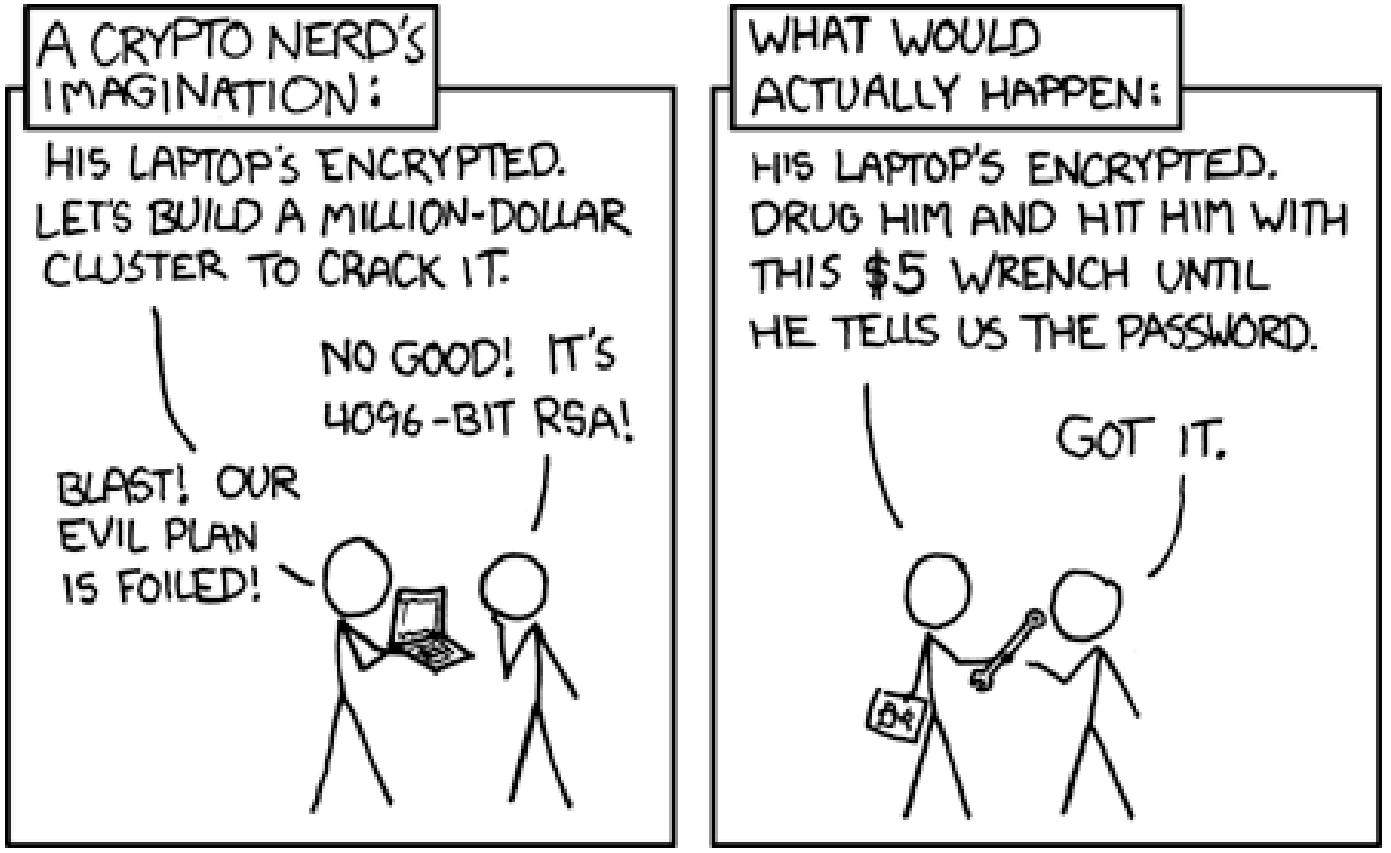
Computer Security and Privacy (CS642)

User Authentication

Earlence Fernandes

earlence@cs.wisc.edu



Admin

- Homework 1 is out: start doing it
- Talk to TAs to figure out any setup issues
- You may discuss high-level ideas but DON'T discuss solutions
  - Piazza is NOT a place to ask for direct solutions to the homework
  - Piazza is NOT a place to answer homework questions with direct solutions
  - Homework is an individual exercise (unless I tell you otherwise)

- Homework 1 is due Feb 13th at 11am
- See the website; It has everything you need

Human authentication

Prove to a computer who you are

What you know

Passwords, PINs, passphrases, life questions

What you are

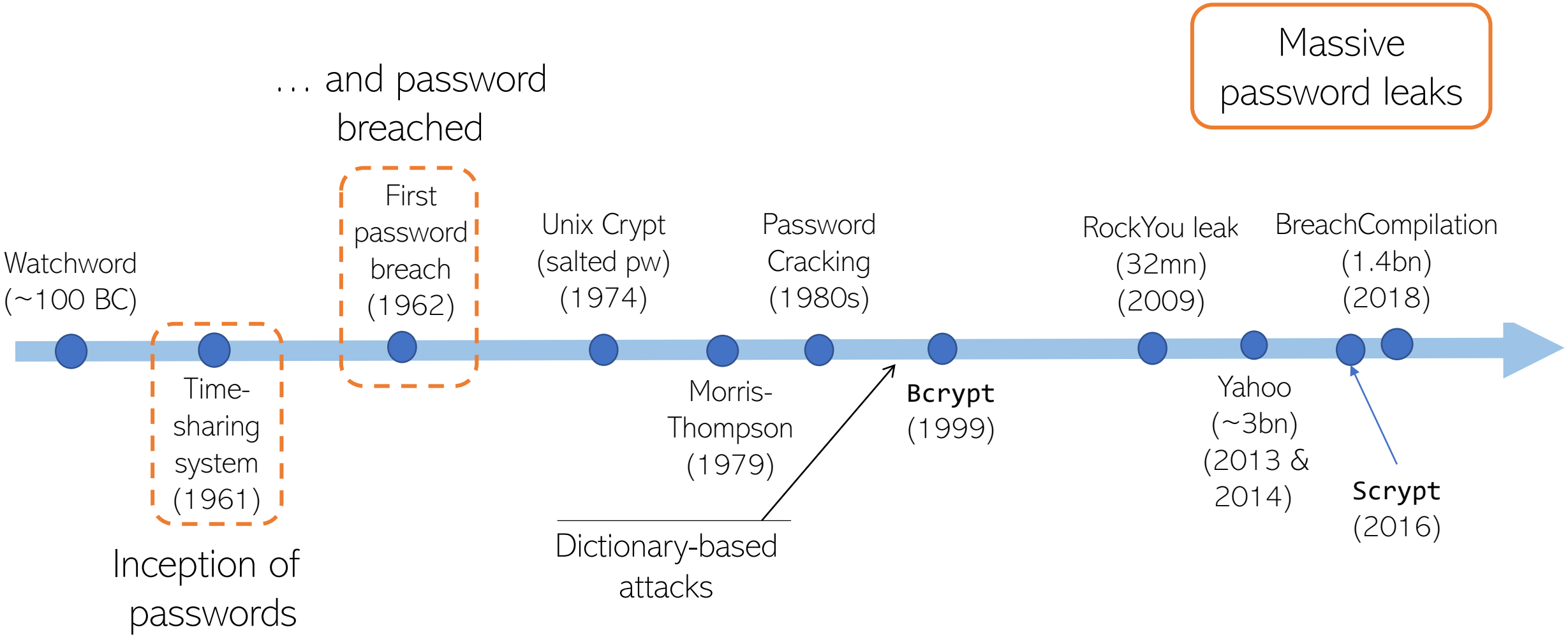
Fingerprints, iris scan, hand geometry, heartbeat

What you have

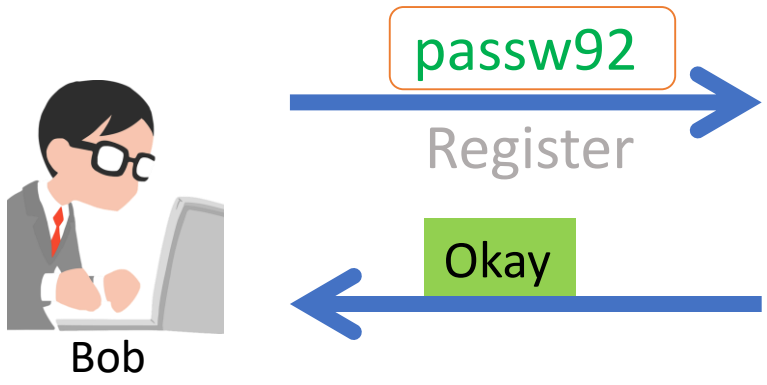
Smartcard, email, mobile phone, RSA keys,

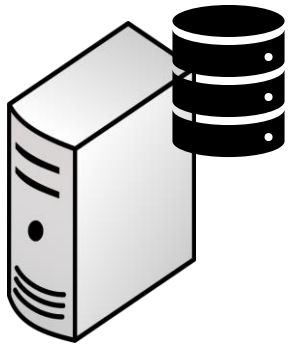
Passwords

A brief history of passwords



Passwords-based authentication (PBA)





Username	Password
Bob	passw92

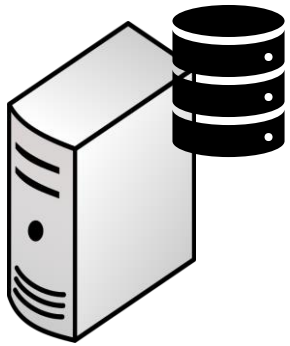
Passwords-based authentication (PBA)

Bob

passw92

Login

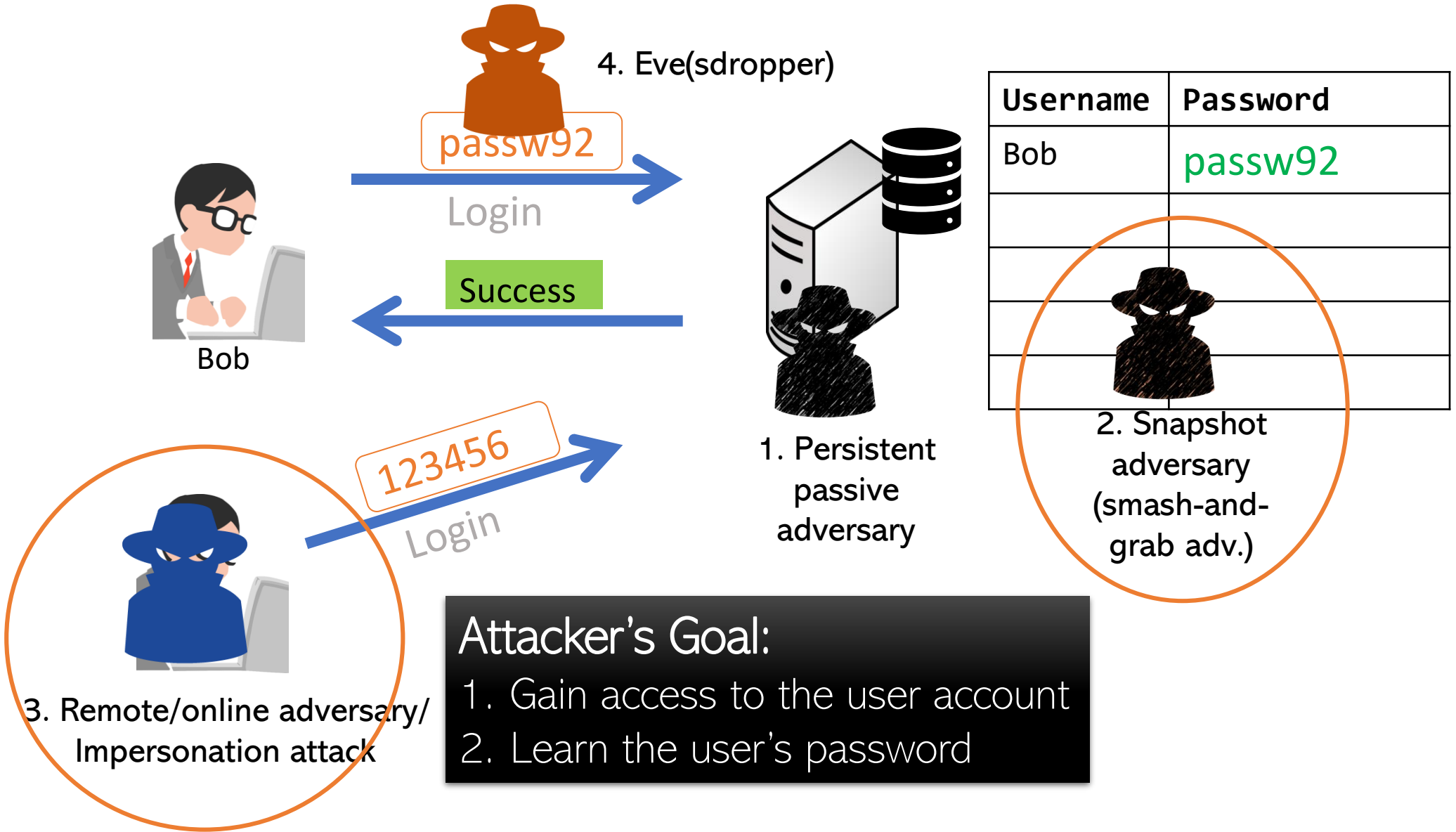
Success



Username	Password
Bob	passw92

passw92 $\stackrel{?}{=}$ passw92

Threats against PBA systems



Attacker's Goal:

1. Gain access to the user account
2. Learn the user's password

Remote guessing attack (a.k.a, impersonation attack, or online attack)

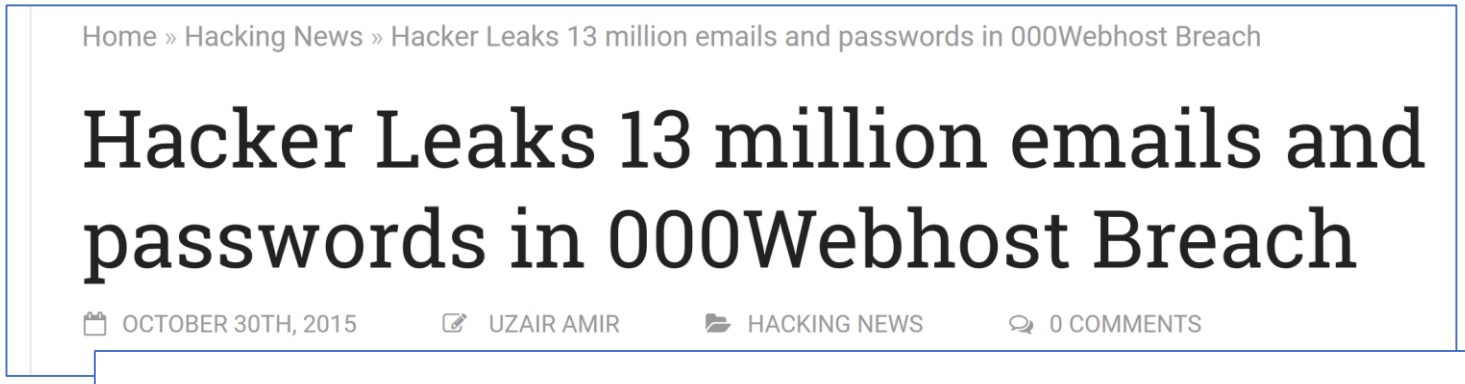
- Horizontal / untargeted attack
  - Guess most probable passwords against all the user accounts
  - Distribute the "resources" across all users
  - Want to compromise **a** user account
  - Guesses are independent of the user under attack

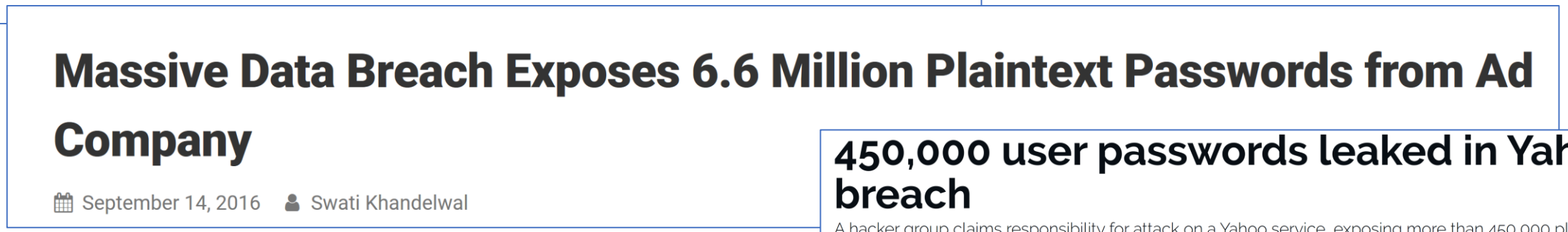
- Vertical / targeted attack
  - Target a user account and direct all resources towards compromising that account
  - Tailored guesses, based on user's name, email, DoB, location

Defense against remote guessing attack

- Throttle or block account if too many incorrect passwords submitted against an account: *query budget* (q)
- Throttling
  - "Your account is locked for too many incorrect attempts, try again after 15 min"
  - Throw captcha to stop automated bots
- Blocking
  - "Your account is blocked for suspicious activity, please contact customer care to unblock"
  - Blocking malicious IPs
  - Problem of Denial-of-Service (attacker can block legitimate users)
- Throttling by IP/browser is not useful
  - Browser = User-agent; easy to fake
  - IPs are cheap; e.g., EC2

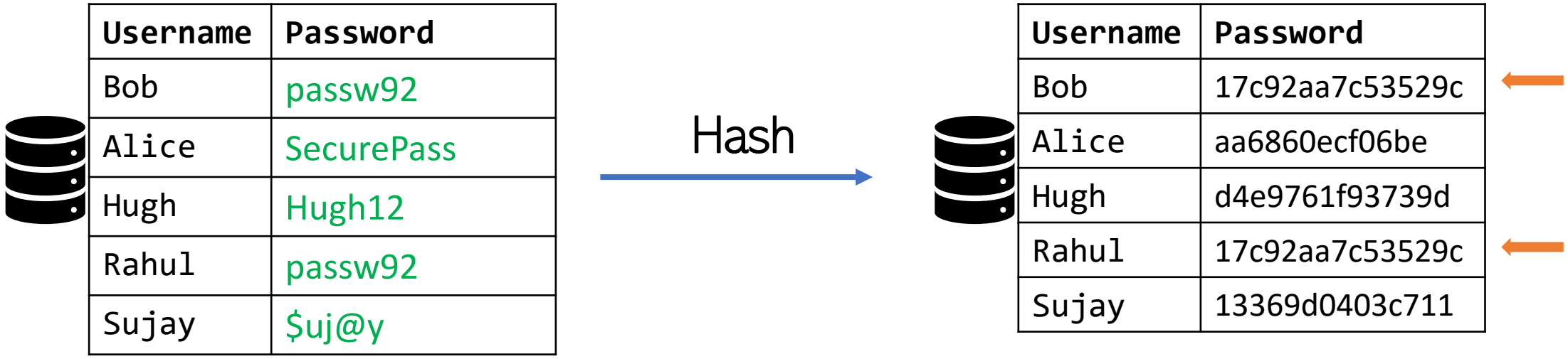
Defense against smash-and-grab attacker: "We will never get hacked"





Checkout this for the full list:
https://haveibeenpwned.com/PwnedWebsites

Defense against smash-and-grab attacker: Hash passwords

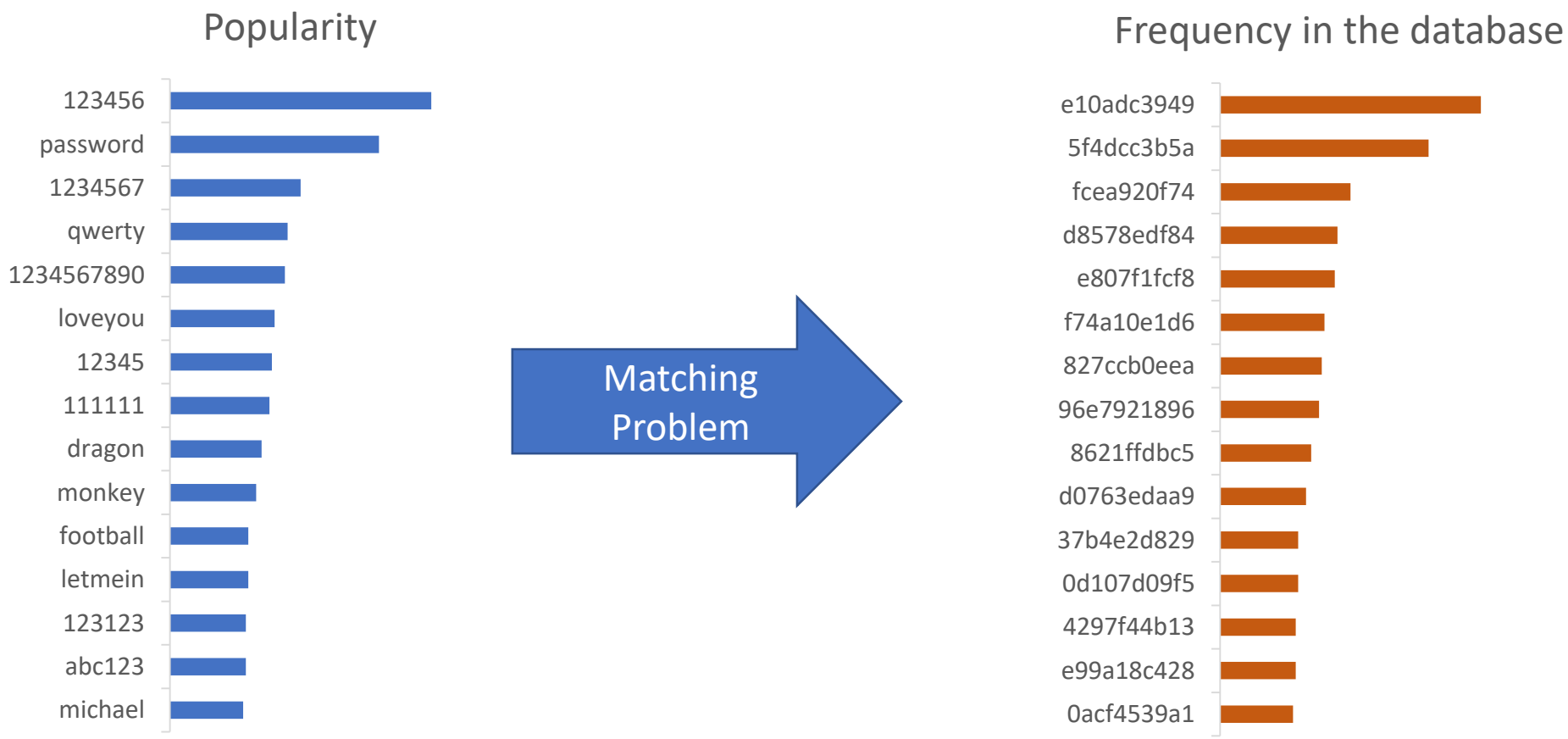


Just hashing is not enough

1. Reveals users with the same password
2. Hashes are normally fast, easy to brute-force

1. Less work for the attacker
2. Easier to crack popular password

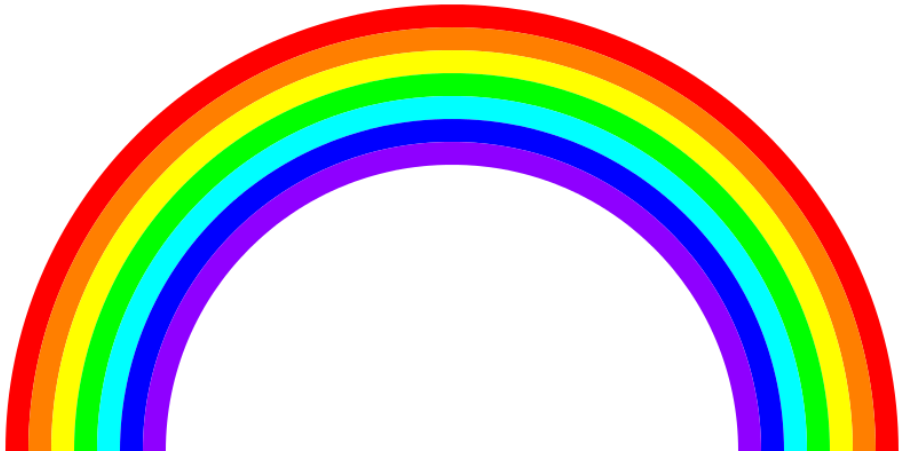
Password frequency to password recovery



From previous password leaks

Another attack: Rainbow Table

- Database of precomputed hashes of passwords
- Search through the database for matches



CrackStation

Defuse.ca

CrackStation » Password Hashing Security » Defuse Security »

Free Password Hash Cracker

Enter up to 20 non-salted hashes, one per line:

```
ab15b87dbcb022c6d5503141c16f115973ae9f6c345cbe20b20e468d85cbbc04
```

I'm not a robot — reCAPTCHA Privacy - Terms

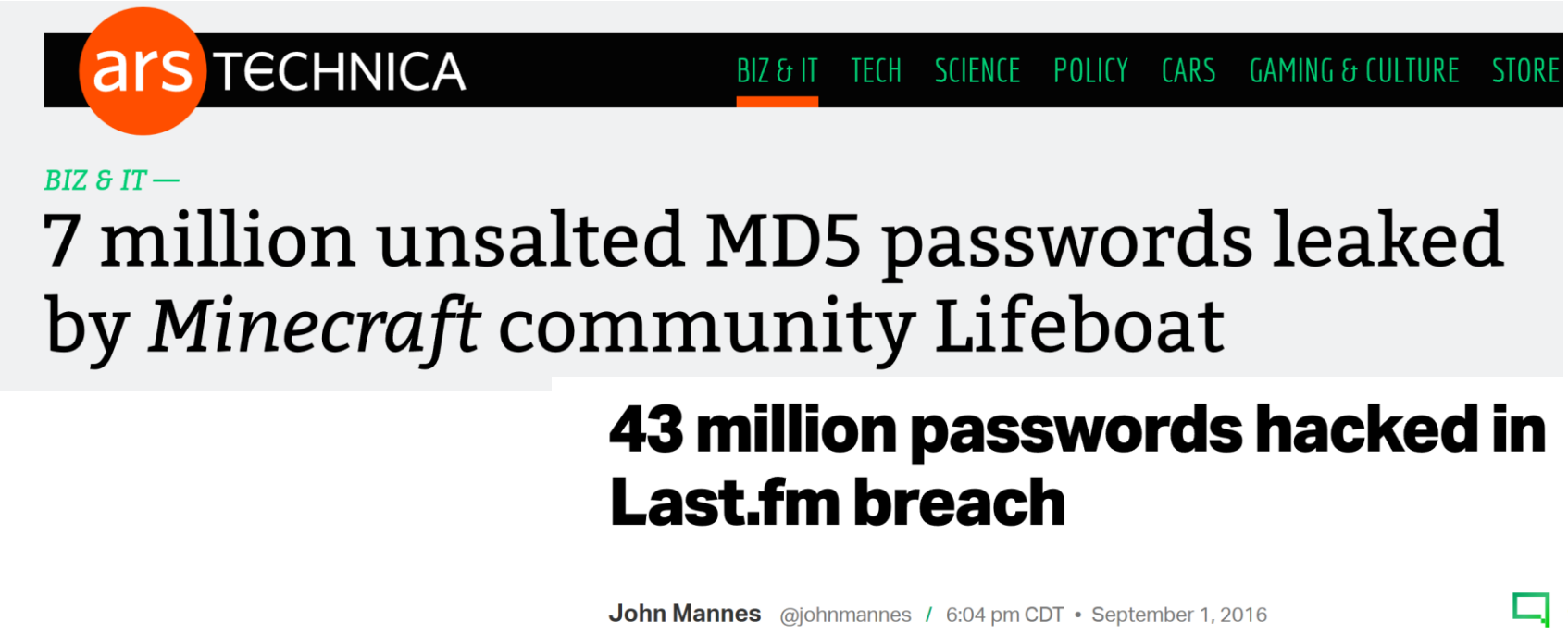
Crack Hashes

Supports: LM, NTLM, md2, md4, md5, md5(md5_hex), md5-half, sha1, sha224, sha256, sha384, sha512, ripeMD160, whirlpool, MySQL 4.1+ (sha1(sha1_bin)), QubesV3.1BackupDefaults

Hash	Type	Result
ab15b87dbcb022c6d5503141c16f115973ae9f6c345cbe20b20e468d85cbbc04	sha256	everythingisbroken

Color Codes: Green: Exact match, Yellow: Partial match, Red: Not found.

… but we will never lose the password database



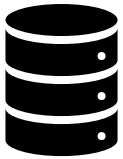
Add salt (and pepper)

Pepper:

65db2ad3f98db40

```
h  = H(pw + sa)
```

Per user random values



Username	Salt	Password
Bob	70af7d6c23	4699ce4e7b1dac7d
Alice	17a7dc97de	74418729b9f206e7
Hugh	6d7d52cba3	22fa3a5288aa1bb5
Rahul	2ef7d06331	1b34a1b436fc21da
Sujay	884948ef85	1fc13443a0b77b0b

```
h  = H(pw + sa + pepper)
```

If pepper is long and not stolen, brute-force cracking is not possible

A global value stored separately from the database

Advantages of Salting

- Without salt, attacker can precompute hashes of all dictionary words once for ALL password entries
  - Same hash function on different machines
  - Identical password: identical hash; one table of hash values can be used for all password files that get leaked
- With salt, attacker must compute hashes of all dictionary words for EACH password entry
  - With 12-bit random salt, same password can hash to 2^12 different values
  - Attacker must try dictionary words for EACH SALT VALUE in password file

Hashes can be brute-force cracked

Hash all possible passwords of certain length and see if the hash output matches with the hash value present in the database

- Well crafted password dictionaries are available
- Lots of leaked passwords available

- Software to speedup cracking:

HASHCAT
Advanced Password Recovery

John the Ripper

123456
password
12345
1234567890
letmein
loveyou
password1
qwerty
qwerty1
Password
hereyougo
michael
something
random

2.7 billion records

Collection of 1.4 Billion Plain-Text Leaked Passwords Found Circulating Online

December 12, 2017 Mohit Kumar

SECURITY & PRIVACY + SECURITY NEWS

Collection #1 (and #2–5) are the latest massive password dumps

Posted on January 18th, 2019 by Joshua Long

Cryptographic hashes are quite fast

- As you may be discovering in your homework…

- Parallelization: Graphics Processing Unit (GPU)
- Dedicated hardware: Application-specific Integrated Circuit (ASIC), FPGA

~1 Giga SHA hashes/sec / GPU (=10^9 hashes/sec)

- How much time will take to exhaustively search all possible 6-character alpha-numeric passwords? ($52^6/10^9 < 20$ sec)

Use slow hash function

- Password based key derivation function (PBKDF): Iterate to slow down

```
import hashlib
hashlib.pbkdf2_hmac?

Docstring:
pbkdf2_hmac(hash_name, password, salt, iterations, dklen=None) -> key
```

Remember HMAC ?

e.g., 1,000,000

No need to worry how to concatenate the salt

Still not slow enough

- Servers use typically commodity hardware (EC2 machines)

- Attackers can use state-of-the art machines, ASICs, FPGAs, GPUs
  - Not fair!

Memory-hard hash function: `scrypt`

- Time + memory (costly resource)
- Bcrypt (does not provide guarantee)
- Scrypt

Specify how much memory is required to compute the hash

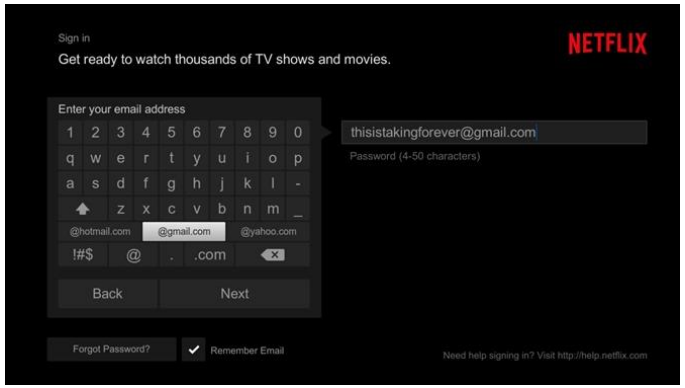
```
hashlib.scrypt?
Signature:
hashlib.scrypt(password, salt, …, maxmem, dklen=64)
```

- With less than the threshold memory, it will take more time
  - GPUs have limited memory; ASICs can have more memory, but it's costly to have so much memory

- Not easy to expedite using dedicated hardware

Usability problem with passwords

- Too many passwords to remember; cognitive burden
- Hard to type long complicated passwords

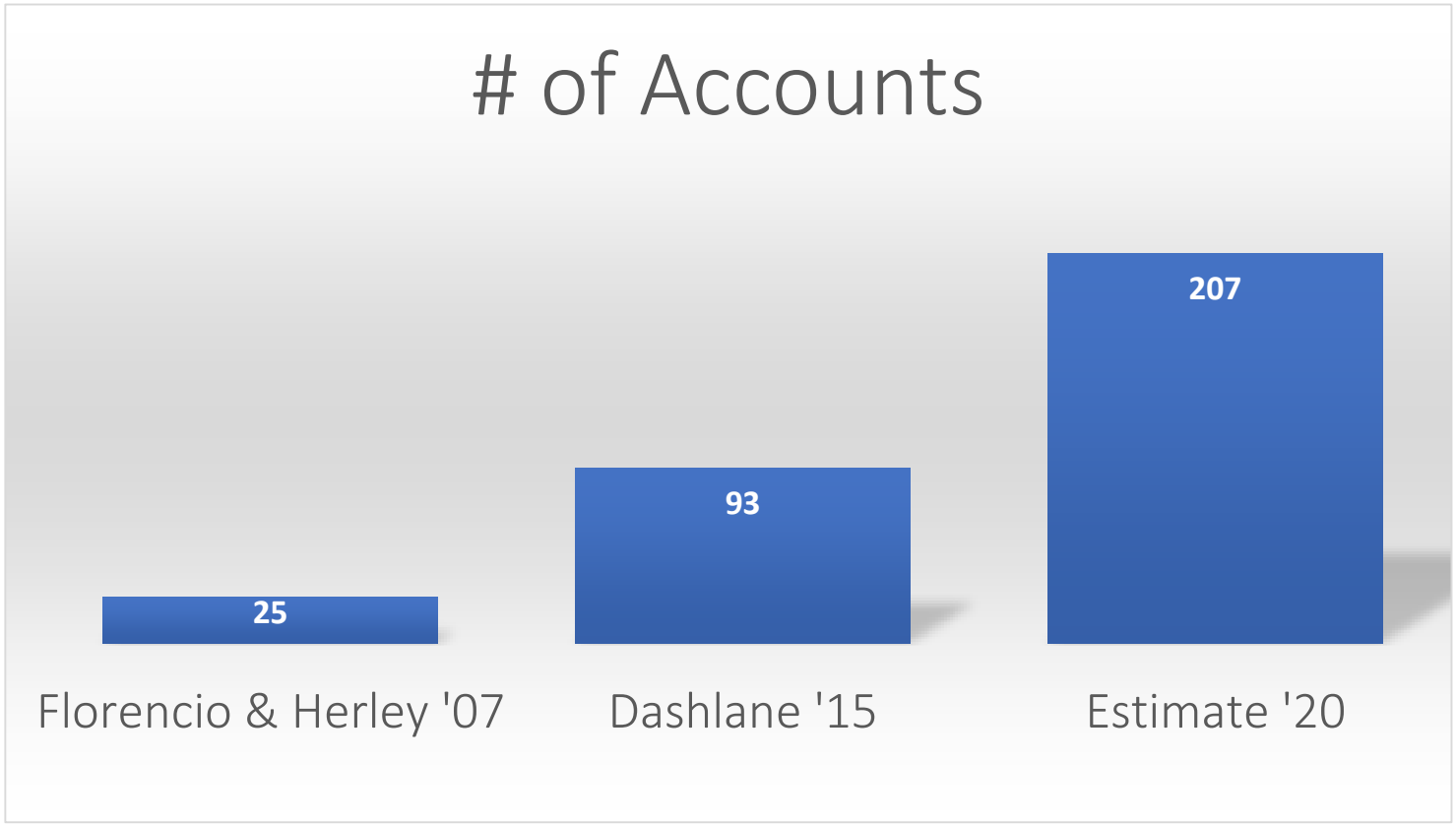


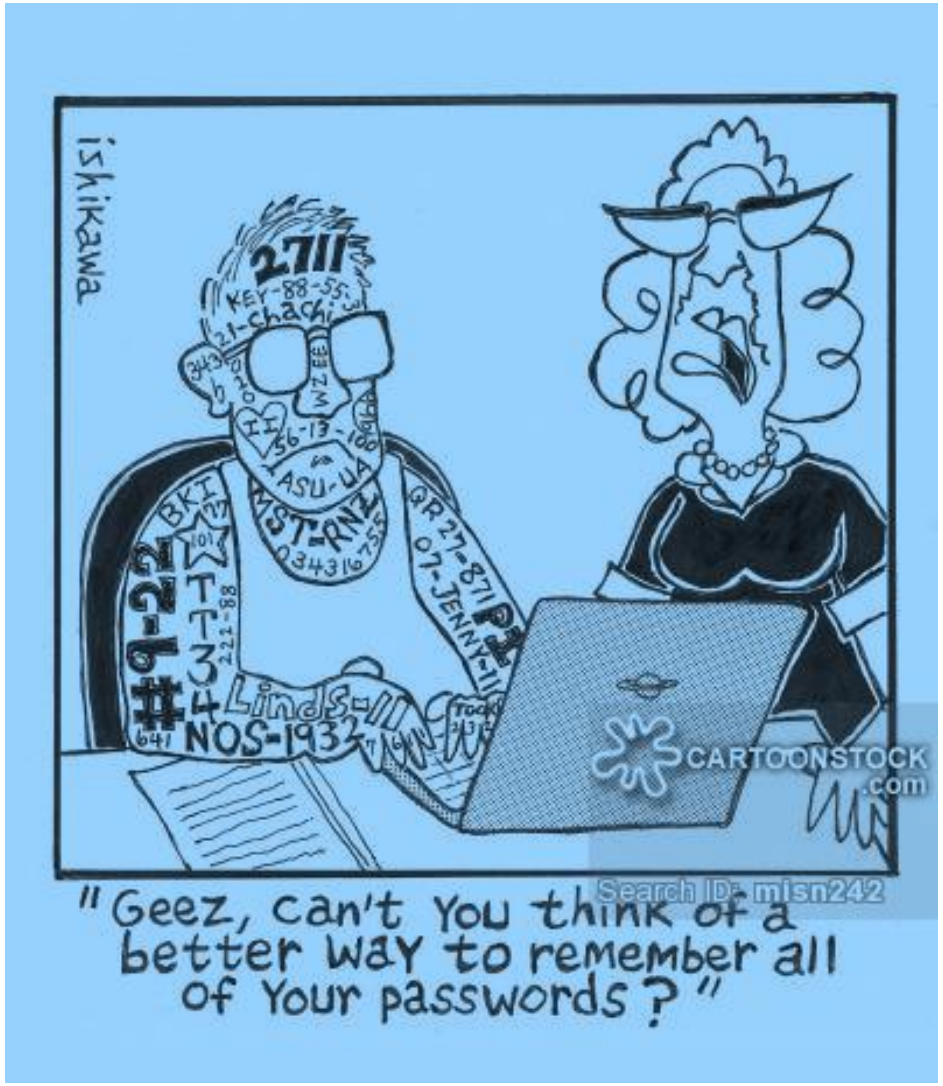
- As a result
  - Users pick weak passwords
  - Users reuse password for multiple web services

What exactly do I mean by “weak password”?

- Is **643107** is weaker than **1234567890** or **password?**
- Need to fix the attack method: dictionary-based or brute-force
- Dictionary-based methods are clearly better
- Popular passwords are weaker
  - P[w] = Probability that a password is chosen by a randomly chosen user
  - Higher the probability, weaker the password
- How to measure the probability?
  - There are several language-based model to estimate the probability
  - E.g., zxcvbn (https://lowe.github.io/tryzxcvbn/)

Too many passwords!





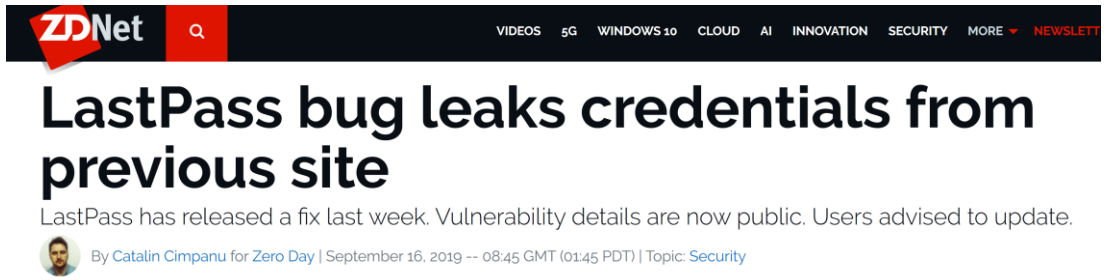
Solution: Password manager

Store all passwords in the password manager and encrypt using the master password.

and many more…

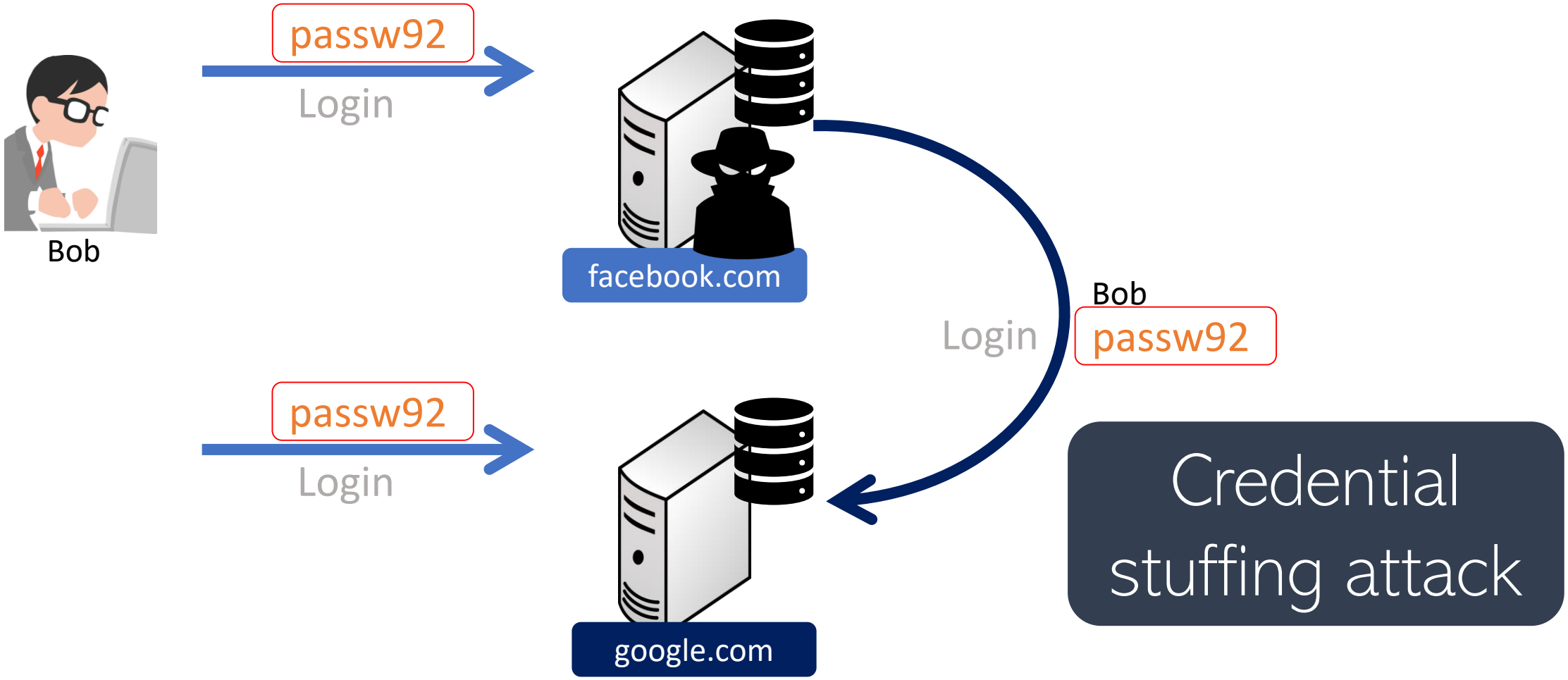
Low acceptance of password managers

- Trust issue:
  - Many people don't trust password managers with their passwords
  - "What if they get hacked",
  - "what if they loose my passwords"
  - "What if I forget my master password"



- UI issue. Does not work everywhere.
  - E.g., during ssh login, bios password

Perils of password reuse



Billions of passwords leaked

December 18, 2017

4iQ Discovers 1.4 Billion Clear Text Credentials in a Single Database

BRIAN BARRETT SECURITY 01.16.19 08:12 PM

HACK BRIEF: AN ASTONISHING 773 MILLION RECORDS EXPOSED IN MONSTER BREACH

2.2 Billion Accounts Found In Biggest Ever Data Dump -- How To Check If You're A Victim

Davey Winder Contributor ⓘ

Cybersecurity

I report and analyse breaking cybersecurity and privacy stories

Over 2.5 Billion User Accounts Have Been Hacked This Year Alone. Here are 4 Things You Can Do to Protect Yourself

Credential stuffing attack is one of the most prevalent form account compromise.

[Verizon data breach report, 2018]

Two-factor authentication

+

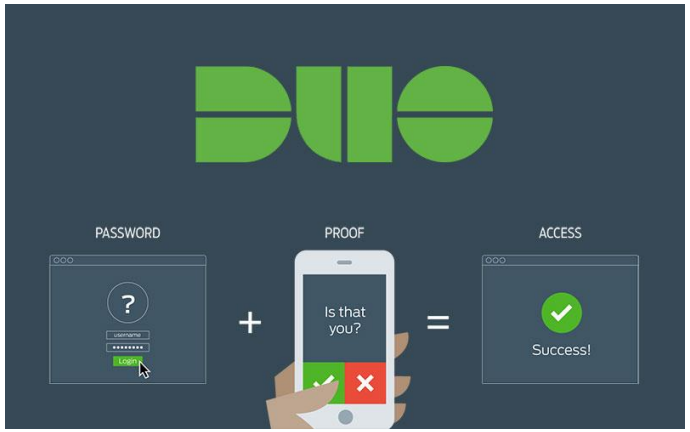
What you know

What you have

1. Stops automated password guessing attacks (including credential stuffing attacks)
2. Protect against remote attacker/online adversary; Study suggests stops ~90% of attacks

But, usability burden!
- Sim hijacking attack
- Phishing

https://krebsonsecurity.com/tag/sim-swapping/

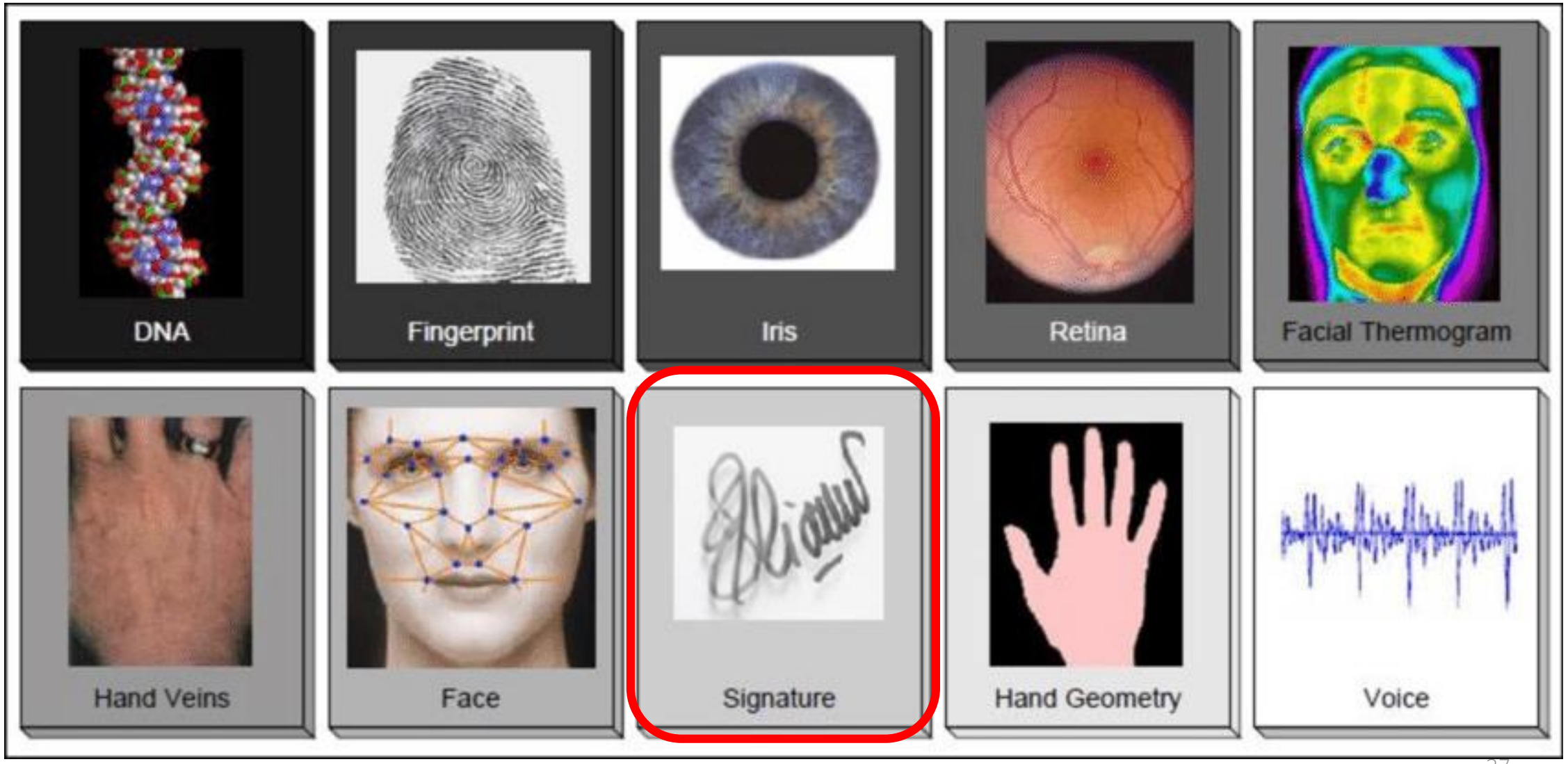


Password summary

- Human-generated secret
- Used to authenticate a user – only the user supposed to know the secret
- Threat model
  - Online attack (remote guessing attack)
  - Offline attack (smash-and-grab attacker)
- Usability is key
- Some philosophical questions/thoughts:
  - Who should be allowed to see your password for google.com? (Ideally only you)

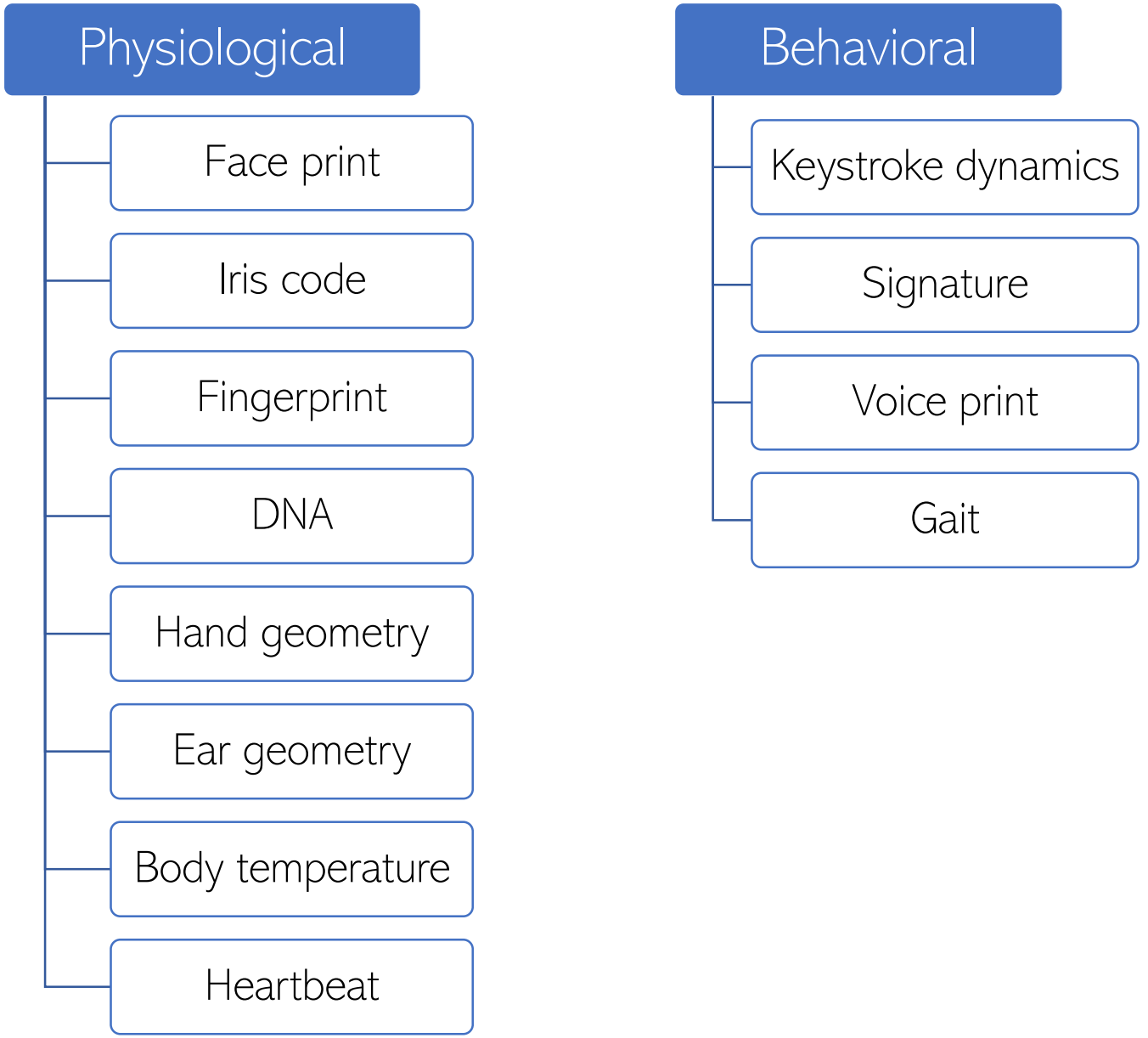
Biometrics

Alternative: Biometrics



Biometric Fingerprint Architecture for Home Security System, Siswanto et al., 2016

Biometrics


Physiological
Face print
Iris code
Fingerprint
DNA
Hand geometry
Ear geometry
Body temperature
Heartbeat
Behavioral
Keystroke dynamics
Signature
Voice print
Gait

Attributes of a "good" biometric feature

1. **Universality**: Does everyone have it?
2. **Distinctiveness**: Is it different for everyone?
3. **Permanence**: Does the feature change over time/age?
   - bad: face, good: fingerprint
4. **Collectability**: How easy it is to collect/measure the feature?
   - Very hard: DNA, relatively easy: fingerprint
5. **Performance**: How difficult to match?
6. **Acceptability**
7. **Circumvention**: How easy to spoof?
   - Voice recognition

Nothing about secrecy?

https://www.cse.msu.edu/~rossarun/pubs/RossBioIntro_CSVT2004.pdf

Fingerprint: History

- Prehistoric potters identify their works with an impressed fingerprint
- 200 BC: Chinese sign legal documents using fingerprints
- 1400 AD: Persia used fingerprint for identification
- 1685: Marcello Malpighi (University of Bologna), formalized fingerprint, introduced ridges, minutiae points
- 1858: The British started using fingerprint in India (Hoogly district, Bengal) to sign contracts
- 1880s, scientists (including Charles Darwin) began observing fingerprints for identification
- 1903: NYC State Prison started using fingerprinting inmates
- 1905: US army started using fingerprints for personal identification
- 1924: FBI Identification Division to collect and consolidate fingerprints
- 2012: Automated Fingerprint Identification System (AFIS)

Source: http://onin.com/fp/fphistory.html

Identification vs Authentication

- Identification: Claiming an identity, uniquely identifying a person

- Authentication: proving an identity
  - E.g., via passwords or biometrics

Fingerprint: How does it work?

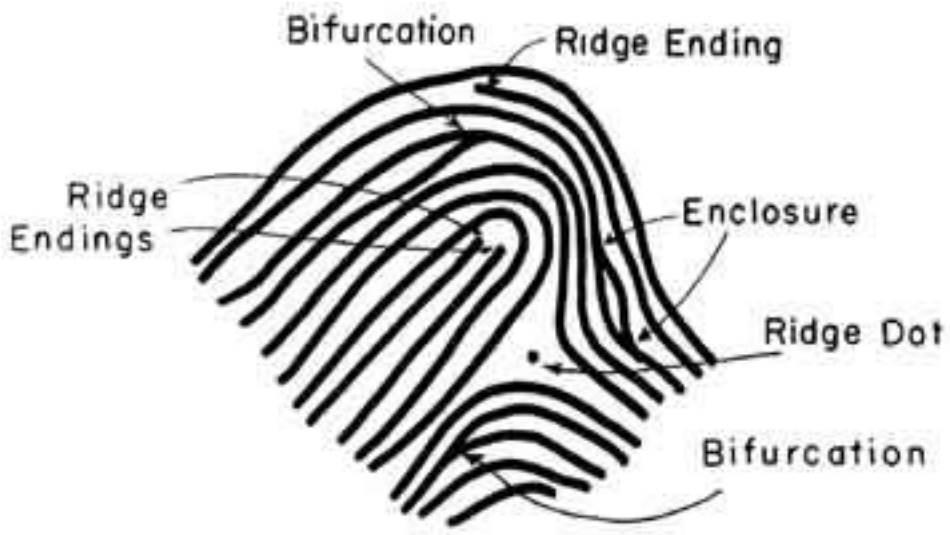


Image acquisition

Optical, Capacitive, Ultrasound sensors to read fingerprint

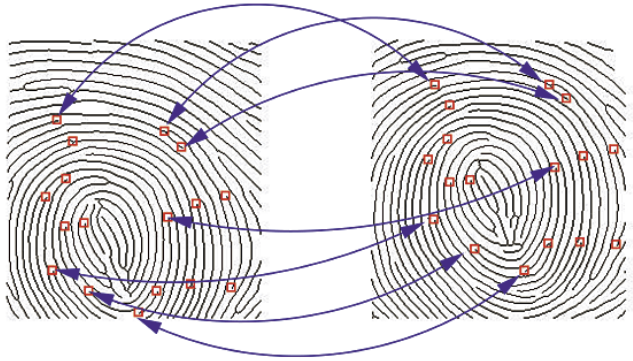
Image processing

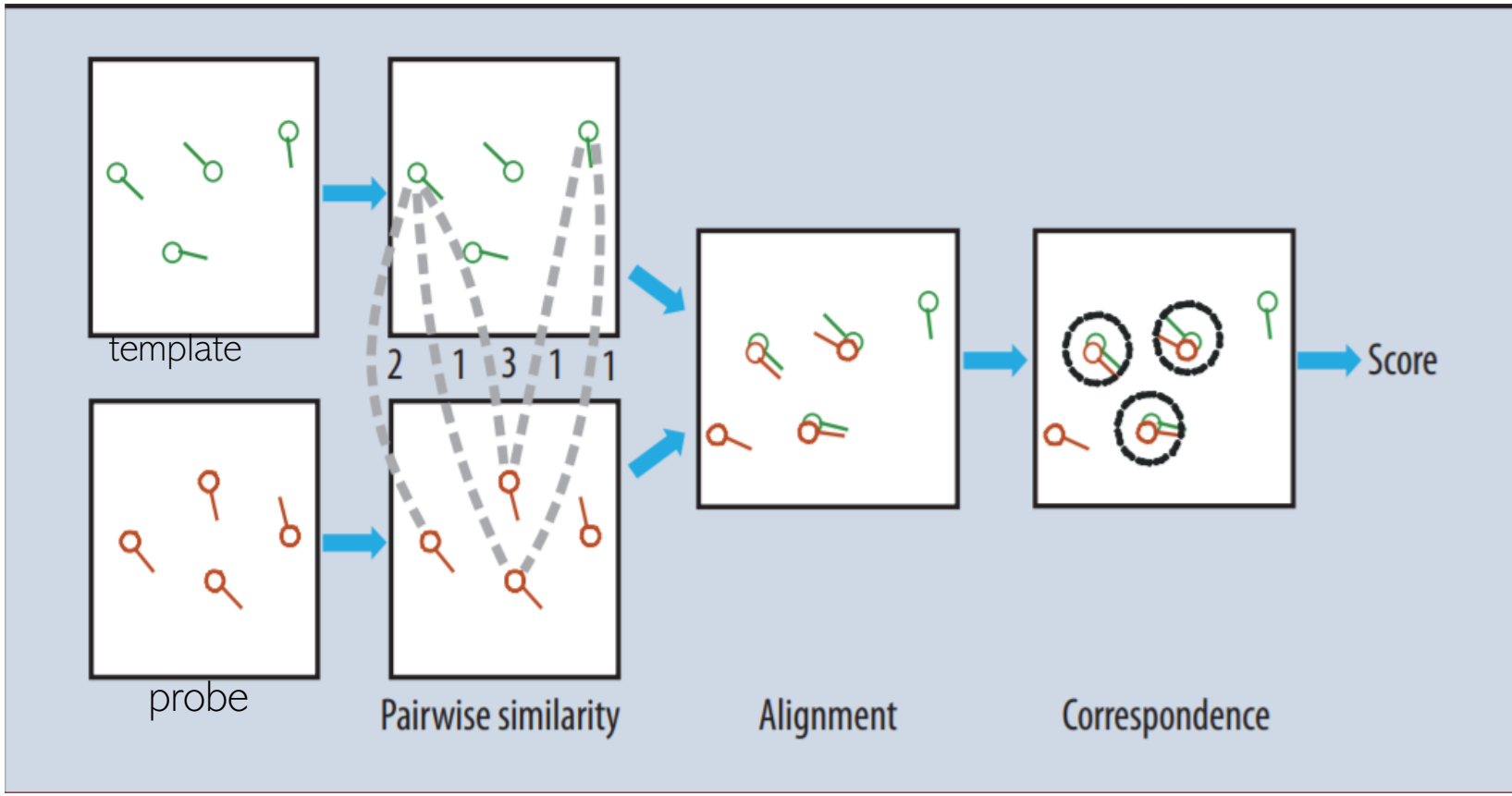
Minutiae extraction

Store f in the database with the username (u)

$$f = \{(x_1, y_1, \theta_1), (x_2, y_2, \theta_2), \ldots (x_n, y_n, \theta_n)\}$$

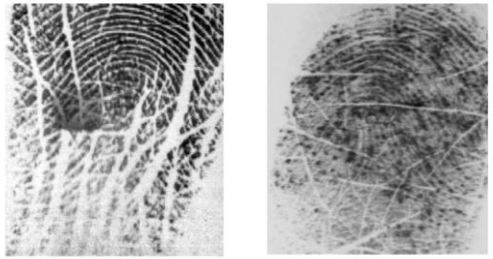
Fingerprint matching





Challenges with Biometrics

- Low accuracy
  - High False Non-Matching Rate (FNMR) or False Rejection Rate (FRR)
  - iPhone fingerprint matching has 1 in 50,000 false matching rate (FMR)
- Noise from biometric readers
- High error rate for some users



- Speed and scale matching process is slow

- Cannot be hashed

Secure storage of biometrics

- Threat model:
  - What happened: storage compromise
  - Goal: Learn user's biometrics (to use it to hack other web services)
- Passwords are **hashed**, fingerprints are _____?
- Cannot hash, because every reading of the same fingerprint is different
  - Hash output will be completely different, and therefore cannot match
  - Cryptographic hash functions reveal *nothing* beyond strict equality

- Encrypted?
  - Where to store the key for the encryption?
  - Trusted Platform Module (TPM), or Secure Enclave for Apple TouchID
    - Key is hidden in a tamperproof hardware

Attacks against biometric authentication

- Spoofing
  - Fake fingerprint
  - Spoof from latent fingerprint
  - Possible defense: liveness detection

- MasterPrints
  - Dictionary attack using synthetic fingerprints
  - Exploit the error in matching algorithms – generate fingerprints that are likely to be matched with other fingerprints in a dataset.
  - 1 masterprint matches with 8.6% – 22.5% of users [Bontrager et al. 2017]

Other privacy issues with biometrics

- Cannot be changed if compromised
  - Fundamentally “reused”!
- Surveillance, tracking
- Digital and physical identities are tied

Token-based authentication

Limitation of passwords and biometrics

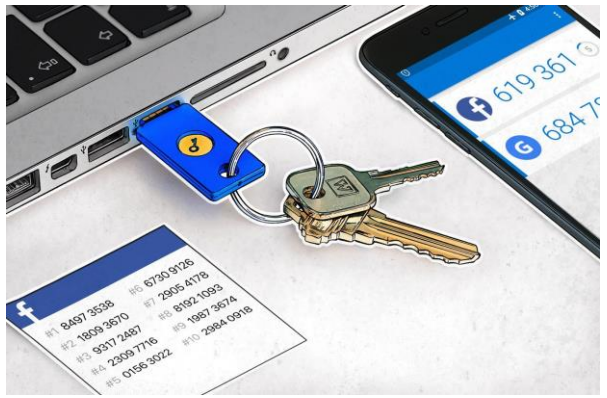
Passwords are

- Weak
- Users reuse in different websites
- Hard to remember, especially for hundreds of accounts
- Hard to type

Biometrics

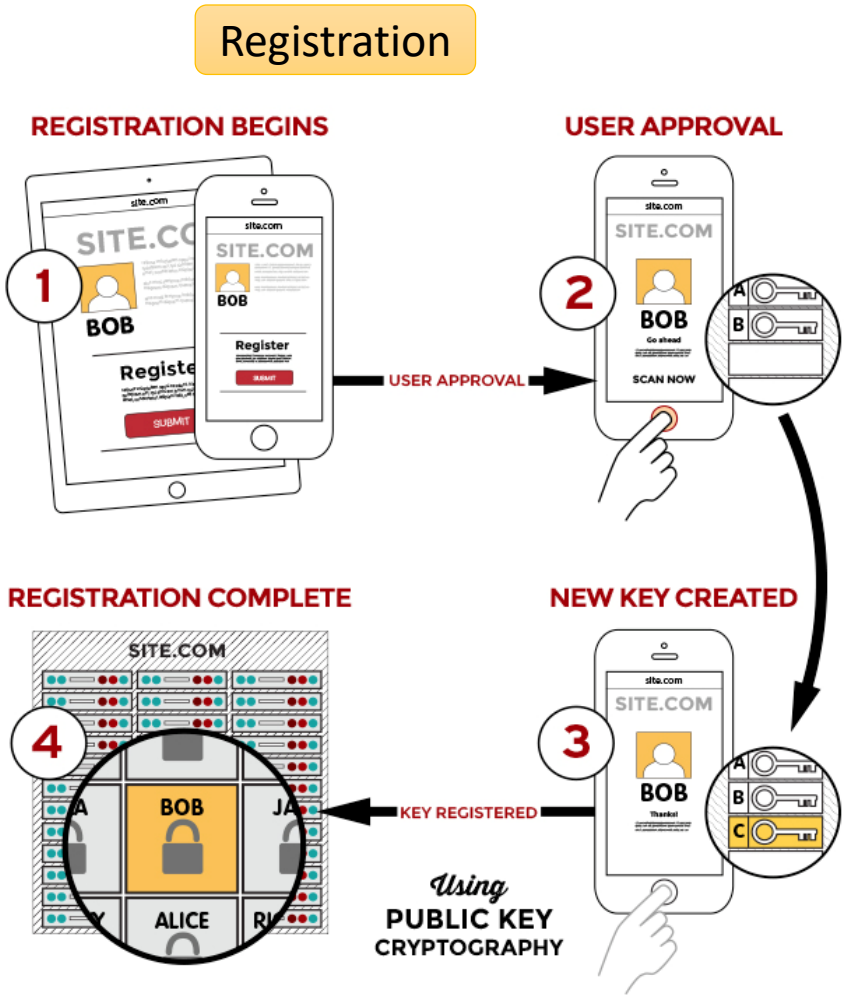
- Privacy violating
- Not a secret
- High error rate, low accuracy
- Spoofing, masterprint attacks

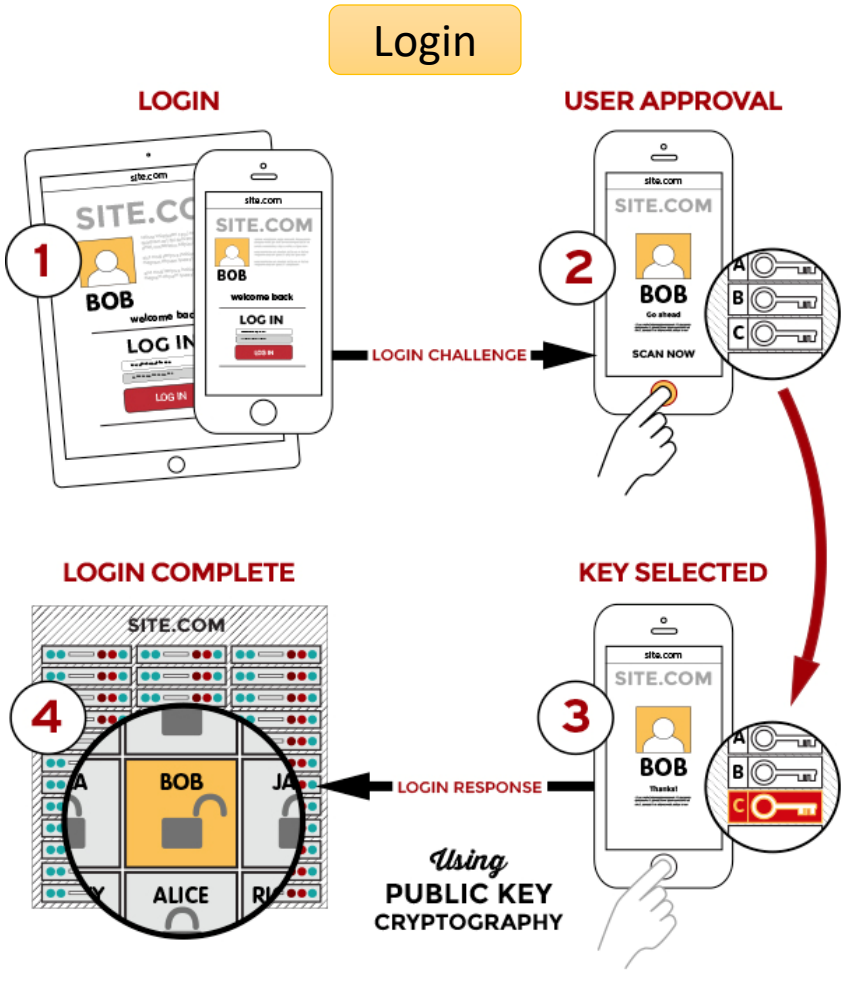
Use hard-tokens: what you have type authentication



FIDO fingerprint-based web authentication

Fast Identity Online





Source: https://fidoalliance.org/how-fido-works/

Multi-factor authentication

+

+

What you know

What you are

What you have

Risk-based authentication / continuous auth

- Analyze the risk profile of the user requesting access
- Challenge the user with additional authentication if the risk is high (confidence is low)
  - Second factor when logging in from a different browser, or geo location.

Problem

- Constantly monitor and update the risk profile
- Privacy issues with monitoring personal information – such as IP, user-agent, mouse movement, maybe browser fingerprinting

Authentication recap

- What you know – passwords, pin, passphrases
- What you are – biometrics
- What you have – hardware token, mobile phone

- Complicated problem, given diversity of users, devices, services
  - *No silver bullet*

- Next class: Asymmetric Crypto