06. User Authentication

이형태

2019학년도 2학기

User Authentication

- User authentication defined by RFC 4949: The process of verifying an identity claimed by or for a system entity
- Consist of the following two steps:
  1. **Identification step:** Presenting an identifier to the security system
  2. **Verification step:** Presenting or generating authentication information that corroborates the binding between the entity and the identifier
- Fundamental building block and the primary line of defense
- Basis for most types of access control and for user accountability

Electronic User Authentication: Model

- Electronic user authentication defined by NIST SP 800-63-2:
  - The process of establishing confidence in user identities that are presented electronically to an information system
  - Systems can use the authenticated identity to determine if the authenticated individual is authorized to perform particular functions

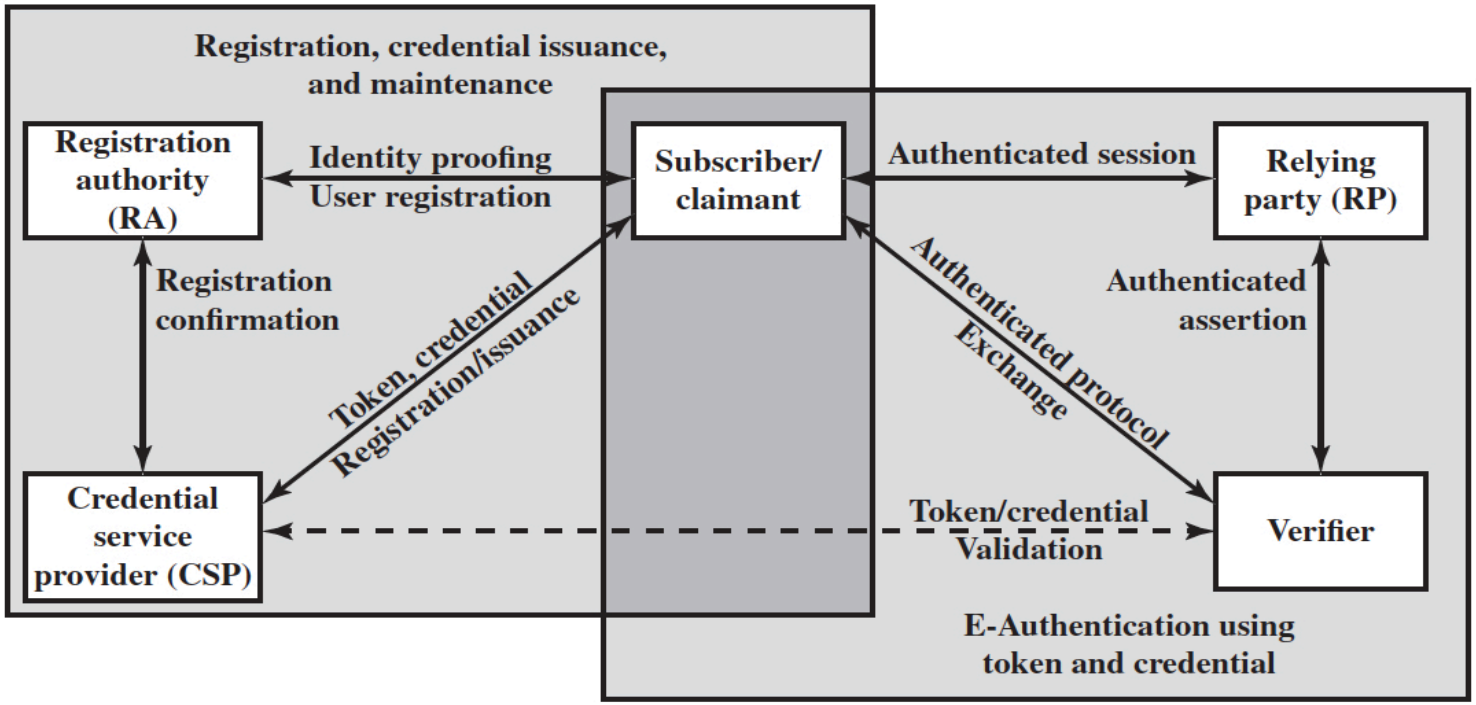


Figure 3.1 **The NIST SP 800-63-2 E-Authentication Architectural Model**

Picture from [SB15]

Means of Authentication

- Means of authentication: Can be used alone or in combination
  - Something the individual knows, e.g., password, personal identification number (PIN), answers to a prearranged set of questions
  - Something the individual process (token), e.g., keycards, smart cards, and physical keys
  - Something the individual is (static biometrics), e.g., recognition by fingerprint, retina and face
  - Something the individual does (dynamic biometrics), e.g., recognition by voice pattern, handwriting characteristics, and typing rhythm
- But, each method has problems, e.g.,
  - Attacks by guessing or stealing a password, and forging or stealing a token
  - Heavy costs for managing password and tokens
  - False positive, false negative, user acceptance, costs, and convenience

Password-Based Authentication

Password-Based System

- Widely used line of defense against intruders
- Compare the password to a previously stored password for the ID and maintain in a system password file
- Roles of ID:
  - Determine whether the user is authorized to gain access to a system
  - Determine the privileges accorded to the user, e.g. superuser and guest
  - Be used in what is referred to as discretionary access control
- Role of password: Serve to authenticate the ID of the individual on to the system

Vulnerability of Passwords

- Offline dictionary attack
- Specific account attack
- Popular password attack
- Password guessing against single user
- Workstation hijacking
- Exploiting user mistakes
- Exploiting multiple password use
- Electronic monitoring

Table: Top 25 most common passwords[a]

1	123456
2	password
3	123456789
4	12345678
5	12345
6	111111
7	1234567
8	sunshine
9	qwerty
10	iloveyou
11	princess
12	admin
13	welcome
14	666666
15	abc123
16	football
17	123123
18	monkey
19	654321
20	!@#$%^&*
21	charlie
22	aa123456
23	donald
24	password1
25	qwerty123

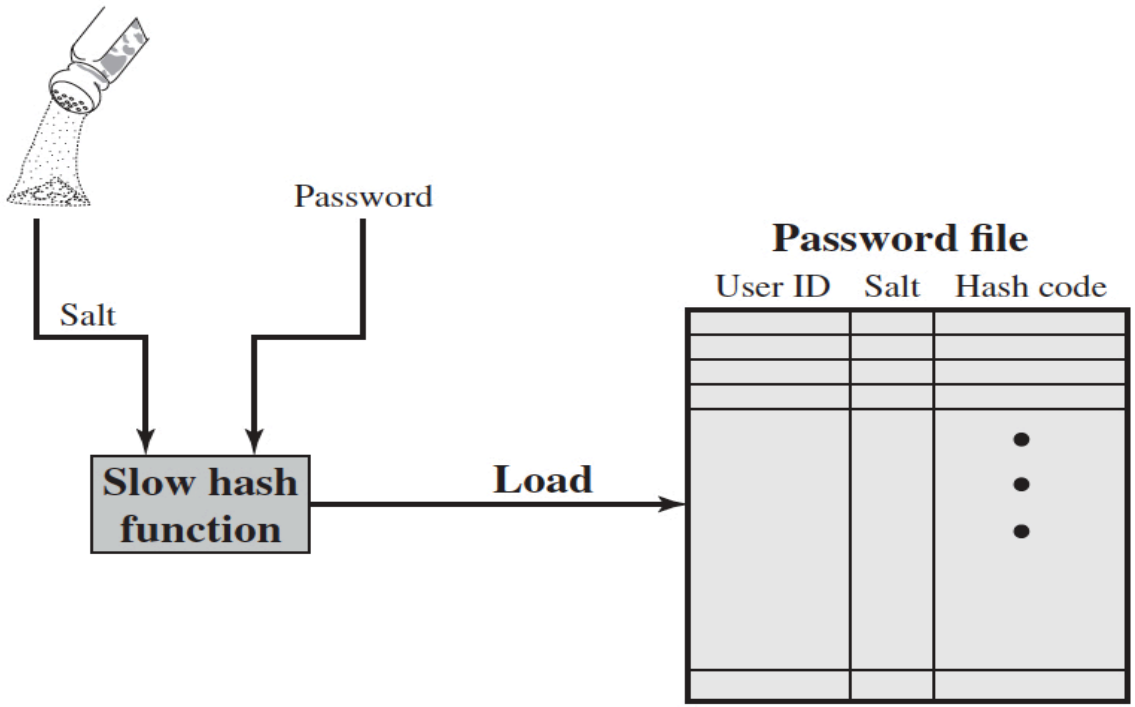
[a] Top 25 Worst Passwords You Should Never Use (by SplashData)
https://en.wikipedia.org/wiki/List_of_the_most_common_passwords

Hashed Passwords

- The use of hashed passwords and a salt value
- Widely used password security technique
- Found on all UNIX variants as well as on a number of other operating systems
- Roles of the salt
  1. Prevent duplicate passwords from being visible in the password file
  2. Increase the difficulty of offline dictionary attacks
  3. Hard to find out whether a person with passwords on two or more systems has used the same password on all of them

UNIX Password System I

- To load a new password into the system
  - The user selects a new password.
  - It is combined with a fixed-length salt value related to the time in the older system or pseudorandom/random numbers in the newer system
  - The system stores the hash value of the password and the salt value, together with the corresponding ID and the salt value.

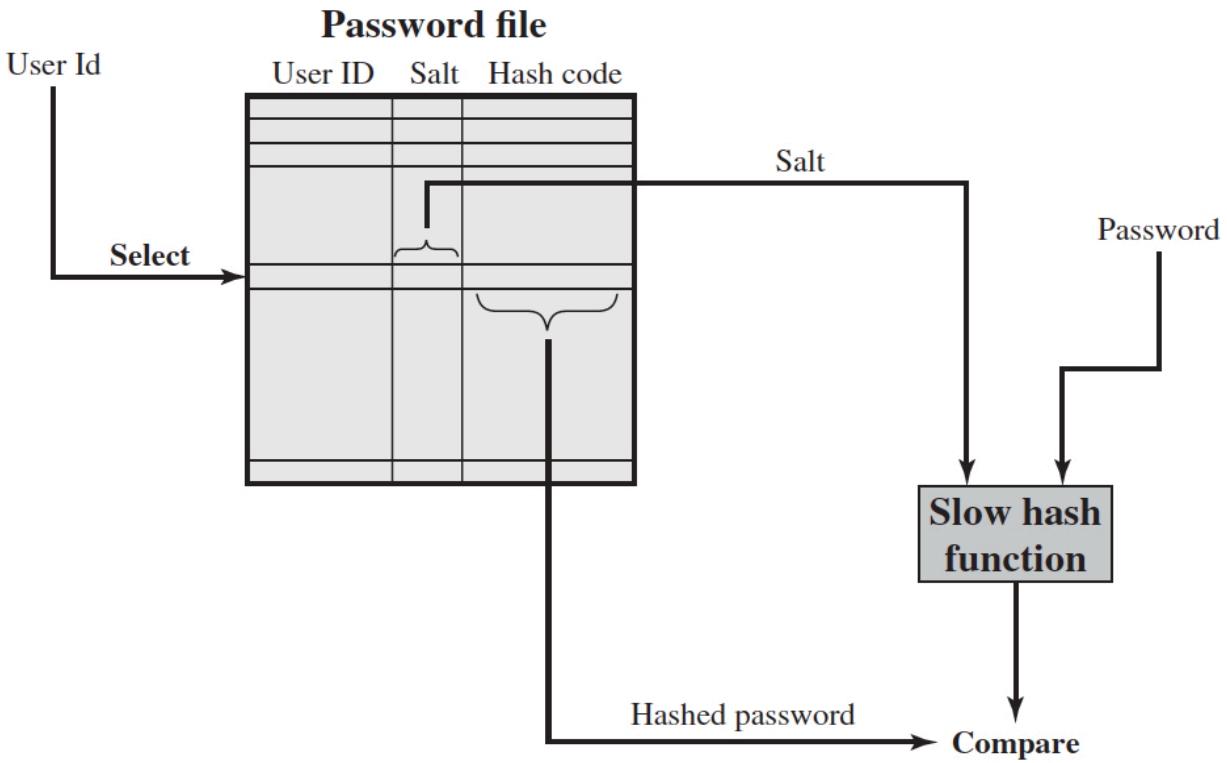


(a) Loading a new password

Picture from [SB15]

UNIX Password System II

- To log on the system
  - The user provides an ID and a password
  - The system retrieves the salt value and the hashed password from the password file using the ID.
  - The system checks the hashed password using the retrieved salt value and the received password.



Picture from [SB15]

UNIX Implementation I

- UNIX Implementation I (Original version)
  - Password: $\leq$ 8 printable characters in length (= 56-bit value using 7-bit ASCII) $\Rightarrow$ DES Key
  - Using this key, encrypt a 64-bit block of zeros
  - Repeat 25 times with output
  - Salt: 12-bit
  - Slow hash: Obtained by modifying DES to convert into a one-way function
- Attack: Dictionary attack over 50 million password guesses in about 80 minutes using supercomputer in 2003

UNIX Implementation (Cont.)

- UNIX Implementation II (Linux, Solaris, and FreeBSD)
  - Slow hash: Hash function such as MD5 (with 1000 iterations)
  - Password: arbitrary length
  - Salt: $\leq$ 48-bit
  - Output: 128-bit hash value
- UNIX Implementation III (OpenBSD)
  - Slow hash: Based on Blowfish symmetric block cipher (called Bcrypt), encrypt the 192-bit magic value "OrpheanBeholderScryDoubt" 64 times using Bcrypt in ECB mode
  - Salt: 128-bit
  - Expensive key schedule
- Other hash functions, like SHA-256 and SHA-512, can also be applied.

Password Cracking

- Dictionary attack
  - Develop a large dictionary of possible passwords and try each of these against the password file
  - Check a hash value of each password and salt, and variations
- Time-memory-trade-off using rainbow table
  - Generate a large dictionary of possible password with each possible salt $\Rightarrow$ **rainbow table**
  - In 2003, using 1.4 GB of data, 99.9% of all alphanumeric Windows password hashes can be cracked in 13.8 seconds.
- Password cracking techniques have improved due to
  - the improvement of machines and
  - the leakage of passwords by hackers and their pattern analysis

Password File Access Control

- Deny the opponent access to the password file to prevent a password attack
  - Allow privileged users only to access
  - Shadow password file: The hashed passwords are kept in a separate file from the use IDs
- Remain potential vulnerabilities
  - Unanticipated break-ins (by hackers)
  - Accident of protection that makes it readable
  - Use of the same password on different machines
  - Access to an unprotected backup device
  - Catching passwords by sniffing network traffic

Password Selection Strategies

- Goal: To eliminate guessable passwords while easy to remember
- Basic strategies
  - User education
    - ★ e.g., Use the first letter of each word of a phrase
  - Computer-generated passwords
    - ★ A password looks like random, but is hard to remember.
  - Reactive password checking
    - ★ The system periodically runs its own password cracker to find guessable passwords.
    - ★ The system cancels any passwords that are guessed and notifies the user.
  - Complex password policy (Proactive password checker)
    - ★ A user selects his or her own password.
    - ★ The system checks to see if the password is allowable and, if not, rejects it.

Proactive Password Checker

- Need a balance between user acceptability and strength
- Method I: Rule enforcement, e.g.,
  - All passwords must be at least eight characters long.
  - In the first eight characters, the passwords must include at least one each of uppercase, lowercase, numeric digits, and punctuation marks.
  - Not perfectly secure against password cracking.
- Method II
  - Compile a large dictionary of possible "bad" passwords and reject if a password is in the dictionary
  - Has two problems: Space to store a dictionary, Time to search a dictionary
- Method III: Use Bloom filter

Bloom Filter

- A **Bloom filter** of order k consists of a set of k independent hash functions $H_1(x), \ldots, H_k(x)$, where each function maps an element in a set of cardinality D into a hash value in the range 0 to $N-1$, i.e,

$$H_i(X_j) = y$$

  where $1 \leq i \leq k$, $1 \leq j \leq D$, and $0 \leq y \leq N-1$.

- Utilized in many applications in the security area

Password Checker Based on Bloom Filter

- Procedure
  1. A hash table of N bits is defined, with all bits initially set to 0.
  2. For each password, its k hash values are calculated, and the corresponding bits in the hash table are set to 1; if the bit already has the value 1, it remains at 1.
  3. Once a new password is given, compute its k hash values. If all the corresponding bits of the hash table are equal to 1, then the password is rejected.
- All password in the dictionary will be rejected.
- There are some "false positive", e.g.,
  - Suppose that the passwords *undertaker* and *hulkhogan* are in the dictionary.
  - *xG%#jj98* is not in the dictionary.
  - If

$$H_1(undertaker) = 25, H_1(hulkhogan) = 83, H_1(xG\%\#jj98) = 665$$
$$H_2(undertaker) = 998, H_2(hulkhogan) = 665, H_2(xG\%\#jj98) = 998,$$

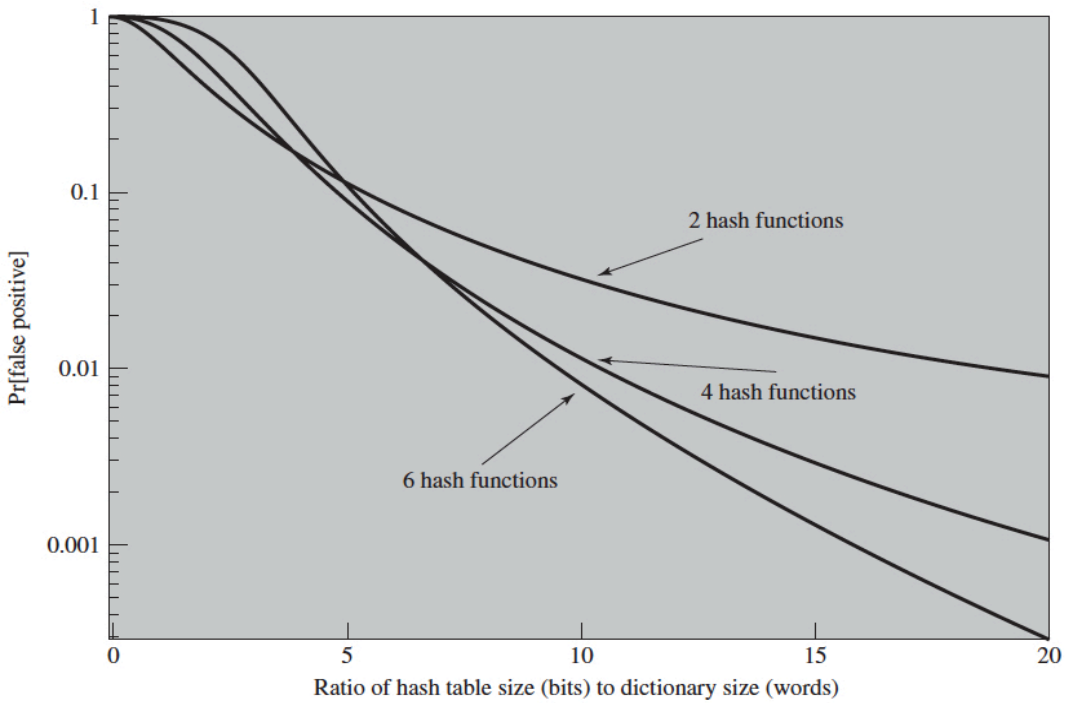
    then *xG%#jj98* will be rejected.

Performance off Bloom Filter

- The probability of a false positive

$$P = \left(1 - \left(1 - \frac{1}{N}\right)^{kD}\right)^k \approx (1 - e^{-kD/N})^k$$

where k is the number of hash functions, N is the number of bits in hash table, and D is the number of words in dictionary.



Picture from [SB15]

Token-Based Authentication

Types of Cards Used as Tokens

- Token: objects that a user possesses for the purpose of user authentication

Card Type	Defining Feature	Example
Embossed	Raised characters only, on front	Old credit card
Magnetic stripe	Magnetic bar on back, characters on front	Bank card
Memory	Electronic memory inside	Prepaid phone card
Smart	Electronic memory and processor inside	Biometric ID card
Contact	Electrical contacts exposed on surface	
Contactless	Radio antenna embedded inside	

Memory Cards

- Can store, but do not process data
- e.g) Bank card with a magnetic stripe on the back
- Can be used alone for physical access, e.g., a key for hotel room
- Provided by combining with password or personal identification number (PIN)
- Advantage: Adversary must obtain a physical possession of the card and know the password/PIN.
- Drawbacks:
  - Need a special reader
  - Token loss
  - User dissatisfaction

Smart Cards

- Categorized along four dimentions
  - Physical characteristics: Include an embedded microprocessor.
  - User interface: Include a keypad and display for human/token interaction
  - Electronic interface: Communicate with a reader/writer
    - ⋆ Contact
    - ⋆ Contactless
  - Authentication protocol
    - ⋆ Static: The user authenticates himself or herself to the token and the token authenticates the user to the computer.
    - ⋆ Dynamic: The token generates a unique password periodically and it is entered into the computer system.
    - ⋆ Challenge-response: The computer system generates a challenge a challenge and a smart token generates a response based on the challenge.

Biometric Authentication

Biometric Authentication

- Based on user's unique physical characteristics
- Complex and expensive

- Kinds of physical characteristics
  - Facial characteristics
  - Fingerprints
  - Hand geometry
  - Retinal pattern
  - Iris

  - Signature
  - Voice

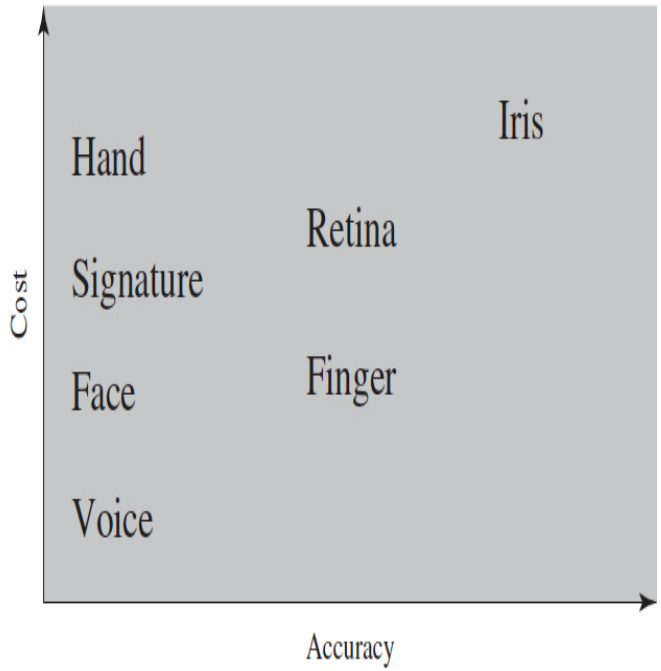


Figure 3.7 Cost versus Accuracy of Various Biometric Characteristics in User Authentication Schemes

Picture from [SB15]

Fingerprint

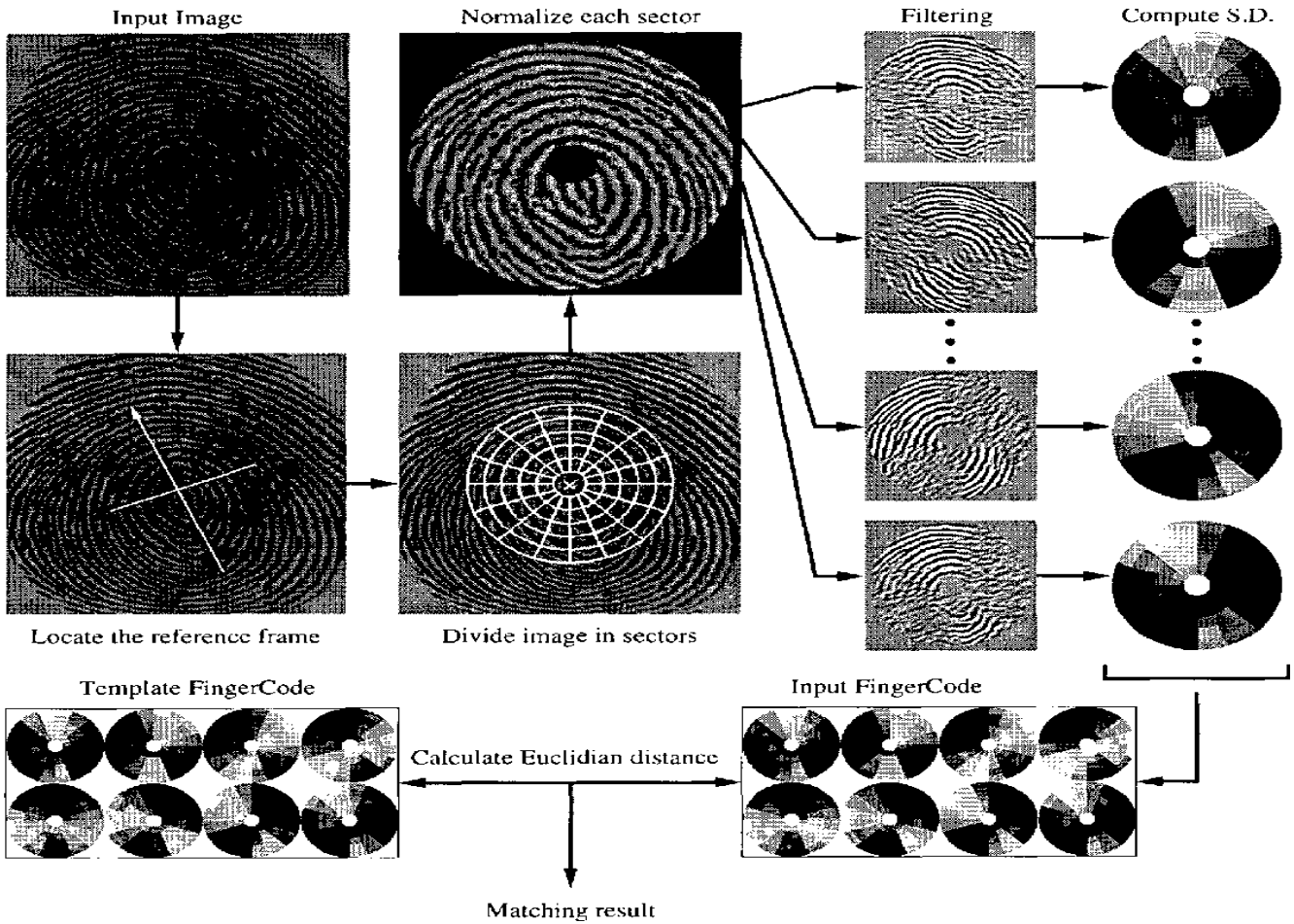


Image from https://www.semanticscholar.org/paper/FingerCode%3A-A-Filterbank-for-Fingerprint-and-Jain-Prabhakar/a91eca9d11755108c8c1a4354ed5b6c5a89ca4f8

Image from https://www.bayometric.com

Iris

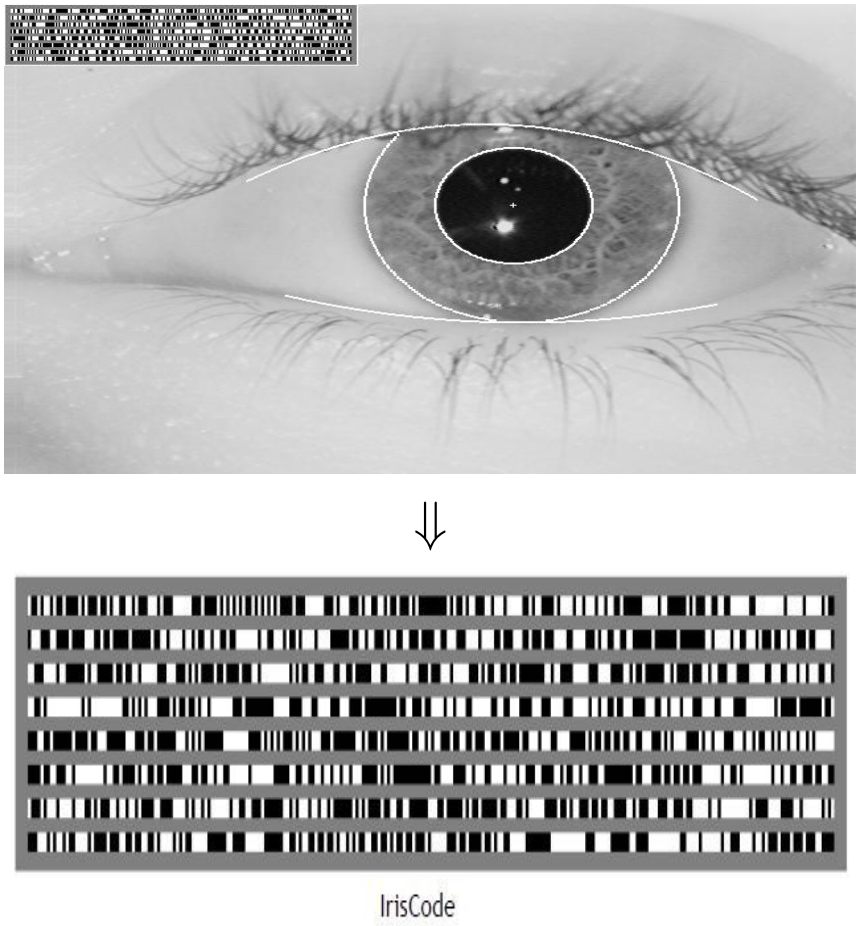
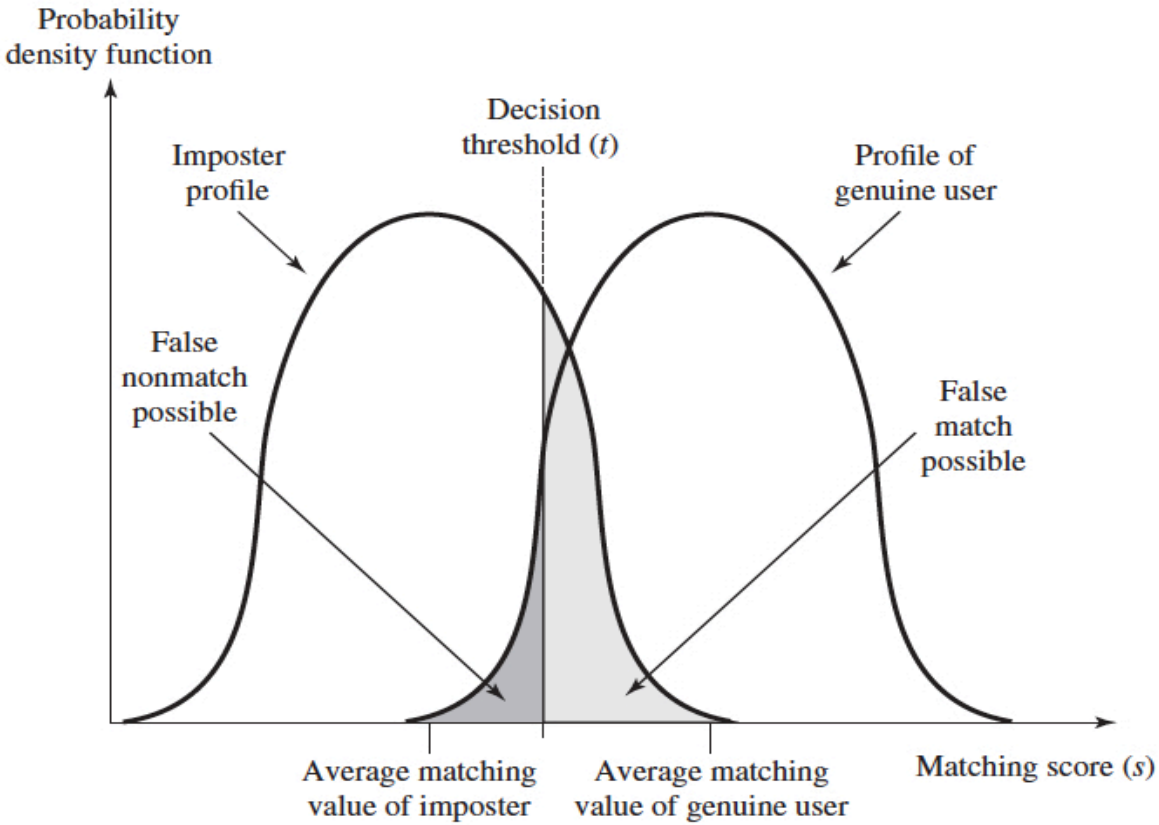


Image from https://www.cl.cam.ac.uk/ jgd1000/

Image from http://www.iritech.com/iris-biometric-barcode

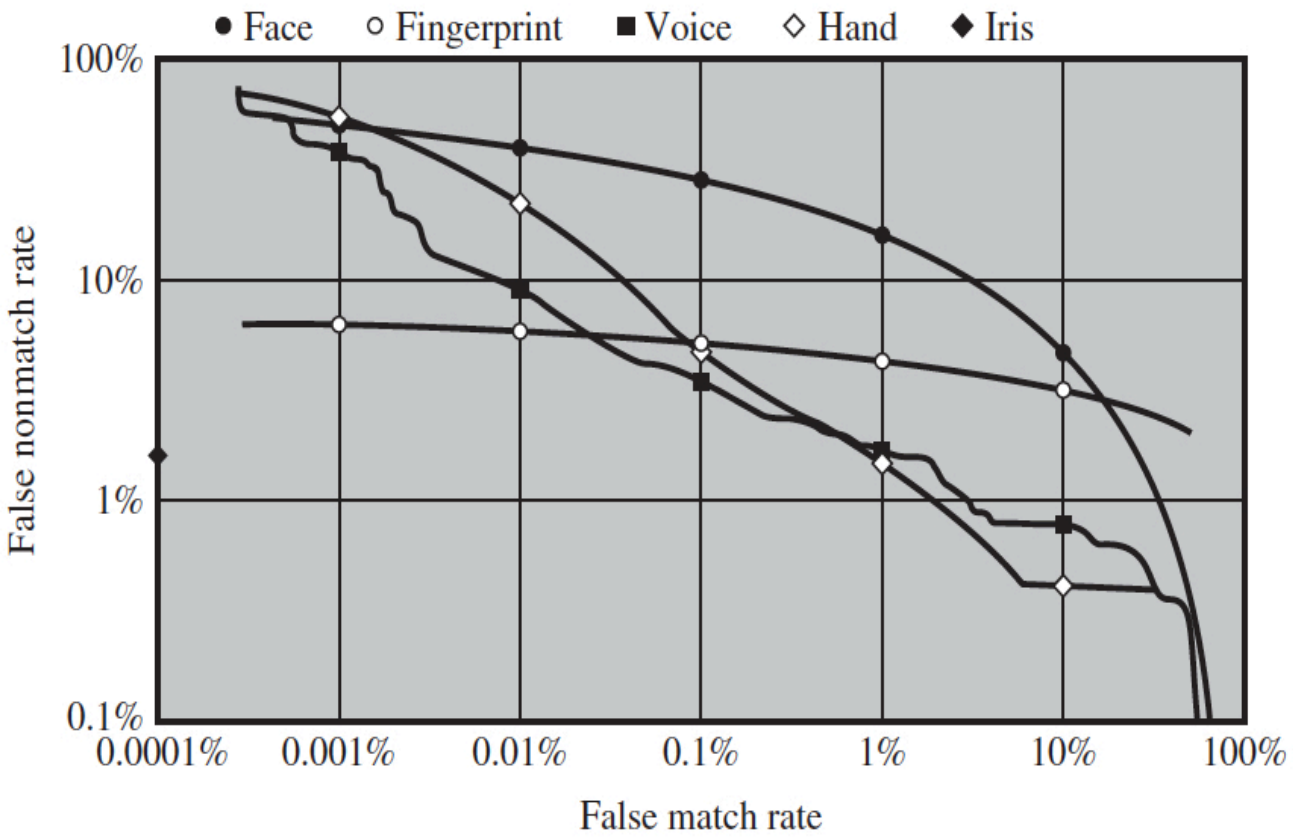
Biometric Accuracy



- False nonmatch possible = False positive
- False match possible = False negative

Picture from [SB15]

Actual Biometric Measurement Operating Characteristic Curves



- Reported in [MANSO1]

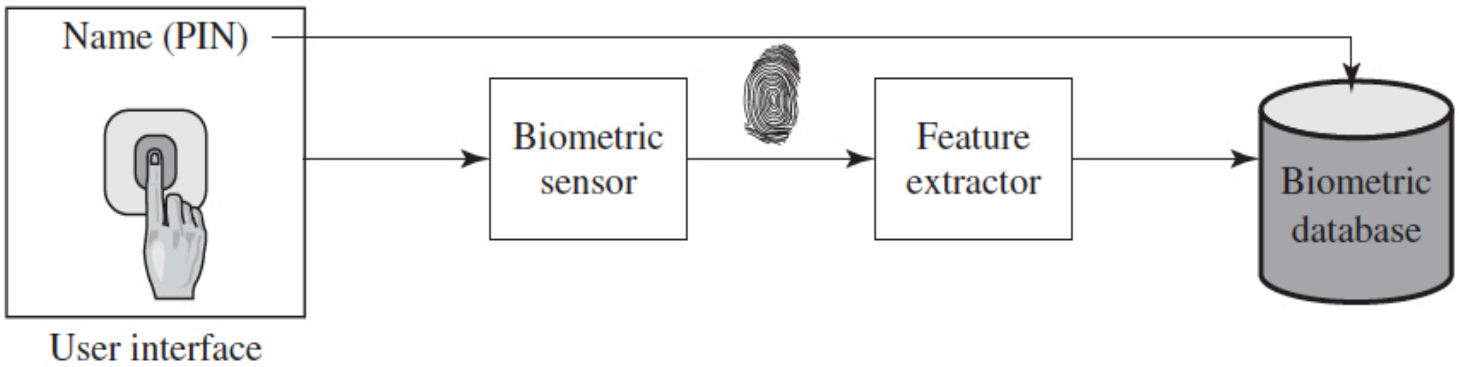
Picture from [SB15]

Specification of Biometric Information

	Iris	Fingerprint	Face	Voice
Template Type	Binary	Integer (0-255)	Floating	Floating
Distance	Hamming Distance	Euclidean Distance	Cosine Similarity	Cosine Similarity
Dimension	2048/24000	640	448	50
FAR	0.01 %	1 %	1 %	1%
FRR	2 %	12 %	5 %	5%
Threshold	0.31	190000	0.79	0.05

Generic Biometric System: Enrollment

- The user present a name and some types of password or PIN to the system.
- At the same time, the system senses biometric characteristics of this user.
- The system digitalizes the input and then extract a set of features (called template) that can be stored.

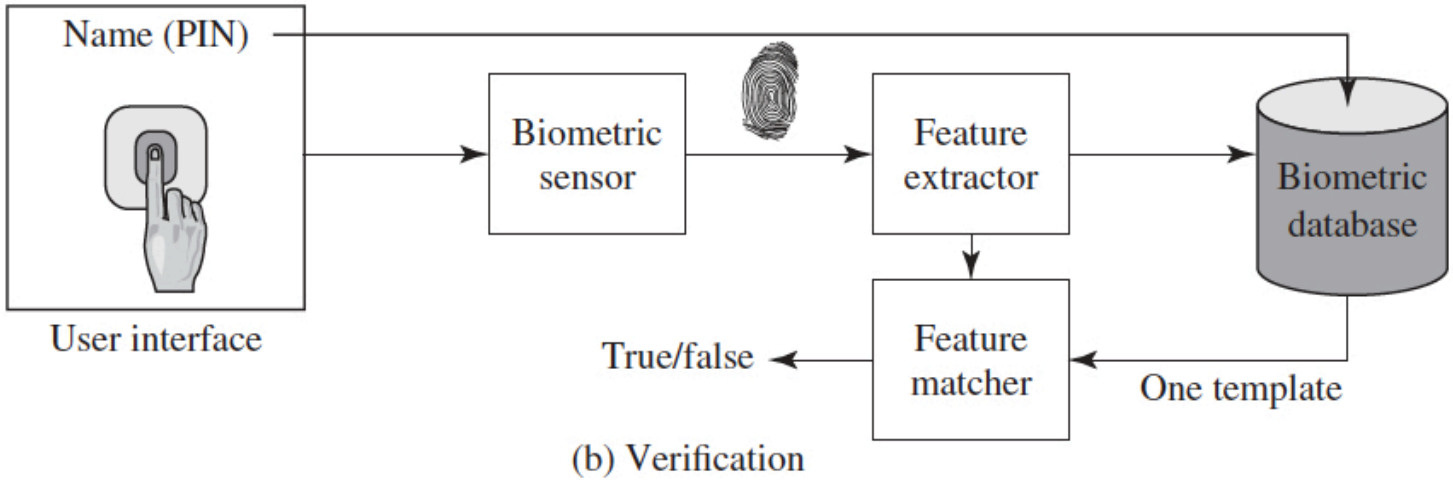


(a) Enrollment

Picture from [SB15]

Generic Biometric System: Verification

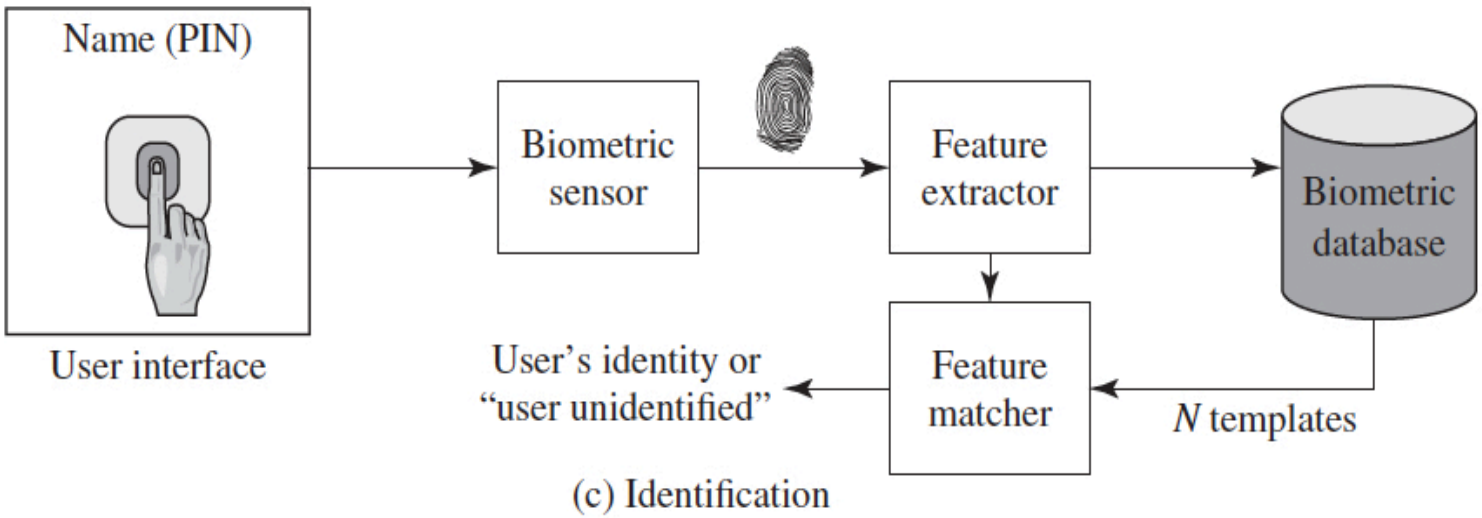
- The user enters a password/PIN and also uses a biometric sensor.
- The system extracts the corresponding feature and compares that to the template stored for this user.



Picture from [SB15]

Generic Biometric System: Identification

- The individual uses the biometric sensor, but does not provide additional information.
- The system compares the presented template with the set of stored templates.



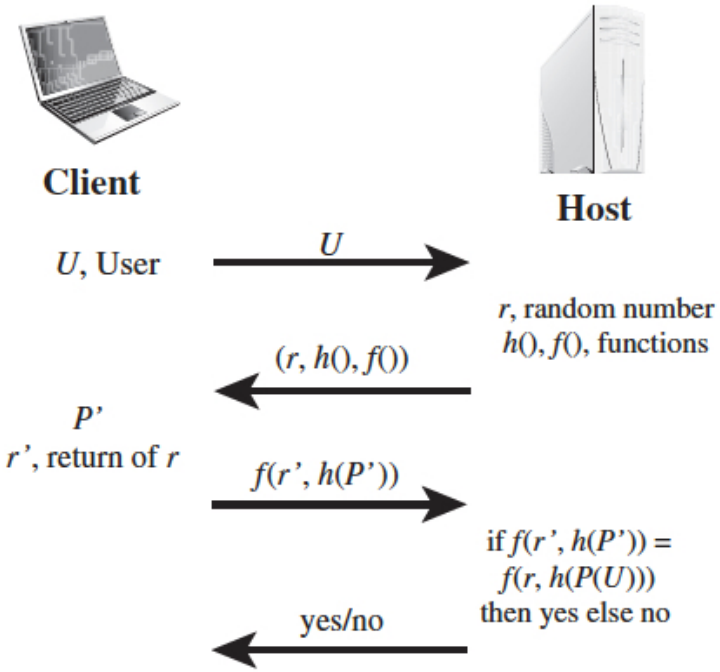
Picture from [SB15]

Remote User Authentication

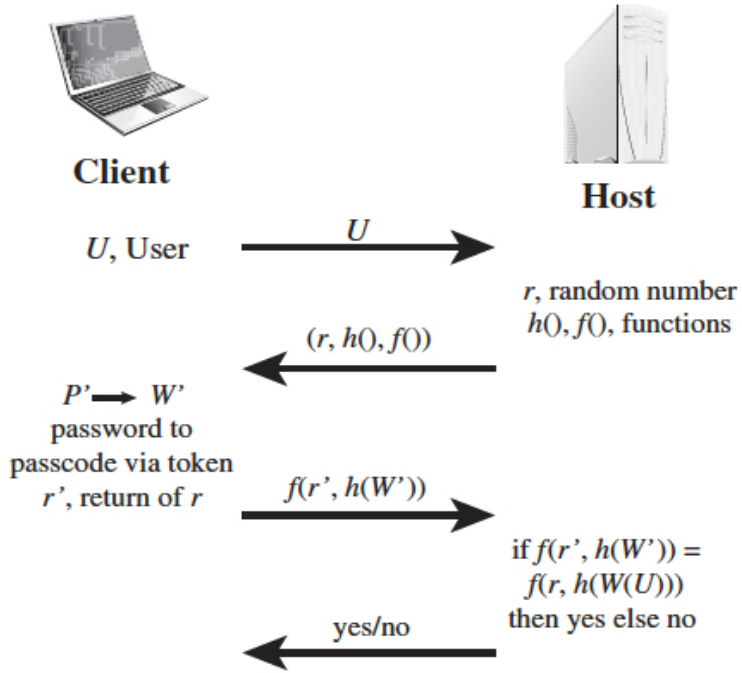
Remote User Authentication

- Remote user authentication: authentication that takes place over the Internet, a network, or a communications link, without local presence
- Additional security threats are raised, e.g., eavesdropping and capturing a password, replaying an authentication sequence that has been observed
- Apply some form of challenge-response protocol

Password Protocol/Token Protocol



(a) Protocol for a password

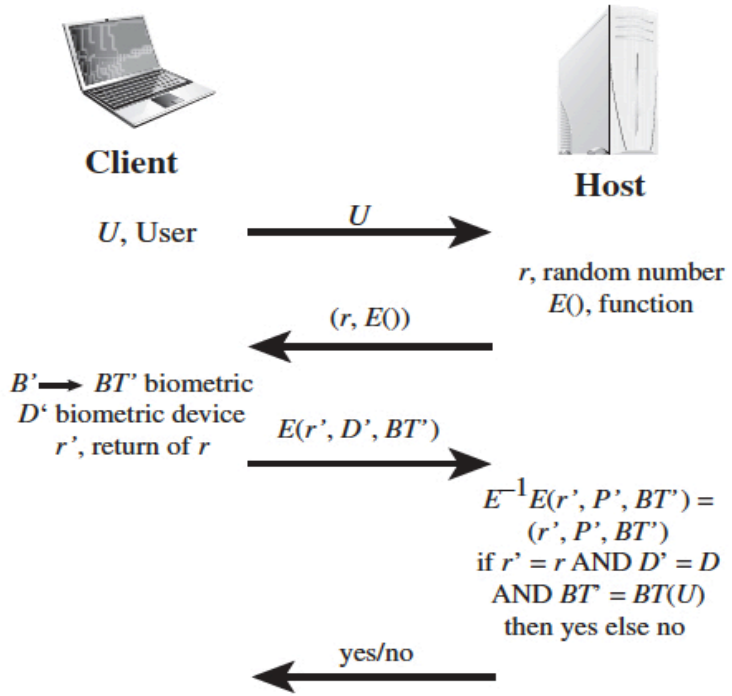


(b) Protocol for a token

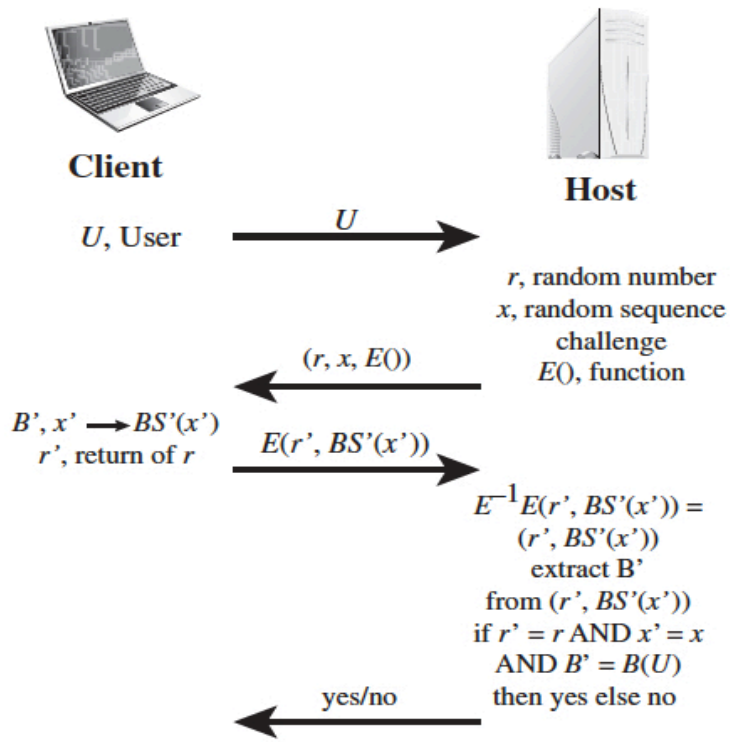
- Defend against
  - intruder attack
  - electronic monitoring attack
  - replay attack, in which the adversary captures the user's transmission and attempts to log on to a system by retransmitting the user's message

Picture from [SB15]

Static/Dynamic Biometric Protocol



(c) Protocol for static biometric



(d) Protocol for dynamic biometric

- Exploit an encryption scheme
- A protocol for dynamic biometric additionally uses a random sequence x to generate a biometric signal

Picture from [SB15]

References

SB15 W. Stallings and L. Brown, Computer Security: Principles and Practice, 3rd edition, Pearson Prentice Hall, 2015