ECE 443/518 – Computer Cyber Security
Lecture 14 Password Security

Professor Jia Wang
Department of Electrical and Computer Engineering
Illinois Institute of Technology

October 1, 2025

Outline

Password Security

Passwordless Authentication

Midterm Exam

- Lecture 1 $\sim$ Lecture 14, see Homework 1 and 2 for sample.
  - Points may be deducted if key steps are missing.
- Students registered for main campus section: Wed. 10/8, 11:25 AM – 12:40 PM, in class.
  - A physical calculator is allowed. Laptop or any other electronic device or calculator apps running on them are not allowed.
  - Closed book/notes. A letter-size page of cheat sheet is allowed.
- Online students may take the exam as above, or contact Charles Scott (scott@iit.edu) to make arrangement and confirm with me.
  - No make-up exam will be offered if you fail to do so.
- ADA Accommodations: contact Center for Disability Resource (disabilities@iit.edu)
- Emergency/extraordinary reasons for make-up midterm exams are accepted only with documented proof like docter's notes.

Reading Assignment

- This lecture: Password Security
- Next lecture: OpenSSL

Outline

Password Security

Passwordless Authentication

Password Authentication

- A cryptography system based on symmetric cryptography, e.g. Kerberos, inevitably depends on shared secrets between the system and its users.
    - Password: a string that could be memorized by human beings.
- Setup: Alice comes up with a password, and shares it with Bob via a secure channel.
    - This is a secret that none of Alice and Bob should disclose.
- Authentication: Bob asks whoever claims to be Alice to show knowledge of the password.
    - Directly on a secure channel created by public-key cryptography. But what if Bob is not Bob?
    - Or via challenge-response and authenticated encryption.

Why Password Authentication?

- Apparent “advantages”
  - Simple to implement for Bob: compare strings.
  - No additional hardware for Alice: memorize strings.
  - Provide mutual authentication between Alice and Bob.

Why NOT Password Authentication

- "compare strings"
  - Phishing scams: Oscar may pretend to be Bob and obtain the password from Alice directly.
  - Without proper protection, Oscar may obtain Alice's password from where Bob stores passwords, and reuse this password to access other systems Alice is using.
- "memorize strings"
  - Alice may need to authenticate to many Bob's – easy-to-remember passwords are easy for Oscar to guess.
  - Using a password manager that depends on Internet or a device introduces other concerns on usability and security.
- "mutual authentication"
  - Nonrepudiation does not hold – is Alice the only one that has the password to access her bank account?
  - Auditing requires additional evidences.

Password Storage

- Assume Bob need to store many passwords for his customers.
  - What if Oscar stole the file containing these many passwords?
- Hash and salting
  - Instead of storing *password* directly in a file, Bob stores both a random *salt* and $F(salt, password)$ for some function F.
  - So Oscar cannot recover *password* from Bob's password file easily.
  - The function F generally works like hash/MAC so we call it password hashing.
- What if Bob does not use *salt*?
  - Rainbow table attacks: Oscar may precompute hashes for popular passwords and then easily identify them from the file.
- How to design F?
  - Usually by using existing hash and MAC algorithms.
  - Slow them down by running multiple rounds to resist brute-force attacks that recovers *password*.

Password Storage Implementations

- `crypt(3)`, available since 1970's
  - Standard utility used by UNIX/Linux systems for password hashing.
  - Allow to used different hash/MAC algorithms internally, e.g. DES in 70's and SHA512 now.
- `scrypt`, 2009
  - Published as RFC 7914 (2016)
  - Specifically designed to resist brute-force attacks using GPU and ASIC machines by requiring large amount of memory to run efficiently.
- But guess how many websites still store your passwords in plaintext now!

Password Policy and Usability

- Bob may apply password policy to require his customers to use better passwords.
- Rules: length restrictions, no dictionary word, must contain uppercase/lowercase/digits/symbols, etc.
- Aging: require to replace passwords half year, one year, etc.
- Unfortunately, they do impact usability.
  - How about write passwords down on sticky notes?

Multi-Factor Authentication

- Use multiple methods like phone numbers, emails, devices, biometrics, and location to determine the identity
- Trade-off between usability, privacy, security, and regulations.
  - Can a company collect and store biometric data like fingerprints from its customers for authentication purpose?
- The process to reset authentication could be the weakest link!

Outline

Password Security

Passwordless Authentication

Passwordless Authentication

- Multi-factor authentication generally requires to use personal devices like smartphones to obtain necessary information.
- Can we make better use of the device to eliminate password during the authentication process?
  - Use public-key cryptography to provide better usability and security.
  - Reduce the frequency to input passwords by using them only for device setups.
  - Or eliminate the need to create passwords by using mechanisms like emails and phone numbers that don't depend on devices .

FIDO2 (Fast IDentity Online 2)

- An open standard for user authentication without using passwords.
  - Developed during 2010's
  - Mainly consisting of CTAP and WebAuthn.
  - Usually known by consumers as passkeys.
- Client to Authenticator Protocol (CTAP)
  - Define how software (applications, browsers, OS) interact with authenticator hardware (smartphone, fingerprint reader, usb key, etc.)
- Web Authentication (WebAuthn)
  - Define how web applications interact with web services for registration and authentication.

FIDO2 Device Registration

- Registration allows a user to use a web service on a device at a later time without inputing a password.
- The device first creates a public/private key pair.
  - The device should store the private key locally, and protect it by device pins or biometrics.
- The public key is sent to the web service via an authentic channel.
  - With the help of web applications.
  - The authentic channel means that the web service should use some mechanism to validate the identity of the user, preventing man-in-the-middle attacks.
- The web service stores the public key and associates it with the user account.

FIDO2 User Authentication

- User gains access to the device by providing pin or biometrics.
- The device uses the private key to authenticate with the web service via challenge-response.
  - With the help of web applications.
  - Neither man-in-the-middle nor replay attacks are possible.
- What about phishing scams?
  - The attacker cannot reuse the challenge-response to authenticate with the actual web service.
  - However, the attacker may trick the user to reveal sensitive information, e.g. "your passkeys are expired, please provide your password again to generate new ones."

Additional Discussions

- Lost devices or compromised private keys
  - Users need to revoke existing public keys with the web service if device pins or biometrics may be revealed.
  - This is much simpler than PKI: since public keys for PKI are usually used for server authentication, clients are responsible to obtain updates via certificate revocation list (CRL).
- Store private keys online.
  - To share the private key with multiple devices so that user doesn't need to go through the registeration process for multiple times.
  - Improve usability but increase risk of compromised private keys.

Summary

- It seems trivial to make passwords more secure but it isn't.
- Industries are moving toward passwordless authentication using passkeys.