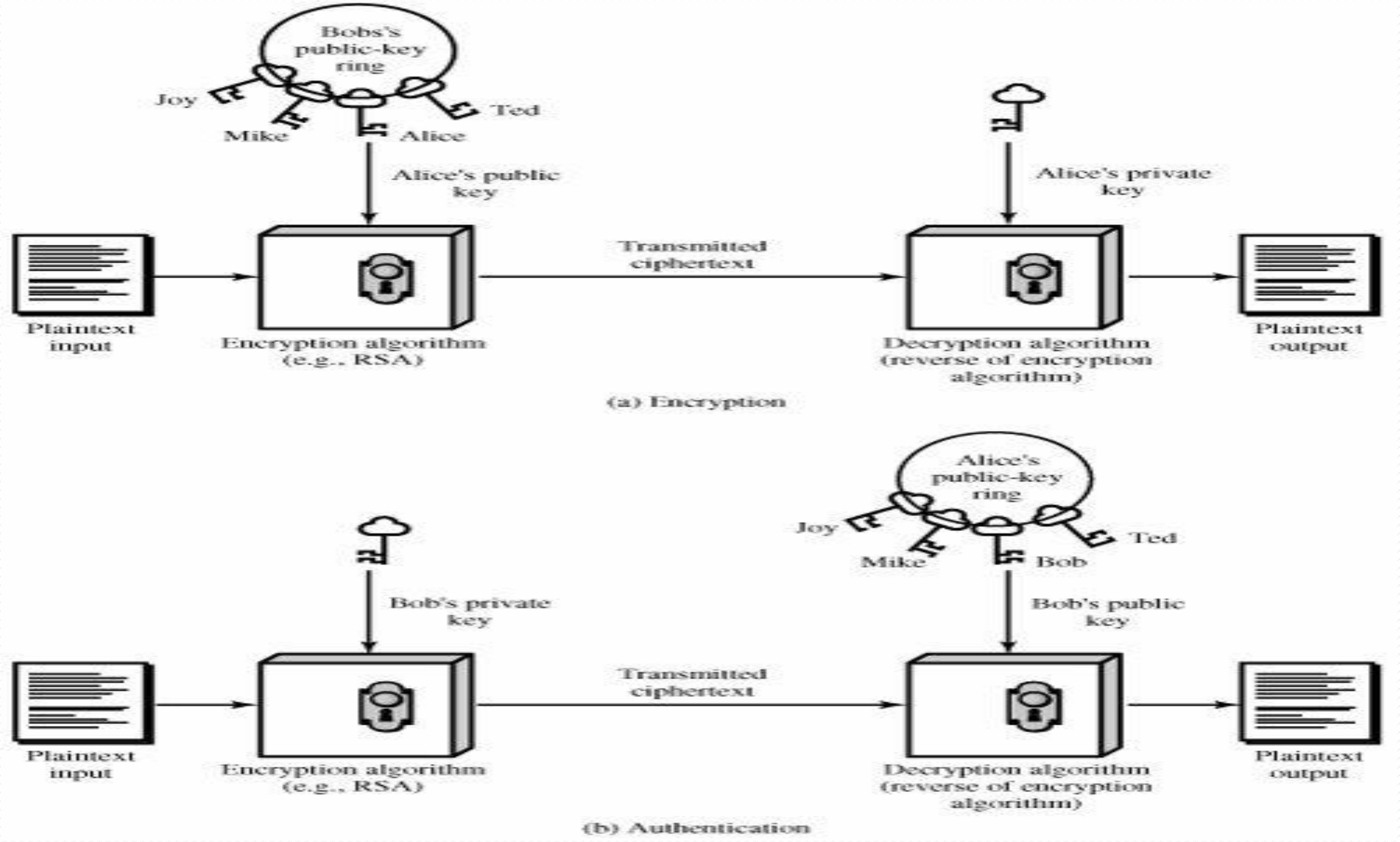
CS ??? Computer Security
User Authentication

Yasser F. O. Mohammad

REMINDER 1:Public Key Encryption



REMINDER 2: RSA Algorithms

Key Generation

Select p, q	p and q both prime, $p \neq q$
Calculate $n = p \times q$	
Calculate $\phi(n) = (p-1)(q-1)$	
Select integer e	$\gcd(\phi(n), e) = 1;\ 1 < e < \phi(n)$
Calculate d	$d \equiv e^{-1} \pmod{\phi(n)}$
Public key	$PU = \{e, n\}$
Private key	$PR = \{d, n\}$

Encryption

Plaintext:	$M < n$
Ciphertext:	$C = M^e \bmod n$

Decryption

Ciphertext:	C
Plaintext:	$M = C^d \bmod n$

REMINDER 3: Diffie-Hellman

- The point is that users A and B will be able to calculate the secret key using only:
  1. His private key
  2. Other's public key

- Eve needs to do a discrete logarithm because she does not have any of the private keys.

Global Public Elements

q	prime number
α	$\alpha < q$ *and* α a primitive root of q

User A Key Generation

Select private X_A	$X_A < q$
Calculate public Y_A	$Y_A = \alpha^{X_A} \bmod q$

User B Key Generation

Select private X_B	$X_B < q$
Calculate public Y_B	$Y_B = \alpha^{X_B} \bmod q$

Calculation of Secret Key by User A

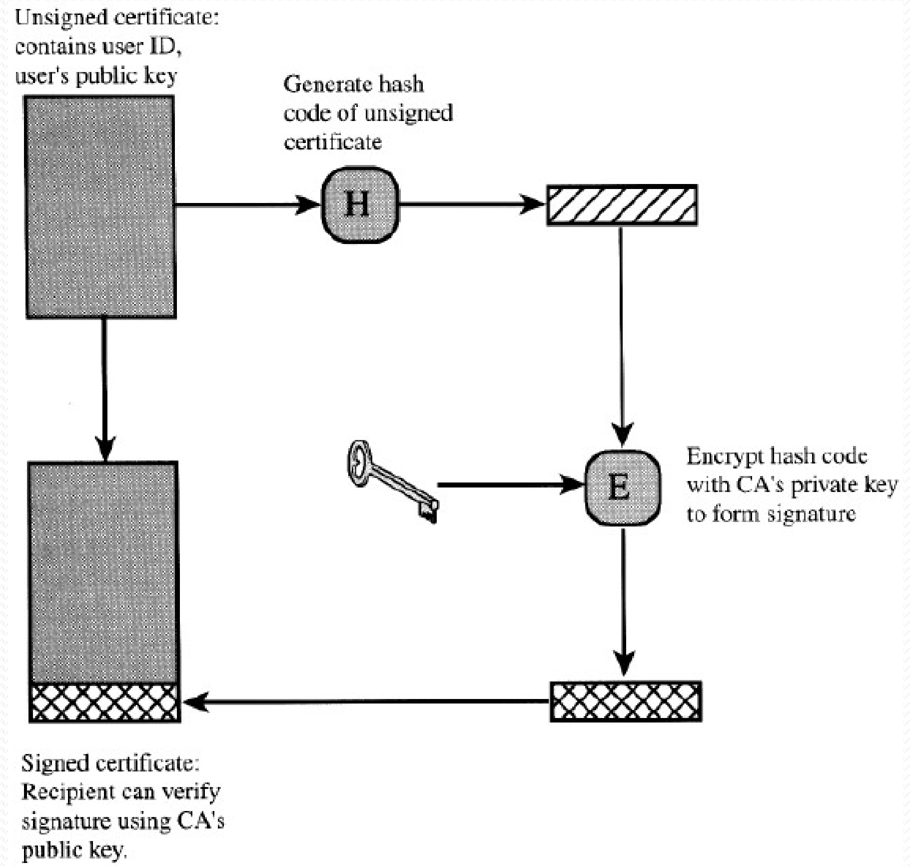
$K = (Y_B)^{X_A} \bmod q$

Calculation of Secret Key by User B

$K = (Y_A)^{X_B} \bmod q$

REMINDER 4: Distribution of Public Keys

- Public Key Certificates
- CA=Certification Authority
- CA's sign public keys of users with its private key
- X.509 standard
- Used in SSL, Secure Electronic Transaction (SET), S/MIME



Authentication

- Message Authentication
  - Who generated this message?
- User Authentication
  - Who am I dealing with?

User Authentication

- Basis of most other security services
  - Access Control
  - User Accountability
  - etc
- Verifying the identity claimed by some entity

- Two steps:
  - Identification: presenting credentials
  - Verification: binding entity to ID

How to authentication a user?

- Something you know
  - Passwords, passphrases
- Something you have
  - Smart cards
- Something you are (static biometrics)
  - Fingerprint
  - Retina recognition
  - etc
- Something you do (dynamic biometrics)
  - Signature
  - Voice pattern
  - etc

DISCUSSION POINT

- What are the problems of each of these methods:
  - Something you know
  - Something you have
  - Something you are
  - Something you do

Password based authentication

- Simplest Approach
  - The system challenges the user
    - S→U: C
  - User presents a function of the password and challenge information
    - U→S: F(P,C)
  - The system processes the reply to confirm the identity of the user (ID)

- The ID can then be used for other security purposes

Password Vulnerabilities

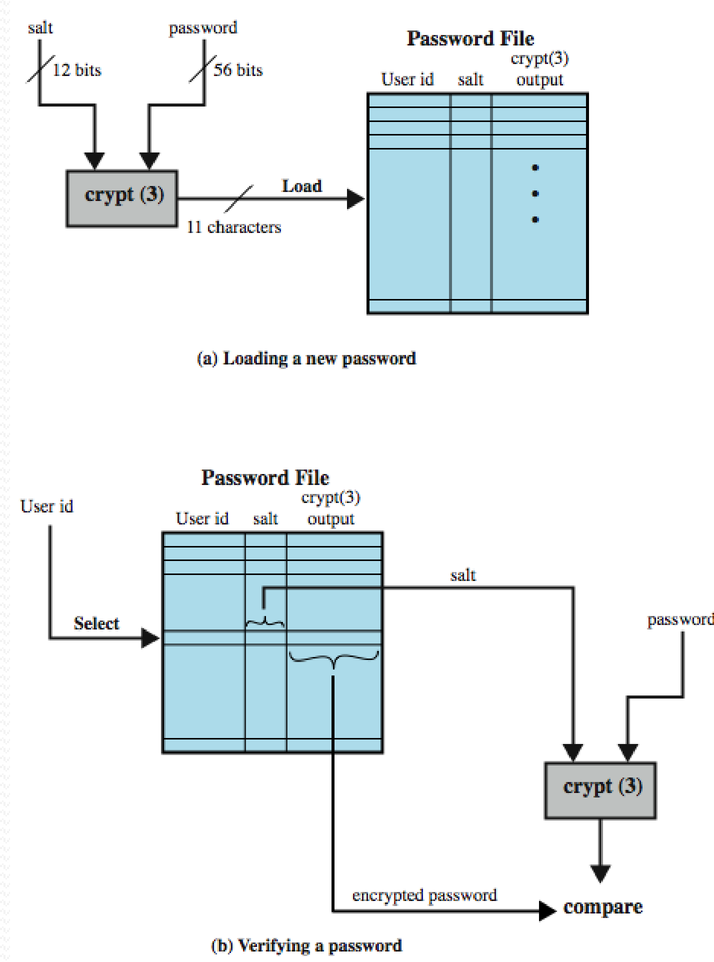
- Offline dictionary attack
  - Keep the password file secure
- Specific account attack
  - Limit the number of failed attempts
  - Intrusion detection
- Popular password attack
  - Do not use popular passwords
  - Account lockout
- Password guessing against single user
  - Training not to use your name as your password!!!

Password Vulnerabilities 2

- Workstation hijacking
  - Do not leave your session
  - Frequent checking
- Exploiting user mistakes
  - Do not write passwords , do not do mistakes!!
- Exploiting multiple password use
  - Use a different password for every occasion
- Electronic monitoring
  - Do not transfer passwords

How not to store the password?

- Uses of salt:
  - Prevents duplicate password discovery
    - In the same pass file
    - In different machines
  - Increases difficulty of offline attacks
- The hashing MUST BE SLOWWWWWWW!!



(a) Loading a new password

(b) Verifying a password

UNIX scheme

- Original scheme
  - 8 character password → 56-bit key
  - 12-bit salt used to modify DES encryption into a one-way hash function
  - Zero repeatedly encrypted 25 times
  - Output translated to 11 character sequence
- Now regarded as insecure
  - e.g. supercomputer, 50 million tests, 80 min
  - $10,000 can do the same with a uniprocessor system in few months
- *sometimes still used for compatibility*

Newer Implementations

- 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 hash algorithm called Bcrypt
  - uses 128-bit salt to create 192-bit hash value

Cracking Passwords

- Dictionary attacks
  - Try each word then obvious variants in large dictionary against hash in password file
- Rainbow table attacks
  - Precompute tables of hash values for all salts
  - e.g. 1.4GB table cracks 99.9% of alphanumeric Windows passwords in 13.8 secs
  - Not feasible if larger salt values used

Problems with password choice

- Short passwords
  - 6% of users use less than 4 chars passwords if allowed
- Guessable passwords
  - 24.2% of passwords used are easily guessable

How to protect password files?

- Use a separate shadow file
- Deny access except for privileged users

- FOR CRACKERS: How to get the pass word file??
  - Exploit O/S bug
  - Accident with permissions making it readable
  - Users with same password on other systems
  - Unprotected backup media
  - Unprotected network traffic

How to complicate passwords?

- User education
  - Do not use your birthday as your password?
- Computer-generated passwords
  - Needs to be memorable
- Reactive password checking
  - Periodically try to crack yourself
- Proactive password checking
  - Check upon password registration

Proactive Password Checking

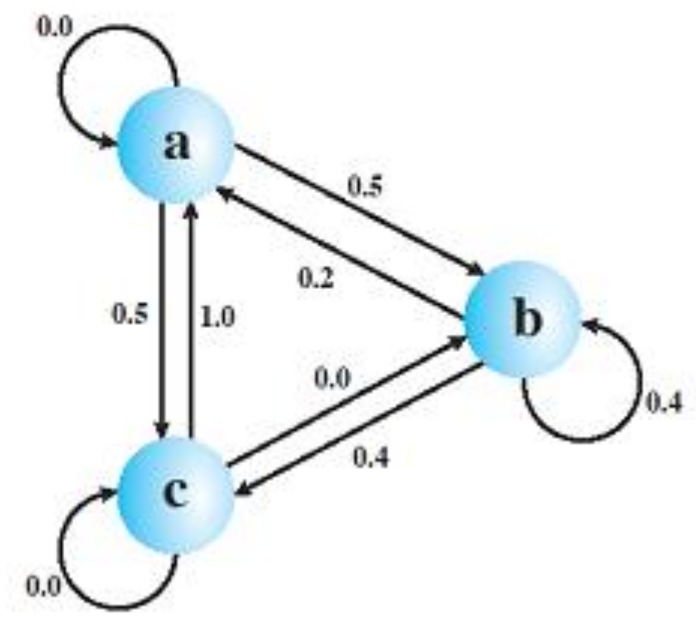
- Simple rules
  - 8+ characters
  - Upper, lower, numeric, punctuation marks
  - Change periodically

- Password Cracker
  - Needs a large dictionary (30MB at least!!!!)
  - Requires sometime to do the crack

  - In general EVE will have more time to crack the system

Proactive Password Checking 2

- Hidden Markov Models
  - Learn a HMM from a dictionary
  - Reject passwords with high probability of being generated from this dictionary
  - Usually uses bigrams as basic units and trigrams to find frequencies



$M = (3, \{a, b, c\}, T, 1)$ where

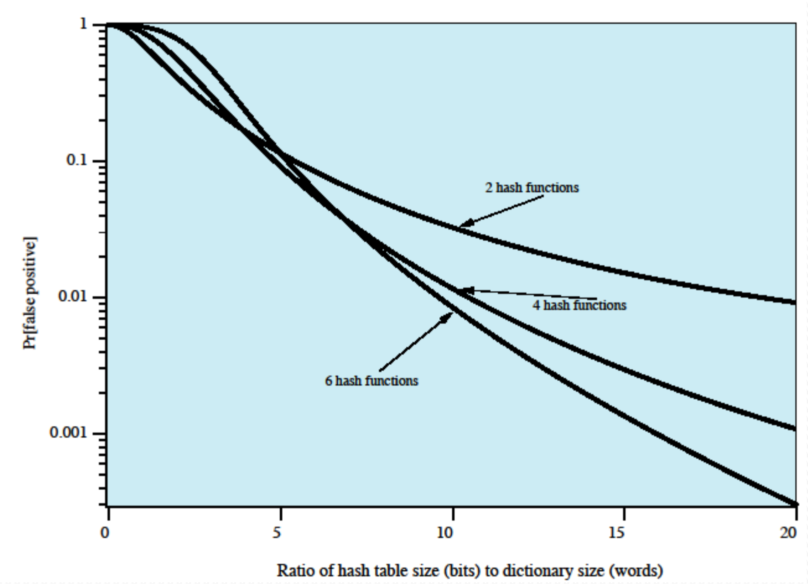
$$T = \begin{bmatrix} 0.0 & 0.5 & 0.5 \\ 0.2 & 0.4 & 0.4 \\ 1.0 & 0.0 & 0.0 \end{bmatrix}$$

e.g., string probably from this language: abbcacaba

e.g., string probably not from this language: aaccebaaa

Proactive Password Checking 2

- Bloom Filter
  - Uses k independent hash functions H_i each gives a value from 0 to N-1
  - Initialization:
    - Calculate H_i for all words in the dictionary
    - Initialize HashTbl of size N to all zeros
    - $H_i(D_i)=j \rightarrow$ HashTbl[j]=1
  - Checking:
    - Calculate H_i for it
    - Reject it if all HashTbl[H_i(P)]==1
  - Has false positives
  - P(false positive)= $\left(1-e^{kD/N}\right)^k$



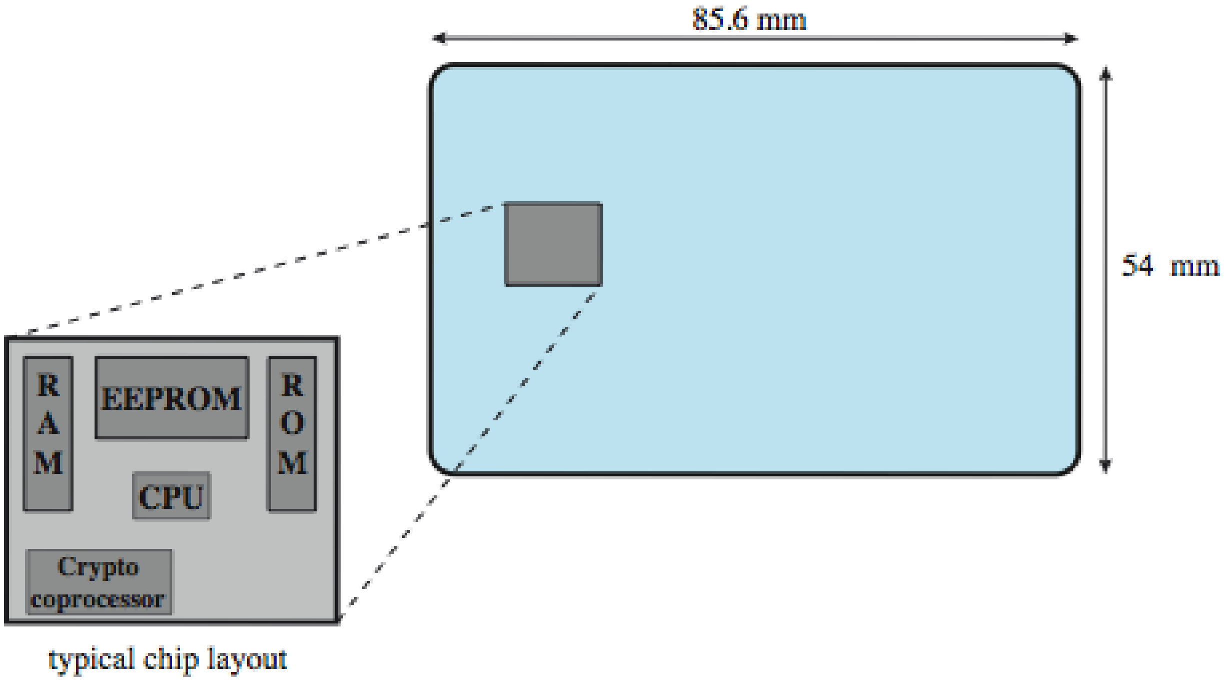
Token Based Authentication

- Problems:
  - Special reader
  - Loss
  - User dissatisfaction!!!

Types of cards usually used

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

Smart Cards



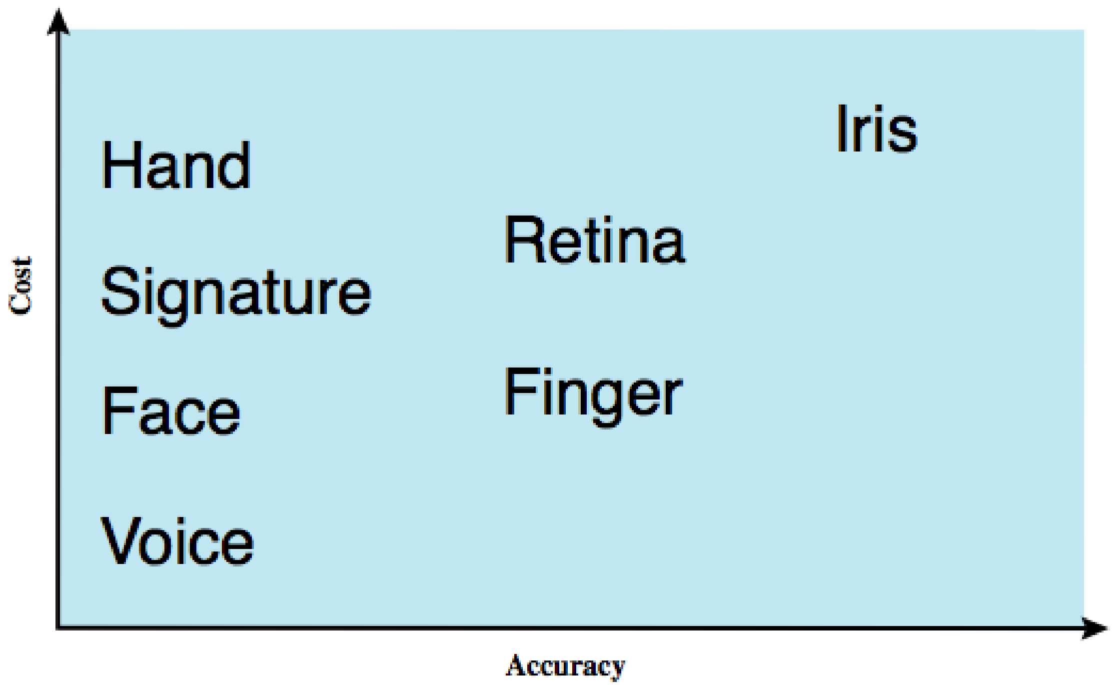
85.6 mm
54 mm
R
A
M
EEPROM
R
O
M
CPU
Crypto
coprocessor
typical chip layout

Authentication Protocols

- Static
  - Something stored in the token
- Dynamic Password Generator
  - Periodically generate passwords
  - Must be synchronized with the Computer
- Challenge Response
  - System→Token: Challenge
  - Token→System: Response

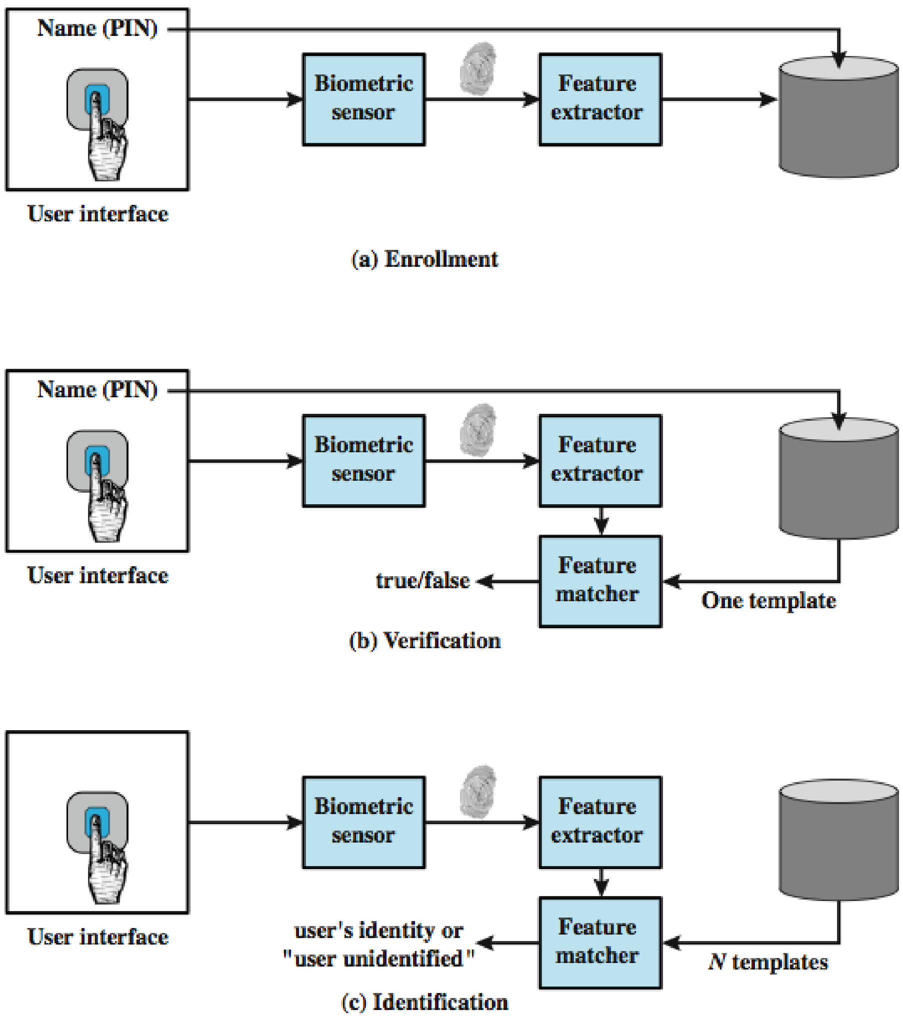
Biometric Authentication

- Both Static and Dynamic



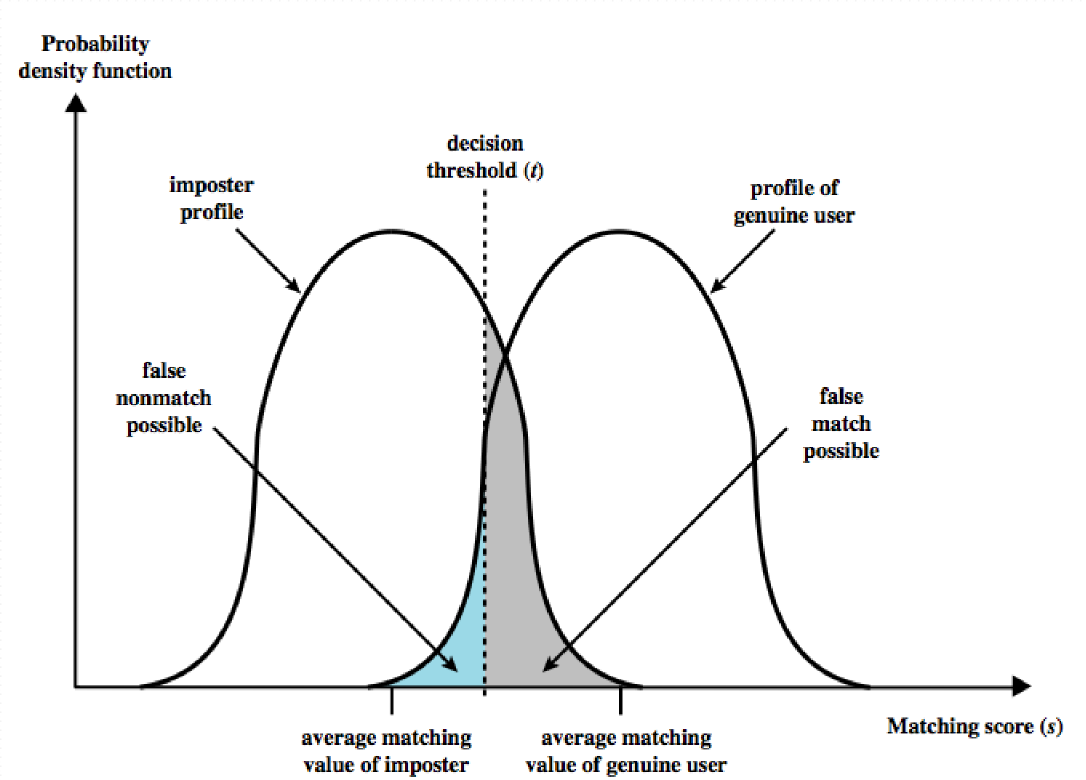
General Operation

- Enrollment
- Verification
- Identification

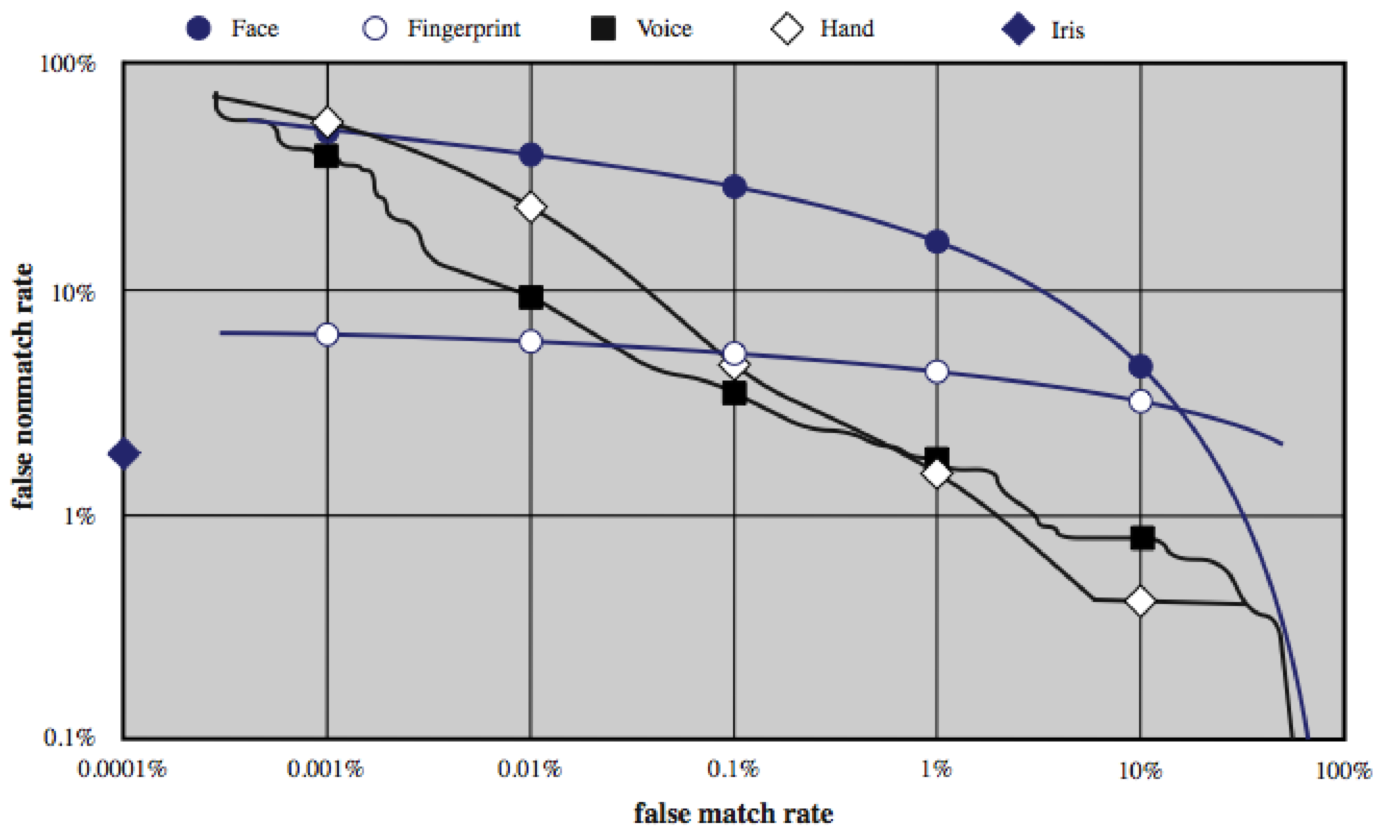


Life is not easy

- After some limit
  - To reduce false negatives you increase false positives



Characteristic Curve



What do you care about

- Finding terrorists in airports using vision
  - False negatives
  - A false positive just causes one extra check by the officer
  - A false negative may cause you hundreds of lives, an airplane (**and your job**)

- Access control for employees
  - False positives
  - A false negative just causes another retrial or officer attention
  - A false positive may cause you company secrets (**and your job**)

Remote User Authentication

- Passwords must never be transferred in clear

Client	Transmission	Host
U, user	$U \rightarrow$	
	$\leftarrow \{r, \mathrm{h}(), \mathrm{f}()\}$	random number h(), f(), functions
P' password r', return of r	$\mathrm{f}(r', \mathrm{h}(P')) \rightarrow$	
	$\leftarrow$ yes/no	if $\mathrm{f}(r', \mathrm{h}(P')) =$ $\mathrm{f}(r, \mathrm{h}(P(U)))$ then yes else no

(a) Protocol for a password

Client	Transmission	Host
U, user	$U \rightarrow$	
	$\leftarrow \{ r, \mathrm{h}(), \mathrm{f}()\}$	r, random number h(), f(), functions
$P' \rightarrow W'$ password to passcode via token r', return of r	$\mathrm{f}(r', \mathrm{h}(W')) \rightarrow$	
	$\leftarrow$ yes/no	if $\mathrm{f}(r', \mathrm{h}(W')) =$ $\mathrm{f}(r, \mathrm{h}(W(U)))$ then yes else no

(b) Protocol for a token

Client	Transmission	Host
U, user	$U \rightarrow$	
	$\leftarrow \{ r, \mathrm{E}()\}$	r, random number E(), function
$B' \rightarrow BT'$ biometric D' biometric device r', return of r	$\mathrm{E}(r', D', BT') \rightarrow$	$\mathrm{E}^{-1}\mathrm{E}(r', P', BT') =$ (r', P', BT')
	$\leftarrow$ yes/no	if $r' = r$ and $D' = D$ and $BT' = BT(U)$ then yes else no

(c) Protocol for static biometric

Client	Transmission	Host
U, user	$U \rightarrow$	
	$\leftarrow \{ r, x, \mathrm{E}()\}$	r, random number x, random sequence challenge E(), function
$B', x' \rightarrow BS'(x')$ r', return of r	$\mathrm{E}(r', BS'(x')) \rightarrow$	$\mathrm{E}^{-1}\mathrm{E}(r', BS'(x')) =$ $(r', BS'(x'))$ extract B' from $BS'(x')$
	$\leftarrow$ yes/no	if $r' = r$ and $x' = x$ and $B' = B(U)$ then yes else no

(d) Protocol for dynamic biometric

Security Issues

- client attacks
  - No access to server
- host attacks
  - Try to get to the DB
- Eavesdropping
  - Listen to transmissions
- Replay
  - Replay
- Trojan horse
  - Appear as a nice guy
- denial-of-service
- فيها لاخفيها

Attacks	Authenticators	Examples	Typical defenses
Client attack	Password	Guessing, exhaustive search	Large entropy; limited attempts
	Token	Exhaustive search	Large entropy; limited attempts, theft of object requires presence
	Biometric	False match	Large entropy; limited attempts
Host attack	Password	Plaintext theft, dictionary/exhaustive search	Hashing; large entropy; protection of password database
	Token	Passcode theft	Same as password; 1-time passcode
	Biometric	Template theft	Capture device authentication; challenge response
Eavesdropping, theft, and copying	Password	"Shoulder surfing"	User diligence to keep secret; administrator diligence to quickly revoke compromised passwords; multifactor authentication
	Token	Theft, counterfeiting hardware	Multifactor authentication; tamper resistant/evident token
	Biometric	Copying (spoofing) biometric	Copy detection at capture device and capture device authentication
Replay	Password	Replay stolen password response	Challenge-response protocol
	Token	Replay stolen passcode response	Challenge-response protocol; 1-time passcode
	Biometric	Replay stolen biometric template response	Copy detection at capture device and capture device authentication via challenge-response protocol
Trojan horse	Password, token, biometric	Installation of rogue client or capture device	Authentication of client or capture device within trusted security perimeter
Denial of service	Password, token, biometric	Lockout by multiple failed authentications	Multifactor with token

REST of Chapter

- SELF READ

Sheet 3

- Text book Problems
  - Review Questions:
    - All
  - Problems:
    - MUST: 1,3,5,7,10
    - OPTIONAL: rest of them

In the next episode!!

- Access Control

- *How to prevent them from getting what they want, if you do not want them to get it*