *template*

- A template is data that can be used to verify your biometrics during authentication
    - Fingerprint
    - Facial features
    - Iris and retina features
    - Hand geometry

- The template may contain ***highly sensitive information***

# PRIVACY PITFALLS OF AUTHENTICATION

# PRIVACY CONCERNS

- Authentication requires that the authentication system to learns the identity of an individual
  - Remember, identity is defined as a set of attributes
  - As seen earlier, some of these attributes may be sensitive
- Privacy
  - Individual’s right to determine by herself how data must be handled
    - To whom it can be communicated
    - For which purposes it might be used
    - For long it can be stored or used

# PRIVACY CONCERNS

- Individuals may not want to disclosed sensitive attributes
  - Such as biometric data
- Individuals may not want their identity to be bounded to an action
  - Accessing a room
  - Buying an item
- Some attributes may be analyzed to learn infer others
  - Electrocardiogram (ECG) information may reveal underlying health conditions
- Aggregating identifiers may lead to disclosure of sensitive data

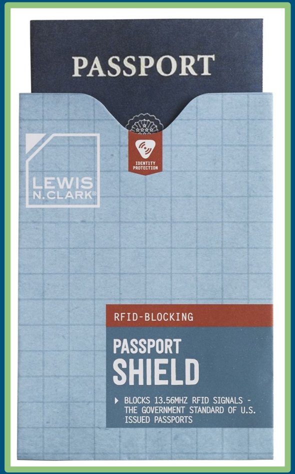
# GUIDELINES TO PREVENT PRIVACY PITFALLS

- Seek Consent
  - Authentication must only be carried out after the principal giving explicit consent
- Select Minimal Identity
  - Collect identities which require minimum amount of attributes
- Limit Storage
  - Do not save authentication information unless it is necessary
- Avoid Linking
  - Do not reuse identifiers for identities in different systems

# SIMILAR TO GDPR

- Seek Consent (Explicit consent)
    - Authentication must only be carried out after the principal giving explicit consent
- Select Minimal Identity (Data minimisation)
    - Collect identities which require minimum amount of attributes
- Limit Storage (Data minimisation + purpose of usage)
    - Do not save authentication information unless it is necessary
- Avoid Linking (purpose of usage)
    - Do not reuse identifiers for identities in different systems

# RFID CHIPS




- Constantly ready for authentication
  - Attacker may place an authentication device nearby
- No notification
  - Individuals may be unexpectedly authenticated as no notification is provided after being authenticated
- Thus, RFID alone violate the seek consent guideline
  - There exist some solutions
    - E.g., US passports have a cover of foil which creates a Faraday cage when being closed
    - Opening the password is interpreted as giving consent for being authenticated

# HEARTBEAT AUTHENTICATION

- Template information may be correlated to health state

- In conflict with limit data storage
  - Attributes contain additional information which is irrelevant for authentication

- Additional risk (not proven, MSc thesis topic):
  - Electrocardiogram information may be synthesized from old samples
  - E.g., fitbit, garmin, strava workout data
    - These companies could **impersonate individuals**

# SUMMARY

- Identities
  - Identification vs Authentication
- Protocol Design (important for assignments and exercises)
- Authentication Methods
  - Something you know
  - Something you have
  - Something you are
- Privacy Pitfalls

# ACKNOWLEDGEMENTS


- Michael’s Clarkson slides on human authentication, passwords and tokens have inspired some parts of this lecture