3. Authentication

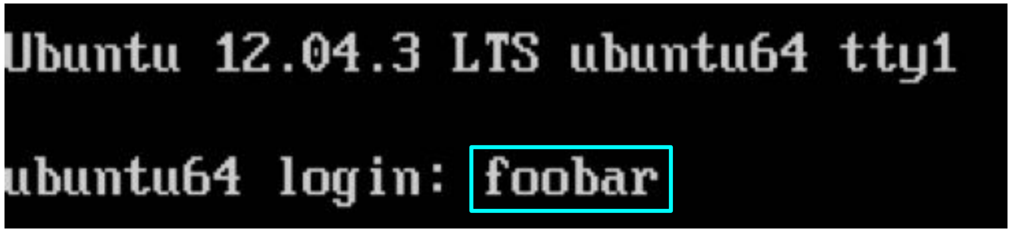
Computer Security Courses @ POLIMI

Prof. Carminati & Prof. Zanero

Identification *vs.* Authentication

Identification: an entity declares its identifier

- **Examples:** *"I am Stefano"*, *"I am Michele"*

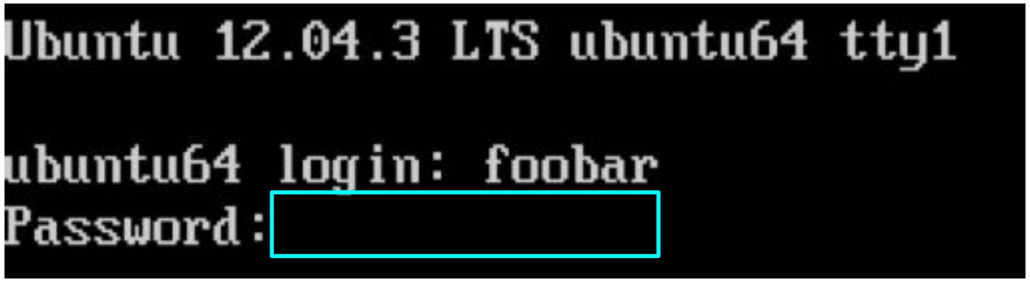


```
Ubuntu 12.04.3 LTS ubuntu64 tty1

ubuntu64 login: foobar
```

Authentication: the *entity* provides a *proof* that verifies its identity.

- **Examples:** *"Here is Stefano's ID card"*

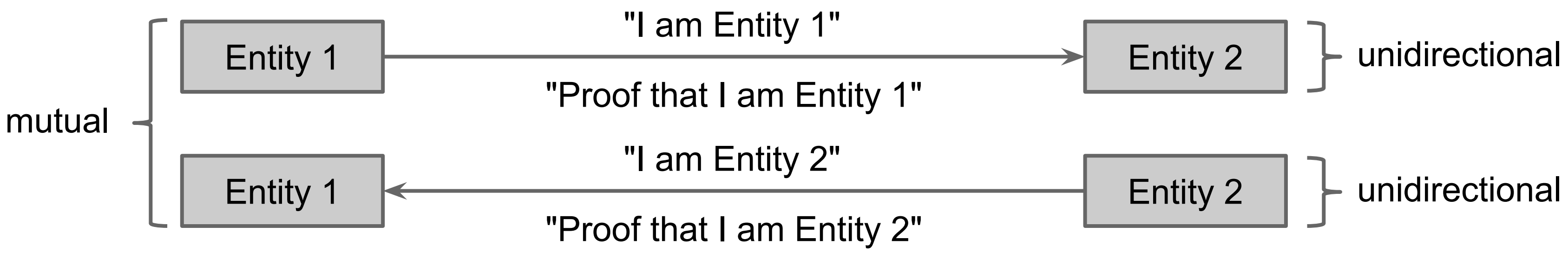


```
Ubuntu 12.04.3 LTS ubuntu64 tty1

ubuntu64 login: foobar
Password:
```

Authentication

Can be *unidirectional* or *bidirectional (mutual)*.



Can happen between any entity:

- Human to human
- Human to computer
- Computer to computer

Foundation for the subsequent *authorization* phase (see next class).

Three Factors of Authentication

Something that the entity ***knows*** (to know)

- **Example:** password, PIN, secret handshake.

Something that the entity ***has*** (to have)

- **Example:** Door key, smart card, token.

Something that the entity ***is*** (to be)

- **Example:** Face, voice, fingerprints.

Humans: (3) more used than (2), more used than (1).

Machines: (1) more used than (2), more used than (3).

Multi-factor authentication uses two or three factors.

The "to know" Factor

Passwords and PINs

The *"to know"* Factor: Passwords

Advantages

- Low cost,
- ease of deployment,
- low technical barrier.

Disadvantages

Secrets can be

a. stolen/snooped
b. guessed
c. cracked (enumerated)

Countermeasures *(i.e., costs)*

Enforce passwords that

- *change/expire* frequently
- are *long* and have a *rich character set*
- are *not related to the user*

Estimate the most likely attack in the scenario. ⟺ Accordingly choose the countermeasure(s) that are worth asking users to adhere to (remember indirect costs?)

Why are Countermeasures Costs?

Humans are not machines

- Inherently *unable to keep secrets*
- Hard to *remember* complex passwords
- Can’t pick *unlimited* countermeasures
  - how to choose?

Countermeasure guideline: 🟩 important, 🟧 may help, 🟥 unimportant

Against **snooping**	Against **cracking**	Against **guessing**
🟥 *complexity*	🟩 *complexity*	🟧 *complexity*
🟩 *change*	🟧 *change*	🟧 *change*
🟥 *being related to users*	🟥 being related to users	🟩 *not being related to users*

User Education and Password Complexity

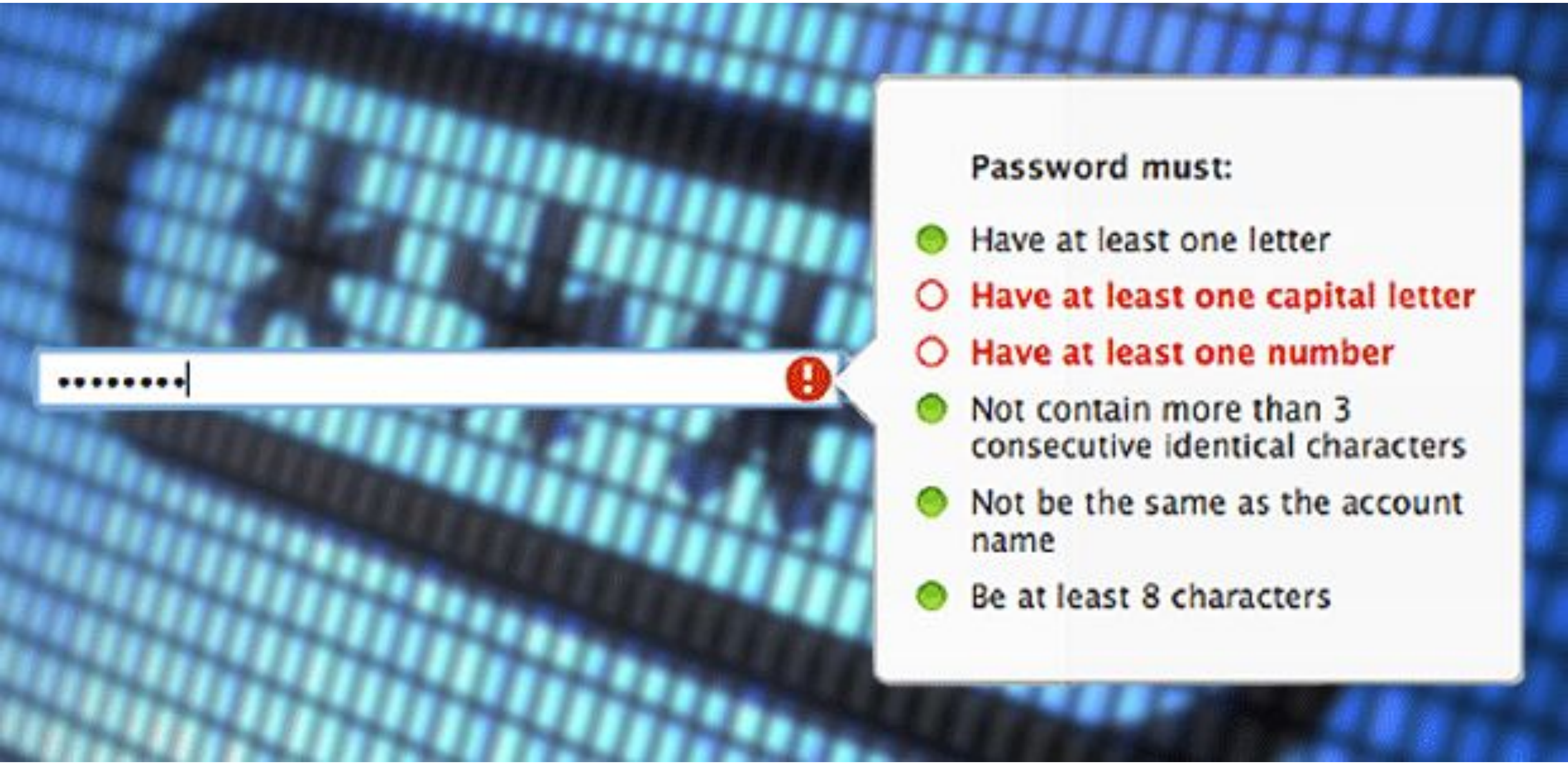
User education: "human" == "weak link".

- enforce *strong* passwords in the process.
- enforce password *expiration/change* policies
- use password meters to balance *usability*.

Password complexity

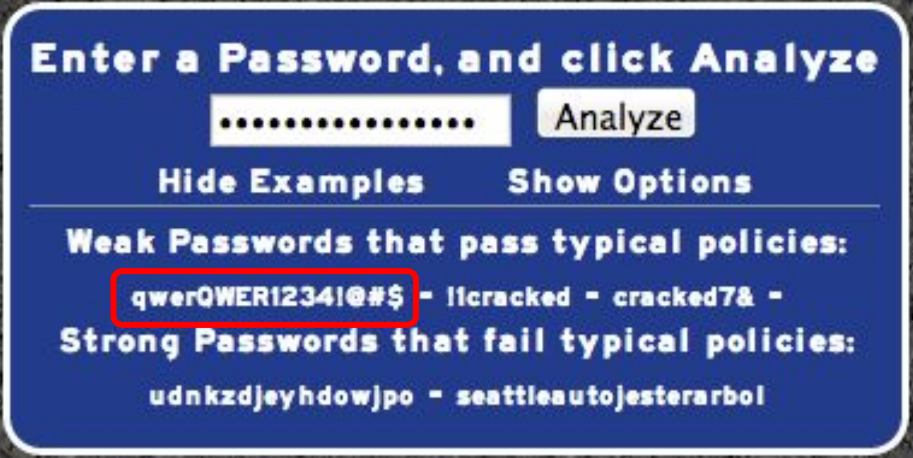
- must h4v3 4 r1ch, ch4r4ct3r, s3t!
- mUsT hAvE a MiXeD cAsE
- muuuuuust beeeeeeeee looooooong enooooogh

Password Meters



Password must:

- ● Have at least one letter
- ○ Have at least one capital letter
- ○ Have at least one number
- ● Not contain more than 3 consecutive identical characters
- ● Not be the same as the account name
- ● Be at least 8 characters



Enter a Password, and click Analyze

••••••••••••••••• Analyze

Hide Examples   Show Options

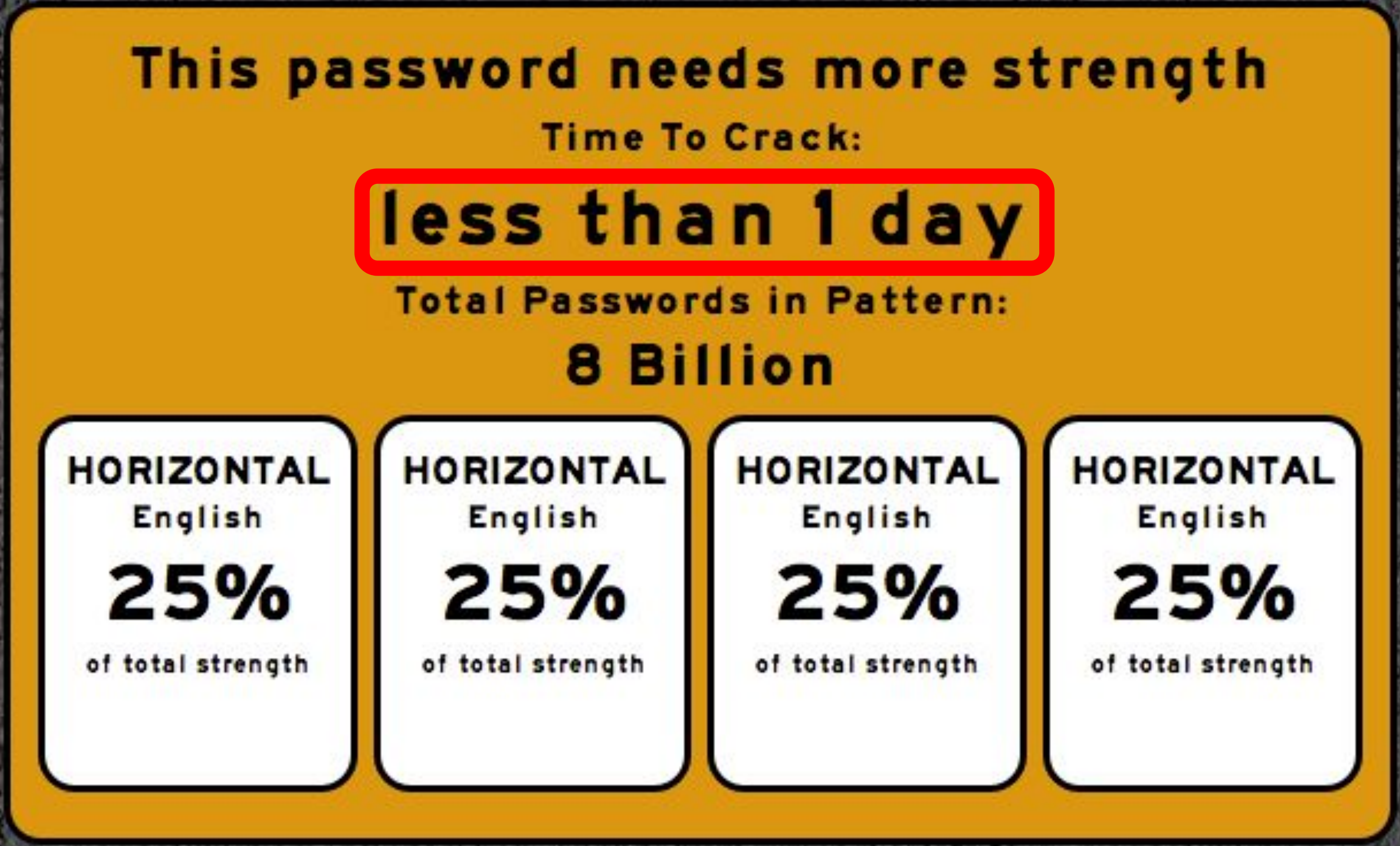
Weak Passwords that pass typical policies:

qwerQWER1234!@#$ - !!cracked - cracked7& -

Strong Passwords that fail typical policies:

udnkzdjeyhdowjpo - seattleautojesterarbol

http://passfault.com/



This password needs more strength

Time To Crack:

less than 1 day

Total Passwords in Pattern:

8 Billion

HORIZONTAL English 25% of total strength	HORIZONTAL English 25% of total strength	HORIZONTAL English 25% of total strength	HORIZONTAL English 25% of total strength

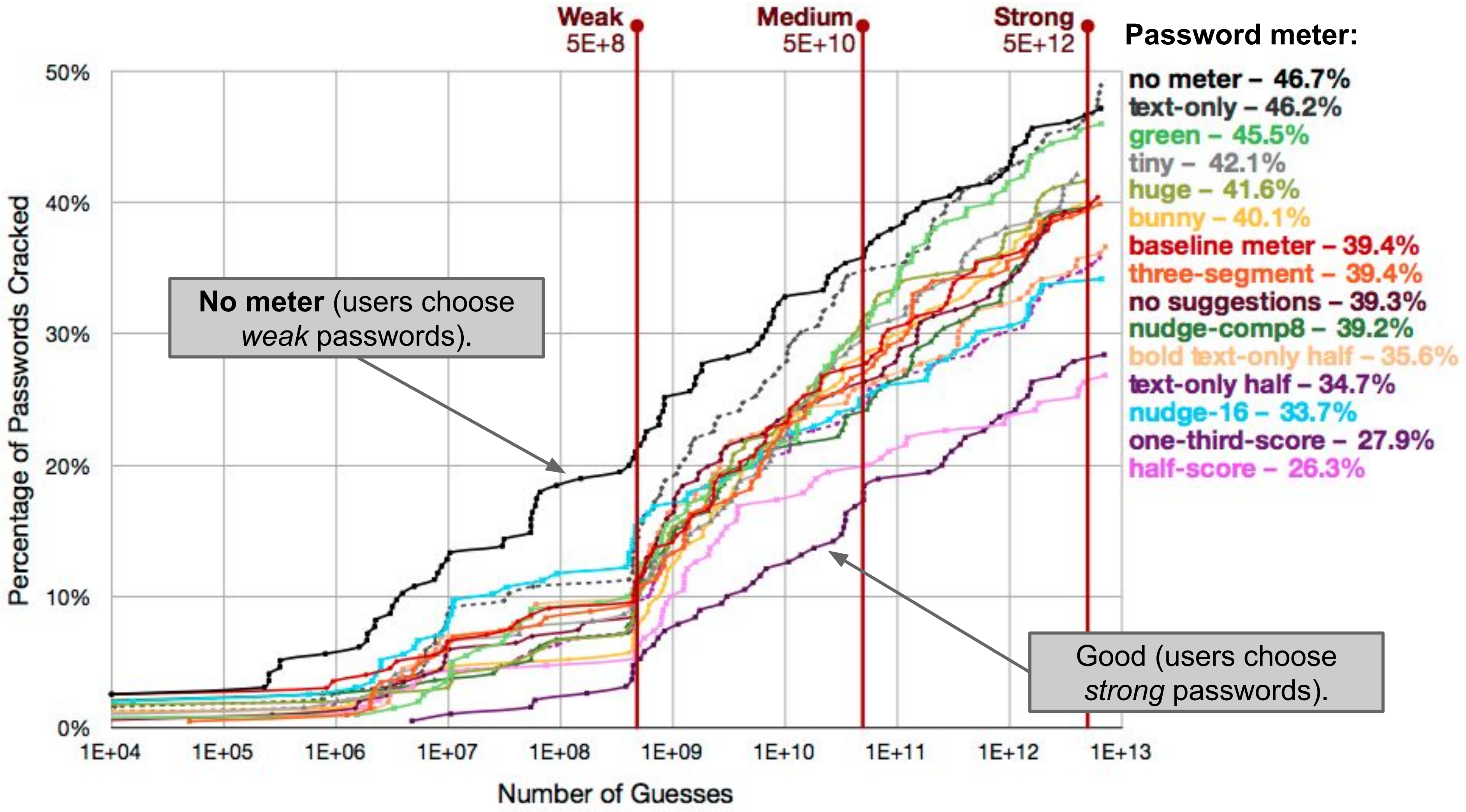


Figure 3: This graph contrasts the percentage of passwords that were cracked in each condition. The x-axis, which is logarithmically scaled, indicates the number of guesses made by an adversary, as described in Section 2.4. The y-axis indicates the percentage of passwords in that condition cracked by that particular guess number.

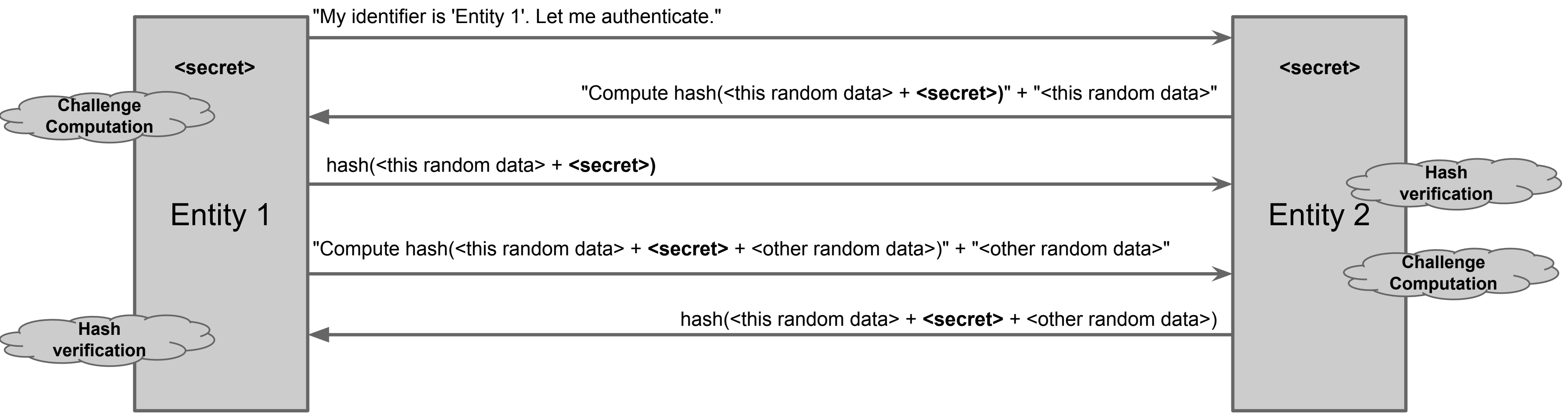
B. Ur, P.G. Kelley, S. Komanduri, J. Lee, M. Maass, M. Mazurek, T. Passaro, R. Shay, T. Vidas, L. Bauer, N. Christin, and L.F. Cranor. How does your password measure up? The effect of strength meters on password creation. USENIX Security 2012.

Secure Password Exchange

Authentication is about sharing a *secret*.

How to minimize the risk that secrets get *stolen*?

- use *mutual authentication* if possible
- use a ***challenge-response*** scheme


Entity 1
<secret>
Challenge Computation
Hash verification
Entity 2
<secret>
Hash verification
Challenge Computation
"My identifier is 'Entity 1'. Let me authenticate."
"Compute hash(<this random data> + **<secret>)**" + "<this random data>"
hash(<this random data> + **<secret>)**
"Compute hash(<this random data> + **<secret>** + <other random data>)" + "<other random data>"
hash(<this random data> + **<secret>** + <other random data>)

- use random data to avoid *replay attacks*

Secure Password Storage

OS store a file with usernames and passwords. An attacker could try to compromise the confidentiality and integrity of this password file

How to minimize the risk that secrets get *stolen*?

- Cryptographic protection
  - Never store passwords in clear (see next classes: encryption + salting to mitigate dictionary attacks)
- Access control policies (privileges to w/r)
- Never disclose secrets in password-recovery schemes.

Caching problem

The "to have" Factor

Tokens, smart cards, smart phones.

The *"to have"* Factor

User must prove that it *possesses* something.

Advantages

- Human factor (less likely to hand out a key),
- relatively low cost,
- good level of security.

Disadvantages

- Hard to deploy,
- can be lost or stolen.

Countermeasures

- none
- use with second factor.

Example Classic Technologies

One-time password generators:

- *Private key* + *counter* synchronized with the host.
- Client: password = Encrypt(counter, private key).
- Host: counter = Decrypt(password, public key).
- Check that the counter is the expected one.
- The counter changes every 30–60 seconds.

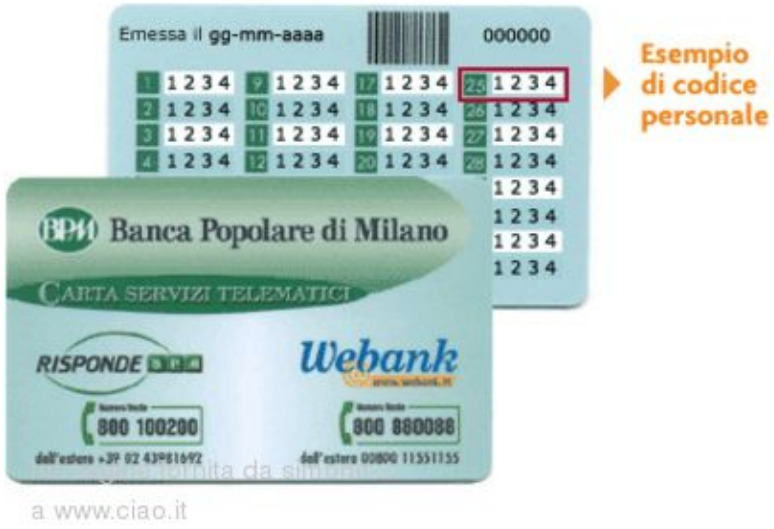
Application examples: online banking, admin console (e.g., Amazon AWS).

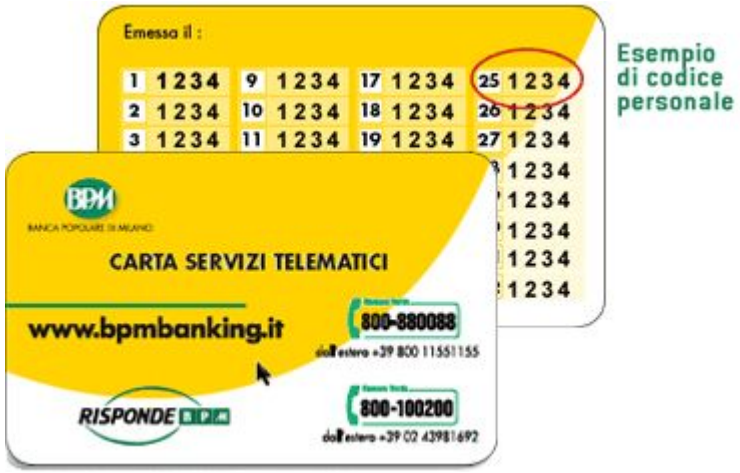
Smart cards (or USB keys)

- CPU + non-volatile RAM with a *private key*.
- The smart card authenticates itself to host via a ***challenge-response*** protocol.
- Uses the *private key* to encrypt the challenge.
- The *private key* does not leave the device.
- Should be **tamper proof** to some extent.

Application examples: credit cards (+PIN).

Dyno Technologies





Static Password Lists:

- Known to both client and host.
- Host chooses a challenge: random numbers (e.g., "second digit of the 14th cell").
- The client transmits the response (hopefully, over an encrypted channel).
- The host should not keep the list in clear (e.g., hashing).

Modern Technologies

Software that implement the same functionality of password generators:

- Key difference
  - password generators are *closed, embedded* systems.
  - password-generation *apps* work on *general-purpose* sw/hw platforms.
- What if the device is infected by a malicious app: Dmitrienko et al., When More Becomes Less: On the (In)Security of Mobile Two-Factor Authentication, FC 2014

The "to be" Factor.

Biometric authentication.

The *"to be"* Factor: Biometric.

User must prove that it has some specific *characteristics*.

Advantages

- high level of security,
- requires no extra hardware to carry around.

Disadvantages	**Countermeasures**
● Hard to deploy,	● none
● non-deterministic matching,	● none
● invasive measurement,	● none
● can be cloned,	● none (see next slides)
● bio-characteristics change,	● re-measure often,
● privacy sensitivity,	● secure the process,
● users with disabilities.	● need alternate (weaker?)

Technology examples

Extract the characteristics (i.e., features) of:

- Fingerprints (http://www.freedesktop.org/wiki/Software/fprint/)
- Face geometry (https://code.google.com/p/pam-face-authentication/)
- Hand geometry (palm print)
- Retina scan
- Iris scan
- Voice analysis
- DNA
- Typing dynamics (http://flyer.sis.smu.edu.sg/ndss13-tey.pdf)

Example: Fingerprint

- *Enrolment*: reference sample of the user’s fingerprint is acquired at a fingerprint reader.
- Features are derived from the sample.
  - *Fingerprint minutiae*: end points of ridges, bifurcation points, core, delta, loops, whorls, ?
- For higher accuracy, record features for more than one finger and different positions.
- Feature vectors are stored in a secure database.
- When the user logs on, a new reading of the fingerprint is taken; features are compared against (similarity) the reference features. User is accepted if match is above a predefined threshold.

Main issue: false positives and false negatives

Consumer-level Biometric Auth

http://cryptome.org/fake-prints.htm

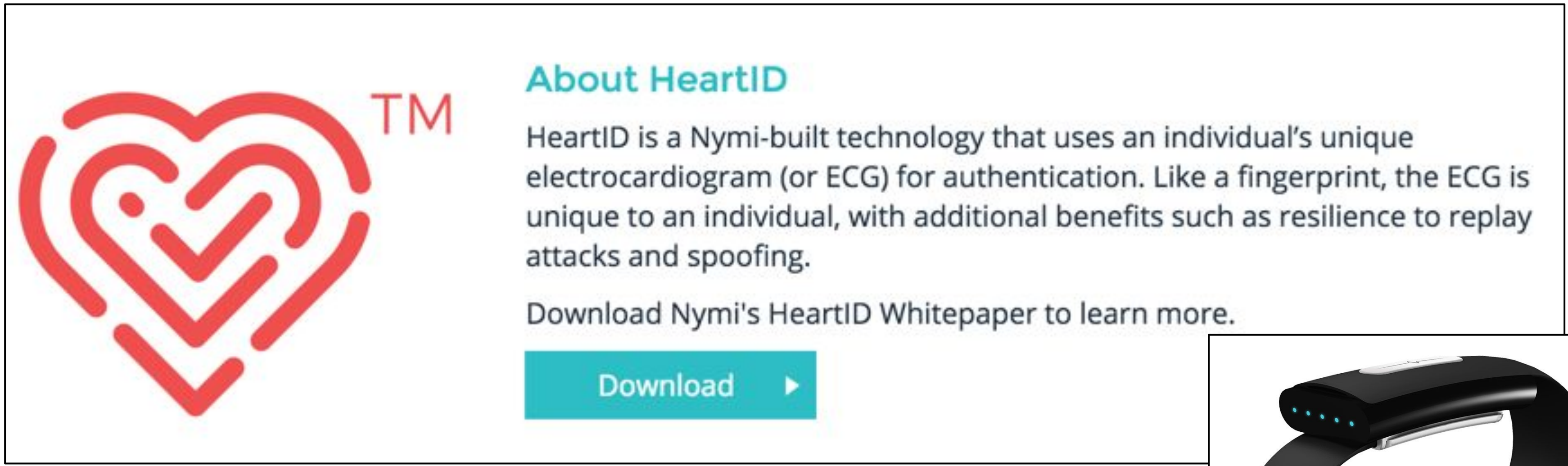
Manufacturer	Model	Technology	Date	Difficulty
Identix	TS-520	Optical	Nov. 1990	First attempt
Fingermatrix	Chekone	Optical	Mar. 1994	Second attempt
Dermalog	DemalogKey	Optical	Feb.1996	First attempt
STMicroelectronics	TouchChip	Solid state	Mar. 1999	First attempt
Veridicon	FPS110	Solid state	Sept.1999	First attempt
Identicator	DFR200	Optical	Oct. 1999	First attempt

20 september 2013 RELEASED

21 september 2013 CRACKED
http://www.ccc.de/en/updates/2013/ccc-breaks-apple-touchid

Consumer-level Biometric Auth

- how unique your heartbeat is?
    - Nymi believes it's unique enough for authentication



About HeartID

HeartID is a Nymi-built technology that uses an individual's unique electrocardiogram (or ECG) for authentication. Like a fingerprint, the ECG is unique to an individual, with additional benefits such as resilience to replay attacks and spoofing.

Download Nymi's HeartID Whitepaper to learn more.

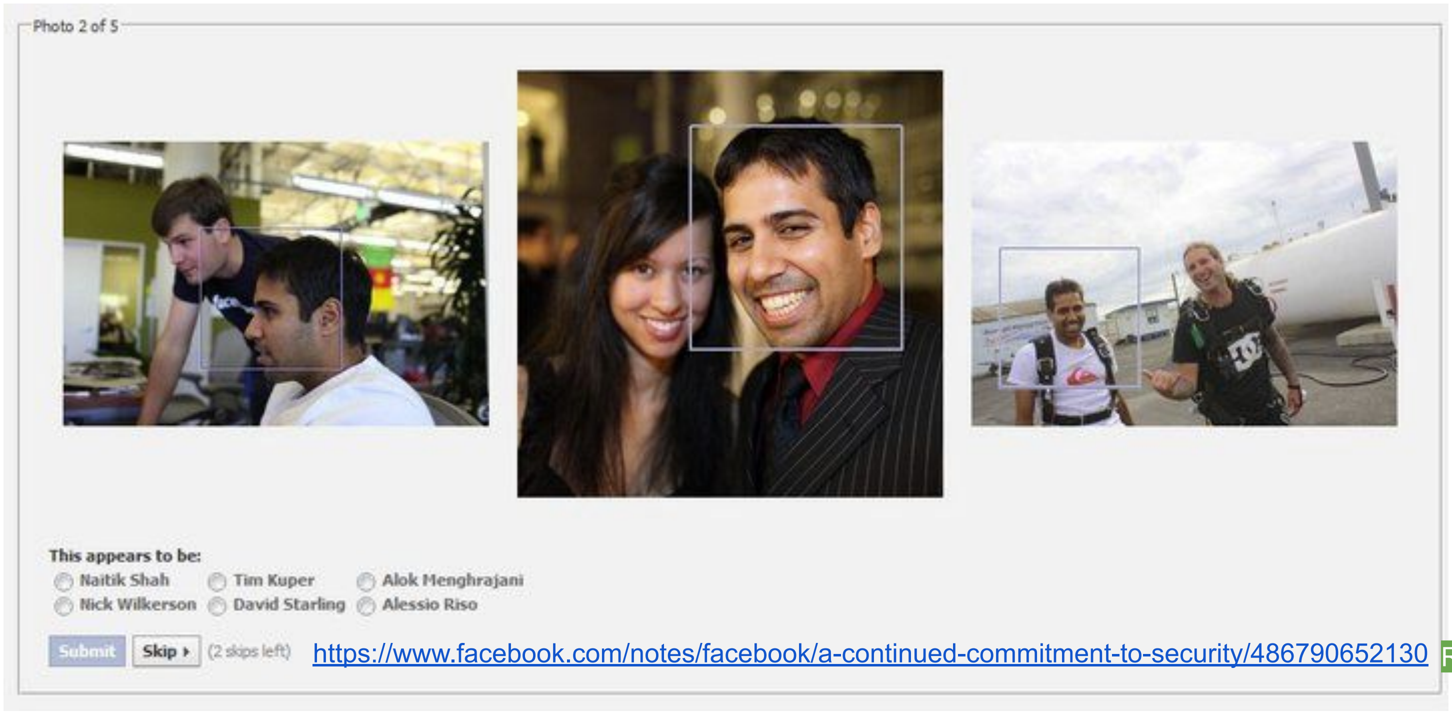
More details at https://nymi.com/product_overview

The "social" Factor

Experimental factors of authentication.

The *"social"* Factor: Who you know.

Alice must prove that she *knows someone.*



https://www.facebook.com/notes/facebook/a-continued-commitment-to-security/486790652130 RELEASED

Papers

- H. Kim, J. Tang, and R. Anderson. *Social authentication: harder than it looks.* In Proceedings of the 2012 Financial Cryptography and Data Security conference.

CRACKED J. Polakis, M. Lancini, G. Kontaxis, F. Maggi, S. Ioannidis, A. Keromytis, S. Zanero, *All Your Face Are Belong to Us: Breaking Facebook's Social Authentication.* In Proceedings of 2012 Annual Computer Security Applications Conference.

Single Sign On

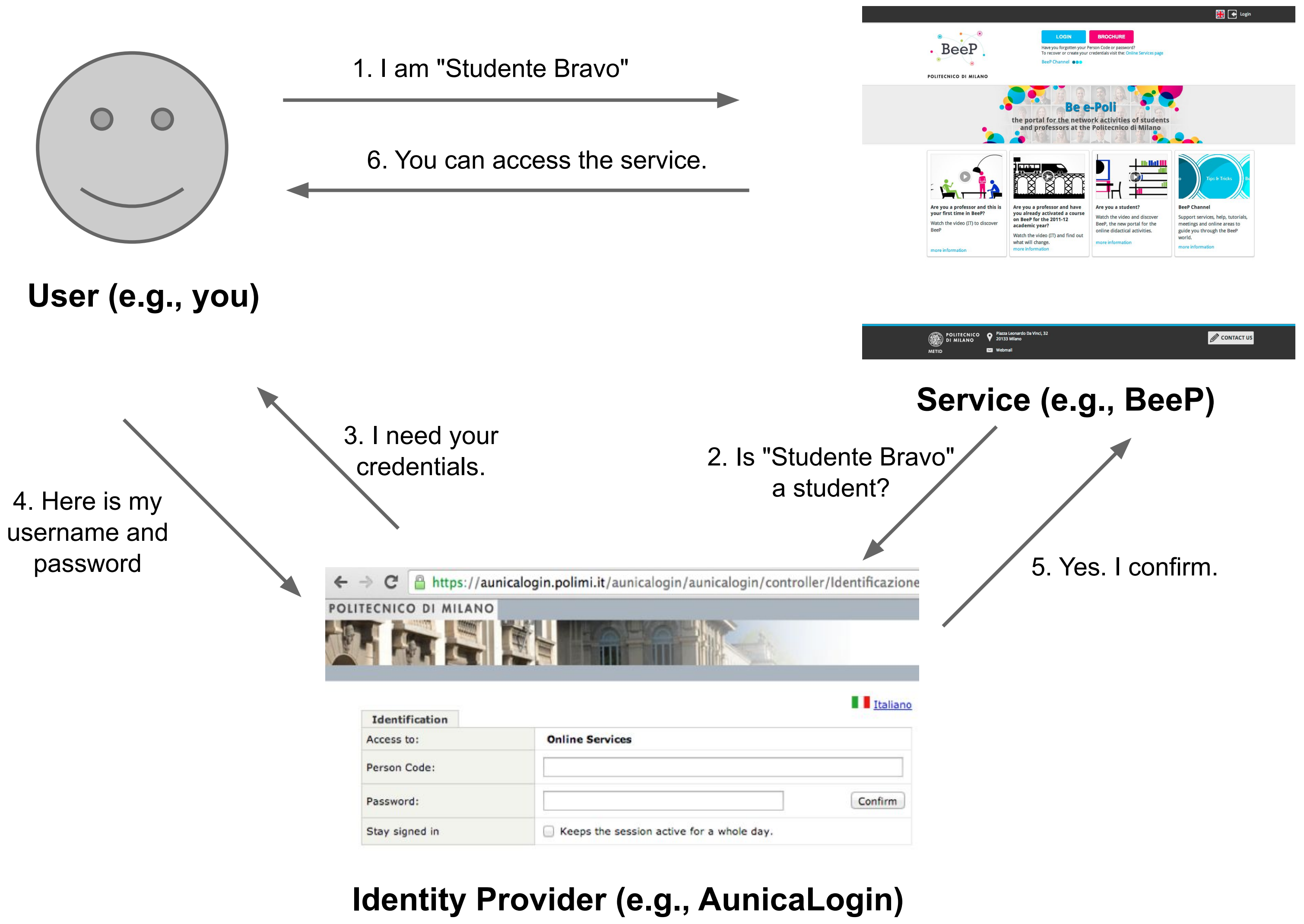
Problem: managing and remembering multiple passwords is complex.

- Users re-use passwords over multiple sites,
- Password policies replicated ($$$).

Solution: 1 identity, 1-2 auth. factors, 1 trusted host.

- elect a trusted host,
- users authenticate (sign on) on the trusted host,
- other hosts ask the trusted host if a user is authenticated.

Example: Shibboleth (AunicaLogin)


1. I am "Studente Bravo"
6. You can access the service.
User (e.g., you)
Service (e.g., BeeP)
BeeP
LOGIN
BROCHURE
Be e-Poli
the portal for the network activities of students and professors at the Politecnico di Milano
CONTACT US
3. I need your credentials.
2. Is "Studente Bravo" a student?
4. Here is my username and password
5. Yes. I confirm.
Identity Provider (e.g., AunicaLogin)

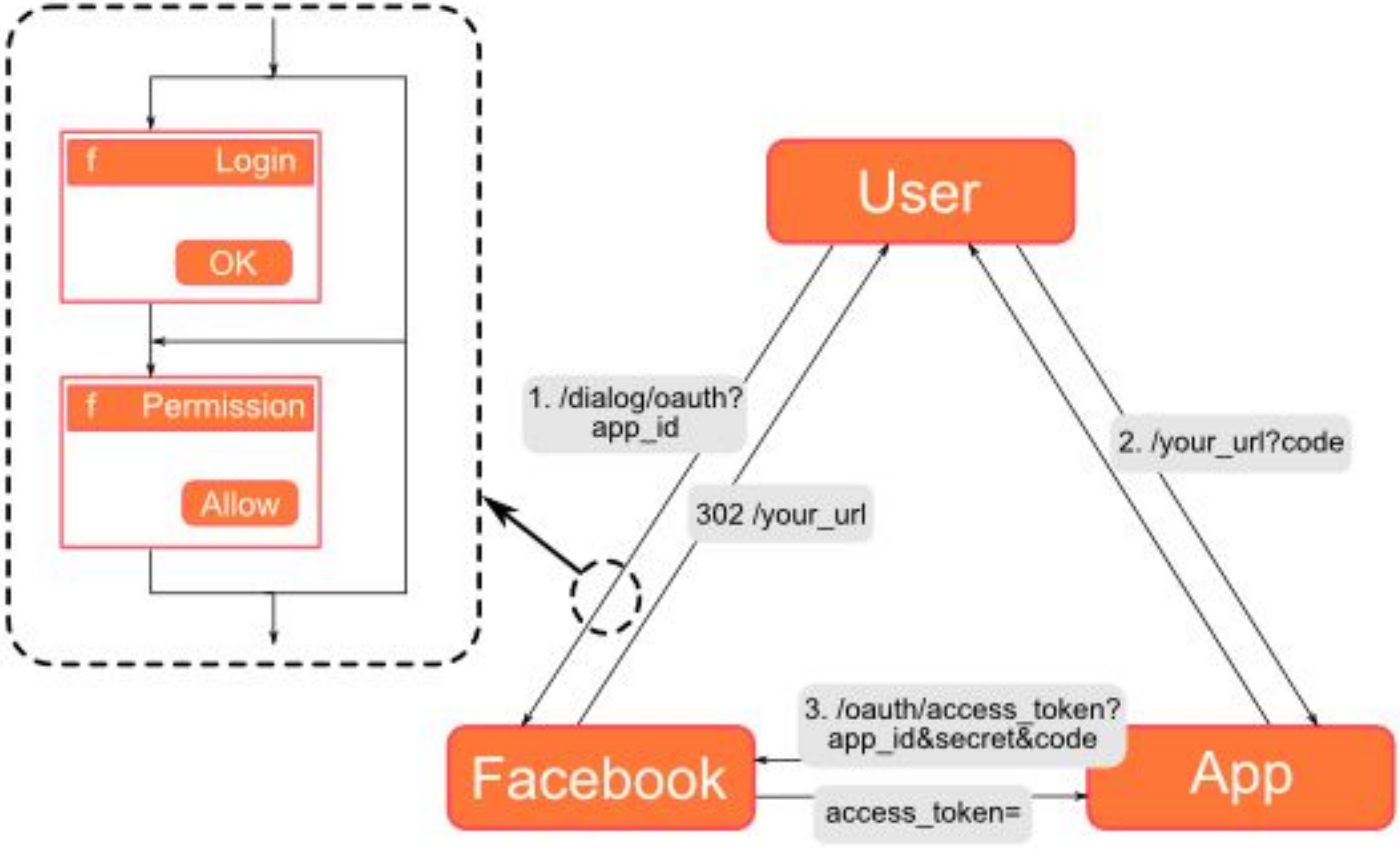
https://aunicalogin.polimi.it/aunicalogin/aunicalogin/controller/Identificazione

POLITECNICO DI MILANO

Italiano

Identification	
Access to:	Online Services
Person Code:	
Password:	Confirm
Stay signed in	Keeps the session active for a whole day.

Example: OAuth2 Flow (Facebook)



Single Sign On Insecurities

Single point of *trust*: the trusted server.

- If compromised, all sites are compromised.
- Password reset scheme must be bulletproof.
  - Email is the trusted element

    Kontaxis G. et al., *SAuth: Protecting User Accounts from Password Database Leaks*. In Proceedings of the 20th ACM Conference on Computer and Communications Security (CCS), 2013.

Difficult to get it right for the developers.

- The flow is complex to implement.
- Libraries exist, but they can be bugged.

http://homakov.blogspot.it/2014/02/how-i-hacked-github-again.html

Conclusions

Identification, *authentication* and *authorization* are three distinct, yet inter-dependent, concepts.

There are *three* types of authentication *factors*, which should be used in *combination*.

Passwords are increasingly showing their limits.

New authentication schemes are promising, but should be used with care.