NETWORK AND SYSTEM SECURITY (CORE ELECTIVE - 5) CS424

Unit 3 : SYSTEM SECURITY

Syllabus

- **INTRODUCTION** **(04 Hours)**

  Introduction to Network and System Security, Security Attacks, Security Requirements, Confidentiality, Integrity, and Availability, Security Mechanisms, NIST Security Standards, Assets and Threat Models.

- **REVIEW OF CRYPTOGRAPHIC TOOLS** **(04 Hours)**

  Number Theory, Prime Numbers, Modular Arithmetic, Confidentiality with Symmetric Encryption, Message Authentication and Hash Functions, Public-Key Encryption, Digital Signatures and Key Management, Random and Pseudorandom Numbers.

- **SYSTEM SECURITY** **(10 Hours)**

  User Authentication - Means of Authentication, Password-Based Authentication, Token-Based Authentication, Biometric Authentication, Remote User Authentication, Access Control-Access Control Principles, Subjects, Objects, and Access Rights, Discretionary Access Control, Example: UNIX File Access Control, Role-Based Access Control, Database Security-The Need for Database Security, Database Access Control, Inference, Statistical Databases, Database Encryption, Cloud Security, Malicious Software, Intruders, Denial of Service and Distributed Denial of Service attacks, Intrusion Detection and Prevention.

Syllabus...

- **SOFTWARE SECURITY AND TRUSTED SYSTEMS** **(12 Hours)**

  Buffer Overflow-Stack Overflows, Defending Against Buffer Overflows, Other Forms of Overflow Attacks, Software Security-Software Security Issues, Handling Program Input, Writing Safe Program Code, Interacting with the Operating System and Other Programs, Handling Program Output, Operating System Security-System Security Planning, Operating Systems Hardening, Application Security, Security Maintenance, Linux/Unix Security, Windows Security, Virtualization Security, Trusted Computing and Multilevel Security-The Bell-LaPadula Model for Computer Security, Other Formal Models for Computer Security,

  The Concept of Trusted Systems, Application of Multilevel Security, Trusted Computing and the Trusted Platform Module, Common Criteria for Information Technology Security Evaluation, Assurance and Evaluation.

Syllabus...

- **NETWORK SECURITY** **(10 Hours)**

  Internet Security Protocols and Standards-Secure E-mail and S/MIME, Pretty Good Privacy (PGP), Domain Keys Identified Mail, Secure Sockets Layer (SSL) and Transport Layer Security (TLS), HTTPS, IPv4 and IPv6 Security, IPSec Protocol, Internet Authentication Applications-Kerberos, X.509, Public-Key Infrastructure, Federated Identity Management, Wireless Network Security-Wireless Security Overview, IEEE 802.11 Wireless LAN Overview, IEEE 802.11i Wireless LAN Security, Network Management Security-SNMP Protocol.

- **ADVANCED TOPICS** **(02 Hours)**

(Total Contact Time = 42 Hours)

Unit 3 Overview

- **User Authentication**
- Access Control
- Database and Cloud security
- Malicious Software
- Denial of Service Attack
- Intrusion Detection
- Intrusion Prevention

User Authentication

Learning objectives

- Discuss the four general means of authenticating a user's identity
- Explain the mechanism by which hashed passwords used for user authentication
- Understand the use of the Bloom filters in password management
- Present an overview of token-based user authentication
- Discuss the issues involved and the approaches for remote user authentication

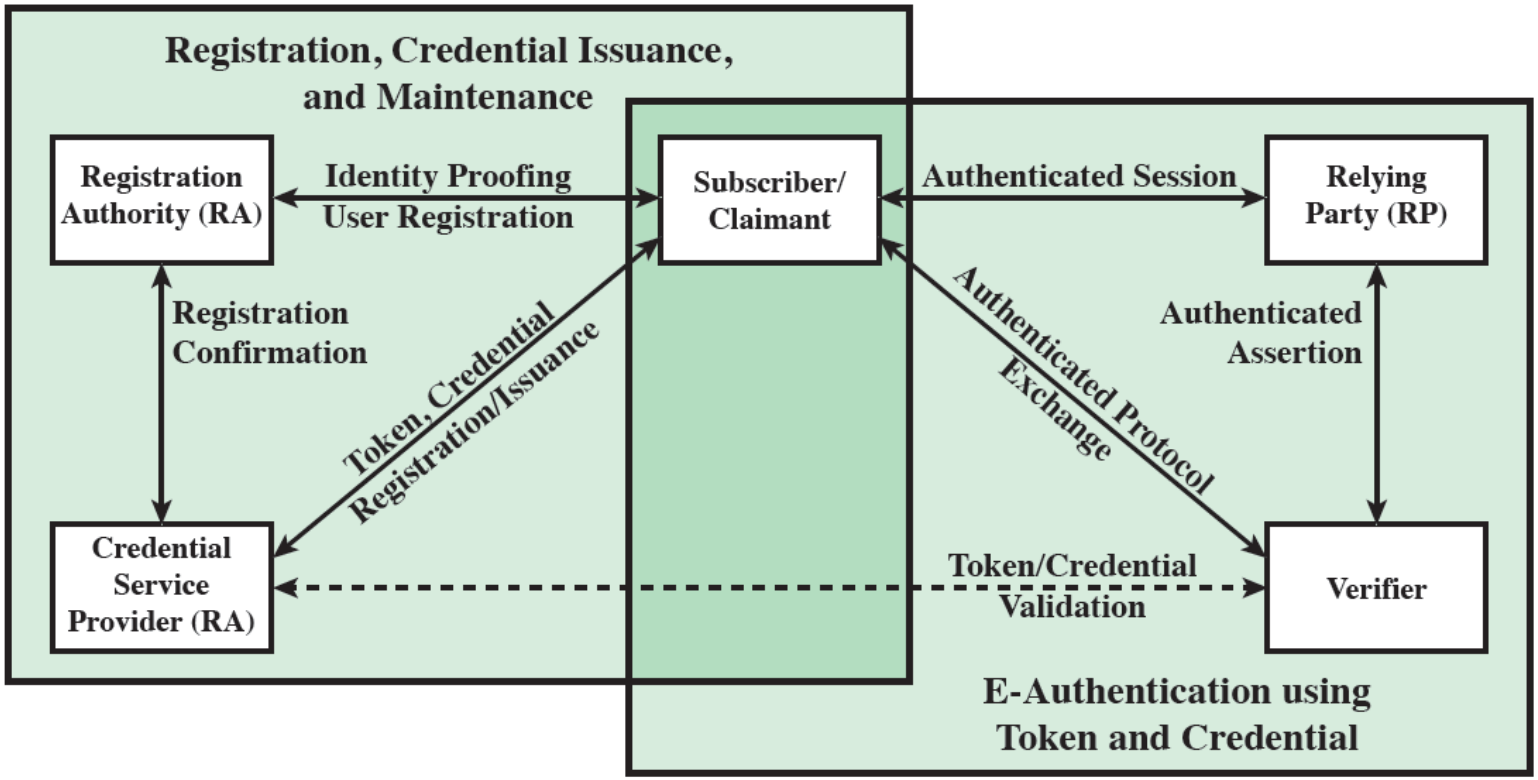
User Authentication

- Fundamental security building block
  - basis of access control & user accountability
- The process of verifying an identity claimed by or for a system entity
- Two steps:
  - identification: specify identifier
  - verification: bind entity (person) and identifier
- Distinct from message authentication (when communicating parties are concerned with the integrity of the exchanges messages)

A model for electronic user authentication

- NIST SP 800-63-2 defines EUA as: the process of establishing confidence in user identity that are electronically presented
- The NIST SP 800-63-2 model
  - User applies to **registration authority (RA)** and becomes a **subscriber** of a **credential service provider (CSP)**
  - RA is a trusted entity
  - The CSP exchanges with the subscriber
  - The credential (a data structure) binds an identity to a token possessed by the subscriber
  - **Claimant**: the party to be authenticated
  - **Verifier**: the party verifying
  - The verifier passes an assertion about the subscriber to the **relaying party (RP)**

A model for electronic user authentication



Means of user authentication

- Four means of authenticating user's identity
- Based one something the individual
  - knows, e.g. password, PIN
  - possesses, e.g. key, token, smartcard
  - is (static biometrics), e.g. fingerprint, retina
  - does (dynamic biometrics), e.g. voice, sign
- Can use alone or combined
- All can provide user authentication
- All have issues

Risk assessment for user authentication

- Assurance level: the degree of certainty that a user has presented a credential that refers to his/her identity
  - Level 1: little confidence (an online forum)
  - Level 2: some confidence (professional organizations)
  - Level 3: High confidence (patent office applicants)
  - Level 4: Very high confidence (employees accessing restricted/sensitive services)(law enforcement officials For example, a law enforcement official accesses a law enforcement database containing criminal records. Unauthorized access could raise privacy issues and/or compromise investigations.)
- Potential impact: low, moderate, high

Risk assessment for user authentication

	Assurance Level Impact Profiles			
Potential Impact Categories for Authentication Errors	1	2	3	4
Inconvenience, distress, or damage to standing or reputation	Low	Mod	Mod	High
Financial loss or organization liability	Low	Mod	Mod	High
Harm to organization programs or interests	None	Low	Mod	High
Unauthorized release of sensitive information	None	Low	Mod	High
Personal safety	None	None	Low	Mod/ High
Civil or criminal violations	None	Low	Mod	High

Password authentication

- Widely used user authentication method
  - user provides name/login and password
  - system compares password with that saved for specified login
- Authenticates ID of user logging and
  - that the user is authorized to access system
  - determines the user's privileges
  - is used in discretionary access control

Password vulnerabilities

- offline dictionary attack – password file, hashes of password
- specific account attack – Guesses of password, no more than 5 attempts
- popular password attack – popular pw, easily remembered pw
- password guessing against single user - previous knowledge about the user, length, special char
- workstation hijacking – inactivity of workstation, ID can be used
- exploiting user mistakes – pw on paper, sharing with colleagues, social engineering
- exploiting multiple password use – similar pw
- electronic monitoring - eavesdropping

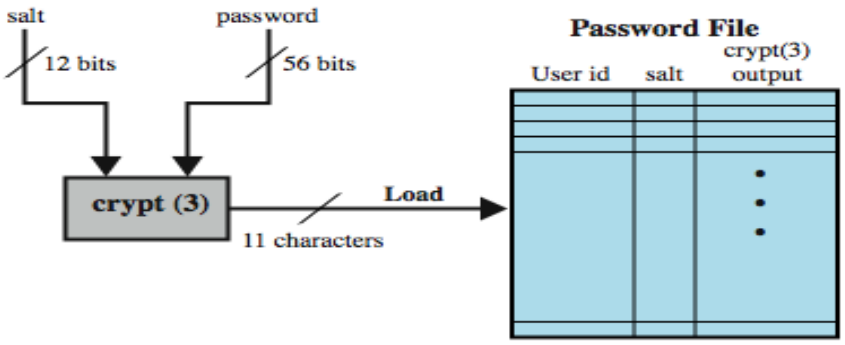
Countermeasures for password vulnerability

- stop unauthorized access to password file
- intrusion detection measures
- account lockout mechanisms
- policies against using common passwords but rather hard to guess passwords
- training & enforcement of policies
- automatic workstation logout
- encrypted network links

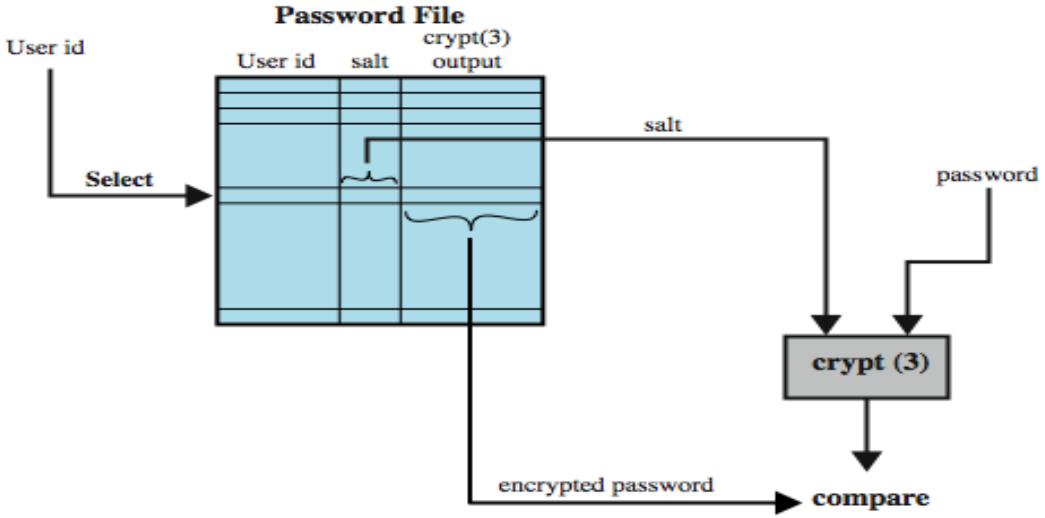
Countermeasures for password vulnerability

- It is worthwhile to study/research password and password vulnerabilities
  - Most common
  - Still the most efficient

Use of hashed passwords



(a) Loading a new password



(b) Verifying a password

Why a salt value?

- Prevents duplicate passwords from being visible in the password file
- Increases the difficulty of offline dictionary attacks - For a salt of length b bits, the number of possible passwords is increased by a factor of 2^b, increasing the difficulty of guessing a password in a dictionary attack
- Nearly impossible to tell if a person used the same password on multiple systems

UNIX Implementation

- Original scheme
  - 8 character password form 56-bit key
  - 12-bit salt used to modify DES encryption into a one-way hash function
  - The o/p of the algorithm then serves as input for a second encryption.
  - This process is repeated for a total of 25 encryptions.
  - The resulting 64-bit output translated to 11 character sequence
- Now regarded as woefully insecure
  - e.g. supercomputer, 50 million tests, 80 min
- Sometimes still used for compatibility

Improved implementations

- Have other, stronger, hash/salt variants
- 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 Cracking

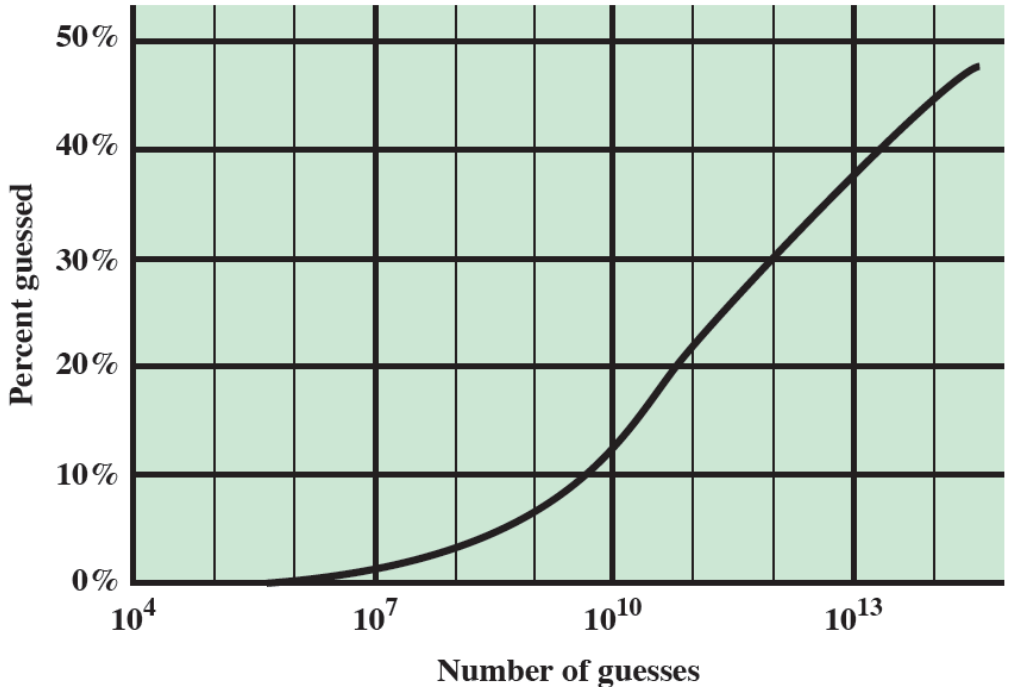
- Dictionary attacks
  - try each word then obvious variants in large dictionary against hash in password file
- **Rainbow** table attacks
  - a large dict of possible passwords
  - for each password:
    - precompute tables of hash values for all salts
    - a mammoth table of hash values: e.g. 1.4GB table cracks 99.9% of alphanumeric Windows passwords in 13.8 secs
  - not feasible if larger salt values used

Password choices/concerns

- users may pick short passwords
  - e.g. 3% were 3 chars or less, easily guessed
  - system can reject choices that are too short
- users may pick guessable passwords
  - so crackers use lists of likely passwords
  - e.g. one study of 14000 encrypted passwords guessed nearly 1/4 of them
  - would take about 1 hour on fastest systems to compute all variants
- The first big breakthrough came in late 2009, when an SQL injection attack against online games service RockYou.com exposed 32 million plaintext passwords used by its members to log in to their accounts

Another case study

- An analysis of passwords used by 25,000 students
- Over 10% recovered after 10^10 guesses



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 (for su only)
- Still have vulnerabilities
  - exploit O/S bug
  - accident with permissions making it readable
  - users with same password on other systems
  - access from unprotected backup media
  - sniff passwords in unprotected network traffic

Using Better Passwords

- Clearly have problems with passwords
- Goal to eliminate guessable passwords
  - Still easy for user to remember
- Techniques
  - user education
  - computer-generated passwords
  - reactive password checking (periodic checking) - strategy is one in which the system periodically runs its own password cracker to find guessable passwords.
  - proactive password checking (at the time of selection) a user is allowed to select his or her own password.

Sample Password

- However, do not pick a well-known phrase like “An apple a day keeps the doctor away” (Aaadktda).
- pick something like “My dog’s first name is Rex” (MdfniR)
- “My sister Peg is 24 years old” (MsPi24yo).

Proactive Password Checking

- Rule enforcement plus user advice, e.g.
  - 8+ chars, upper/lower/numeric/punctuation
  - may not suffice
- Password cracker
  - list of bad passwords
  - time and space issues
- Markov Model
  - generates guessable passwords
  - hence reject any password it might generate
- Bloom Filter
  - use to build table based on dictionary using hashes
  - check desired password against this table

Bloom Filter

- A Bloom filter of order *k* consists of a set of *k* independent hash functions
- *H*1(*x*), *H*2(*x*),c, *Hk*(*x*), where each function maps a password into a hash value in the range 0 to *N* – 1. That is,
- *Hi*(X*j*) = *y*, 1 <=*i*<= *k*; 1 <= *j* <= *D*; 0 <= *y* <= *N* - 1

Where X_j = *j*th word in password dictionary,

D = number of words in password dictionary

The following procedure is then applied to the dictionary:

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. Thus, if *Hi* (X*j*) = 67 for some (*i*, *j*), then the sixty-seventh bit of the hash table is set to 1; if the bit already has the value 1, it remains at 1.

Bloom Filter…

- When a new password is presented to the checker, its *k* hash values are calculated.
- If all the corresponding bits of the hash table are equal to 1, then the password is rejected.
- All passwords in the dictionary will be rejected. But there will also be some “false positives” (that is, passwords that are not in the dictionary but that produce a match in the hash table).
- Suppose that the passwords *undertaker* and *hulkhogan* are in the dictionary, but *xG%#jj98* is not. Further suppose that
- *H*1(undertaker) = 25 *H*1 (hulkhogan) = 83 *H*1 (xG%#jj98) = 665
- *H*2(undertaker) = 998 (hulkhogan) = 665 *H*2 (xG%#jj98) = 998
- If the password xG%#jj98 is presented to the system, it will be rejected even though it is not in the dictionary.

Token-based authentication

- Object user possesses to authenticate, e.g.
  - memory card (magnetic stripe)
  - smartcard

Types of Cards used as Tokens

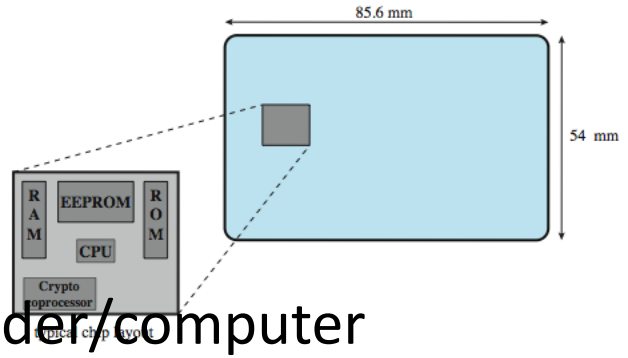
Card Type	Defining Feature	Example
Embossed	Raised characters only, on front	Old credit card
Magnetic stripe	Magnetic bar on back, characters on front	Bank card
Memory	Electronic memory inside	Prepaid phone card
Smart Contact Contactless	Electronic memory and processor inside Electrical contacts exposed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

Memory Card

- store but do not process data
- magnetic stripe card, e.g. bank card
- electronic memory card
- used alone for physical access (e.g., hotel rooms)
- some with password/PIN (e.g., ATMs)
- Drawbacks of memory cards include:
  - need special reader
  - loss of token issues
  - user dissatisfaction (OK for ATM, not OK for computer access)

Smartcard

- credit-card like
- has own processor, memory, I/O ports
  - ROM, EEPROM, RAM memory
- executes protocol to authenticate with reader/computer
  - static: similar to memory cards
  - dynamic: passwords created every minute; entered manually by user or electronically
  - challenge-response: computer creates a random number; smart card provides its hash (similar to PK)
- also have USB dongles



Smartcard

- **Physical characteristics:** Smart tokens include an embedded microprocessor. A smart token that looks like a bank card is called a smart card. Other smart tokens can look like calculators, keys, or other small portable objects.
- **User interface:** Manual interfaces include a keypad and display for human/token interaction.
- **Electronic interface:** A smart card or other token requires an electronic interface to communicate with a compatible reader/writer.

1. **Contact:** A contact smart card must be inserted into a smart card reader
2. **Contactless:** A contactless card requires only close proximity to a reader. Both the reader and the card have an antenna, and the two communicate using radio frequencies.

Smartcard…

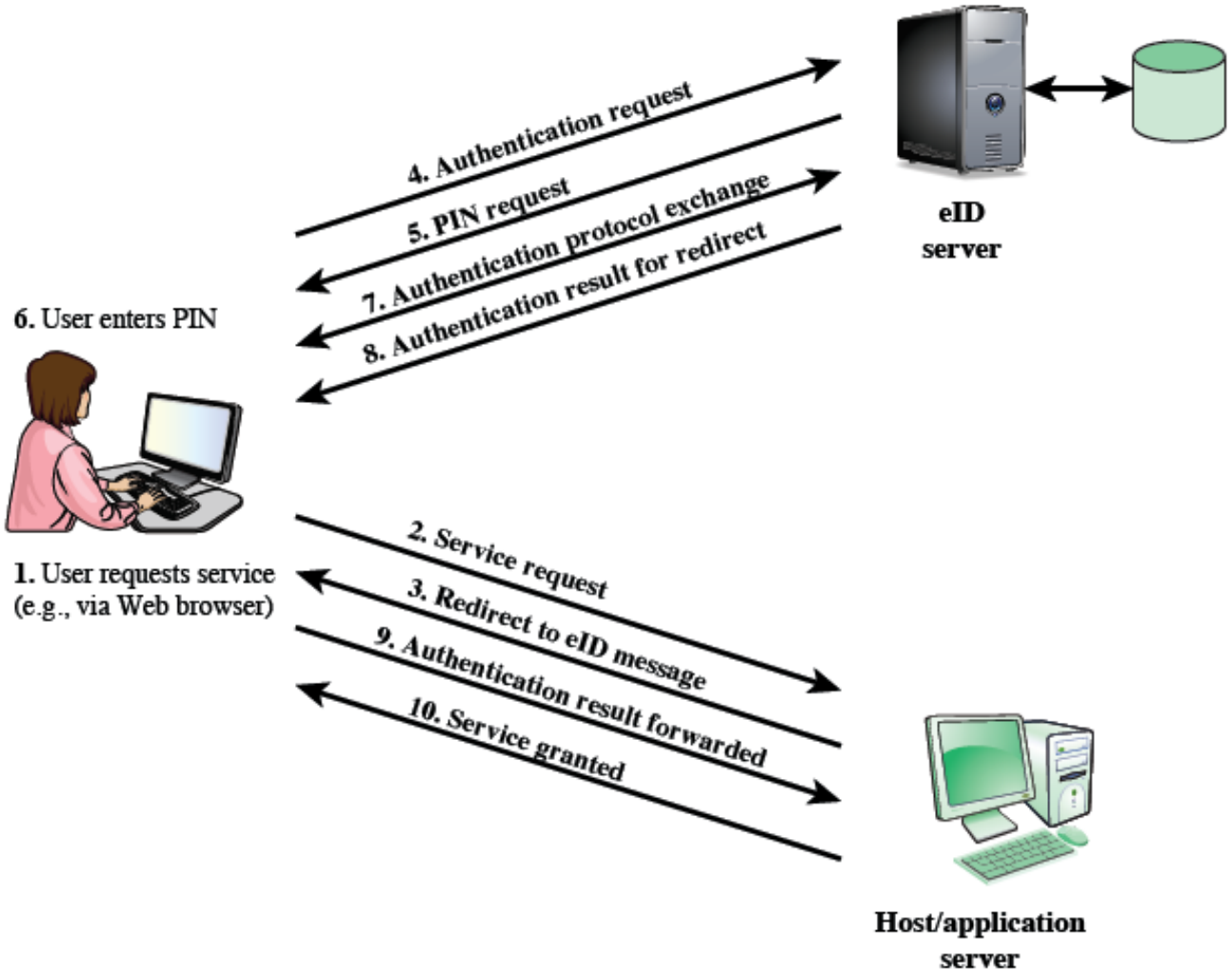
- A typical smart card includes three types of memory.

1. Read-only memory (ROM) stores data that does not change during the card's life, such as the card number and the cardholder's name.
2. Electrically erasable programmable ROM (EEPROM) holds application data and programs, such as the protocols that the card can execute. It also holds data that may vary with time. For example, in a telephone card, the EEPROM holds the talk time remaining.
3. Random access memory (RAM) holds temporary data generated when applications are executed.

Electronic identify cards

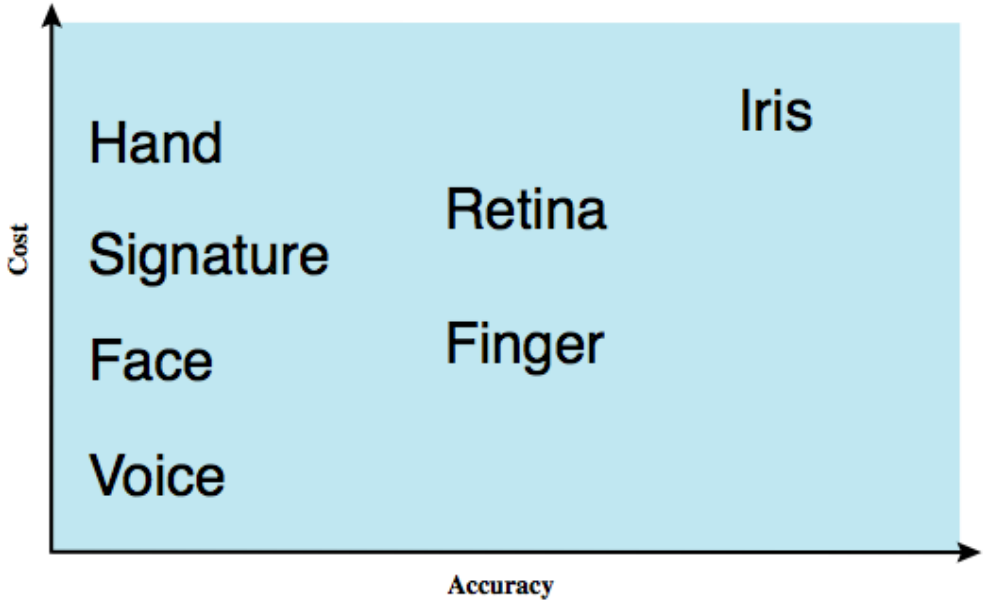
- An important application of smart cards
- A national e-identity (eID)
- Serves the same purpose as other national ID cards (e.g., a driver's licence)
  - Can provide stronger proof of identity
  - A German card
    - Personal data, Document number, Card access number (six digit random number), Machine readable zone (MRZ): the password
    - Uses: ePass (government use), eID (general use), eSign (can have private key and certificate)

User authentication with eID



Biometric authentication

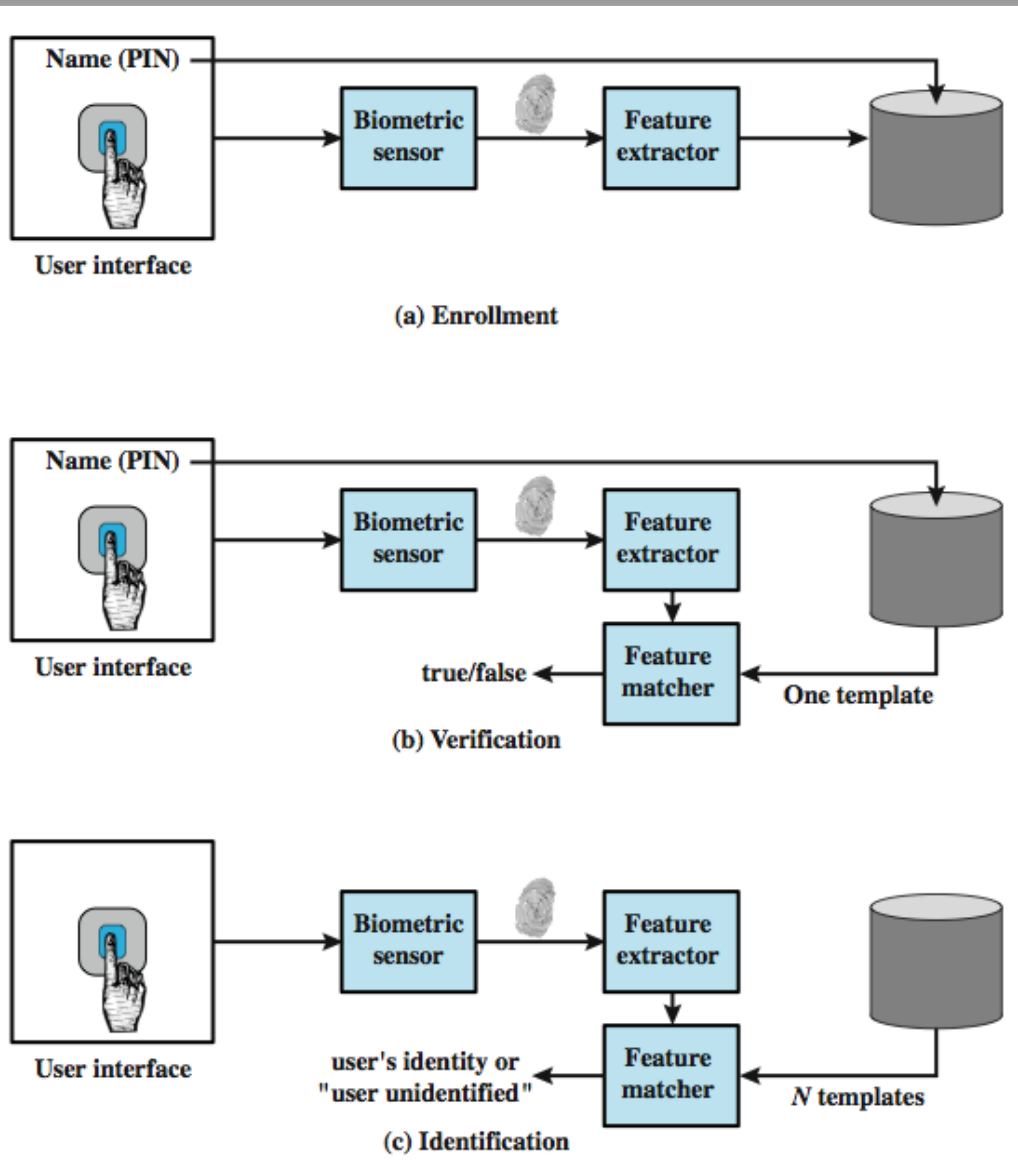
- Authenticate user based on one of their physical characteristics:
  - facial
  - fingerprint
  - hand geometry
  - retina pattern
  - iris
  - signature
  - voice



Operation of a biometric system

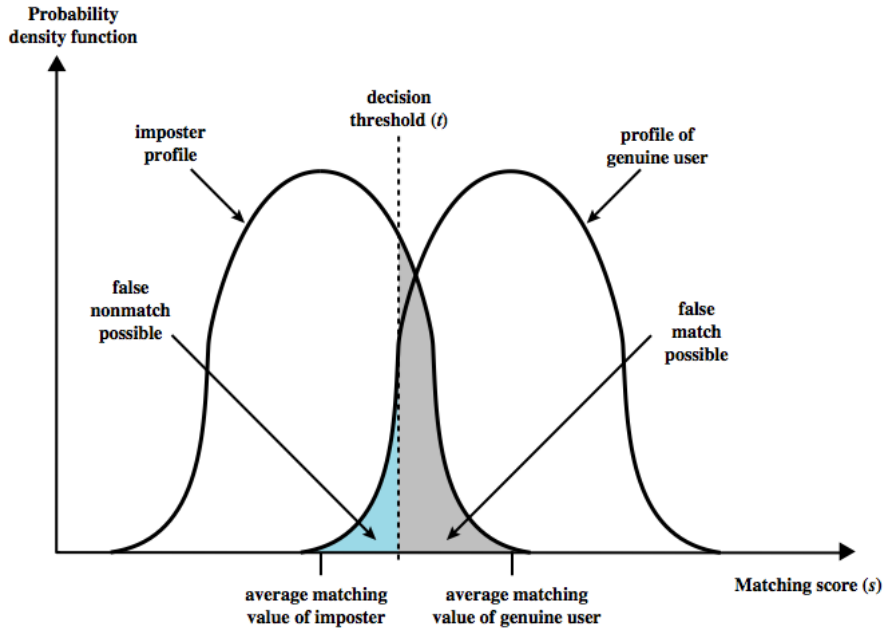
Verification *is analogous to user login via a smart card and a PIN*

Identification *is biometric info but no IDs; system compares with stored templates*



Biometric Accuracy

- The system generates a matching score (a number) that quantifies similarity between the input and the stored template
- Concerns: sensor noise and detection inaccuracy
- Problems of false match/false non-match

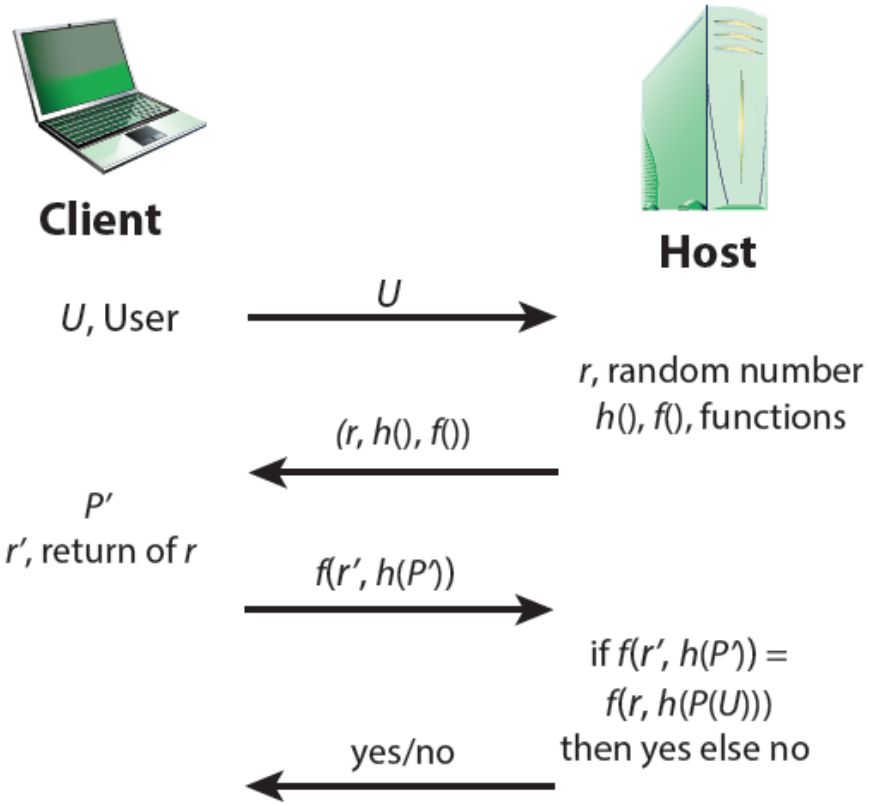


Remote User Authentication

- Authentication over network more complex
  - Problems of eavesdropping, replay
- Generally use challenge-response
  - user sends identity
  - host responds with random number *r*
  - user computes *f(r,h(P))* and sends back
  - host compares value from user with own computed value, if match user authenticated
- Protects against a number of attacks

Protocol for a password verification

- Similar approach for token and biometric verification



Thank You !!!