Authentication

- Authentication may be based on
  - What you know
  - What you have
  - What you are
  - Examples? Tradeoffs?
  - Others?

- Can also consider two-factor authentication
What you are -- biometrics

- Tradeoff of cost vs. accuracy
  - Face (low accuracy, low cost)
  - Fingerprint/hand print (good accuracy, moderate cost)
  - Iris scan (high accuracy, high cost)
  - ...

Verification vs. identification

- Verification: send (id, biometric) and check whether this 'matches' the stored biometric for user id
  - Better suited for authentication
- Identification: Send biometric, find the user whose biometric is the closest match
  - Comes up in law enforcement
Challenges in using biometrics

- Reproducibility
  - How much entropy \((= \log_2(\text{expected #att}))\) is there?
    - Difficult to estimate
  - How private are they?
  - Revocation?
    - Difficult to use securely
      - Reproducibility
      - Non-uniform
      - Still need a secure protocol...

Reproducibility

- Biometric data is not exactly reproducible
  - Need to check for *closeness* rather than an exact match
  - Implies the existence of false positives and negatives
    - Must trade off one vs. the other...
  - Implies a reduction in entropy; easier for an attacker to guess a value close to your biometric data
How can you securely authenticate yourself to a remote server using your fingerprint?

Trivial solution:

Biometric authentication

User \[\rightarrow\] Server

close?

Can work for 'local' authentication…
…but completely vulnerable to eavesdropping!

Better(?) solution

User \[\leftarrow\] nonce \[\rightarrow\] Server

h = MAC(, nonce)

A single-bit difference in the scanned fingerprint results in a failed authentication!
Authentication using biometrics

- There exist techniques for secure authentication using biometric data
  - Resilient to error!
  - Establish random, shared key
- An active research area...
Password selection

- User selection of passwords is typically very poor
  - Low-entropy password makes dictionary attacks possible

- Typical passwords:
  - Derived from account names or usernames
  - Dictionary words, reversed dictionary words, or small modifications of dictionary words

- Users typically use the same password for multiple accounts
  - Weakest account determines the security!
  - Can use programs to correct this

Password strength

- Several empirical studies of password strength, using compromised passwords

- “Most” (> 80%) passwords have fewer than 22 bits of entropy
  (Weir et al., “Testing Metrics for Password-creation Policies by Attacking Large Sets of Revealed Passwords”)
Better password selection

- Non-alphanumeric characters
- Longer phrases
- Can try to enforce good password selection…
- …but these types of passwords are difficult for people to memorize and type!
- Security/usability tradeoff

Mandating password changes

- Many sites now force a password change at regular intervals

- What does this accomplish?
  - Off-line attacks?
  - Adversary who breaks in and passively monitors a user’s account?
Password storage

- In the clear...

- Hash of password
  - Makes adversary’s job (slightly) harder
  - Potentially protects users who choose good passwords

- “Salt”-ed hash of password
  - No harder to attack any single user’s password, but bulk dictionary attacks are harder
  - Prevents using pre-computed ‘rainbow tables’
  - Prevents password duplication from being detected

Password storage

- Encrypted passwords? (What attack is this defending against?)

- Centralized server stores password...
Password-based protocols

- Password-based authentication
  - Any system based on low-entropy shared secret

- Distinguish on-line attacks vs. off-line attacks

From passwords to keys?

- Can potentially use passwords to derive symmetric or public keys
- What is the entropy of the resulting key?
- Allows off-line dictionary attacks on the password
Password-based protocols

- Any password-based protocol is potentially vulnerable to an "on-line" dictionary attack
  - On-line attacks can be detected and limited

- How?
  - "Three strikes"
  - Monitor ratio of successful to failed logins
  - Gradually slow login-response time

- Potential DoS

---

Password-based protocols

- Off-line attacks can never be ‘prevented’, but protocols can be made secure against such attacks

- Any password-based protocol is vulnerable to off-line attack if the server is compromised
  - Once the server is compromised, why do we care?
Basic password protocols…

- Server stores $H(pw)$; user sends $pw$
  - Insecure against replay attacks
  - If $pw$ is a password, not secure against server compromise or eavesdropping (off-line attack)

- Server stores $pw$, sends $R$; user sends $MAC_pw(R)$
  - If $pw$ is a password, not secure against server compromise or eavesdropping (off-line attack)

Mutual authentication

- None of the password protocols we have seen so far offer mutual authentication
Authentication with password + public key

- Say that only the server has a known public key (e.g., SSL)
- Server sends R
- Client sends $E_{pk}(R, \text{password, session-key})$

- Secure if encryption scheme is CCA-secure
- Can be extended to give mutual authentication

“Do Strong Passwords Accomplish Anything?”
Basic points

- Weak passwords suffice if account locking is used
- Strong passwords are overly burdensome
- Strong passwords do nothing to protect users from most common attacks: phishing or keylogging

Cost/benefit analysis
- Are strong passwords worth the effort?

Attack taxonomy

- Phishing
- Keylogging
- On-line password guessing for one userID
- On-line password guessing for many userIDs
- Off-line password guessing
- Other
  - Social engineering
  - Password cached on machine
<h2>Attack taxonomy</h2>

- Phishing/keylogging/other attacks unaffected by password strength
- On-line attacks against one userID are preventable using moderate-strength passwords (next slide)
- Off-line attacks are preventable by using a good protocol
  - crypto must be good: don’t use El Gamal....
- Main advantage of strong passwords is for on-line attacks against many userIDs

<h2>On-line attacks against one user?</h2>

- Assumptions
  - 6-digit PIN
  - 24-hour lockdown after 3 failed login attempts
- Number of passwords an attacker can search in 10 years
  - $3 \times 365 \times 10 \sim 10^4$
- Probability of success
  - $10^4/10^6 = 1\%$
On-line attacks, many users?

- An attack on $10^6$ users would likely succeed in breaking in to one of their accounts
  - Account locking has no effect!

- Note that the number of password guesses depends on the number of users
  - $N$ users => $3N$ password guesses per day (under previous assumptions)

On-line attacks, many users?

- Useful to think in terms of the credential space of (userID, password) pairs
  - The adversary breaks in if it guesses a valid credential

- Say all 25-bit strings are valid userIDs (because userIDs issued sequentially) and 20-bit passwords are used
  - Size of credential space = $2^{45}$
  - Number of valid credentials = $2^{25}$
  - Success probability per attempt = $2^{-20}$
  - Expected attempts to success = $2^{20}$
On-line attacks, many users?

- Could decrease attacker's success probability by making the space of legal userIDs more sparse!

- We usually assume userIDs are public (e.g., sent in the clear during login)...
  - ...but it would be hard for the attacker to collect very many userIDs

On-line attacks, many users?

- Interesting distinction here
  - Users can write down their userIDs
    - Protected against on-line attacks by moderate-strength password and account locking
  - Attacker can get the userID of any particular user
  - Attacker cannot (easily) get the userIDs of many users

- Note that an attacker who can easily get many userIDs can perform a DoS attack on the site (lock out users)
On-line attacks, many users?

- Preceding analysis assumes the adversary cannot distinguish an incorrect password guess from an incorrect guess of a userID
- Be careful in what error messages are returned
- Be careful of timing attacks

Forgotten passwords

- How to deal with users who forget their passwords?
- Traditional approach: user physically requests password reset (after showing ID, etc.)
- This does not work well over the web...
Forgotten passwords

- *Secret questions* are often used

- These are not very good!
  - 33-39% of answers could be guessed by family members or close friends
  - 20% of users could not remember their own answers!

- Can be improved somewhat using multiple questions, and requiring a threshold of correct answers