Information Security

CS3002

Lecture 18

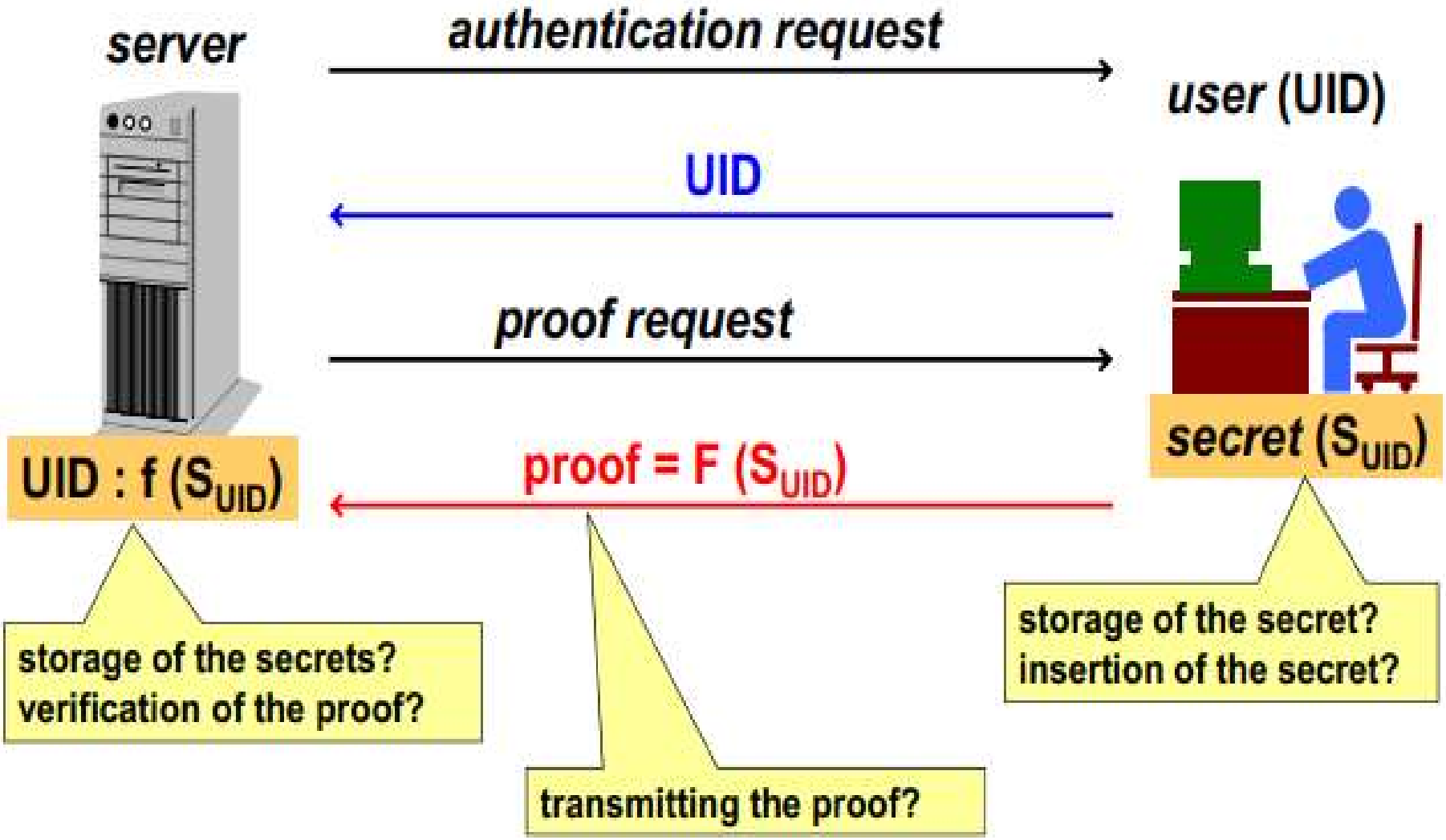
23rd October 2024

Dr. Rana Asif Rehman

Email: r.asif@lhr.nu.edu.pk

Authentication

Authentication



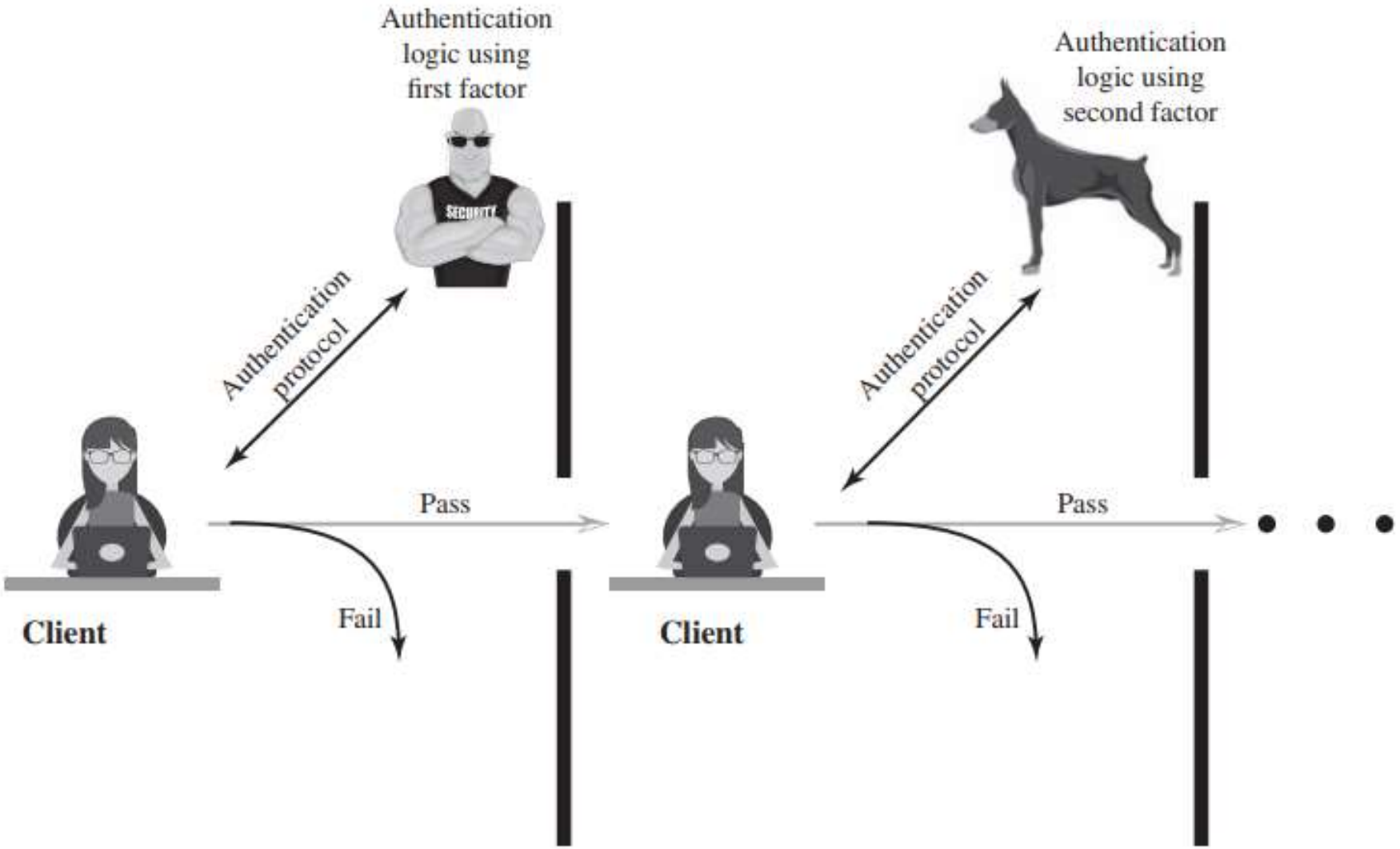
Authentication Methodologies

- Something you know (e.g: password)
- Something you have (e.g: smart card)
- Something you are (e.g: fingerprint)

- Can be based on multiple factors
  - (1/2/3 - factors authentication)

- Multifactor authentication is the combination of the above. E.g: PIN Enabled smart card
- Other methods:
  - Information about a user. E.g: attribute authentication
    - Voice patterns, typing rhythm
  - Location of a user

Multifactor Authentication



Types of Authentication

- There are two basic types of authentication: non-repudiable and repudiable.

- Repudiable Authentication – involves factors, “what you know” and “what you have,” that can present problems to the authenticator because the information presented can be unreliable because such factors suffer from several well-known problems including the fact that possessions can be lost, forged, or easily duplicated.
- Non-repudiable Authentication - involves characteristics whose proof of origin cannot be denied. Such characteristics include biometrics like iris patterns, retinal images, and hand geometry and they positively verify the identity of the individual.

Authentication Mechanisms

- In general authentication takes one of the following three forms:
  - Basic authentication involving a server: The server maintains a user file of either passwords and user names or some other useful piece of authenticating information. This information is always examined before authorization is granted.
  - Challenge-response: in which the server or any other authenticating system generates a challenge to the host requesting for authentication and expects a response.
  - Centralized authentication, in which a central server authenticates users on the network and in addition also authorizes and audits them.

1. Password-based Authentication

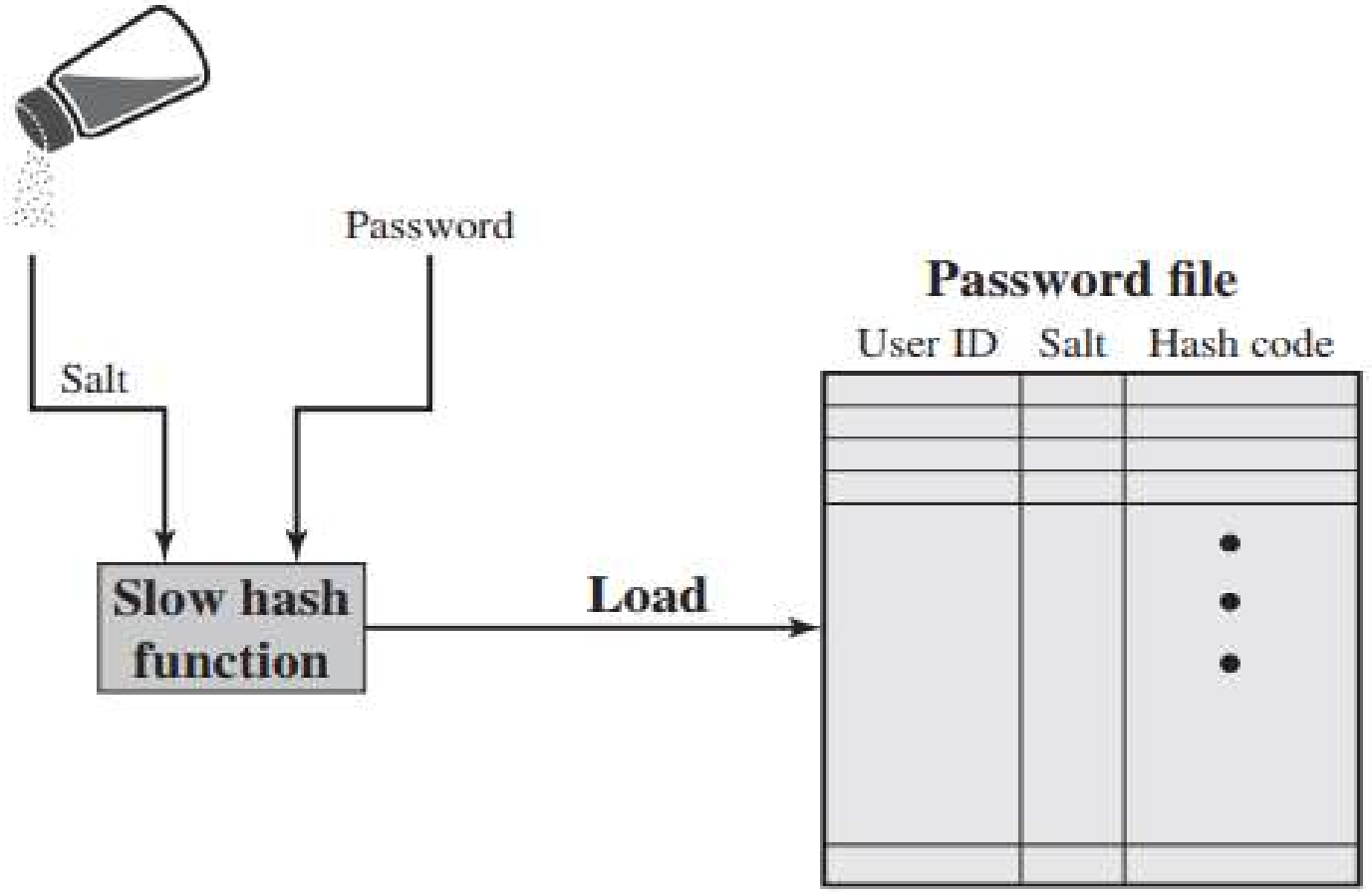
- secret = the user password
- (client) create and transmit proof
  - F = I (the identity function)
  - i.e. proof = password (cleartext!)

- (server) verify the proof:
  - case #1: F = I (the identity function)
    - server knows all passwords in cleartext (!)
    - access control: proof = password ?
  - case #2: F = one-way hash (that is a digest)
    - server knows the passwords' digests, HUID
    - access control: f(proof) = HUID ?

Passwords

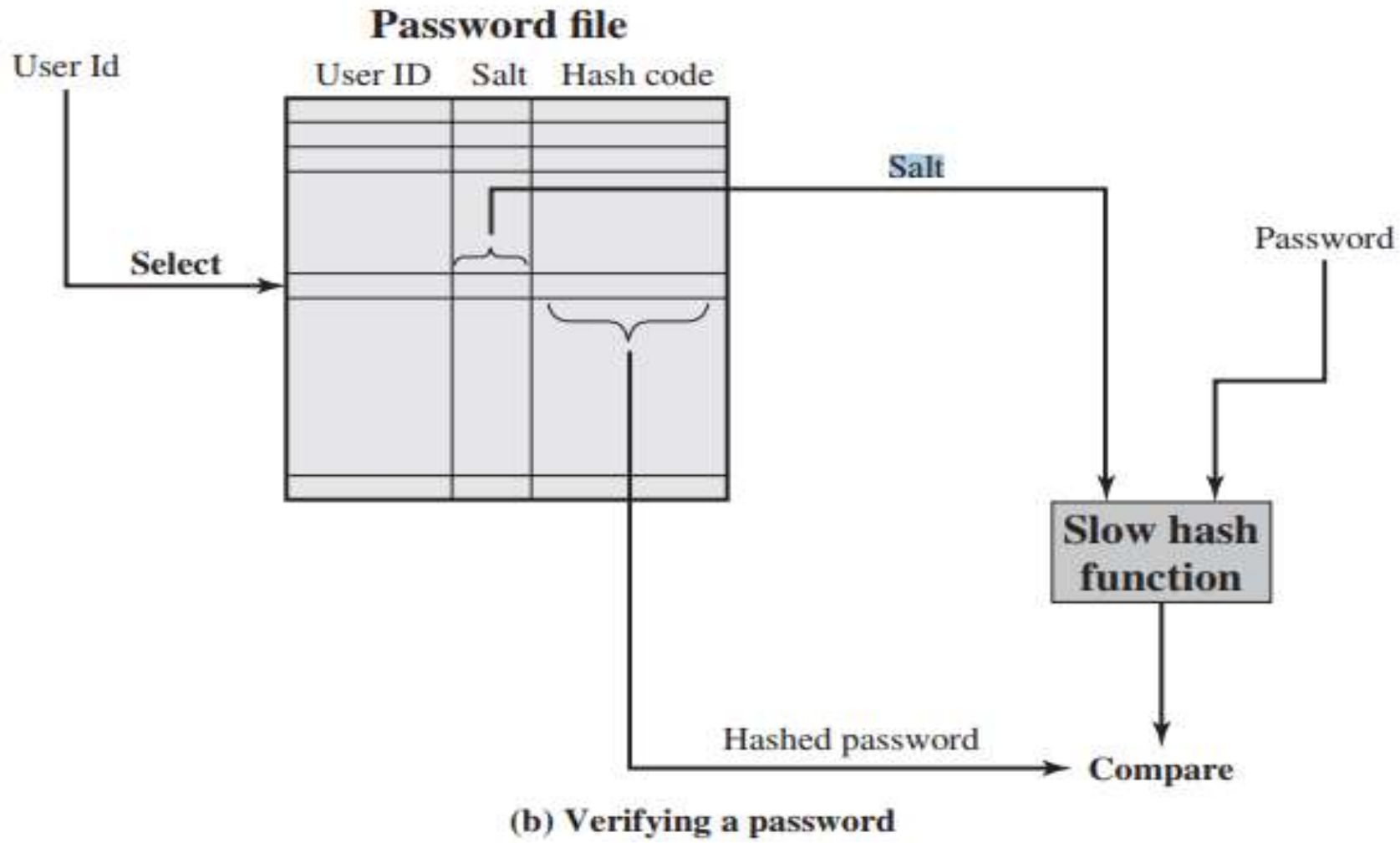
- Authentication based on alphanumeric characters or numbers
  - PROS
    - Easy to remember (if only for one system)
  - CONS
    - User-side password storage:
      - Post-it!
      - Client-side password manager or wallet
    - password guessable (my son's name!)
    - password readable during transmission
    - server-side password storage issues (hashing is must)
      - 35% passwords identified using dictionary attack
      - Use "salt"
    - Shoulder surfing
    - Using same password in multiple places

Using “salt” and hash



(a) Loading a new password

Using “salt” and hash



(b) Verifying a password

Using “salt” and hash

- for each user UID:
  - create / ask the password
  - generate a random salt (should contain rarely used or control characters)
- compute HP = hash (password | salt)
- store the triples { UID, HP, $salt_{UID}$ }

- Advantages:
  - Prevents duplicate passwords from being visible in the password file (different HP for users having the same password)
  - Increases the difficulty of offline dictionary attacks
  - Nearly impossible to tell if a person used the same password on multiple systems

2. Token-based Authentication

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

Memory Cards

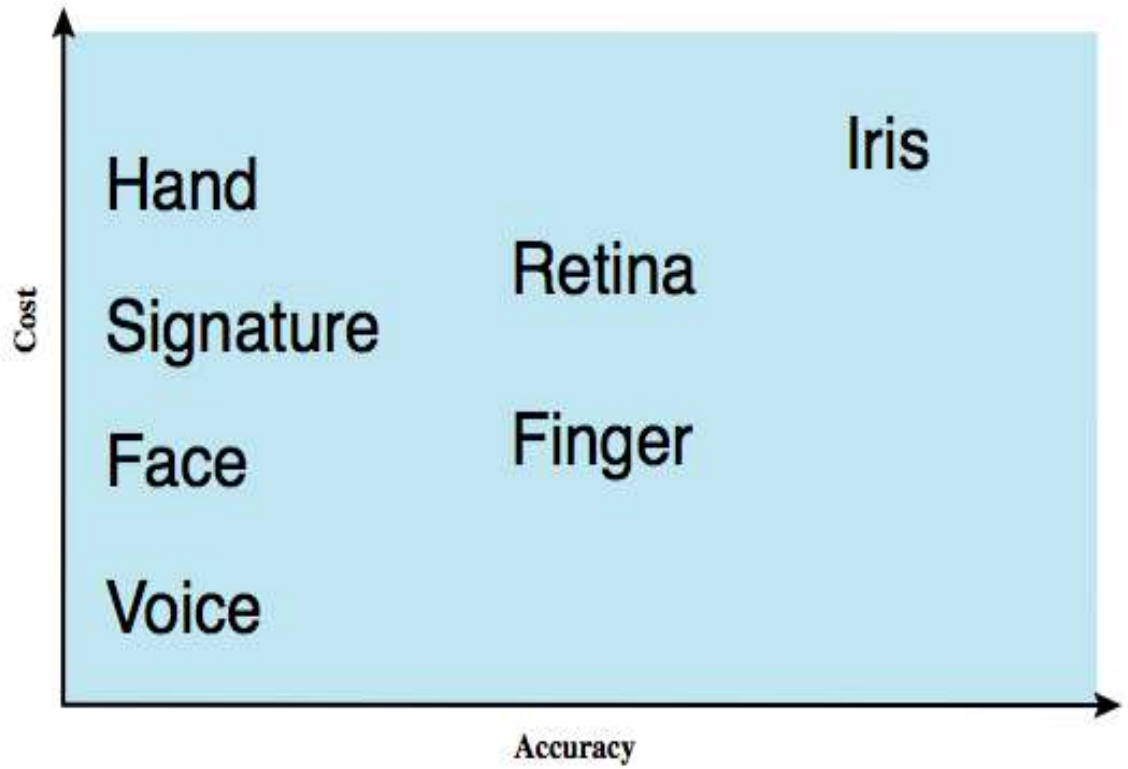
- Memory cards can store but not process data. The most common such card is the bank card with a magnetic stripe on the back
- Memory cards can be used alone for physical access, such as a hotel room

Smart Cards

- A smart card contains within it an entire microprocessor, including processor, memory, and I/O ports
- the most important category of smart token is the smart card, which has the appearance of a credit card, has an electronic interface, and may use any of the type of protocols

3. Biometric Authentication

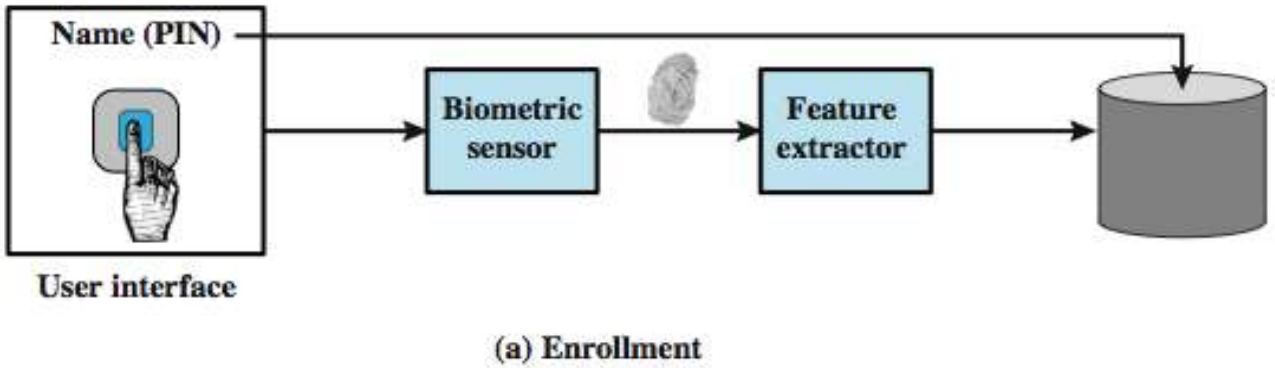
- Authenticate user based on one of their physical characteristics:
  - facial
  - fingerprint
  - hand geometry
  - retina pattern
  - iris
  - signature
  - voice



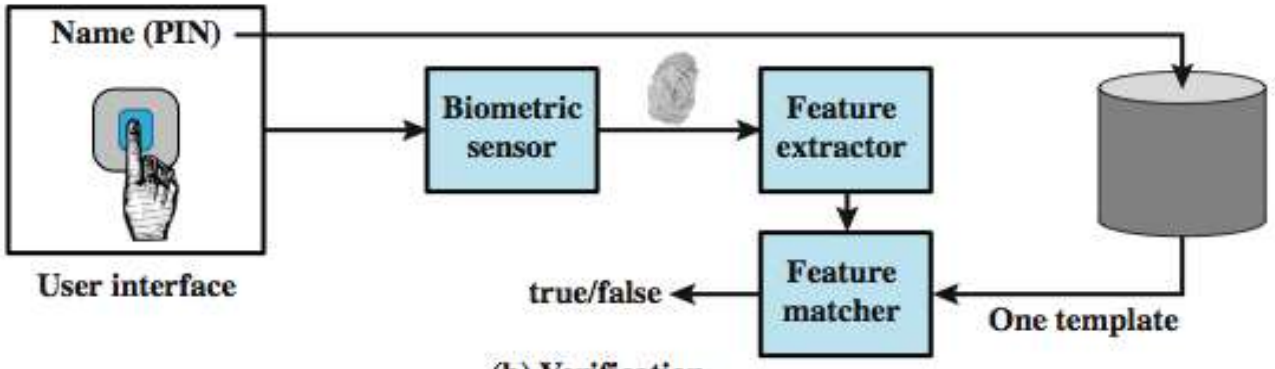
Operation of a Biometric System

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

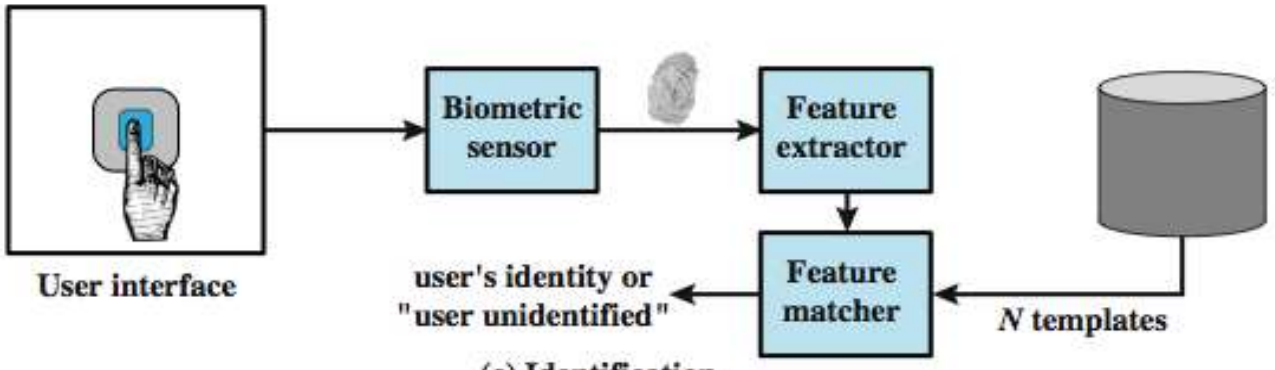
Identification *is biometric info but no IDs; system compares with stored templates*



(a) Enrollment



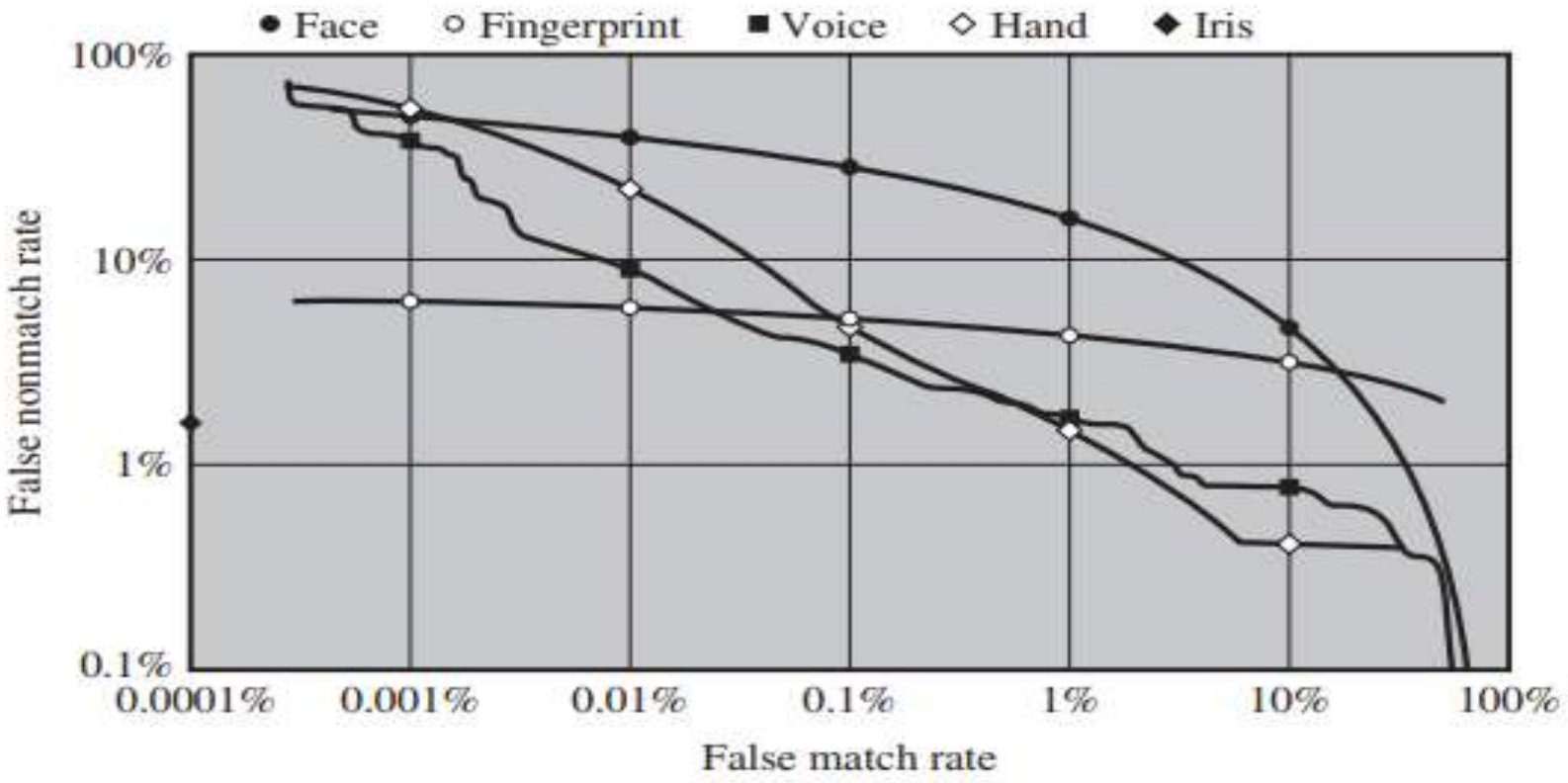
(b) Verification



(c) Identification

Problems of Biometric Systems

- FAR = False Acceptance Rate
- FRR = False Rejection Rate
- FAR and FRR may be partly tuned but they heavily depend on the cost of the device



Problems of Biometric Systems

- Variable biological characteristics:
  - finger wound
  - voice altered due to emotion or injury
  - retinal blood pattern altered due to alcohol or drug

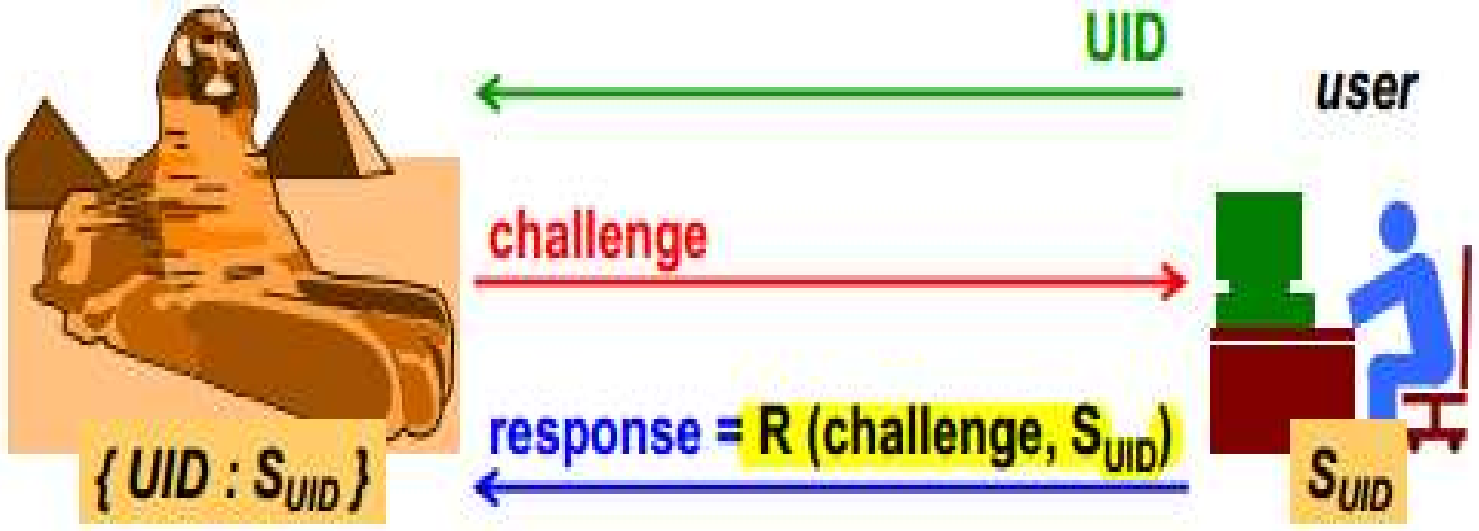
Problems of Biometric Systems

4. Remote User Authentication

- Remote user authentication raises additional security threats, such as an eavesdropper being able to capture a password, or an adversary replaying an authentication sequence that has been observed
- To counter threats to remote user authentication, systems generally rely on some form of challenge-response protocol.

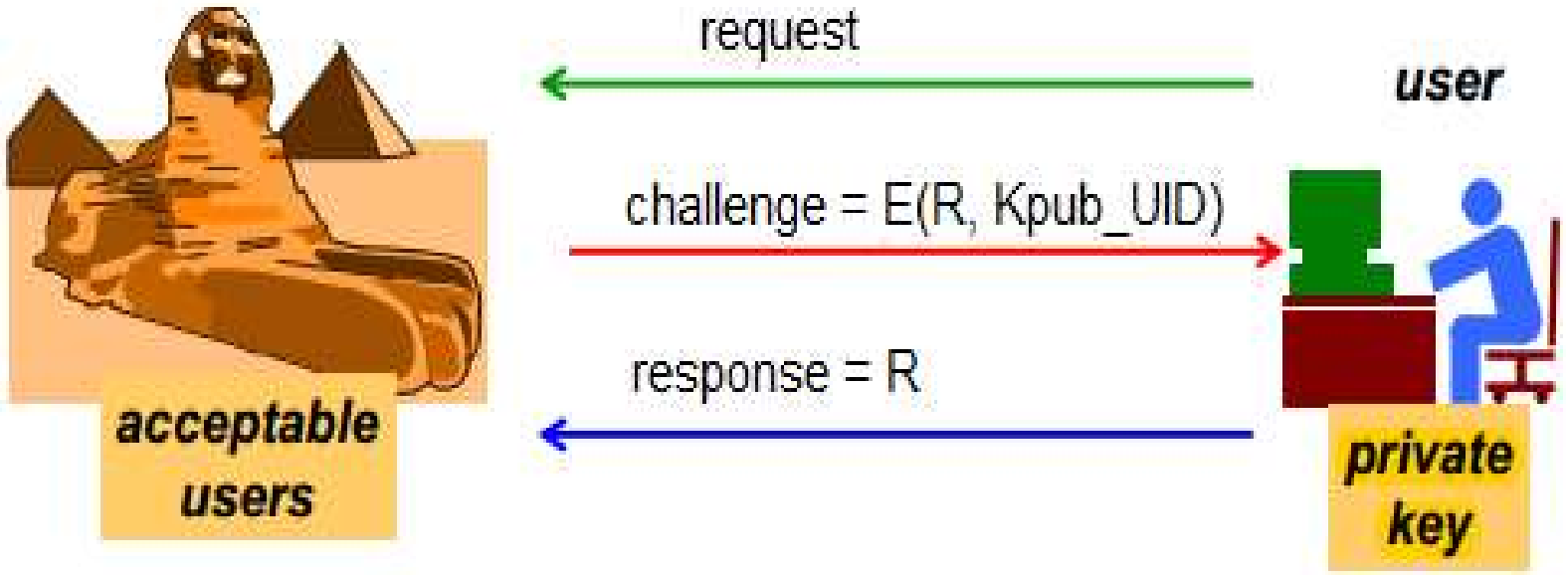
Symmetric Challenge-Response Authentication

- a challenge (typically a random nonce) is sent to the user ...
- … who replies with the solution after a computation
- involving the shared secret and the challenge
- the server must know the secret in clear
- often R is a hash function (can't be encryption)

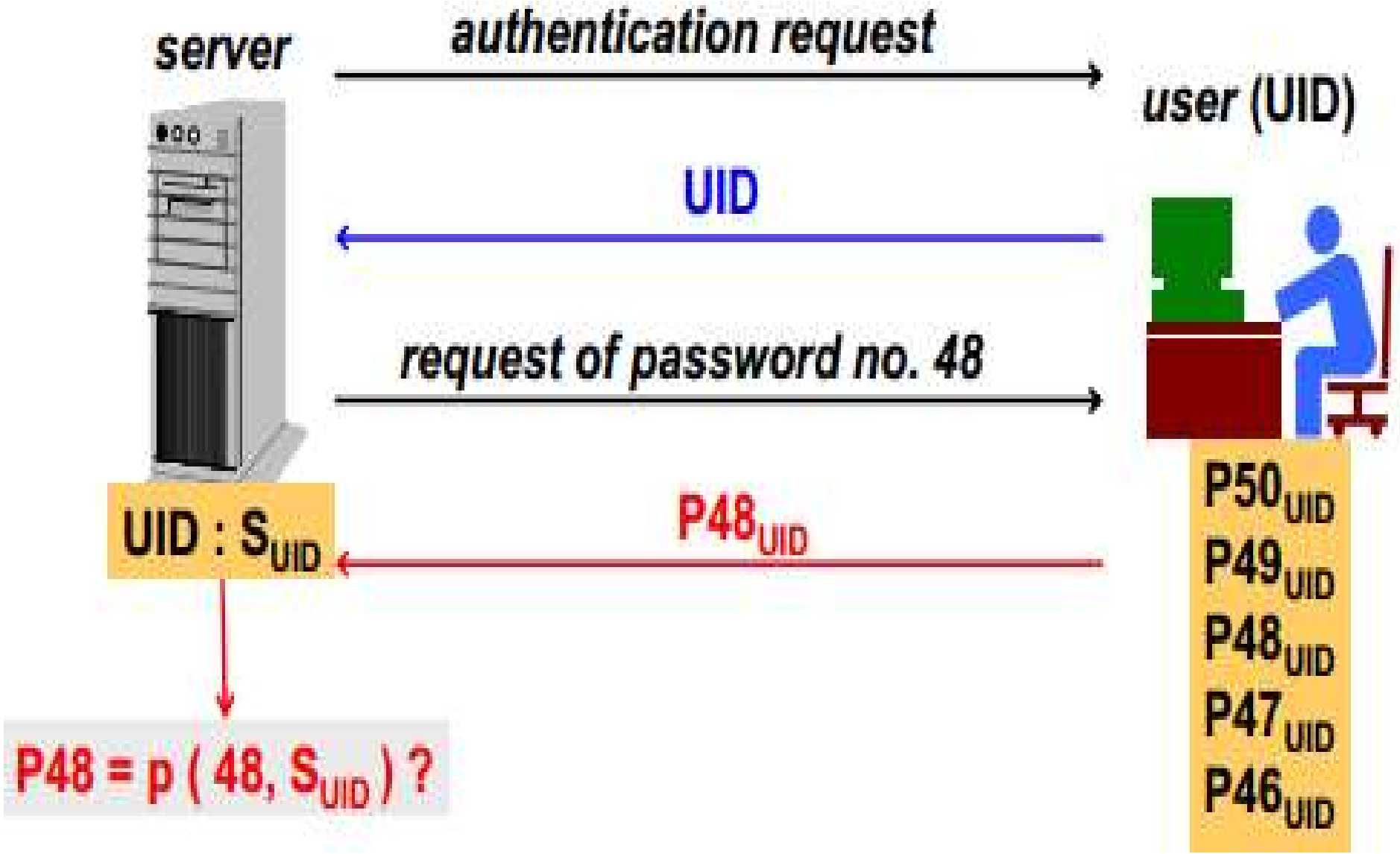


Asymmetric Challenge-Response System

- a random number R is encrypted with the user's public key ...
- and the users replies by sending R in clear thanks to its knowledge of the private key



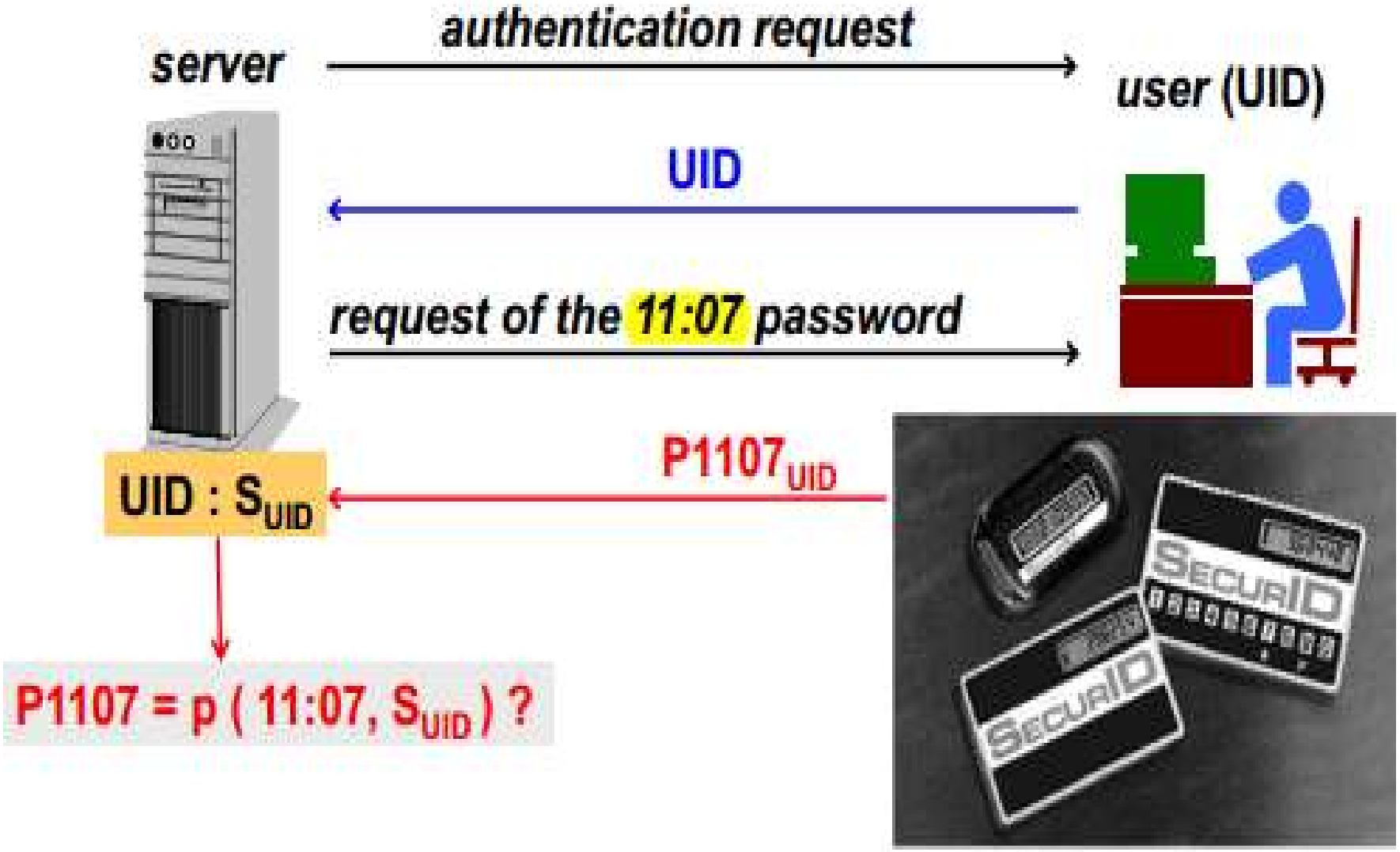
One-Time Passwords



One-Time Passwords

Núm.	Clau	Núm.	Clau	Núm.	Clau	Núm.	Clau	Núm.	Clau
1	1232	13	2768	25	9921	37	5095	49	2132
2	3476	14	2924	26	8132	38	6768	50	4095
3	4874	15	3492	27	7095	39	4924	51	5768
4	5921	16	4195	28	6768	40	3492	52	6924
5	6132	17	5232	29	5924	41	6195	53	7492
6	7095	18	8476	30	4232	42	4232	54	5195
7	8768	19	7874	31	3476	43	2476	55	4232
8	7924	20	6921	32	2874	44	4874	56	4476
9	5492	21	5132	33	2492	45	6921	57	3874
10	4195	22	4095	34	4195	46	2132	58	3921
11	7095	23	5232	35	4232	47	3492	59	7492
12	8768	24	8476	36	3476	48	6195	60	5195

Passwords (one-time token based)

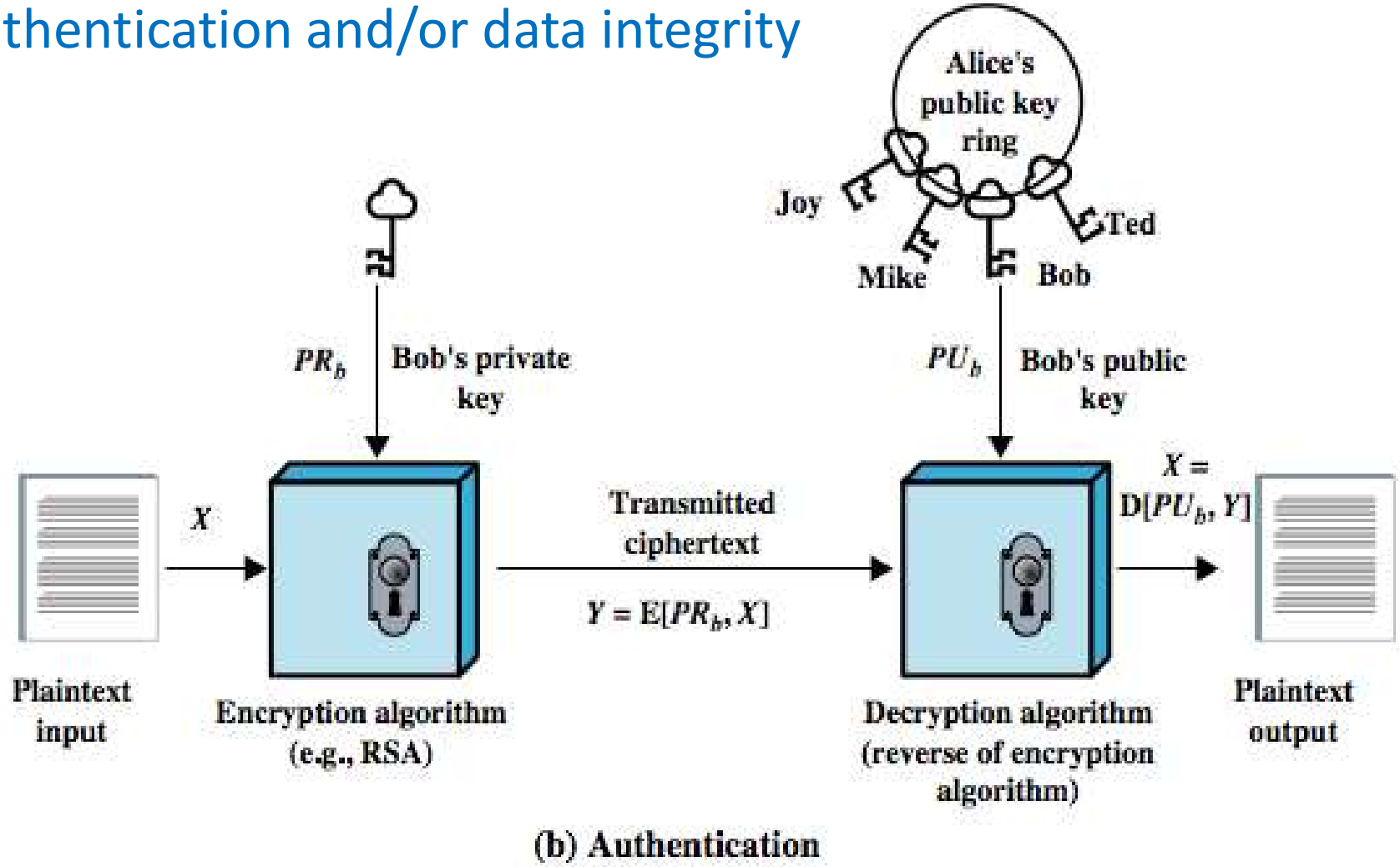


Internet Authentication Applications

1. Authentication using Public-Key

Authentication using Public-Key

Authentication and/or data integrity



(b) Authentication

PKI and Certificate Authorities

- Certificate consists of:
  - A public key plus a User ID of the key owner
  - Signed by a third party trusted by community
  - Often govt/bank **certificate authority** (CA)
- Users obtain certificates from CA
  - Create keys & unsigned cert, gives to CA, CA signs cert & attaches sig, returns to user
- Other users can verify cert
  - Checking sig on cert using CA's public key

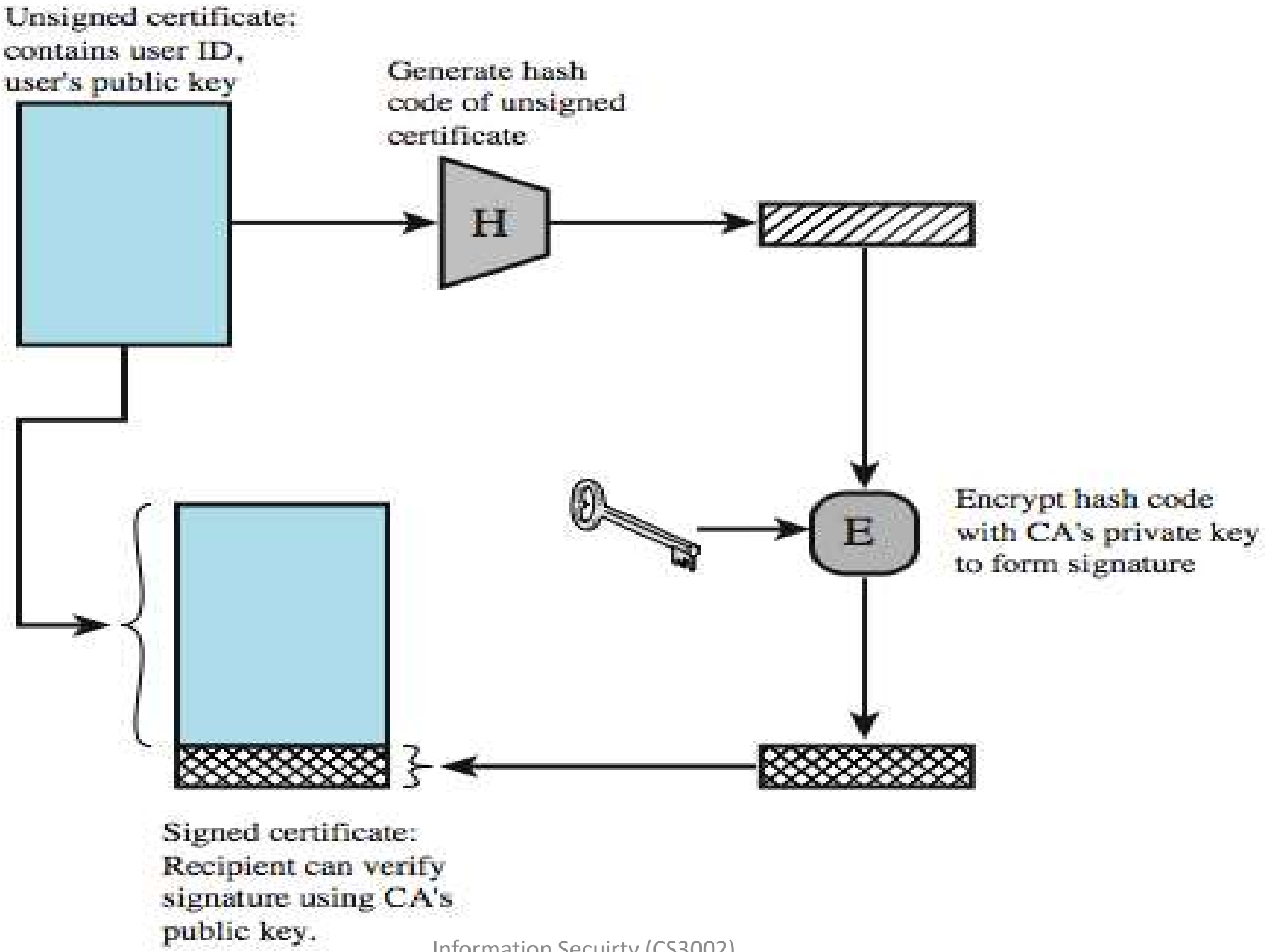
Common Key Steps

1. User software creates a pair of keys: private and public
2. Clients prepares unsigned certificate that includes user ID and public key
3. User provides unsigned certificate to a CA
4. CA creates a signature:
   i. Creates a hash code of the unsigned certificate
   ii. Encrypts the hash code with the CA's private key
5. CA attaches the signature to unsigned certificate to make signed certificate

Key Steps (cont..)

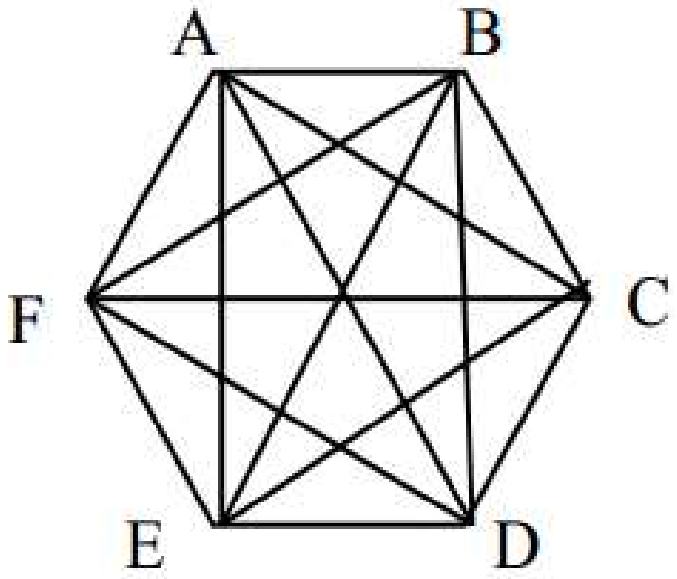
6. CA returns the signed certificate to the client
7. Client may provide signed signature to other users
8. Any user may verify the certificate
   I. Calculate the hash code of certificate (exclude signature)
   II. Decrypt signature using CA's public key
   III. Compare the two

Public Key Certificates



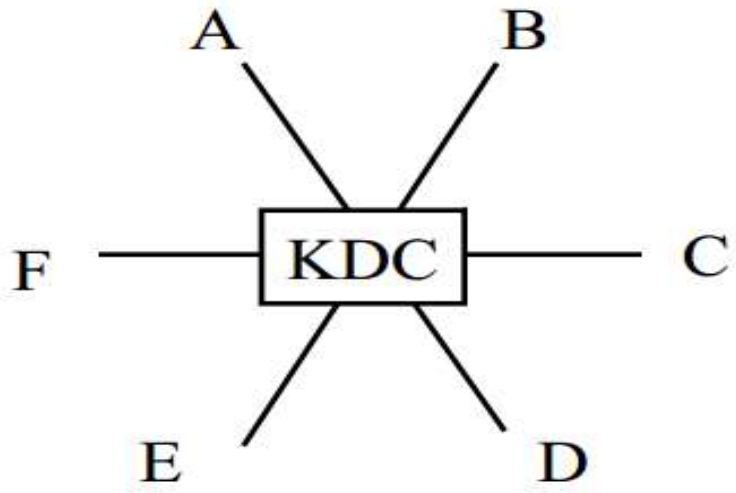
2. Authentication using Secret-Key

Secret keys for N-system network (Problem)



- n system need n(n-1) pairs of secret keys
- Each system remembers n-1 keys.
- If a new system comes in n new key are generated.
- If a system leaves, n-1 keys are removed.

Key Distribution Center (KDC)



- Each node is configured with KDC's key.
- KDC has all the keys.
- A ↔ B communication? KDC sends a key K_{AB} encrypted with A's key to A and B's key to B.

- Issues:
  - If KDC is compromised, all systems are compromised.
  - KDC is single point of failure or performance bottleneck.
  - KDC has to be on-line all the time. Replication!

Kerberos

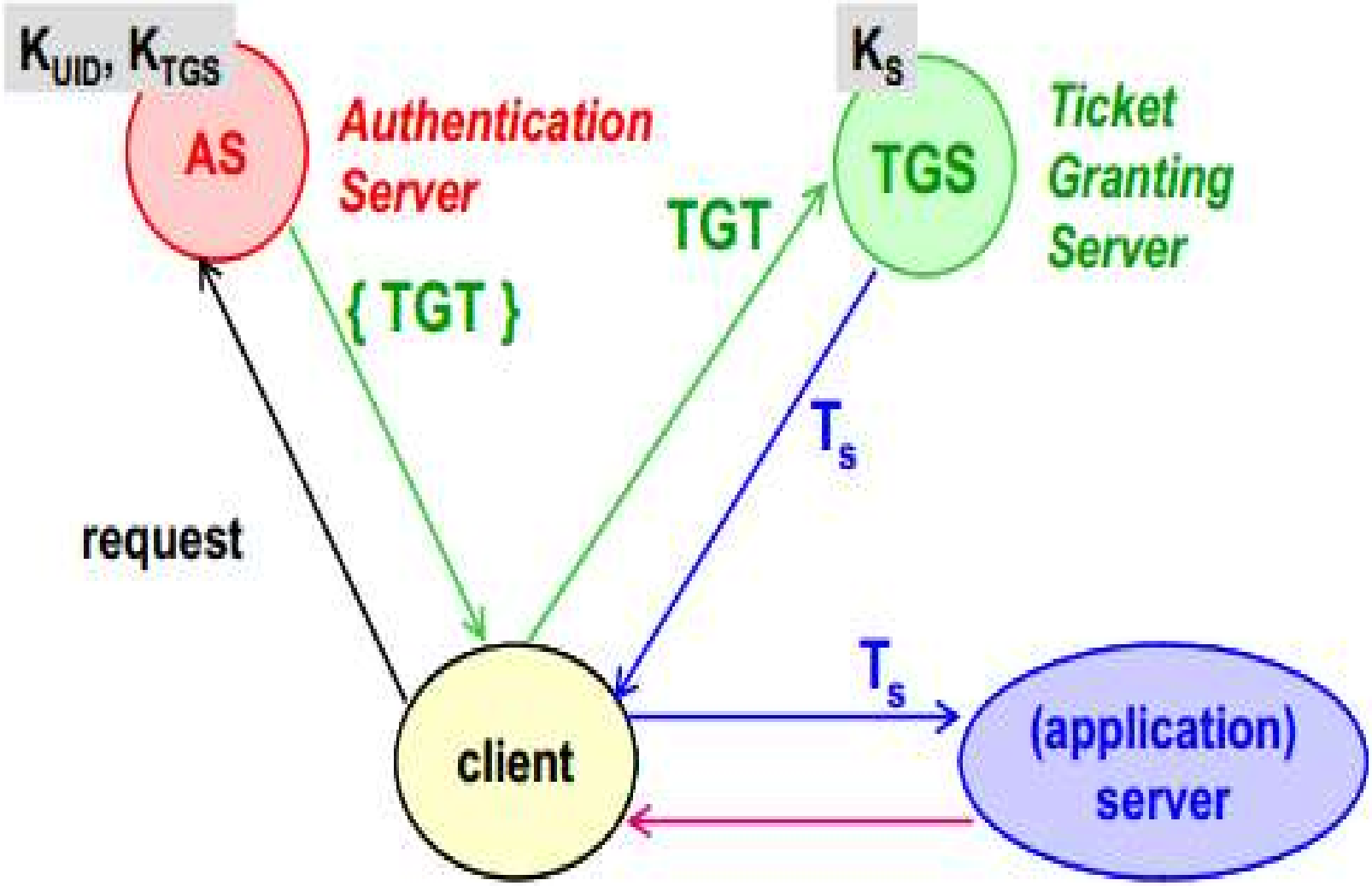
- Network authentication protocol
- Based on Trusted Third Party (TTP) - KDC
- invented by MIT for project Athena

- Named after Greek mythological character “Cerberus” Three headed dog protecting the entrance of Hades
- Used by popular operating systems and servers
- Protect against eavesdropping and firewall limitation to users and replay attacks

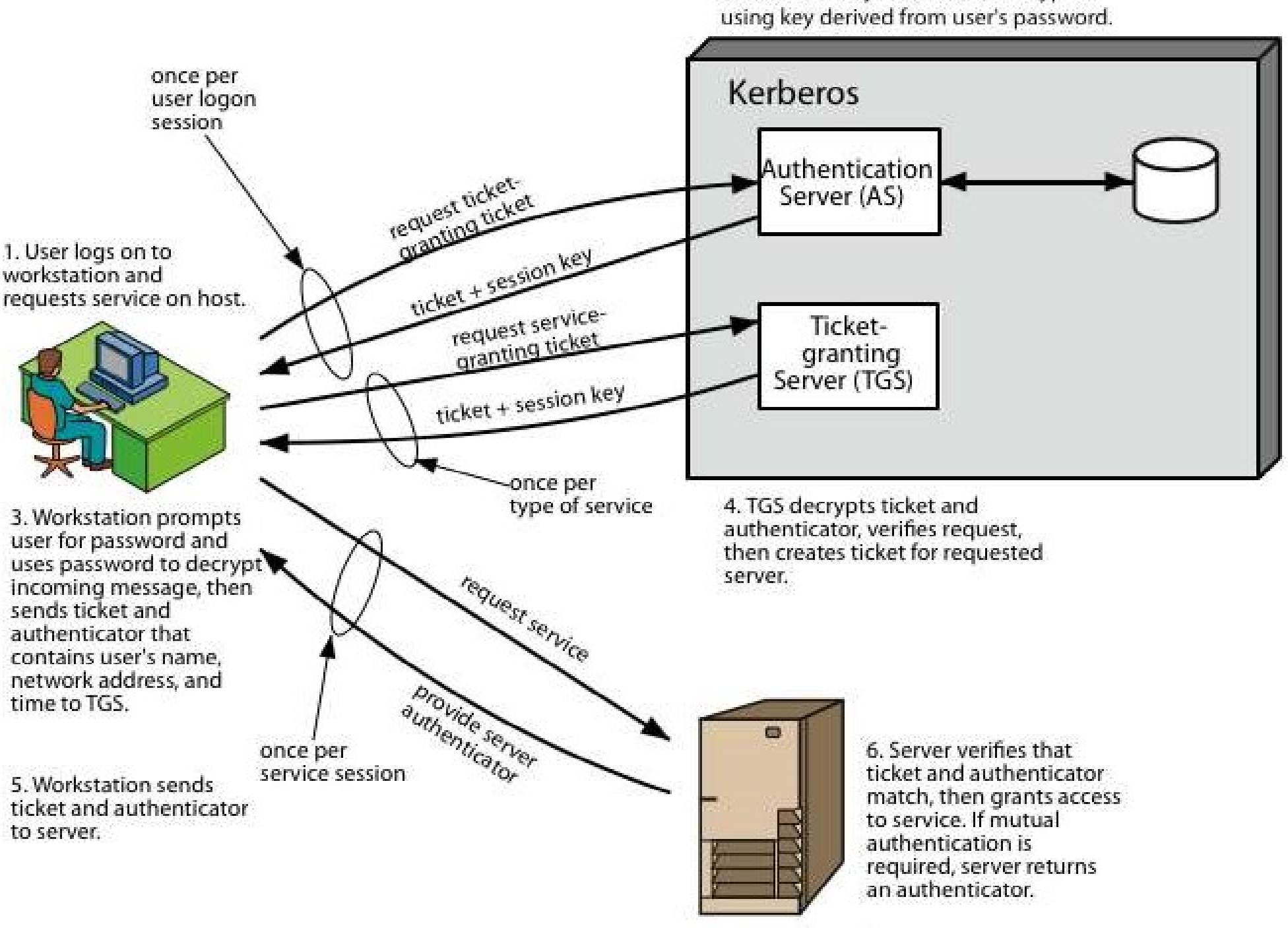
Kerberos Overview

- Authentication server authenticates a user to a specific service in the network
- TGS, Ticket Granting Server, grants ticket to the user
- Authentication server and TGS can be the same system. They work as a single unit.
- Application Server provides the service to the user
- The client/user, Auth. Server & TGS, Application server are the 3 heads of kerberos!

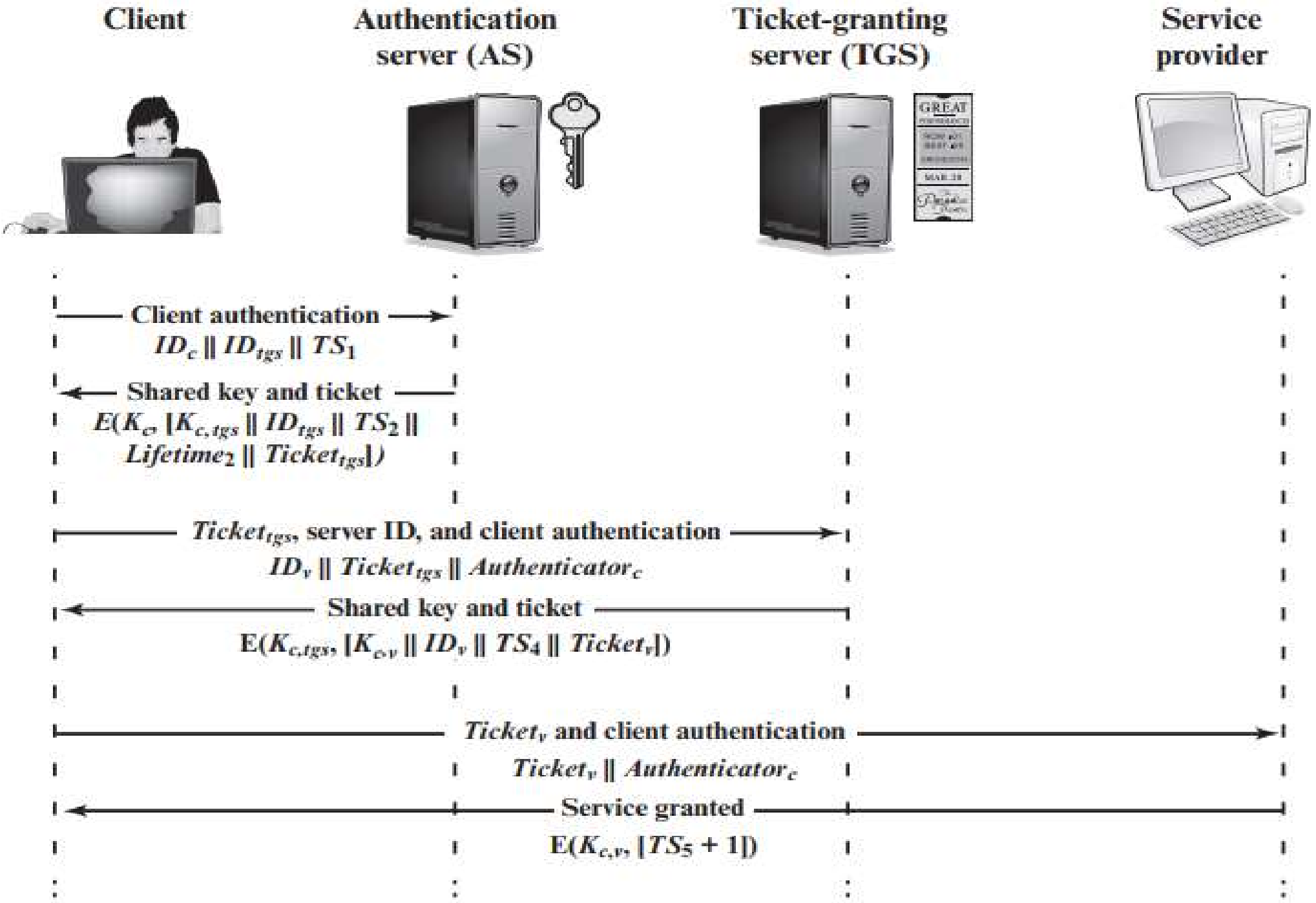
Kerberos High-Level View



Kerberos Protocol



Kerberos Exchanges



Other Authentication Systems

- OATH (open authentication)
  - Interoperability of authentication systems based on OTP, both symmetric and asymmetric

- SSO (single sign-on)
  - Single credential multiple services