CSE 127: Introduction to Security

Lecture 16: Authentication and passwords

George Obaido

UCSD

Spring 2022

Slides from Nadia Heninger and Stefan Savage

Today

- Common techniques for authenticating users, locally and remotely
- Security challenges associated with different authentication methods
- Mitigations designed to address some of the above security challenges

Authentication

- Using cryptography, Alice and Bob can authenticate each other by proving they know respective secret keys
  - Challenge-response authentication: Alice sends a random challenge to Bob. Bob signs (or MACs) the challenge.
  - Switch roles, repeat.

- What exactly did we authenticate?
  - Have Alice and Bob really committed their secret keys to memory?
  - Did they manually perform cryptographic signing operations?

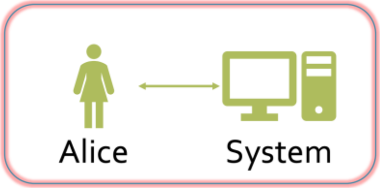
Authentication

- Using cryptography, Alice and Bob can authenticate each other by proving they know respective secret keys
  - Challenge-response authentication: Alice sends a random challenge to Bob. Bob signs (or MACs) the challenge.
  - Switch roles, repeat.

- What exactly did we authenticate?
  - Have Alice and Bob really committed their secret keys to memory?
  - Did they manually perform cryptographic signing operations?
- They authenticated each other's computers.

Authentication

- How do we authenticate a human user to a system?
- System is often remote server
- Authenticate: ascertain who is interacting with the system
  - Necessary to apply appropriate security policy
  - Only the intended subject should be able to authenticate to the system as that subject



Authentication

How do we authenticate a human user to a machine?

- Provide identity and proof of identity
- Identity examples:
  - Name, username, student ID, others?

Authentication

How can Alice prove that she's really Alice?

- Three types of authentication factors
  - Password: Something you know
  - Token: Something you have
  - Biometrics: Something you are

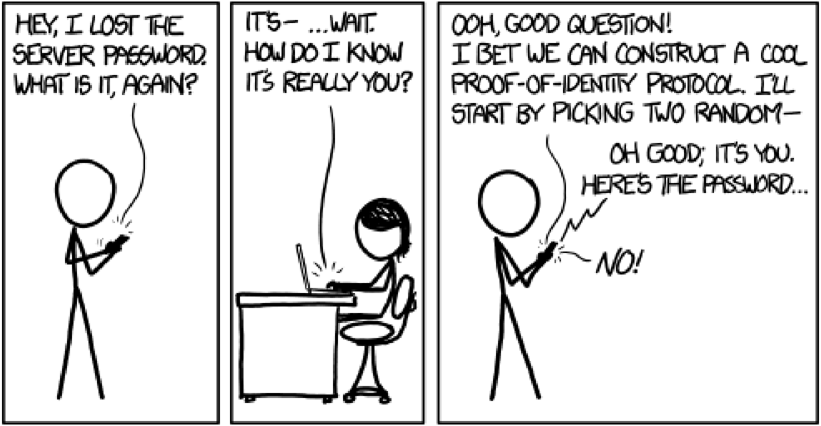
- Each factor can be used independently, or combined for multi-factor authentication.
  - Typically two-factor

@swiftonsecurity

Something you know

Something you know

- A secret that only the real Alice should know
    - A secret passcode.
        - Examples: PIN, password
        - PIN: Personal Identification Number (misnomer. Usually used for authentication, not identification.)
    - A secret about Alice
        - Examples: mother’s maiden name, first pet, mortgage payment
- Technically, only proves knowledge of secret, not that it’s really Alice
    - Secrets leak, can be shared, guessed.



https://xkcd.com/1121/

Passwords

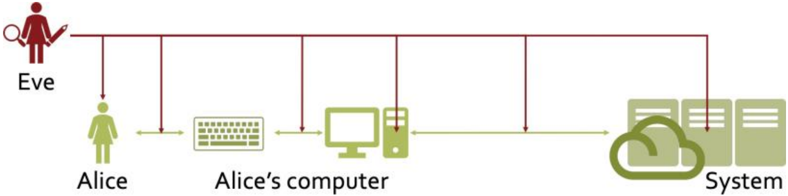
How does Alice prove she knows the password?

- Simplest: Alice provides the password to the system.
- Problems?
  - Passive adversary may observe password in transit
  - Need secure channel to protect confidentiality
  - Active adversary may impersonate the system
  - Alice needs a way of authenticating the system

Setting

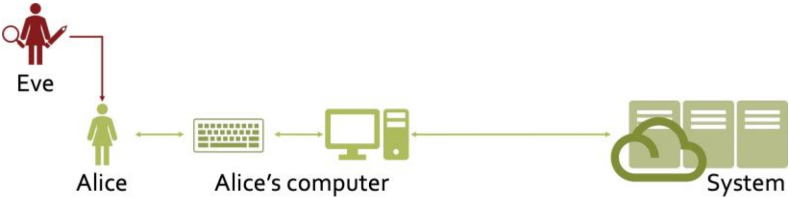
Alice uses a keyboard to type her password into client software that sends it on to the remote system for authentication.

Which points can Eve attack?



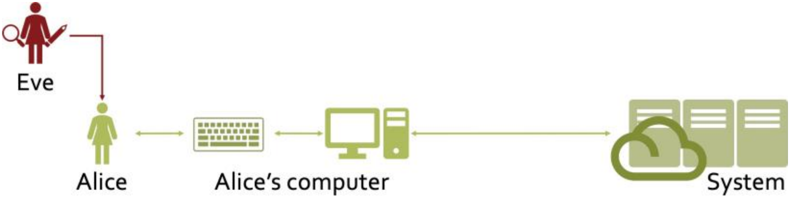
Attacking Passwords

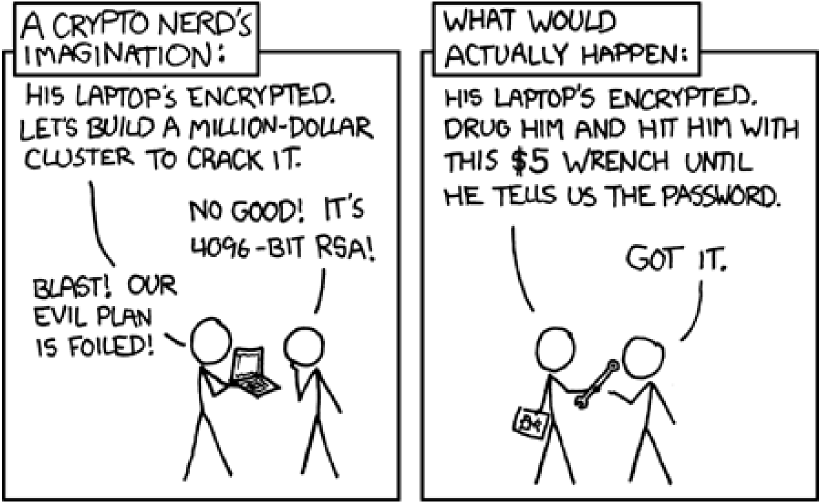
- Get it from Alice
- Intercept it
- Get it from the system



Attacking Passwords

- Is Alice invested in keeping it a secret?
  - Debit card PIN number?
  - Personal email password?
  - Netflix password?
  - Corporate network password?
- Is it written down somewhere?
  - Good against remote attackers
  - Not good against targeted local attacks (co-workers, family, abusers)
  - Know your threat model!
- Can it be guessed based on available knowledge about Alice (birthday, names of pets/kids/spouses, etc)?





https://xkcd.com/538/

Strong passwords

- Challenge: come up with passwords that are hard to guess, but easy to remember.
- Common password rules:
  - Composition: Letters and numbers, mixed case, symbols, banned dictionary
  - Length (char size)
  - Lifetime (monthly, quarterly, yearly, etc)
- Unintended consequences
  - Required letters/symbols → ?
  - Monthly change requirement → ?

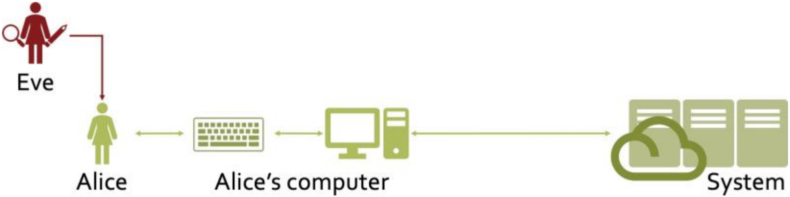
Top 25 most common passwords by year according to SplashData

Rank	2011[6]	2012[7]	2013[8]	2014[9]	2015[10]	2016[5]	2017[11]	2018[12]	2019[13]
1	password	password	123456	123456	123456	123456	123456	123456	123456
2	123456	123456	password	password	password	password	password	password	123456789
3	12345678	12345678	12345678	12345	12345678	12345	12345678	123456789	qwerty
4	qwerty	abc123	qwerty	12345678	qwerty	12345678	qwerty	12345678	password
5	abc123	qwerty	abc123	qwerty	12345	football	12345	12345	1234567
6	monkey	monkey	123456789	123456789	123456789	qwerty	123456789	111111	12345678
7	1234567	letmein	111111	1234	football	1234567890	letmein	1234567	12345
8	letmein	dragon	1234567	baseball	1234	1234567	1234567	sunshine	iloveyou
9	trustno1	111111	iloveyou	dragon	1234567	princess	football	qwerty	111111
10	dragon	baseball	adobe123[a]	football	baseball	1234	iloveyou	iloveyou	123123
11	baseball	iloveyou	123123	1234567	welcome	login	admin	princess	abc123
12	111111	trustno1	admin	monkey	1234567890	welcome	welcome	admin	qwerty123
13	iloveyou	1234567	1234567890	letmein	abc123	solo	monkey	welcome	1q2w3e4r
14	master	sunshine	letmein	abc123	111111	abc123	login	666666	admin
15	sunshine	master	photoshop[a]	111111	1qaz2wsx	admin	abc123	abc123	qwertyuiop
16	ashley	123123	1234	mustang	dragon	121212	starwars	football	654321
17	bailey	welcome	monkey	access	master	flower	123123	123123	555555
18	passw0rd	shadow	shadow	shadow	monkey	passw0rd	dragon	monkey	lovely
19	shadow	ashley	sunshine	master	letmein	dragon	passw0rd	654321	7777777
20	123123	football	12345	michael	login	sunshine	master	!@#$%^&*	welcome
21	654321	jesus	password1	superman	princess	master	hello	charlie	888888
22	superman	michael	princess	696969	qwertyuiop	hottie	freedom	aa123456	princess
23	qazwsx	ninja	azerty	123123	solo	loveme	whatever	donald	dragon
24	michael	mustang	trustno1	batman	passw0rd	zaq1zaq1	qazwsx	password1	password1
25	Football	password1	000000	trustno1	starwars	password1	trustno1	qwerty123	123qwe

https://en.wikipedia.org/wiki/List_of_the_most_common_passwords

Attacking Passwords

- Can Eve trick Alice into revealing her password?
- How does Alice know she is logging into the real system?
- Phishing!
  - Tricking Alice into revealing her password by impersonating the system she is trying to access
- Alice has to be able to authenticate the system before providing her password



Phishing

How can Alice authenticate the system?

HTTPS certificates validate the domain name in the URL.

What does it really tell you?

- That you are communicating with a server owned by UCSD?
- **No**. Only that you are communicating to www.ucsd.edu and your connection is secure (confidentiality and integrity are protected) against passive and active attackers on the link.

Phishing

- How do you know www.ucsd.edu is a legitimate UCSD web site?

- What about:
  - www.cse.ucsd.edu
  - www.ucsd.cse.edu
  - www.cse-ucsd.edu

Phishing

- How do you know www.ucsd.edu is a legitimate UCSD web site?
- A user is expected to know which domains are associated with the entity they are trying to interact with.
- And how to properly parse the URL
- Some browsers now highlight the domain portion

Phishing

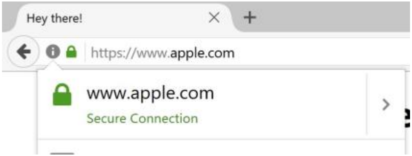
What if the user knows which domain is real?

- Homoglyphs: symbols that appear identical or very similar
- Attack: Register domain names that look just like the victim domain, but using a different character set.

Phishing

- https://www.irongeek.com/homoglyph-attack-generator.php
  - 

- https://www.xudongz.com/blog/2017/idn-phishing/
  - 



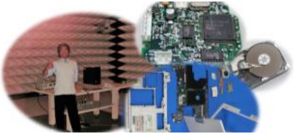
Phishing

- Related: When logging into a machine locally, how does Alice know that she is entering the password into the real login program?
  - Trusted path: Mechanism that guarantees user is interacting with intended component
  - CTRL+ALT+DEL on Windows

Attacking Passwords

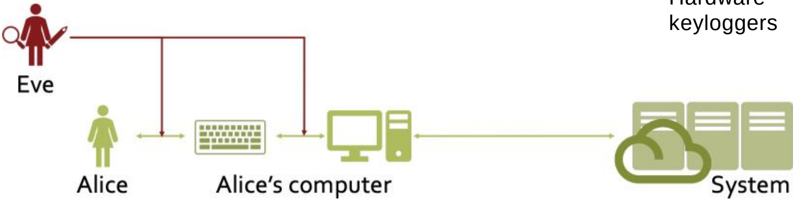
- Shoulder surf
- Side channels
- Hardware keyloggers

Shoulder surf



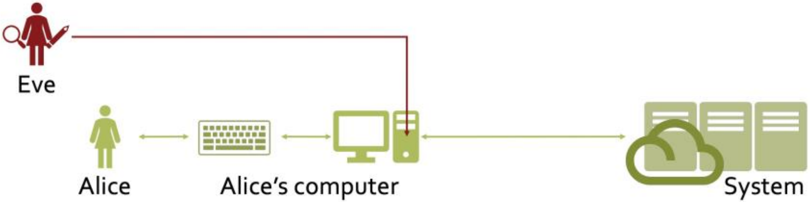
Side channels

Hardware keyloggers



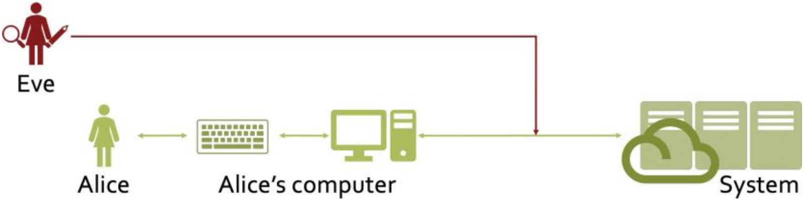
Attacking Passwords

- Software keyloggers
- Passwords in memory
  - Internal buffers
  - Clipboard
- Stored passwords
  - Cached passwords (e.g. browsers)
  - Password managers (e.g Bitwarden, LastPass, etc)
    - Good ones are well protected by master passwords
  - AlicePasswords.txt



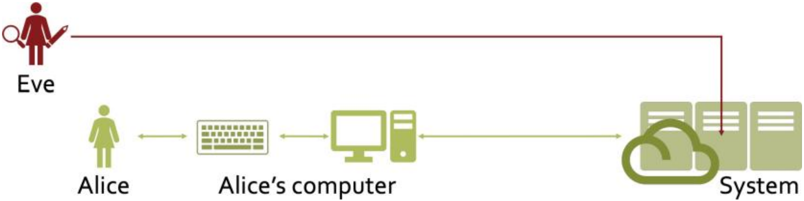
Attacking Passwords

- Monitoring the transmission channel
  - Channel should be encrypted to protect password confidentiality
  - Examples: TLS/SSH/HTTPS



Attacking Passwords

- Use system as an oracle: try to log in with different passwords
  - Defense: Minimize error information
  - Defense: Limit number of login attempts per user
  - Attack: Try different users for common passwords
- Compromise password database
  - Huge yield compared to user-side attacks
  - https://haveibeenpwned.com/
  - Password reuse issues



Protecting Passwords

- How can the system verify that the password Alice entered is correct?

- Naive solution:
  - Store a copy of the password and compare provided copy to the stored one

- Problem?

- If system is compromised, passwords are revealed
  - Same passwords may be used on other systems

Protecting Passwords

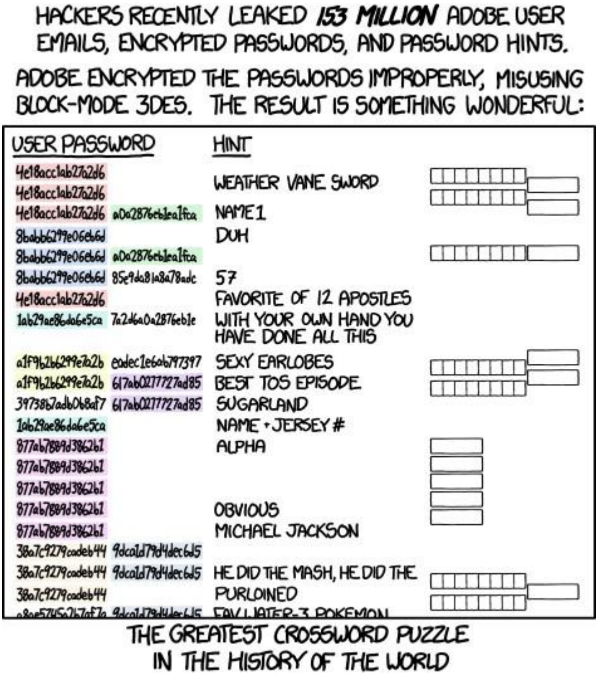
- Other solutions?

Protecting Passwords

- Other solutions?
- Hint: System does not need to know the password, only be able to verify it is correct.
- What if the system stores a cryptographic hash of the password?
  - H(password)
  - Hash must be *pre-image resistant* (hard to invert)
- Better... but still problematic.

Protecting Passwords

- Given a hash of a password, Eve can use it to validate guesses
  - Also, obvious which users have identical passwords
- Dictionary attacks
  - Dictionary: collection of possible, or likely, password strings
  - Try every string in the dictionary until the correct entry is found.
- Pre-compute hashes of all strings in the dictionary, then perform reverse look-ups by hash to find corresponding password.



https://xkcd.com/1286/

Protecting Passwords

Dictionary attack cost example:

- Assume passwords are composed of upper or lower case letters or digits
- $26 + 26 + 10 = 62 \approx 64$ possible values per character
- $64^n = 2^{6n}$ possible passwords of length n
- For $n = 6$, 2^{36} possible password strings
  - ≈ 10 TB to store all possible 6-character passwords and respective SHA-1 hashes

Can be reduced using techniques like rainbow tables.

Protecting Passwords

- **How do we make dictionary attacks harder?**
- Note, the attacker only had to compute one dictionary of hashes that could then be used for any user's password hash from any system.
- We can parameterize, or "salt", password hashes with unique random numbers
  - Instead of storing $H(p)$, store $(r, H(r||p))$, where r is random salt
  - Precomputation is no longer possible. Attacker must compute unique hashes for every target
- Better... but still problematic.

Specialized Password-Cracking Hardware

- 2012: Gosney 25 GPU password cracking cluster
  - 350B NTLM hashes (used by Windows) per sec
  - 180B MD5 hashes/sec, 63B SHA-1 hashes/sec

- State actors can build custom hardware

https://arstechnica.com/information-technology/2012/12/25-gpu-cluster-cracks-every-standard-windows-password-in-6-hours/

Protecting Passwords

- **How do we make dictionary attacks even harder?**

Protecting Passwords

- **How do we make dictionary attacks even harder?**
- Hint: The computation to verify a password for a given user on a legitimate system happens relatively infrequently, but an attacker attempting to crack a password hash must perform many, many attempts

Protecting Passwords

- How do we make dictionary attacks even harder?
- Hint: The computation to verify a password for a given user on a legitimate system happens relatively infrequently, but an attacker attempting to crack a password hash must perform many, many attempts
- Conclusion: Use a deliberately slow and resource-consuming hashing function
- PBKDF2, bcrypt, scrypt

Protecting Passwords

- Building blocks for password protection
  - Hash
  - Salt
  - Slow down
- For Slow down, use one of:
  - PBKDF2
  - bcrypt
  - scrypt

Open-ended question

What is your password hygiene?

Mine: Use strong passwords, don’t reuse passwords, never share passwords, etc.

Go to **www.menti.com** and use the code **1044 1273**

Something you have

https://wallpaperaccess.com/blue-desktop

Something You Have

- Something only Alice should have
  - Examples: key, smartcard, RFID badge, SecurID token
- Frequently used as a second factor (in combination with a passcode)
  - 2FA token
- Technically, only proves possession of the token, not that it's really Alice
  - Tokens get shared, lost, stolen, duplicated

Smartcards

- Idea: Put a secret key into a tiny computer that Alice can carry with her
  - Plastic card with an embedded integrated circuit
  - Provisioned with secret keys
  - Interacts with readers through contact pads or short range wireless (NFC)

https://en.wikipedia.org/wiki/EMV

- Many uses beyond user authentication
  - Stored value payment and transit
  - SIM cards
    - Satellite TV

Use protected sleeves

- Sample authentication protocol:
  1. Interrogate with a random challenge
  2. Verify signed response

NFC hacking

One Time Passcode Tokens

- Same basic idea as a smart card: a tiny computer with a secret
  - Typically without a direct computer interface
- How to provide challenge and get response?
- Response is displayed on token screen, user types it into the authentication system.
  - Typically using current time instead of a challenge (requires time sync)
  - Some variants have keypads to allow the user to type in a challenge as well

https://www.rsa.com/en-us/resources/rsa-securid-hardware-tokens

One Time Passcode Tokens

- Typical protocol:
  - Based on symmetric cryptography (shared secret between token and authenticating server)
  - Periodically (e.g. once a minute) token generates a new single-use code by MACing current time
  - To authenticate, Alice types in her password and current code (two-factor)
- Strengths:
  - Knowing the password is not enough to impersonate Alice
  - Each code is single-use. Eavesdropping (shoulder-surfing, keylogging, etc.) does not enable Eve to impersonate Alice in the future.
  - Observing any number of codes does not help in predicting future ones.

One Time Passcode Tokens

- Weaknesses:
  - Vulnerable to man-in-the-middle and phishing attacks.
  - Server needs to know the secret key to validate token codes. Single point of failure.
  - Does not scale well to multiple accounts.

https://www.techstagram.com/2013/07/27/rsa-passban/

One-Time Passcode without Tokens

- Virtual edition
  - Everybody (in some parts of the world) already carries a tiny computer. Let's just use that.
  - Strength: better scaling, support multiple keys with the same physical device.
  - Weakness: the two authentication factors are not as isolated anymore.

https://vip.symantec.com/

One-time Passcode without Tokens

- Extending the idea of using (possession of) your phone as an authentication factor.
- Authenticating server can send Alice a one-time code via SMS.
  - Alice logs in with her password and received code.
- Often used for step-up authentication or account recovery.
  - Step-up authentication: secondary (stronger) authentication mechanism invoked based on risk level
    - Examples: When attempting to access more sensitive resources, or when behavior patterns do not match routine.
  - Similar solutions use email instead of SMS.
    - Proof that Alice has access to the email account she registered with.
- Widespread use, but weaker against range of threat models (SMS not very secure)

Something you are

https://wallpaperaccess.com/blue-desktop
- https://creativecommons.org/licenses/by-nc-nd/4.0/

Something You Are

- Some unique identifying characteristic that only Alice has (biometrics)
  - Physical feature: fingerprint, iris print
  - Behavioral characteristic: handwriting, typing
  - Combination thereof: voice, gait
- How do you know that I am the same person that was here last week?
  - Did I provide a password?
  - Did I provide a badge?
- Pretty much all trust boils down to biometric authentication of one human by another.

Biometrics

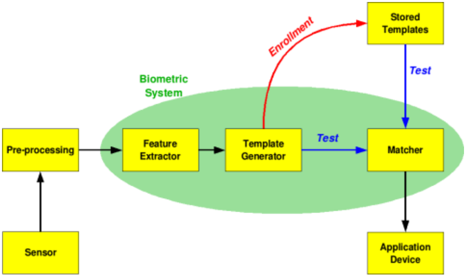
- The only authentication factor that is not designed to be transferable
  - Clear separation of authentication and authorization
- Nothing to remember, nothing to carry around
- Can be very strong differentiator
  - Unique-ish

Biometrics

- Fingerprint
- Handprint
- Retina
- Iris
- Face recognition
- Vein
  - Vascular pattern in back of hand
- Voiceprint
- Signature
- Typing
  - Timing between character sequences
- Gait recognition
- Heartbeat
- DNA

Biometrics

- General approach:
  - Scan an analog sample
  - Convert to set of digital features
  - On enrollment save template of identifiable features



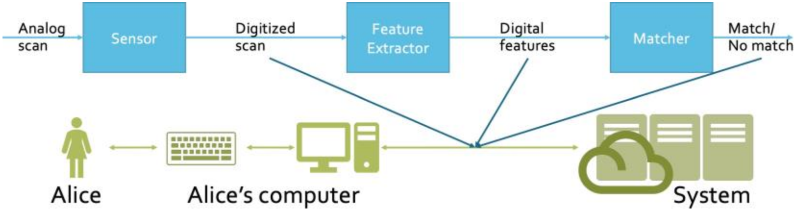
https://en.wikipedia.org/wiki/Biometrics

Biometrics

Simplified flow

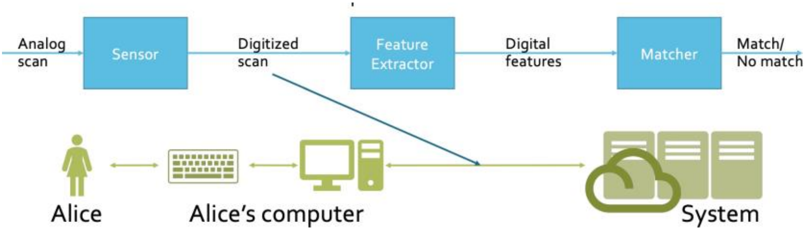
Biometrics

- What happens in a remote authentication setting?
- What does the authenticating system actually get?



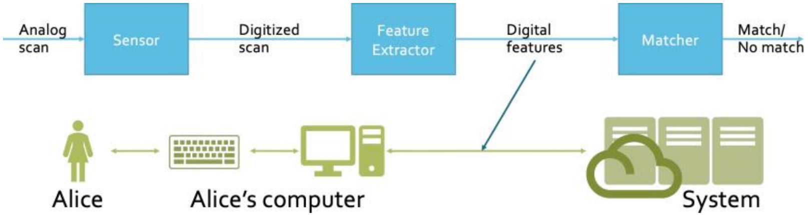
Biometrics

- Scenario A: Only the sensor is local to user.
- Feature extraction and matching happen on authenticating system.
- Authenticating system has to trust Alice's computer to provide sensor data.
- All biometric features and template data are on a central server.



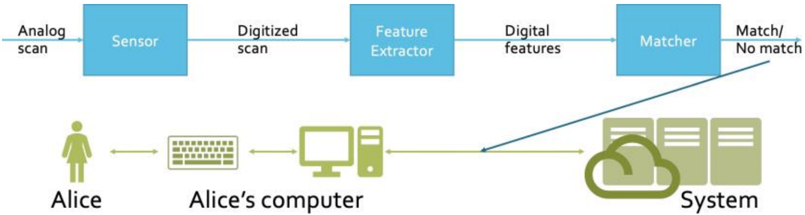
Biometrics

- Scenario B: Sensing and feature extraction are local to user.
- Matching happens on authenticating system.
- Authenticating system has to trust Alice's computer to provide authentic, fresh, unspoofed data.
- All biometric features and template data are still on a central server.



Biometrics

- Scenario C: Sensing, feature extraction, and matching are local to user.
- Only the result is communicated to the authenticating system.
- Authenticating system has to trust Alice's computer to perform authentication.
- All biometric features and template data are isolated on end users' devices.



Biometrics

- Use in distributed systems requires biometric scanner to be trusted and to have secure channel (authenticity, privacy, integrity, no replay) to the server.
- Challenges
    - Accuracy
    - Ease of use (particularly enrollment)
    - User acceptance
    - Feature stability

Enrollment Issues

- Unlike passwords, hard to pre-enroll user
- Users must be enrolled interactively
- For many biometrics, getting good accuracy requires multiple readings
  - Build templates and test against registration
  - Repeat
  - Some templates simply tough (e.g. smooth fingerprint)
  - “Goats”: Subjects who have consistently low match scores against themselves.

How strong is a biometric?

- Non-adversarial
  - False accept rate
  - False reject rate
- Adversarial
  - Intercept
  - Spoofing

Non-adversarial testing

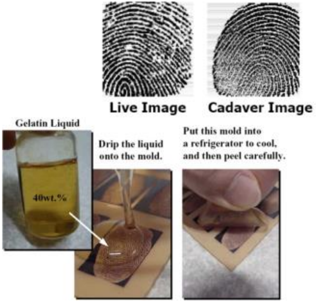
- False accept rate
  - How many random trials before expectation of false accept > 0.5
- False reject rate
  - How many random trials before expectation of false reject > 0.5
- Lower FAR = less tolerant of close matches
  - Harder to attack
  - Necessarily increases FRR
- Lower FRR = more tolerant of close matches
  - Easier to use
  - Necessarily increases FAR
- Since match is approximate can almost always tune for one or other
- Equal error rate point where FAR = FRR
- Note, huge difference between a single false accept and system-wide false accept (more templates means more things you can accept against)

Biometrics Spoofing

- Biometrics are private, but not secret
- Users expose biometric instances everywhere
  - Fingerprints, hand geometry, face, handwriting, iris, gate, etc.
- Allows attacker to create biometric forgery
- Very hard to replace a biometric identifier

Biometrics Spoofing

- There are spoofing techniques for virtually all biometrics



Chaos Computer Club

May 23, 2017

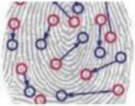
Researchers from the Chaos Computer Club (CCC) have successfully breached the Samsung Galaxy S8's iris recognition system to unlock the...

Continue Reading

Spoofing iris recognition technology with pictures

Mar 9, 2015

In a recent report by Forbes, Chaos Computer Club security researcher Jan "Starbug" Krissler highlighted the vulnerabilities behind some iris-scanning...



German researcher reverse-engineers a fingerprint using photos

Dec 30, 2014

A German researcher from European hackers association Chaos Computer Club recently demonstrated a method to fool standard biometric security software...

Chaos Computer Club claims Touch ID fake fingerprint spoof

Sep 23, 2013

Well that was fast. German hacker collective, Chaos Computer Club, has claimed that it has already spoofed the iPhone 5S's Touch...

Biometric Spoofing Mitigations

- Replay prevention
  - Save previous image and reject if identical
  - Tricky: can pick up and rotate to fool
- Improved validation prevision
  - Verifier should have higher precision than forger
  - Examples: pore detection, perspiration detection
- “Liveness” detection
  - Examples: temperature, pulse, blood flow

Biometric Spoofing Mitigations

- Multi-modal
  - Multiple biometric factors

- Multi-factor
  - Biometric plus password
  - Biometric plus token

Privacy issues

- Biometric identifiers can track your physical activities as well as your virtual activities
  - Some with crisp legal standing (fingerprint, DNA)
- Easy to match (even if can’t spoof)
- Very hard to obscure

Review

- Three types of authentication factors
  - Password: Something you know
  - Token: Something you have
  - Biometrics: Something you are

- Each factor can be used independently, or combined for multi-factor authentication
  - Typically 2-factor

- Use a slow salted hash to store passwords
  - PBKDF2, bcrypt, or scrypt
  - Don’t make up your own!