Information Security
CS 526

Topic 7: User Authentication
Readings for This Lecture

• Wikipedia
  • Password
  • Password strength
  • Salt (cryptography)
  • Password cracking
  • Trusted path
  • One time password
Three A’s of Information Security

• Security is about differentiating among authorized accesses and unauthorized accesses
  – Confidentiality, Integrity, Availability all require this

• Authentication
  – Figures out who is accessing

• Access control
  – Ensure only authorized access are allowed

• Auditing
  – Record what is happening, to identify attacks later and recover
Authentication & Access Control according to Wikipedia

• **Authentication** is the act of establishing or confirming something (or someone) as *authentic*, that is, that claims made by or about the subject are true. This might involve confirming the identity of a person, tracing the origins of an artifact, ensuring that a product is what its packaging and labeling claims to be, or assuring that a computer program is a trusted one.

• **Access control** is a system which enables an authority to control access to areas and resources in a given physical facility or computer-based information system.
User Authentication

• Using a method to validate users who attempt to access a computer system or resources, to ensure they are authorized

• Types of user authentication
  – Something you know
    • E.g., user account names and passwords
  – Something you have
    • Smart cards or other security tokens
  – Something you are
    • Biometrics
Variants of Passwords

- Password
- Passphrase
  - a sequence of words or other text used for similar purpose as password
- Passcode
- Personal identification number (PIN)
Scenarios Requiring User Authentication

• Scenarios
  – Logging into a local computer
  – Logging into a computer remotely
  – Logging into a network
  – Access web sites

• Vulnerabilities can exist at client side, server side, or communications channel.
Threats to Passwords

- Eavesdropping (insecure channel between client and server)
- Login spoofing (human errors), shoulder surfing, keyloggers
- Offline dictionary attacks
- Social engineering (human errors)
  - e.g., pretexting: creating and using an invented scenario (the pretext) to persuade a target to release information or perform an action and is usually done over the telephone
- Online guessing (weak passwords)
Guessing Attacks: Two Factors for Password Strength

• The average number of guesses the attacker must make to find the correct password
  – determined by how unpredictable the password is, including how long the password is, what set of symbols it is drawn from, and how it is created.

• The ease with which an attacker can check the validity of a guessed password
  – determined by how the password is stored, how the checking is done, and any limitation on trying passwords
Password Entropy

- The entropy bits of a password, i.e., the information entropy of a password, measured in bits, is
  - The base-2 logarithm of the number of guesses needed to find the password with certainty
  - A password with, say, 42 bits of strength calculated in this way would be as strong as a string of 42 bits chosen randomly.
  - Adding one bit of entropy to a password doubles the number of guesses required.
  - On average, an attacker will have to try half the possible passwords before finding the correct one

- Aka. Guess entropy
Estimating Password Entropy

- People are notoriously remiss at achieving sufficient entropy to produce satisfactory passwords.
- NIST suggests the following scheme to estimate the entropy of human-generated passwords:
  - the entropy of the first character is four bits;
  - the entropy of the next seven characters are two bits per character;
  - the ninth through the twentieth character has 1.5 bits of entropy per character;
  - characters 21 and above have one bit of entropy per character.
- This would imply that an eight-character human-selected password has about 18 bits of entropy.
Towards Better Measurement of Password Entropy

• NIST suggestion fails to consider usage of different category of characters:
  – Lower-case letters, digits, upper-case letters, special symbols
• Orders also matter:
  – “Password123!” should have different entropy from “ao3swPd!2s1r”
• State of art is to use Markov chains to model probability of different strings as passwords
  – May rank “yqzjx” as very secure
• Fundamental challenge: there are different attack strategies out there
Example of Weak Passwords (from Wikipedia)

- Default passwords (as supplied by the system vendor and meant to be changed at installation time): `password`, `default`, `admin`, `guest`, etc.
- Dictionary words: `chameleon`, `RedSox`, `sandbags`, `bunnyhop!`, `IntenseCrabtree`, etc.
- Words with numbers appended: `password1`, `deer2000`, `john1234`, etc.,
- Words with simple obfuscation: `p@ssw0rd`, `l33th4x0r`, `g0ldf1sh`, etc.
- Doubled words: `crabcrab`, `stopstop`, `treetree`, `passpass`, etc., can be easily tested automatically.
Example of Weak Passwords (from Wikipedia)

• Common sequences from a keyboard row: qwerty, 12345, asdfgh, fred, etc.
• Numeric sequences based on well known numbers such as 911, 314159, or 27182, etc.,
• Identifiers: jsmith123, 1/1/1970, 555–1234, "your username", etc.,
• Anything personally related to an individual: license plate number, Social Security number, current or past telephone number, student ID, address, birthday, sports team, relative's or pet's names/nicknames/birthdays, etc.,
  – can easily be tested automatically after a simple investigation of person's details.
Mechanisms to Avoid Weak Passwords

- Allow long passphrases
- Randomly generate passwords where appropriate
  - Though probably inappropriate for most scenarios
- Check the quality of user-selected passwords
  - use a number of rules of thumb
  - run dictionary attack tools
- Give user suggestions/guidelines in choosing passwords
  - e.g., think of a sentence and select letters from it, “It’s 12 noon and I am hungry” => “I’S12&IAH”
  - Using both letter, numbers, and special characters
Balancing Password Entropy & Usability Concerns

• Forcing randomly generated passwords is often bad.
  – A user needs to remember passwords for tens, if not hundreds of accounts
  – High entropy passwords are difficult to remember

• Often times, guessing passwords is not the weakest link
  – One can use various ways to reduce adversary’s abilities to test password guesses
  – When a user cannot remember the password for an account, there must be a way to allow a user to retrieve it.
    • The recovering method either has low security, or costs lots of money
    • It creates a weaker link.

• Usability matters
Storing Passwords (UNIX Case Study)

• Old UNIX
  – The file /etc/passwd stores $H(password)$ together with each user’s login name, user id, home directory, login shell, etc.
    • $H$ is essentially a one-way hash function
  – The file /etc/passwd must be world readable
  – Brute force attacks possible even if $H$ is one-way
    • how to most effectively brute-force when trying to obtain password of any account on a system with many accounts?
Password Salts

• More modern UNIX
  – Divide /etc/password into two files: /etc/password; and /etc/shadow (readable only by root)
• Store \( [r, H(\text{password}, r)] \) rather than \( H(\text{password}) \) in /etc/shadow
  – \( r \) is randomly chosen for each password
  – \( r \) is public, similar to Initial Vector in CBC & CTR modes
• Benefits
  – dictionary attacks much more difficult
    • cost of attacking a single account remains the same
  – if two users happen to choose the same password, it doesn’t immediately show
Mechanisms to Defend Against Dictionary and Guessing Attacks

- Protect stored passwords (use both cryptography & access control)
- Disable accounts with multiple failed attempts
- Require extra authentication mechanism (e.g., phone, other email account, etc.)
Mechanisms to Defend Against Login Spoofing: Trusted Path

- **Login Spoofing Attacks:**
  - write a program showing a login window on screen and record the passwords
  - put su in current directory

- **Defense: Trusted Path**
  - Mechanism that provides confidence that the user is communicating with the real intended server
    - attackers can't intercept or modify whatever information is being communicated.
    - defends attacks such as fake login programs
  - Example: Ctrl+Alt+Del for log in on Windows
    - Causes a non-maskable interrupt that can only be intercepted by the operating system, guaranteeing that the login window cannot be spoofed
Spoofing & Defenses on the Web

• Phishing attacks
  – attempting to acquire sensitive information such as usernames, passwords and credit card details by masquerading as a trustworthy entity in electronic communication.

• Website forgery
  – Set up fake websites that look like e-commerce sites and trick users into visiting the sites and entering sensitive info

• Defense methods
  – Browser filtering of known phishing sites
  – Cryptographic authentication of servers (will talk about in future)
  – User-configured authentication of servers
    • To ensure that the site is the one the human user has in mind
    • E.g., site key, pre-selected picture/phrases
KeyLogging

- Threats from insecure client side
- Keystroke logging (keylogging) is the action of tracking (or logging) the keys struck on a keyboard, typically in a covert manner so that the person using the keyboard is unaware that their actions are being monitored.
- Software-based
  - key-stroke events, grab web forms, analyze HTTP packets
- Hardware-based
  - Connector, wireless sniffers, acoustic based
- Defenses:
  - Anti-spyware, network monitors, on-screen soft keyboard, automatic form filler, etc.
- In general difficult to deal with once on the system
Using Passwords Over Insecure Channels

- One-time passwords
  - Each password is used only once
  - Defend against passive adversaries who eavesdrop and later attempt to impersonate

- Challenge response
  - Send a response related to both the password and a challenge

- Zero knowledge proof of knowledge
  - Prove knowledge of a secret value, without leaking any info about the secret
How to do One-Time Password

- Shared lists of one-time passwords

- Time-synchronized OTP
  - E.g., use $\text{MAC}_K(t)$, where $K$ is shared secret, and $t$ is current time

- Using a hash chain (Lamport)
  - $h(s), h(h(s), h(h(h(s))))$, ..., $h^{1000}(s)$
  - use these values as passwords in reverse order
Lamport’s One-Time Password: Using a Hash Chain

- **One-time setup:**
  - A selects a value \( w \), a hash function \( H() \), and an integer \( t \), computes \( w_0 = H^t(w) \) and sends \( w_0 \) to B
  - B stores \( w_0 \)

- **Protocol:** to identify to B for the \( i^{th} \) time, \( 1 \leq i \leq t \)
  - A sends to B: \( A, i, w_i = H^{t-i}(w) \)
  - B checks \( i = i_A, H(w_i) = w_{i-1} \)
  - if both holds, \( i_A = i_A + 1 \)
Challenge-Response Protocols

- Goal: one entity authenticates to other entity proving the knowledge of a secret, ‘challenge’
- Approach: Use time-variant parameters to prevent replay, interleaving attacks, provide uniqueness and timeliness
  - e.g., nounce (used only once), timestamps
Challenge-response based on symmetric-key crypto

- **Unilateral authentication, timestamp-based**
  - A to B: $\text{MAC}_K(t_A, B)$

- **Unilateral authentication, nounce-based**
  - B to A: $r_B$
  - A to B: $\text{MAC}_K(r_B, B)$

- **Mutual authentication, nounce-based**
  - B to A: $r_B$
  - A to B: $r_A$, $\text{MAC}_K(r_A, r_B, B)$
  - B to A: $\text{MAC}_K(r_B, r_A)$
Other Defenses

- Alternatives to passwords
  - graphical passwords
- Go beyond passwords
  - security tokens
  - biometrics
  - 2-factor authentication
    - Uses two independent authentication methods
    - US Banks are required to use 2-factor authentication by end of 2006 for online banking
  - Out of band authentication: uses a channel other than the internet
    - E.g., phone
Open Problems

- Better measure of password quality.
- Better ways to make people choose more secure passwords
- Alternatives to passwords?
  - The secret should be easy to remember, difficult to guess, and easy to enter into the system.
- Better ways to make user choose stronger passwords?
- Better ways to use other devices for authentication
- Effective 2-factored and/or out of band authentication for the Web
- Phishing defense
Coming Attractions …

- Web Security