CS 1699
Privacy in the Electronic Society

William Garrison
bill@cs.pitt.edu
6311 Sennott Square
http://cs.pitt.edu/~bill/1699

09: Authentication

First, a threat modeling discussion: Consider airline security

When analyzing the security of airlines, pilots are generally considered trusted, since they are in control of the planes.

Further, people with government security clearances are trusted with national security secrets, so presumably can be considered trusted when flying.

Discuss with the people around you: Design a set of rules that would enable one or both of these groups to bypass TSA gate security, given that it is redundant by the above assumptions.

Today: How do I prove my identity?

Types of identity/name

- Username, public key, pseudonym

Passwords: Something you know

- Complexity policies, storage methods, password managers

Tokens/ID cards: Something you have

- Two-factor, OAuth

Biometrics: Something you are

- Lockscreens, theme parks, unique biometrics

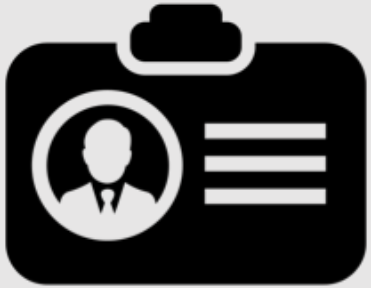
What type of name might I want to authenticate?

Authentication is the process of binding an identity to an interaction

- What does this mean?

User within a domain

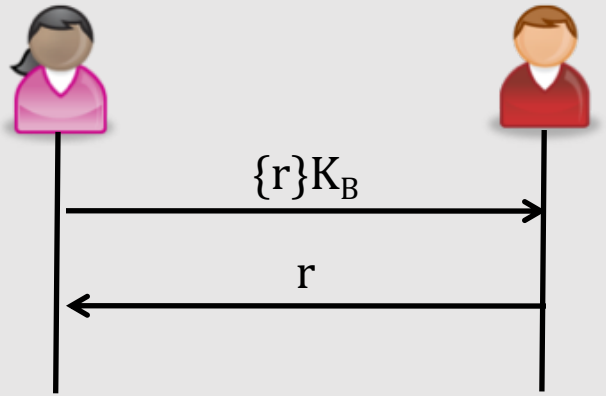
Email address



Qualification/credential

Example of each?

We can also authenticate against keypairs



A type of zero-knowledge proof

What does this prove to Alice?

Dangers / concerns?

- These protocols are much more subtle in practice

Authentication usually involves one or more of three main categories of information

Something you know

Something you have

Something you are

Example of each?

Something you know

Usually, this is a password, but works for any question that can only be answered by the intended recipient

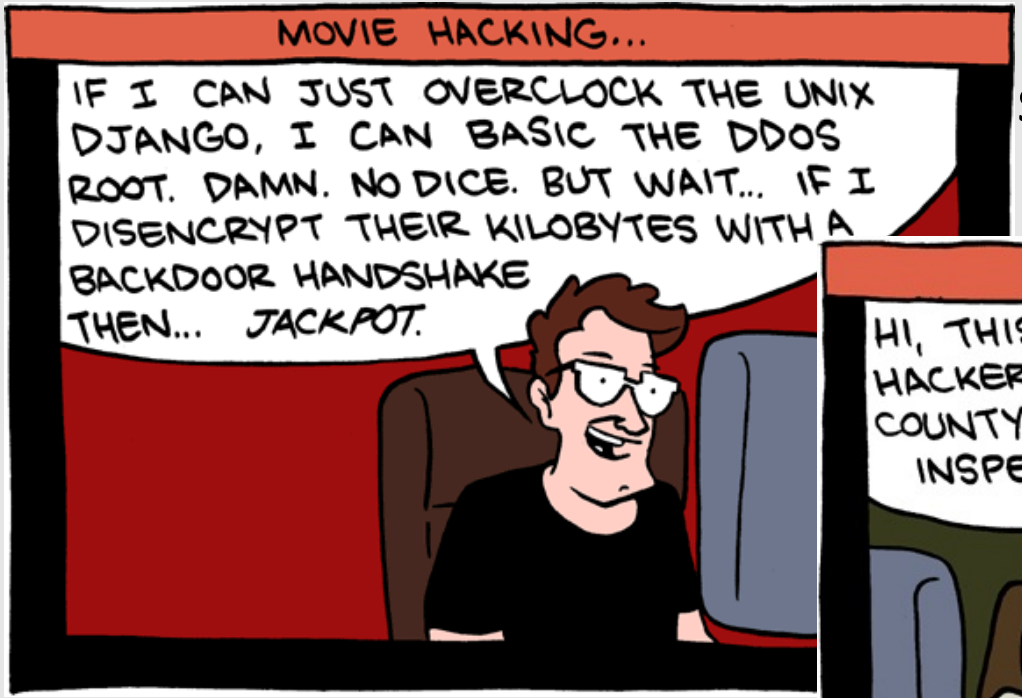
- Assumptions?

Typically, the system simply asks whether the user knows the secret

- Hopefully, over a secure channel
- Compares against the known truth (stored info)
  - Usually complementary—not the password itself
- Why not send, say, hash of the password?

Is it valid to assume only the intended user knows the password?

Attacks against users in password systems



se, phishing, predictable

Common mitigations against humans' password problems

Problem: Humans choose bad passwords, share them often

- How?

Password complexity requirements

- e.g., "Password must contain 3 of 4: uppercase, lowercase, digit, symbol"
- Pros/cons?
- NIST now says: Verifiers SHOULD NOT impose composition rules

Password expiration

- Secrets leak over time, and should be refreshed
- Pros/cons?

Phishing exercises

- Educate to prevent tricky sites stealing passwords

Attacks against passwords, online vs. offline

Online guessing attack: Attempt to authenticate many times

- How can we protect against this?
  - Delay, lock out, etc.
- What if we don't control the gatekeeper? (e.g., mobile phone)

Offline guessing attack: Steal the password database, attempt to recover passwords

- How might this happen?
- Hash functions are preimage resistant; just hash the passwords?
  - What could still go wrong?

Secure password storage

Salt: First, add something unique and random

- NIST: 32 bits

Key derivation: Transform into offline-attack-resistant form

- PBKDF2: Password-Based Key Derivation Function 2
- Idea: Repeatedly hash password (why?)
- $U_1 = \mathrm{PRF}(\mathrm{Password}, \mathrm{Salt})$
  $U_2 = \mathrm{PRF}(\mathrm{Password}, U_1)$
  …
  $U_c = \mathrm{PRF}(\mathrm{Password}, U_{c-1})$
- NIST: Use $c \geq 10{,}000$

Here, PRF is a secure one-way pseudorandom function (e.g., HMAC-SHA2)

This doesn't protect against weak passwords!

How can services protect users from themselves?

- NIST: Monitor break corpuses, notify affected users
  - How, if password is stored securely?

Password managers: Because people are bad at randomness

- Idea: Generate and store uniformly random passwords
- Why is this not subject to "post-it" vulnerability?
- What is required to achieve both security and usability?
- What about phishing?

Something you have

Attacker must be in possession of a trusted device (token)

- Usually, this device stores a long-lived secret that is used to derive one-time PINs
- Server stores the same secret, derives the same codes to verify

Usually, this is part of multi-factor authentication

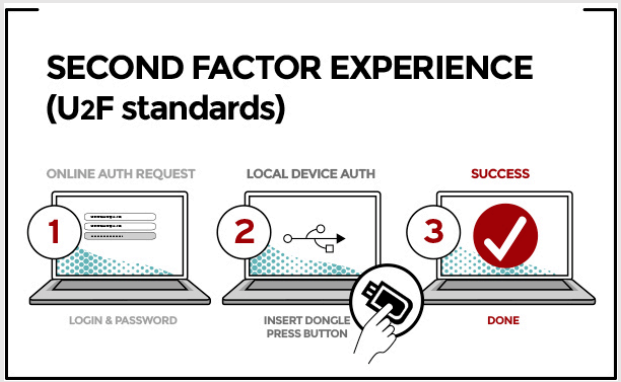
- Require password and code from token
- If you drop your token, not usable alone

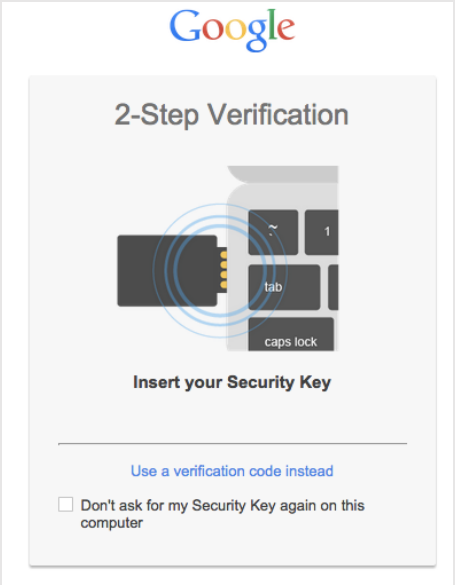
Variants: SMS (issues?), email (sort of), smartphone-as-token

Newer tokens can do mutual authentication to prevent phishing attacks

FIDO Alliance: Open standard for U2F (universal 2-factor) security keys using public-key crypto

- Different key per site, generated and stored with site's public key
- In a phishing attack, public key won't match (or won't be verified), so the "real" info will not be sent
- MITM?





Something you are

Physical authentication has existed since antiquity (and before!)

- Humans and animals are good at recognizing one another
- Instincts and experience cause us to avoid danger
- Uniforms to mark authority

Common biometrics in use today

- Fingerprint
- Iris/retina patterns
- Facial recognition
- Voice recognition

We’re unique in lots of surprising ways

Typing delay patterns

Word choice

Device movement
(while typing, etc.)

Application usage

Biometrics aren't great for everything!

Special care must be used to balance false acceptance rate (FAR) vs. false rejection rate (FRR)

- In what situations have you seen biometrics used? FAR vs. FRR balance for this situation?

Biometrics cannot be revoked or expired!

- How does this change our handling of the secrets?
- Consider secure module's role

Biometrics can seem intrusive

- "Creepy" factor
- Psychological acceptability can be poor

Conclusions

Identities are important, and authentication binds identities to sessions

Usually a combination of something you know/have/are

Different settings call for different approaches!

Next time: Securing data at rest