Chapter 3(3.1 - 3.6)

User Authentication:

- *Identification* → means by which a user provides a claimed identity to the system
- User authentication → means of establishing the validity of the claim

The initial requirement for performing user authentication is that the **user must be registered** with the system.

An applicant applies to a **registration authority (RA)** to become a subscriber of a **credential service provider (CSP).**

- RA is a trusted entity
 - That establishes and vouches for the identity of an applicant to a CSP.
- The CSP then engages in an exchange with the subscriber.
 - The CSP issues some sort of electronic credential to the subscriber.
 - The credential (a data structure) binds an identity to a token possessed by the subscriber

Once a user is registered as a subscriber

- the actual authentication process can take place between the subscriber and one or more systems
 - that perform authentication and, subsequently, authorization

A Model for Digital User Authentication

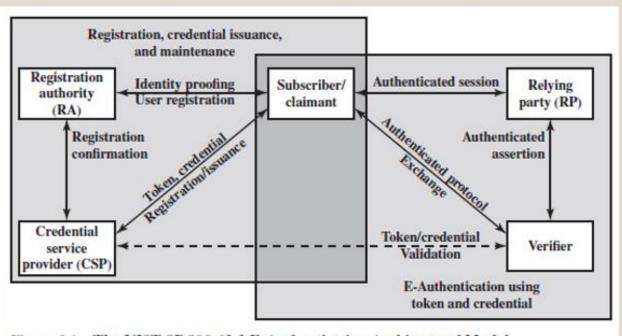


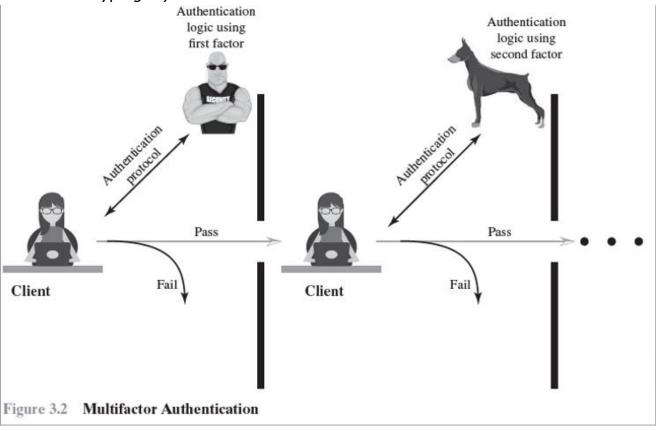
Figure 3.1 The NIST SP 800-63-3 E-Authentication Architectural Model

Authentication is the process of verifying the identity of a user, device, or system. It ensures that someone or something is who or what it claims to be. This is typically done by requiring a user to provide a set of credentials, such as a username and password, or by using other forms of identification such as a fingerprint or a security token.

Authorization, on the other hand, is the process of granting or denying access to resources or actions based on the authenticated identity. Once a user has been authenticated, the system can use authorization to determine what actions the user is allowed to perform.

Means of Authentication:

- Something the individual knows:
 - Examples include a password, a personal identification number (PIN), or answers to a prearranged set of questions.
- Something the individual possesses:
 - Examples include electronic keycards, smart cards, and physical keys.
 - This type of authenticator is referred to as a token.
- Something the individual is (static biometrics):
 - o Examples include recognition by fingerprint, retina, and face.
- Something the individual does (dynamic biometrics):
 - Examples include recognition by voice pattern, handwriting characteristics, and typing rhythm



PASSWORD-BASED AUTHENTICATION:

- A widely used line of defense against intruders is the password system
 - User provides name/login and password
 - System compares password with the one stored for that specified login
- The user ID:
 - Determines that the user is authorized to access the system
 - Determines the user's privileges
 - Is used in discretionary access control

The Vulnerability of Passwords:

1. **Offline dictionary attack:** The attacker obtains the *system password file* and *compares* the password hashes *against* of commonly used passwords.

Countermeasures:

- a. Controls to prevent unauthorized access to the password file
- b. Intrusion detection measures to identify a compromise
- c. Rapid reissuance of passwords should the password file be compromised.
- 2. **Specific account attack:** The attacker targets a **specific account** and **submits password quesses** until the correct password is discovered.

Countermeasure:

- a. An account lockout mechanism, which locks out access to the account after a number of failed login attempts. Typical practice is no more than five access attempts.
- 3. Popular password attack (against a wide range of IDs): A variation of the preceding attack is to use a popular password and try it against a wide range of user IDs.

Countermeasures:

- a. policies to inhibit the selection by users of common passwords and
- b. Scanning the IP addresses of authentication requests and client cookies for submission patterns.
- 4. **Password guessing against single user:** The attacker attempts to **gain knowledge** about **the account holder and system password policies** and uses that knowledge to **guess** the password.

Countermeasures:

- a. Training in and **enforcement of password policies** that make passwords **difficult** to guess.
- b. Such policies address the secrecy
 - i. minimum length of the password
 - ii. character set
 - iii. prohibition against using well-known user identifiers, and
 - iv. length of time before the password must be changed.

- 5. Workstation hijacking: The attacker waits until a logged-in workstation is unattended.

 Countermeasure:
 - a. Automatically logging the workstation out after a period of inactivity.
 - b. Intrusion detection schemes can be used to detect changes in user behavior.
- 6. **Exploiting user mistakes:** If the system assigns a password, then the user is more likely to write it down because it is difficult to remember
 - Countermeasures:
 - i. User training
 - ii. intrusion detection, and
 - iii. Simpler passwords combined with MFA.
- 7. **Exploiting multiple password use:** Attacks can also become much more effective or damaging if different network devices share the same or a similar password for a given user.

Countermeasures

- Include a policy that forbids the same or similar password on particular network devices.
- 8. **Electronic monitoring:** If a password is **communicated across a network** to log on to a remote system, it is vulnerable **to eavesdropping (E.g. sniffing).**

Countermeasures:

• We need additional techniques

UNIX File System:

 already	came in	mid 1	
 aireauv	came in	IIIIQ 1	

- Many systems now use MD5
 - with 48-bit salt
 - password length is unlimited
 - is hashed with 1000 times inner loop
 - produces 128-bit hash
- OpenBSD uses Blowfish block cipher based and hash algorithm called Bcrypt
 - uses 128-bit salt to create 192-bit hash value

Password vulnerabilty of hashed password using salt:

Dictionary attacks

- Develop a large dictionary of possible passwords and try each <u>against</u> the password file
- Each password must be hashed using each salt value and then compared to stored hash values

Exploiting easily guessable passwords

Password crackers exploit the fact that people choose easily guessable passwords

 Shorter password lengths are also easier to crack

Rainbow table attacks

- Pre-compute tables of hash values for all salts
- less computer processing time and more storage than a brute-force attack
- Can be <u>countered</u> by using a <u>sufficiently</u> large salt value and a sufficiently large hash length

John the Ripper

- Open-source password cracker first developed in in 1996
- Uses a combination of brute-force and dictionary techniques

Password File Access Control

- Can block offline guessing attacks by denying access to encrypted passwords
 - make available only to privileged users
 - often using a separate shadow password
- Still have vulnerabilities
 - Weakness in the OS that allows access to the file
 - Accident with permissions making it readable
 - Users with same password on other systems
 - Access from unprotected backup media
 - Sniff passwords in unprotected network traffic

Password select strategies

- Goal to eliminate guessable passwords
 - Still easy for user to remember

Techniques

User education

Users can be told the importance of using hard to guess passwords and can be provided with guidelines for selecting strong passwords

Computer generated passwords

Users have trouble remembering them

Reactive password checking

System periodically runs its own password cracker to find guessable passwords

(the attacker may be faster than the reactive password checker; may crack passwords before reactive password checker determines their weakness)

Complex password policy (proactive)

User is allowed to select their own password, however the system checks to see if the password is allowable, and if not, rejects it

Goal is to eliminate guessable passwords while allowing the user to select a password that is memorable

Proactive Password Checking:

- Rule enforcement plus user advice, e.g.
 - 8+ chars, upper/lower/numeric/punctuation
 - may not suffice
- Password cracker
 - Procedure is simply to compile a large dictionary of possible "bad" passwords.
 - When a user selects a password, the system checks to make sure that it is not on the disapproved list.
 - time and space issues
- Bloom Filter
 - use to build table based on dictionary using hashes
 - check desired password against this table

TOKEN-BASED AUTHENTICATION:

Objects that a user possesses for the purpose of user authentication are called tokens.

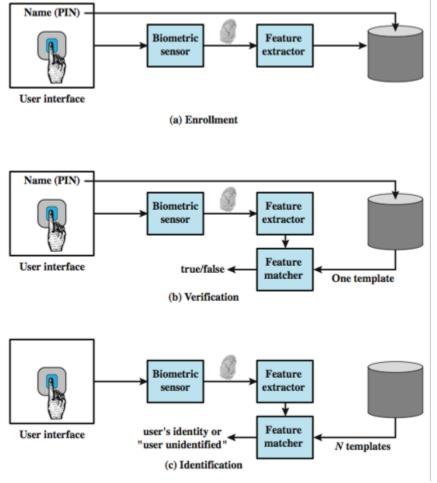
- 1. Memory Card:
 - a. Can store but do not process data
 - b. Provides significantly greater security when combined with a password or PIN
 - c. Drawbacks of memory cards include:
 - i. Requires a special reader
 - ii. Loss of token
 - iii. User dissatisfaction:

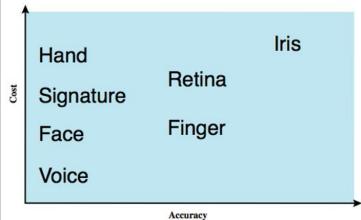
2. Smart token:

- a. Smart tokens include an embedded microprocessor.
- b. Other tokens can look like calculators, keys, or other small portable objects
- c. Has an electronic interface
- d.
- e. Authentication protocol:
 - i. Static:
 - 1. The user authenticates himself or herself to the token and then the token authenticates the user to the computer.
 - ii. Dynamic password generator
 - 1. The token generates a unique password periodically (e.g., every minute).
 - iii. Challenge-response:
 - The computer system generates a challenge, such as a random string of numbers. The smart token generates a response based on the challenge.
 - 2. For example, public-key cryptography could be used and the token could encrypt the challenge string with the token's private key.
 - 3. The simplest example of a challenge-response protocol is <u>password</u> authentication, where the challenge is asking for the password and the valid response is the correct password.

3. Biometric Authentication:

- a. Authenticate an individual based on unique physical characteristics
- b. complex and expensive



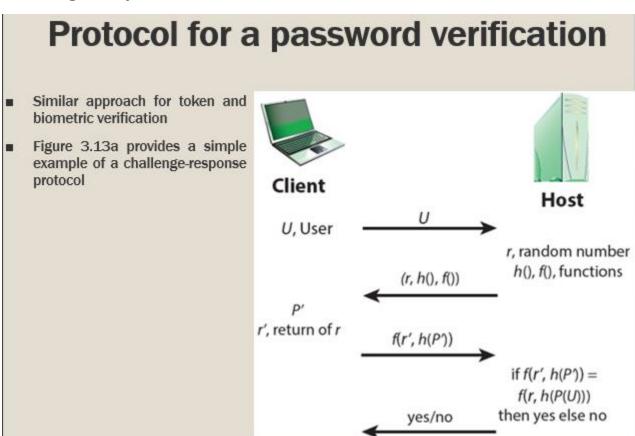


REMOTE USER AUTHENTICATION:

Authentication over a network, the Internet, or a communications link is more complex Additional security threats such as:

■ Eavesdropping, capturing a password, replaying an authentication sequence that has been observed

Challenge Response Protocol:



- The host generates a random number r, often called a **nonce**, and returns this nonce to the user. In addition, the host specifies two functions, h() and f(), to be used in the response. This transmission from host to user is the challenge.
- The user's response is the quantity f(r, h(P)), where r = r and P is the user's password. The function h is a hash function, so the response consists of the hash function of the user's password combined with the random number using the function f.
- The host stores the hash function of each registered user's password, depicted as h(P(U)) for user U.
- When the response arrives, the host compares the incoming f(r, h(P)) to the calculated f(r, h(P(U))). If the quantities match, the user is authenticated.

SECURITY ISSUES FOR USER AUTHENTICATION

Denial-of-Service

Attempts to disable a user authentication service by flooding the service with numerous authentication attempts

Eavesdropping

Adversary attempts to learn the password by some sort of attack that involves the physical proximity of user and adversary

Host Attacks

Directed at the user file at the host where passwords, token passcodes, or biometric templates are stored

AUTHENTICATION SECURITY ISSUES

Trojan Horse
An application or
physical device
masquerades as an
authentic application
or device for the
purpose of capturing a
user password,
passcode, or biometric

Client Attacks

Adversary attempts to achieve user authentication without access to the remote host or the intervening communications path

Replay

Adversary repeats a previously captured user response