JÖNKÖPING UNIVERSITY
School of Engineering

Authentication and access control

Module 2, Information Security, 7,5 ECTS

Erik Bergström
erik.bergstrom@ju.se

Overview of module 2

- Authentication
- Access control

Identification and authentication

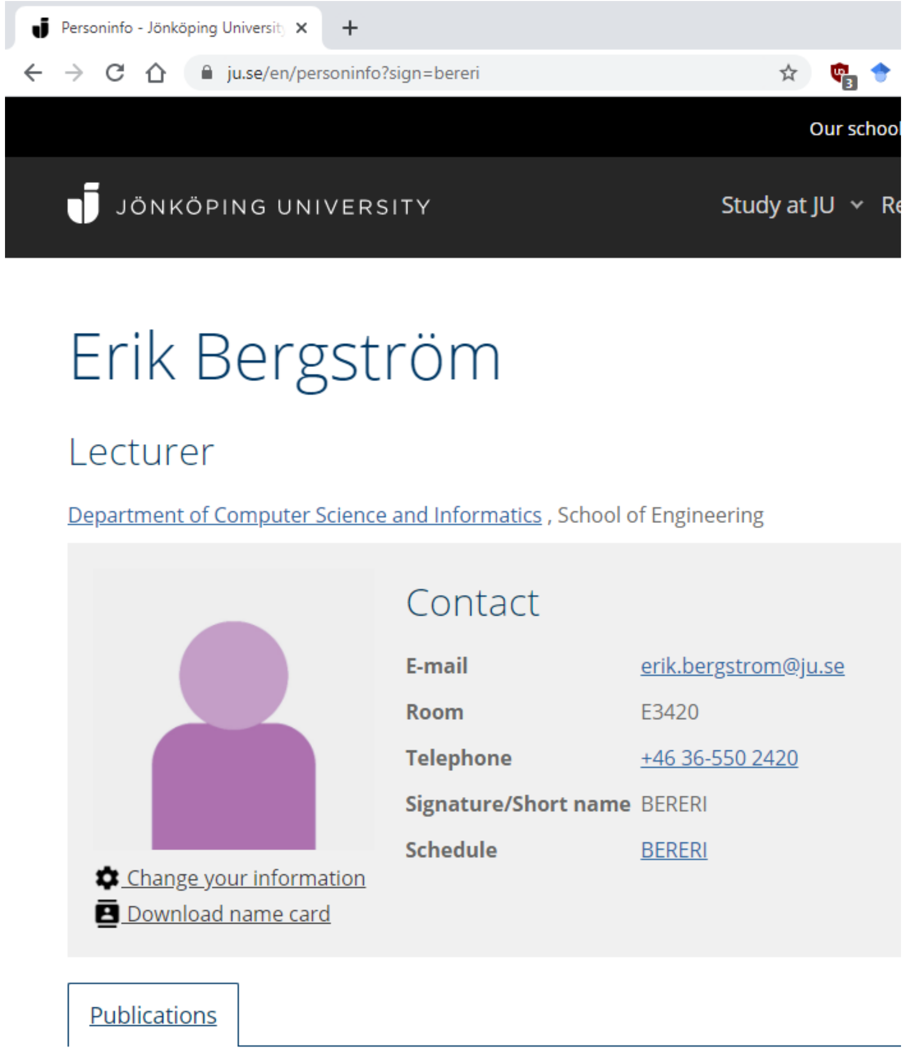
- The terms and concepts identification and authentication are often mixed or confused.

- Identification is the act of indicating a person or thing’s identity
- Authentication is the act of proving that a user is who she says she is

- Identity is often public
- Authentication should be private

Identification

- Establishes the identity of an individual
- Identities are often well-known, predictable, or guessable.
  - Email-addresses
  - Usernames
  - 3 first letters from the surname + 3 first from the first name
- What is my identity@JU? What is Sonnys?



Erik Bergström

Lecturer

Department of Computer Science and Informatics , School of Engineering

Contact

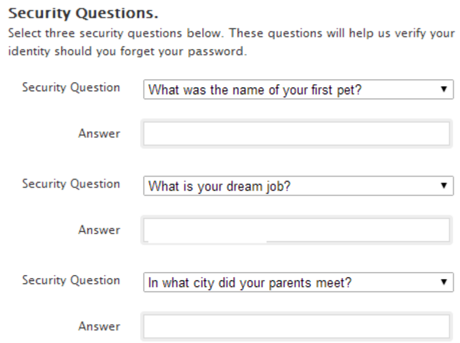
E-mail	erik.bergstrom@ju.se
Room	E3420
Telephone	+46 36-550 2420
Signature/Short name	BERERI
Schedule	BERERI

Authentication

- The act of proving that a user is who she says she is

- Mechanisms:
  - Something the user *knows*
  - Something the user *is*
  - Something the user *has*

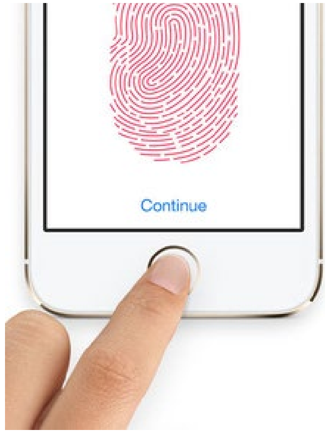
- Can be combined, i.e. two-factor or multi-factor authentication



Security Questions.

Select three security questions below. These questions will help us verify your identity should you forget your password.

Security Question	What was the name of your first pet? ▾
Answer	
Security Question	What is your dream job? ▾
Answer	
Security Question	In what city did your parents meet? ▾
Answer	



Something the user *knows* - passwords

- Passwords
  - Most common method
  - For each user, the system stores both the username and hashed password
    - The hash is non-reversible:
      HP=hash(password) is easy to compute on any input
      From hash(password), password is (extremely) difficult to compute
  - Must be easy to remember – and hard to guess ;-)
- Security questions
  - Don’t use – too much info is available online.
  - Better to rely on other techniques

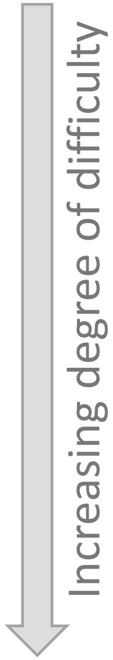
Attacking passwords

- Password authentication is used for anything/everything

- How do we attack?
    - Online
        - Repeated manual or automatic entering of passwords
        - Servers can block and deny access after repeated failures
    - Offline
        - Require access to the hashed password(s)
        - Old Unix: /etc/passwd
        - New Unix /etc/shadow – only readable by root
        - Windows: stored in registry hive in binary format (but still accessible). Hash from SAM file or AD or interception when sent over network
        - Check as much as you want
        - Must be made expensive

Attacking passwords

- Old (1998) but still relevant list of steps for an attacker to try, in order, to determine a password:
    - no password
    - the same as the user ID
    - is, or is derived from, the user’s name
    - on a common word list (for example; password, secret, private) plus common names and patterns (e.g. qwerty, aaaaaa,123, 123456)
    - contained in a short college dictionary
    - contained in a complete English word list
    - contained in common non-English-language dictionaries
    - contained in a short college dictionary with capitalizations (PaSsWorD) or substitutions (digit 0 for letter O, and so forth)
    - contained in a complete English dictionary with capitalizations or substitutions
    - contained in common non-English dictionaries with capitalization or substitutions
    - obtained by brute force, trying all possible combinations of alphabetic characters
    - obtained by brute force, trying all possible combinations from the full character set
- The last step will always work – but time/CPU is limited
- Brute force is systematic – but inefficient



Increasing degree of difficulty

Attacking passwords

- Dictionary attacks
    - Trying all the strings in a pre-arranged listing derived from lists
    - More efficient than brute force since we tend to use names, places,…
    - Many password recovery (/cracking) tools exist, e.g.:
    - Dictionary attacks are best suited for passwords that are not too long
- Guessing attack
    - Exploits human nature to use easy to remember passwords
    - Trial-and-error

Attacking passwords

- Brute force, dictionary and guessing attacks use clear text passwords as input
  - Run the password through the system online or the algorithm offline
  - Hence a slow hashing mechanism wastes time!

- Rainbow tables (simplified here and in the book)
  - Generally an offline attack
  - Uses precomputed lists of hashes
  - Rainbow tables are a compromise between pre-computation and low memory usage

Password salt

- Same password will generate same hash - password salt is used to overcome this problem
  - Salt can be random or generated from clock, process identifier…
  - Salt is 8bytes in UNIX/Linux
  - Salt is stored in the password table with the password and the username

```
HP=hash(password||salt)
```

- Don’t use the same salt or too short salt
  - Long salt counter rainbow tables

Salt example

- Without salt

Username	Password	Hashed value (MD5)	Hashed value (SHA-256)
user1	MySuperPassw0rd	e746e64b281f03f09d5623d97eef5869	95210fefc572ea43e1bee40c52140066a9e0d6f5ebebabd8f920140856d1b017
user2	MySuperPassw0rd	e746e64b281f03f09d5623d97eef5869	95210fefc572ea43e1bee40c52140066a9e0d6f5ebebabd8f920140856d1b017

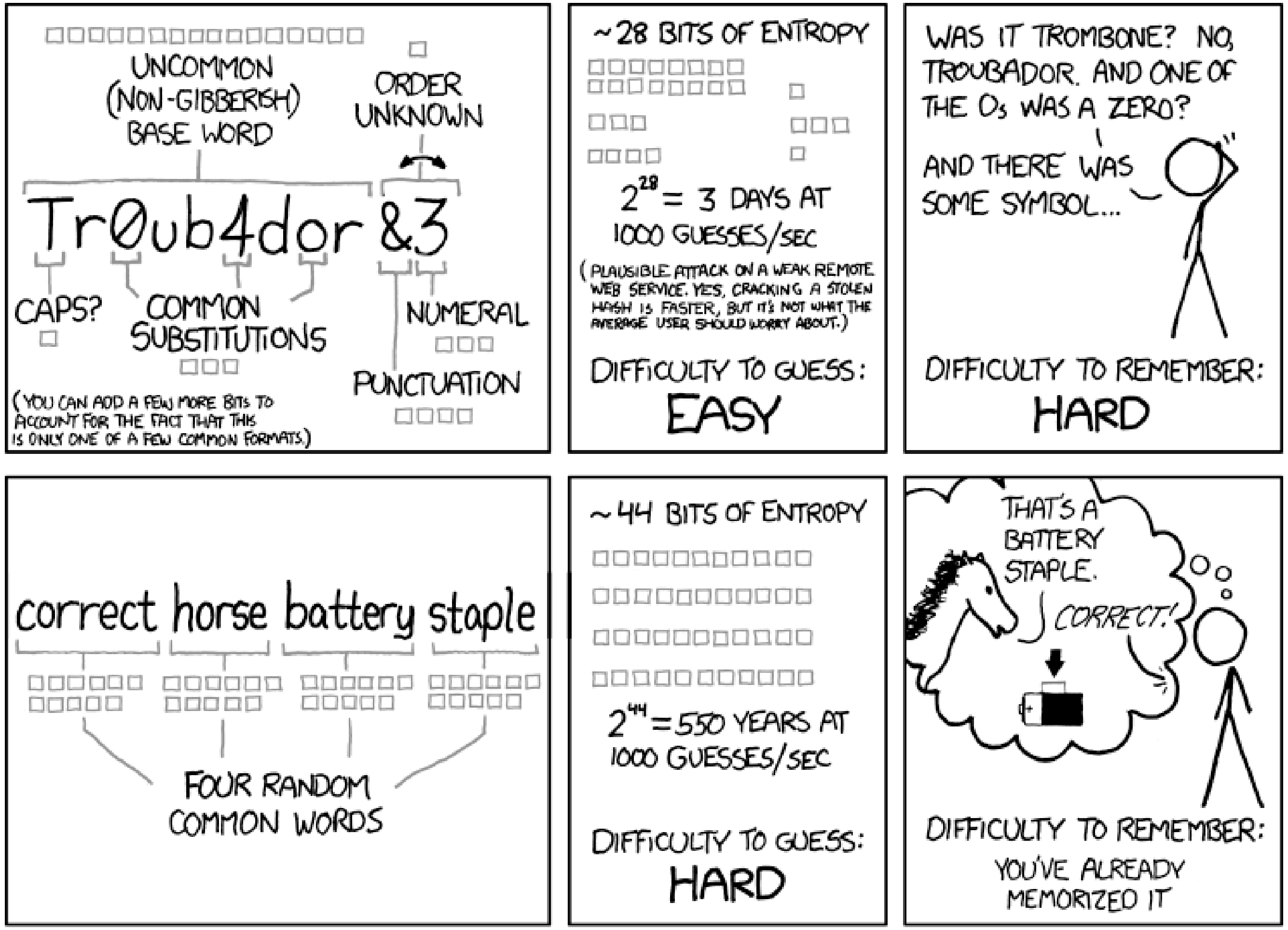
- With salt

Username	Password	Salt (in hex)	String to hash	Hashed value (SHA-256)
user1	MySuperPassw0rd	436f4e7665727431	MySuperPassw0rd436f4e7665727431	fd0cc86c33bd00092270eff52fd6eb9fc36a245fd07c21e25b64ccf8a2c288dc
user2	MySuperPassw0rd	c3b6c3a4c3a5706f	MySuperPassw0rdc3b6c3a4c3a5706f	4626ed723087c03251b431d18b080fa00fb08ee34542a4bc06c0a518d4a69926

How to choose a good password (Pfleeger)

- Use characters other than just a–z
  - a-z is only 26 possibilities. A-Z+a-z+0-9 = 62 possibilities
- Choose long passwords
- Avoid actual names or words
- Use a string you can remember
  - Please do not throw sausage pizza away for real = PdN75pa4r
- Use variants for multiple passwords
  - Like above plus concatenate e.g. fab for Facebook (PdN75pa4rfab)
- Change the password regularly
- Don’t write it down
- Don’t tell anyone else

- Don’t use CorrectHorseBatteryStaple ;-)

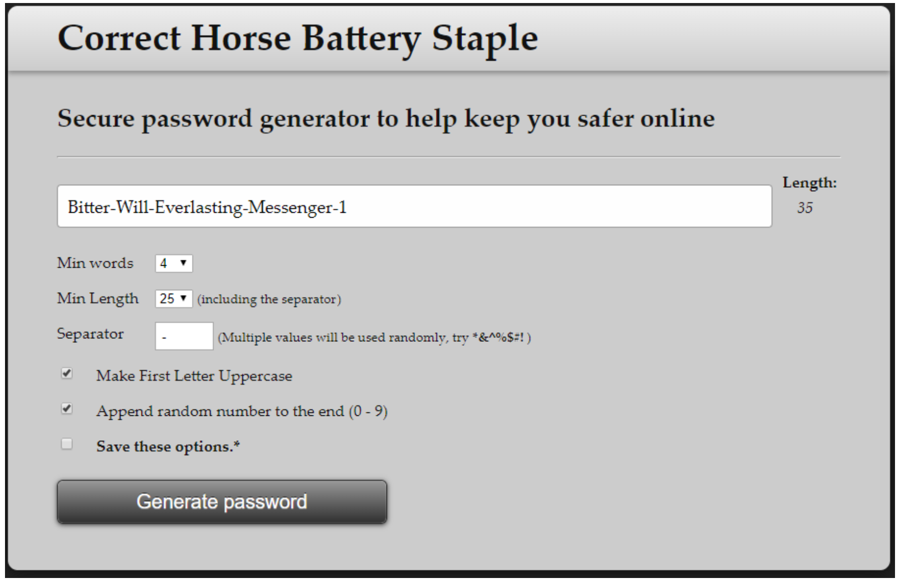

UNCOMMON (NON-GIBBERISH) BASE WORD
ORDER UNKNOWN
Tr0ub4dor&3
CAPS?
COMMON SUBSTITUTIONS
NUMERAL
PUNCTUATION
(YOU CAN ADD A FEW MORE BITS TO ACCOUNT FOR THE FACT THAT THIS IS ONLY ONE OF A FEW COMMON FORMATS.)
~28 BITS OF ENTROPY
2^28 = 3 DAYS AT 1000 GUESSES/SEC
(PLAUSIBLE ATTACK ON A WEAK REMOTE WEB SERVICE. YES, CRACKING A STOLEN HASH IS FASTER, BUT IT'S NOT WHAT THE AVERAGE USER SHOULD WORRY ABOUT.)
DIFFICULTY TO GUESS: EASY
WAS IT TROMBONE? NO, TROUBADOR. AND ONE OF THE Os WAS A ZERO?
AND THERE WAS SOME SYMBOL...
DIFFICULTY TO REMEMBER: HARD
correct horse battery staple
FOUR RANDOM COMMON WORDS
~44 BITS OF ENTROPY
2^44 = 550 YEARS AT 1000 GUESSES/SEC
DIFFICULTY TO GUESS: HARD
THAT'S A BATTERY STAPLE.
CORRECT!
DIFFICULTY TO REMEMBER: YOU'VE ALREADY MEMORIZED IT

THROUGH 20 YEARS OF EFFORT, WE'VE SUCCESSFULLY TRAINED EVERYONE TO USE PASSWORDS THAT ARE HARD FOR HUMANS TO REMEMBER, BUT EASY FOR COMPUTERS TO GUESS.

https://xkcd.com/936/

How to choose a good password

- Need to be hard to guess (dictionary, rainbow tables…)
- Should be easy to remember (otherwise shoulder-surfing, social engineering…)
- Reasonable length
- Bit random
- Not used everywhere
- Password managers are helpful
    - Use as few, good passwords as possible, and let the manager generate different passwords for different services



Correct Horse Battery Staple

Secure password generator to help keep you safer online

Bitter-Will-Everlasting-Messenger-1

Length: *35*

Min words 4 ▾

Min Length 25 ▾ (including the separator)

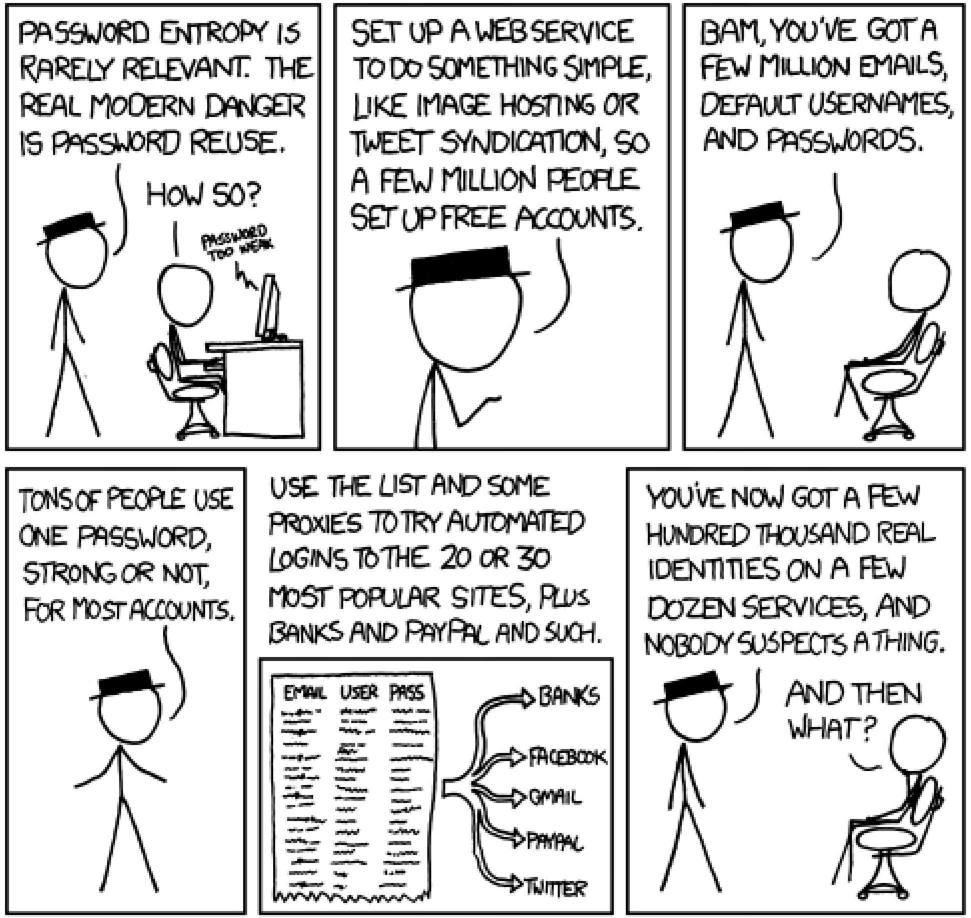
Separator - (Multiple values will be used randomly, try *&^%$#!)

☑ Make First Letter Uppercase

☑ Append random number to the end (0 - 9)

□ **Save these options.***

https://correcthorsebatterystaple.net/


PASSWORD ENTROPY IS RARELY RELEVANT. THE REAL MODERN DANGER IS PASSWORD REUSE.
HOW SO?
PASSWORD TOO WEAK
SET UP A WEB SERVICE TO DO SOMETHING SIMPLE, LIKE IMAGE HOSTING OR TWEET SYNDICATION, SO A FEW MILLION PEOPLE SET UP FREE ACCOUNTS.
BAM, YOU'VE GOT A FEW MILLION EMAILS, DEFAULT USERNAMES, AND PASSWORDS.
TONS OF PEOPLE USE ONE PASSWORD, STRONG OR NOT, FOR MOST ACCOUNTS.
USE THE LIST AND SOME PROXIES TO TRY AUTOMATED LOGINS TO THE 20 OR 30 MOST POPULAR SITES, PLUS BANKS AND PAYPAL AND SUCH.
EMAIL
USER
PASS
BANKS
FACEBOOK
GMAIL
PAYPAL
TWITTER
YOU'VE NOW GOT A FEW HUNDRED THOUSAND REAL IDENTITIES ON A FEW DOZEN SERVICES, AND NOBODY SUSPECTS A THING.
AND THEN WHAT?

Part of https://xkcd.com/792/

Something the user *is* - biometrics

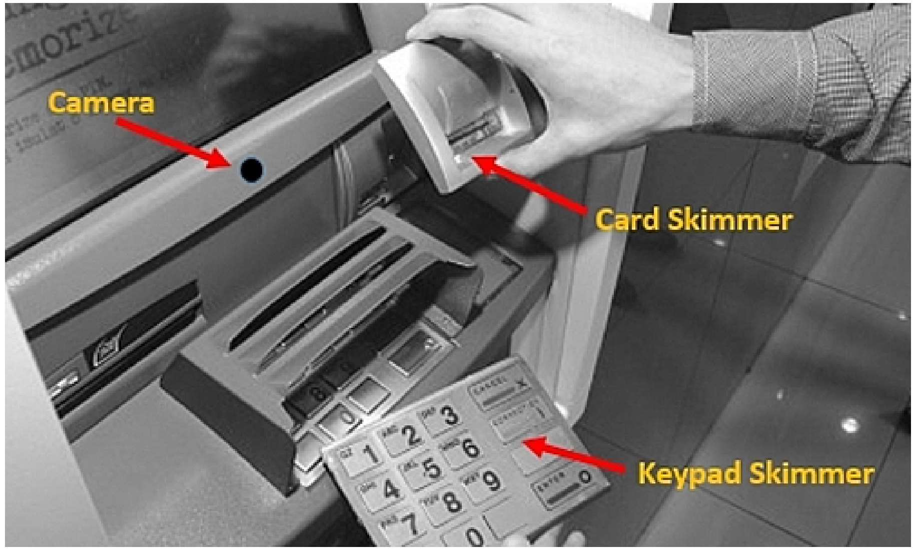
- Many different techniques:
  - Fingerprint
  - Hand geometry (shape and size of fingers)
  - Retina and iris (parts of the eye)
  - Voice
  - Handwriting, signature, hand motion
  - Typing characteristics
  - Blood vessels in the finger or hand
  - Face
  - Facial features, such as nose shape or eye spacing

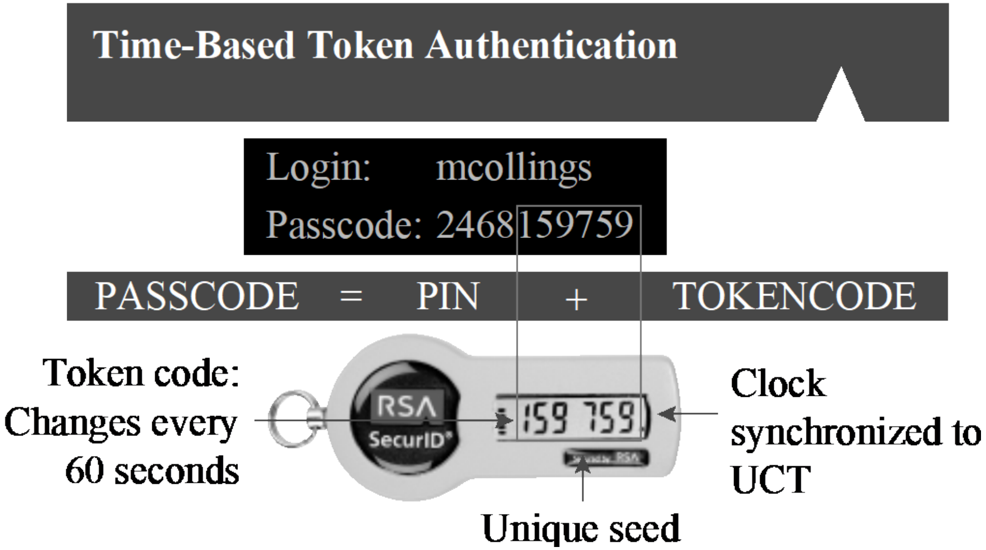
- Fairly new technologies
  - Some find them intrusive
  - Some are expensive
  - Single point of failure
  - Sampling error
  - False readings
  - Speed
    - Need to be accurate but not slow
  - Forgery
    - E.g. fingerprints made by gelatin

Something the user *has*

- Active and passive tokens

- Static and dynamic tokens




Time-Based Token Authentication

Login: mcollings
Passcode: 2468159759

PASSCODE = PIN + TOKENCODE

Token code: Changes every 60 seconds

Clock synchronized to UCT

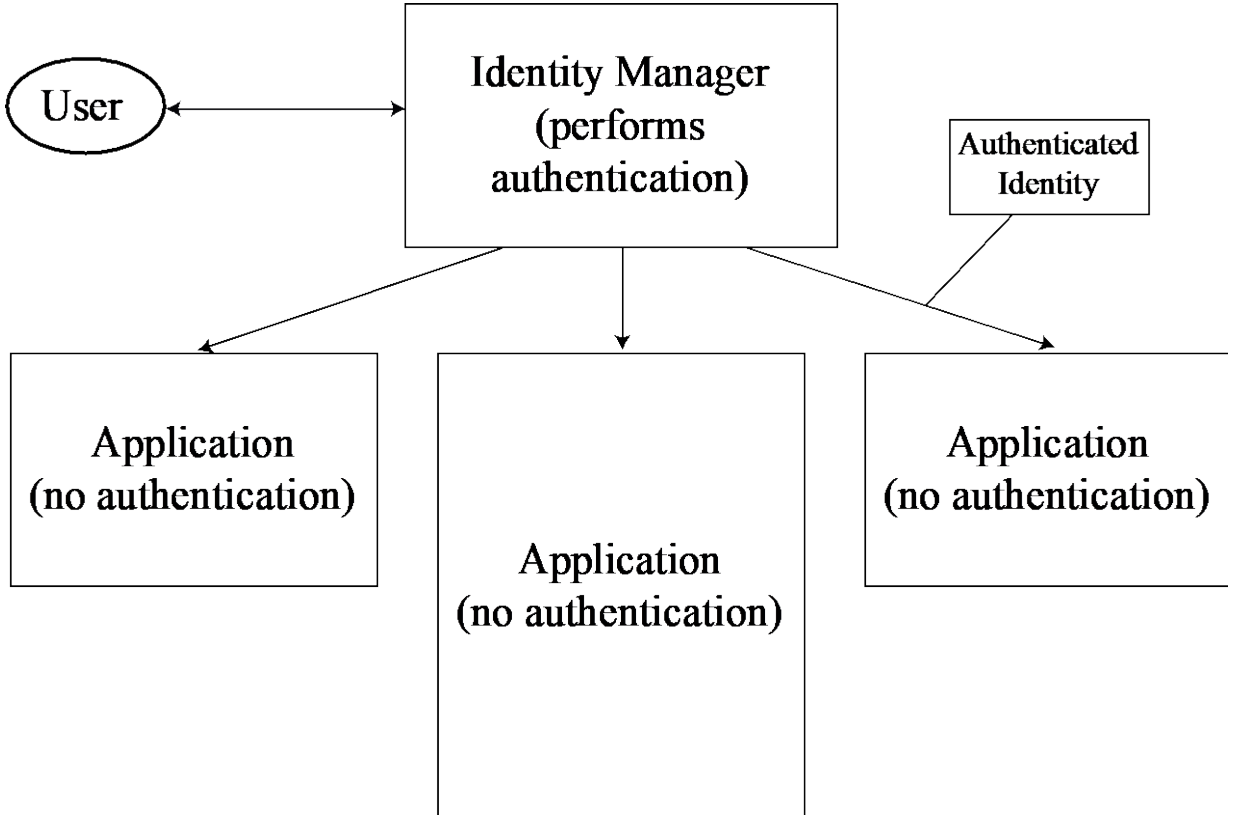
Unique seed

Identity management

- Complicated to keep track of all identities (for users and staff)
- Users use several systems at the same time –> many authentications
- Distributed, heterogeneous domain that needs authentication within an organization
- Solutions include:
  - Federated identity manager
  - Single sign-on

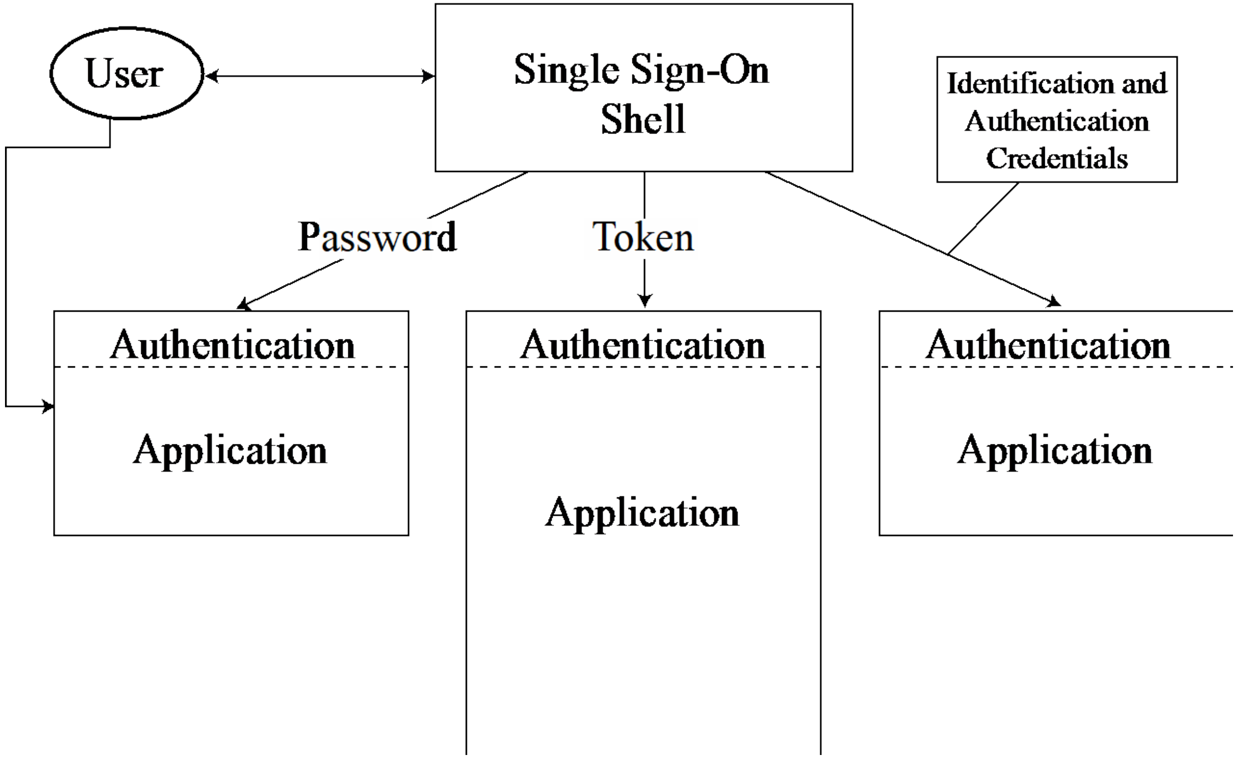
Federated identity manager (FIdM/FIM)

- One profile is used
- Unifies the identification and authentication process for a group of systems
- Authentication is performed in one place
- Systems share access to the central authentication database


User
Identity Manager (performs authentication)
Authenticated Identity
Application (no authentication)
Application (no authentication)
Application (no authentication)

Single sign-on (SSO)

- Single sign-on lets a user log on once per session
  - But access to many different applications/systems
- Often works in conjunction with federated identity management
  - SSO is a subset of FIdM
  - The federated identity provider acts as the source of authentication for all the applications



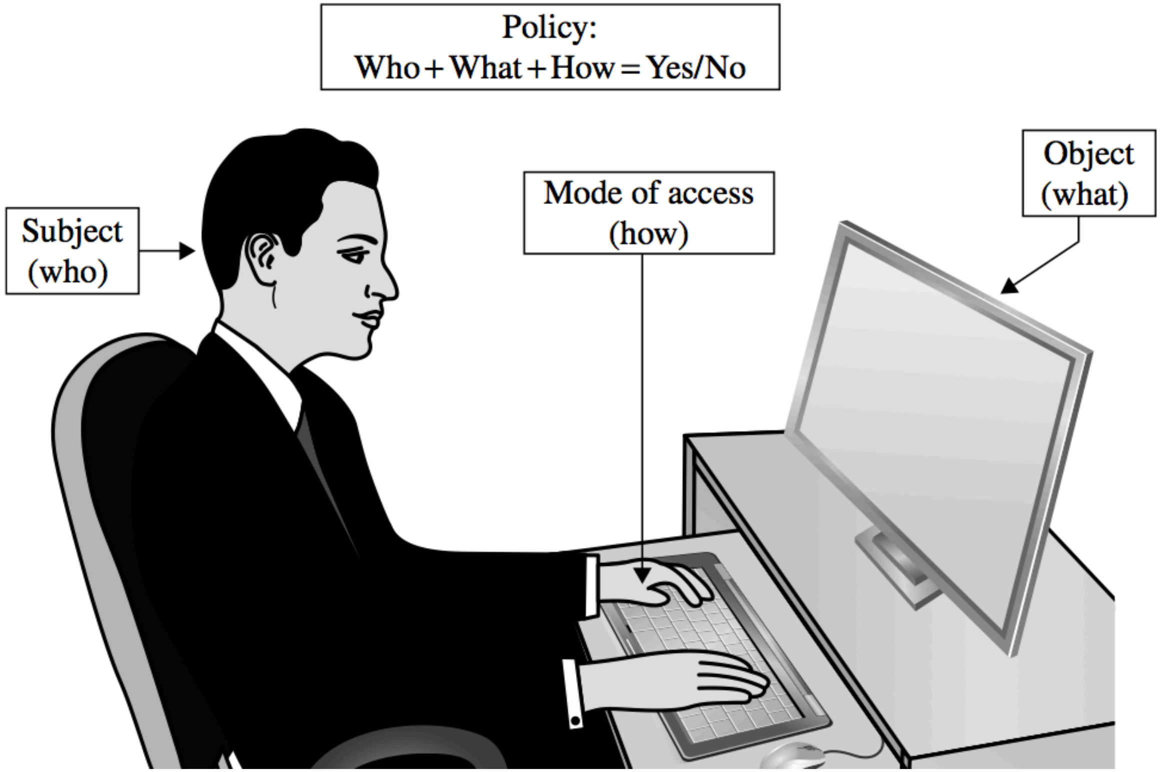
Access control

- Access control: limiting who can access what, and in what ways

- Access control has two components:
  - Authentication
  - Authorization
  - (Sometimes also audit/accounting)

Access control

- A subject is permitted to access an object in a particular mode, and only such authorized accesses are allowed
  - Subjects
  - Objects
  - Access modes



Access policies

- Goals:
  - Check every access
  - Enforce least privilege
  - Verify acceptable usage

Access policies

- Track users’ access
- Enforce at appropriate granularity
- Use audit logging to track accesses
- How do we implement access control? More soon =)

JÖNKÖPING UNIVERSITY

School of Engineering