Information Security
CS3002
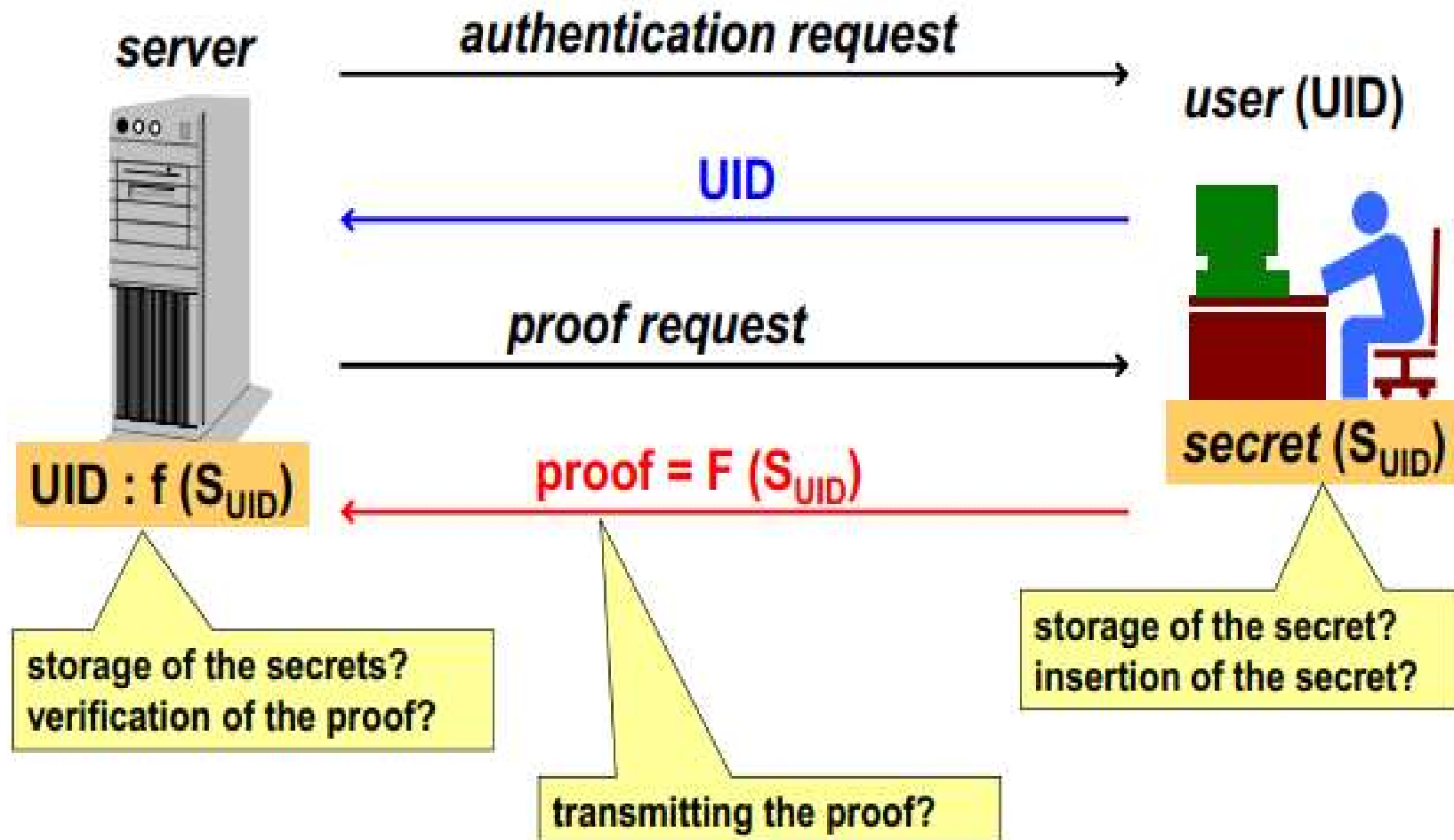
Lecture 18
31st October 2023

Dr. Rana Asif Rehman
Email: r.asif@lhr.nu.edu.pk

Authentication

Authentication

server

authentication request →

← UID

proof request →

UID : $f(S_{UID})$

← proof = $F(S_{UID})$

user (UID)

secret (S_{UID})

storage of the secrets?
verification of the proof?

transmitting the proof?

storage of the secret?
insertion of the secret?

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

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.

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 ?
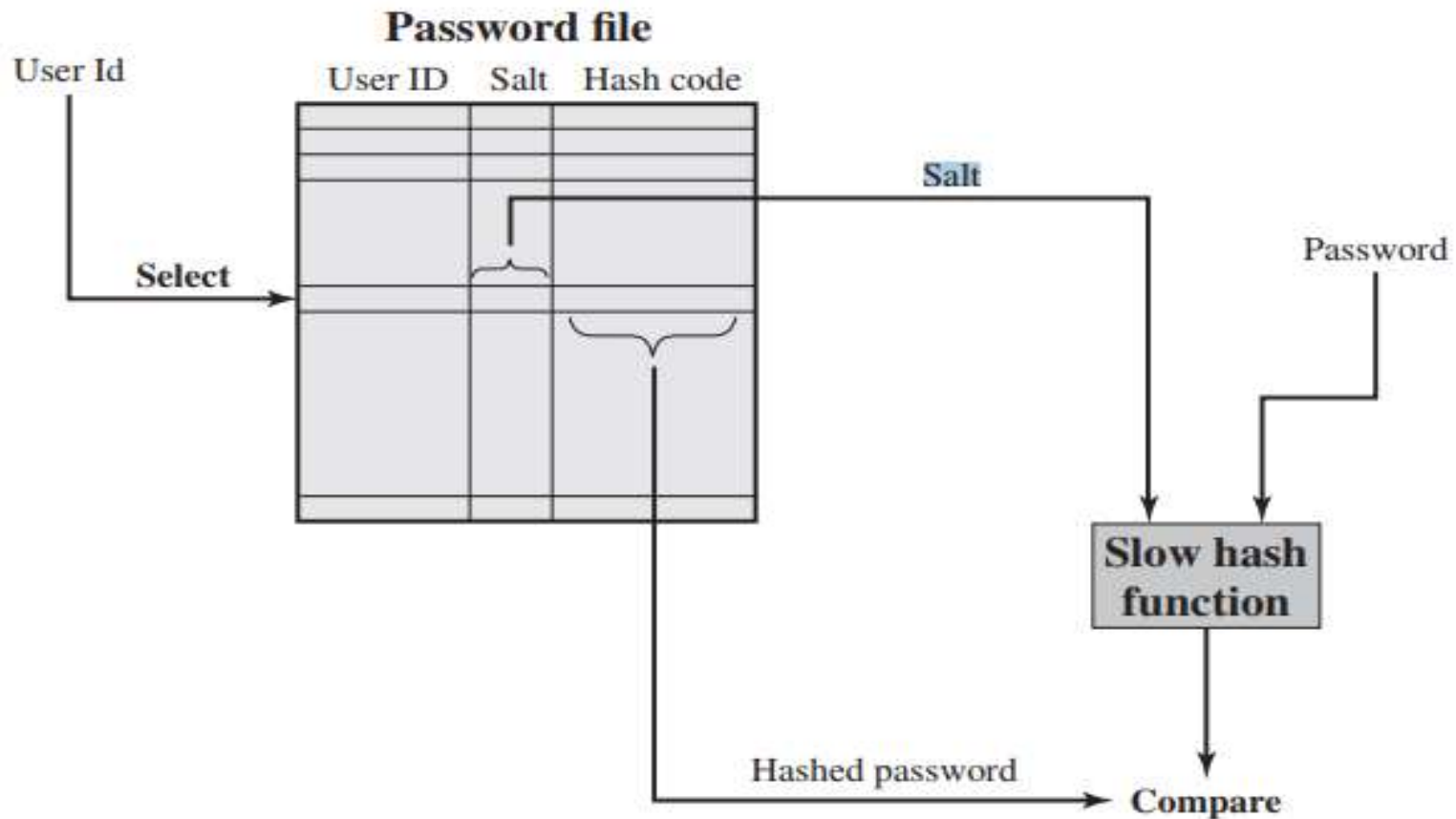
Passwords

- 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
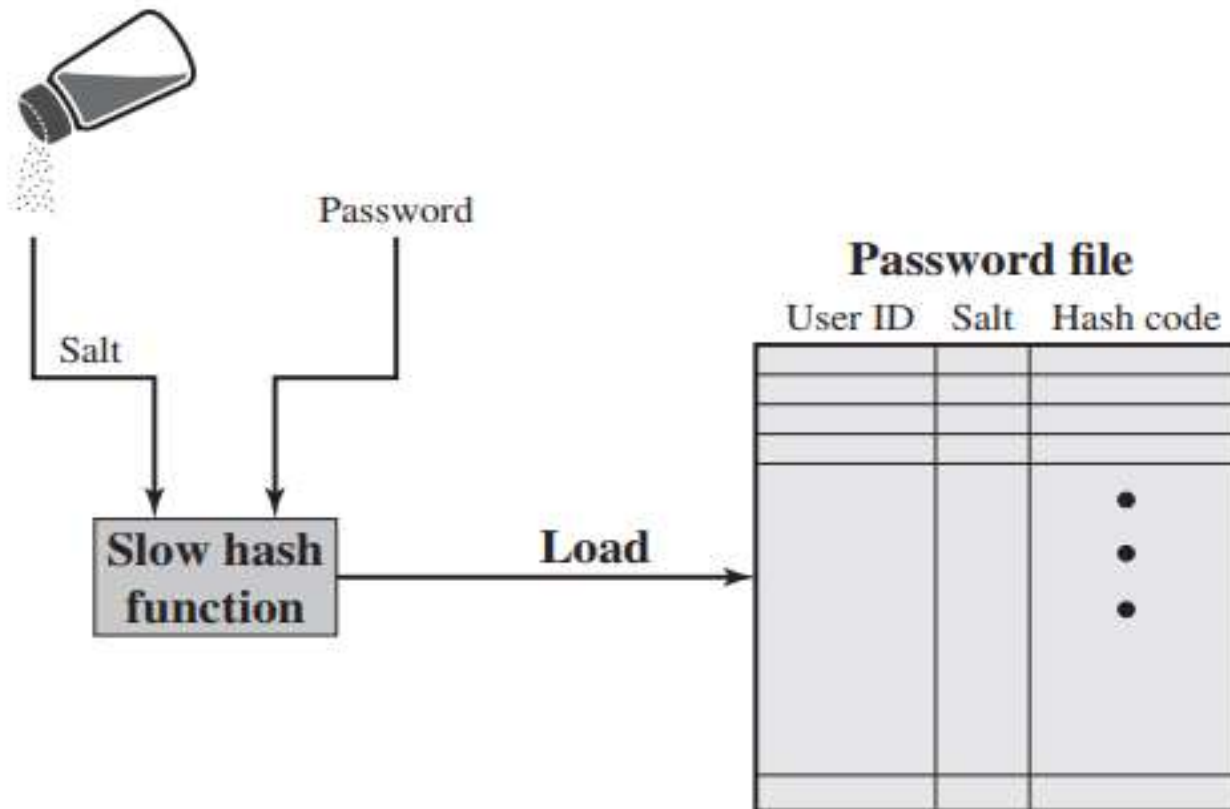
(a) Loading a new password

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

Biometric Authentication

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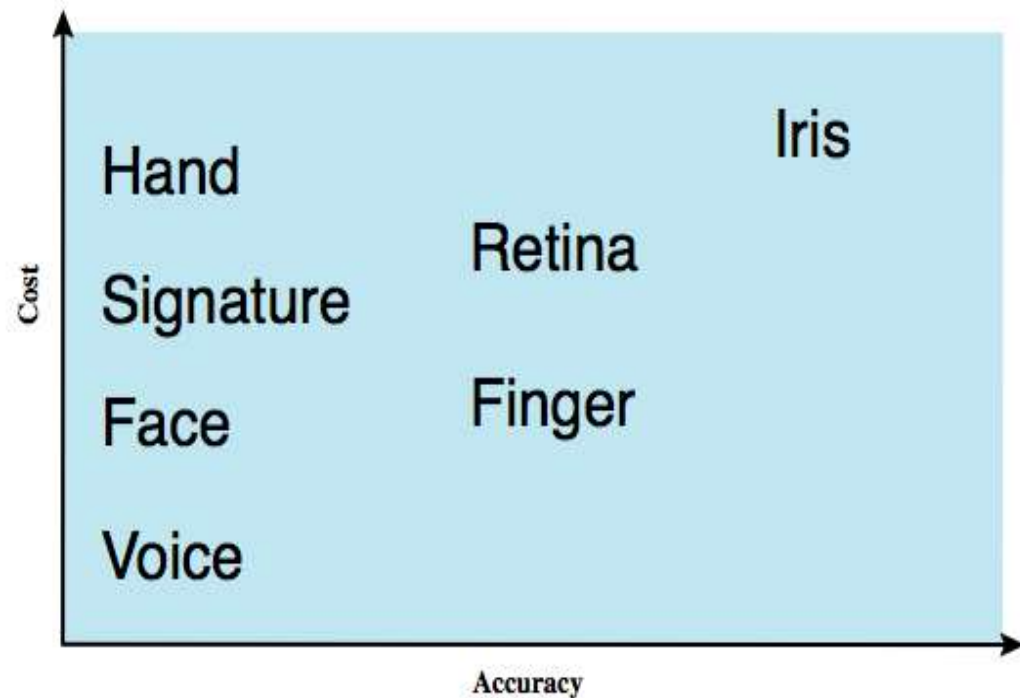
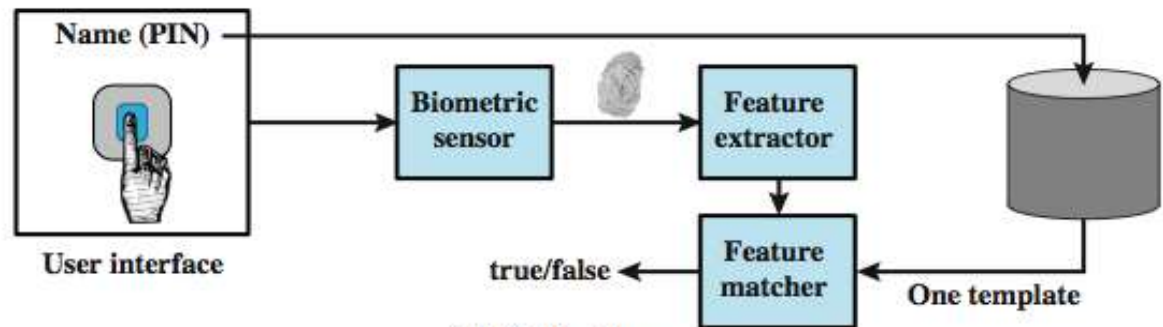# Operation of a Biometric System
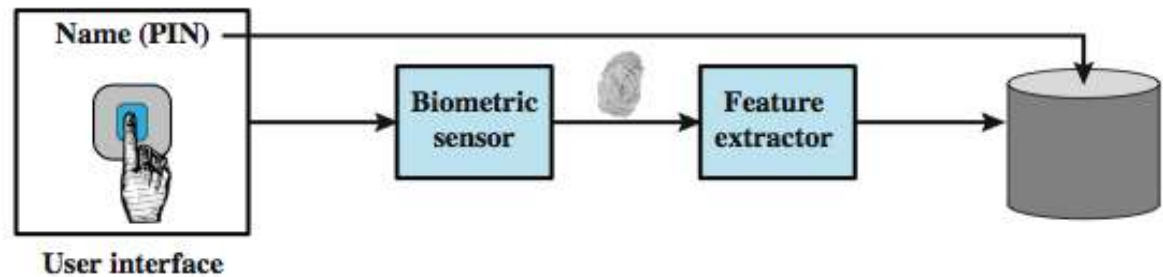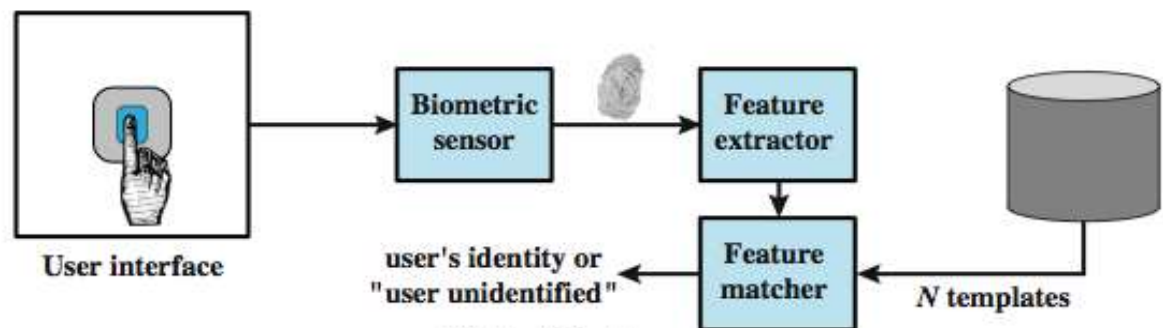

(a) Enrollment

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

(b) Verification

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

(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
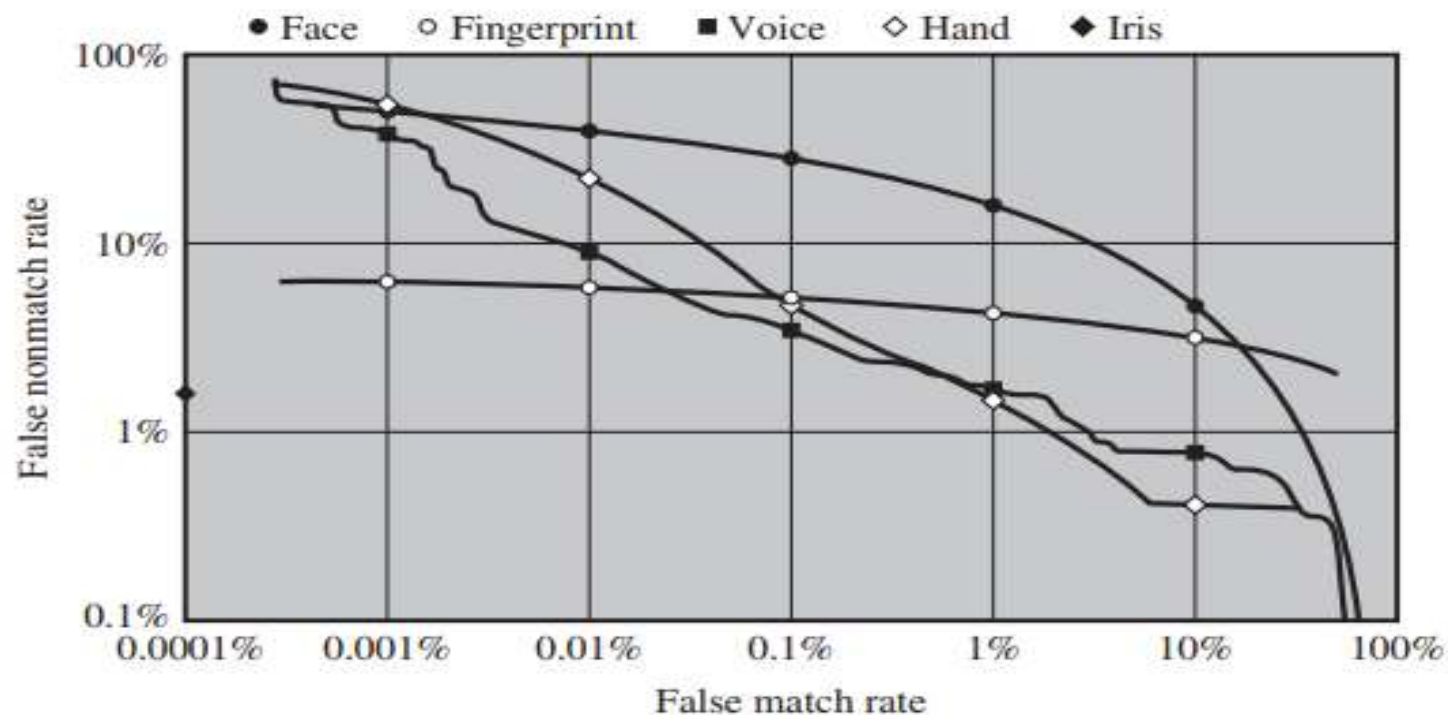
Problems of Biometric Systems

- Variable biological characteristics:
  - finger wound
  - voice altered due to emotion or injury (https://www.youtube.com/watch?v=iYhpbph4sLc)
  - retinal blood pattern altered due to alcohol or drug

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)

UID

user

challenge

response = R (challenge, S_{UID})

{ UID : S_{UID} }

S_{UID}
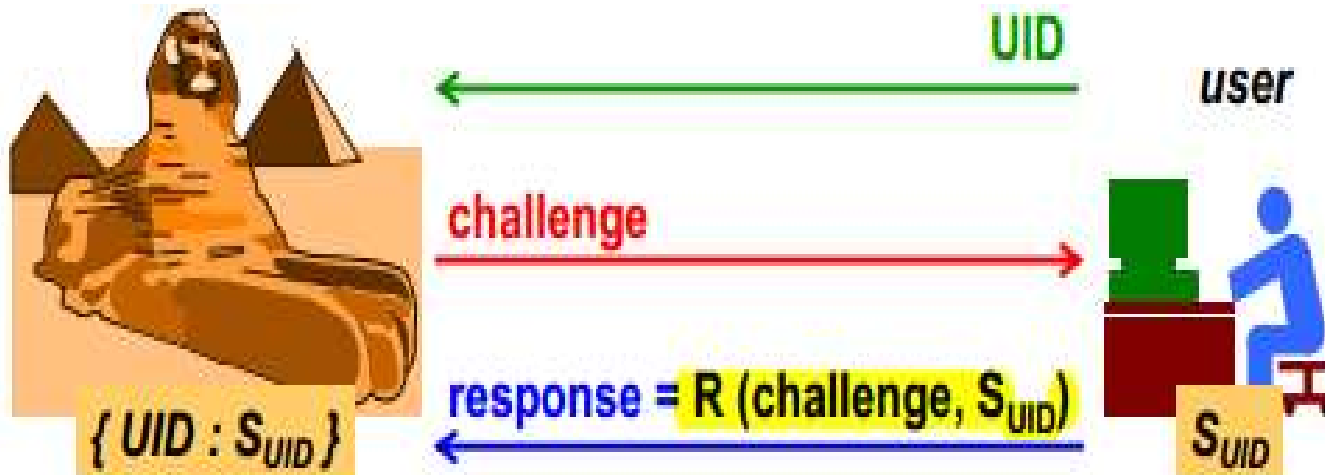
Asymmetric Challenge Response System

- 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

Information Secuirty (CS3002)

One-Time Passwords

server

authentication request →

UID ←

request of password no. 48 →

P48$_{UID}$ ←

UID : S$_{UID}$

P48 = p (48, S$_{UID}$) ?

user (UID)

P50$_{UID}$
P49$_{UID}$
P48$_{UID}$
P47$_{UID}$
P46$_{UID}$
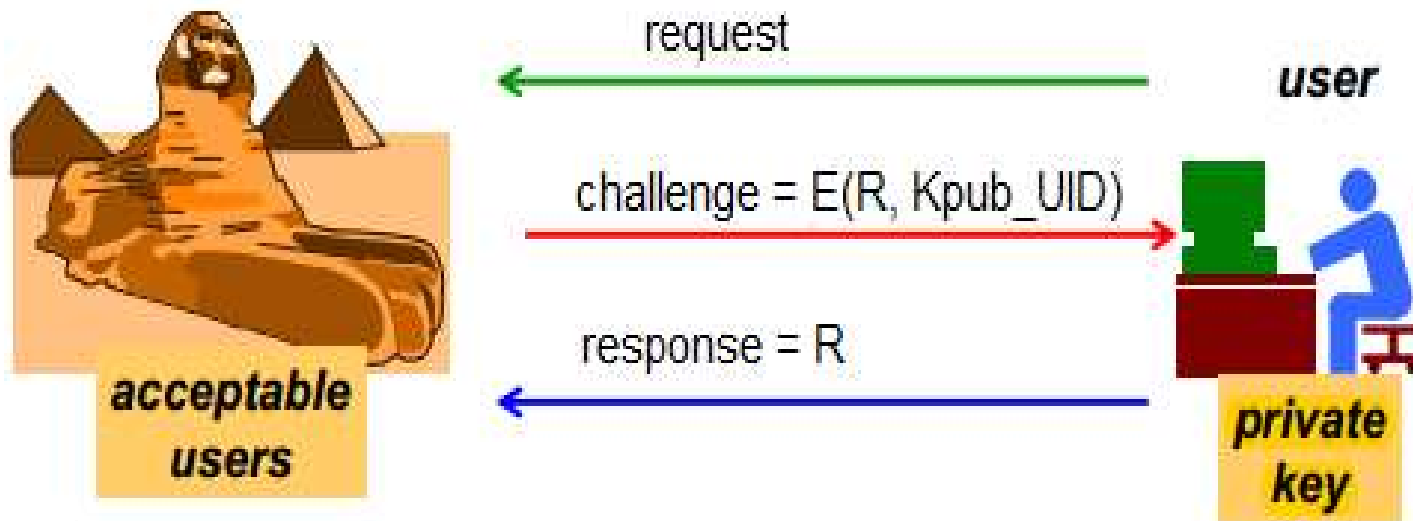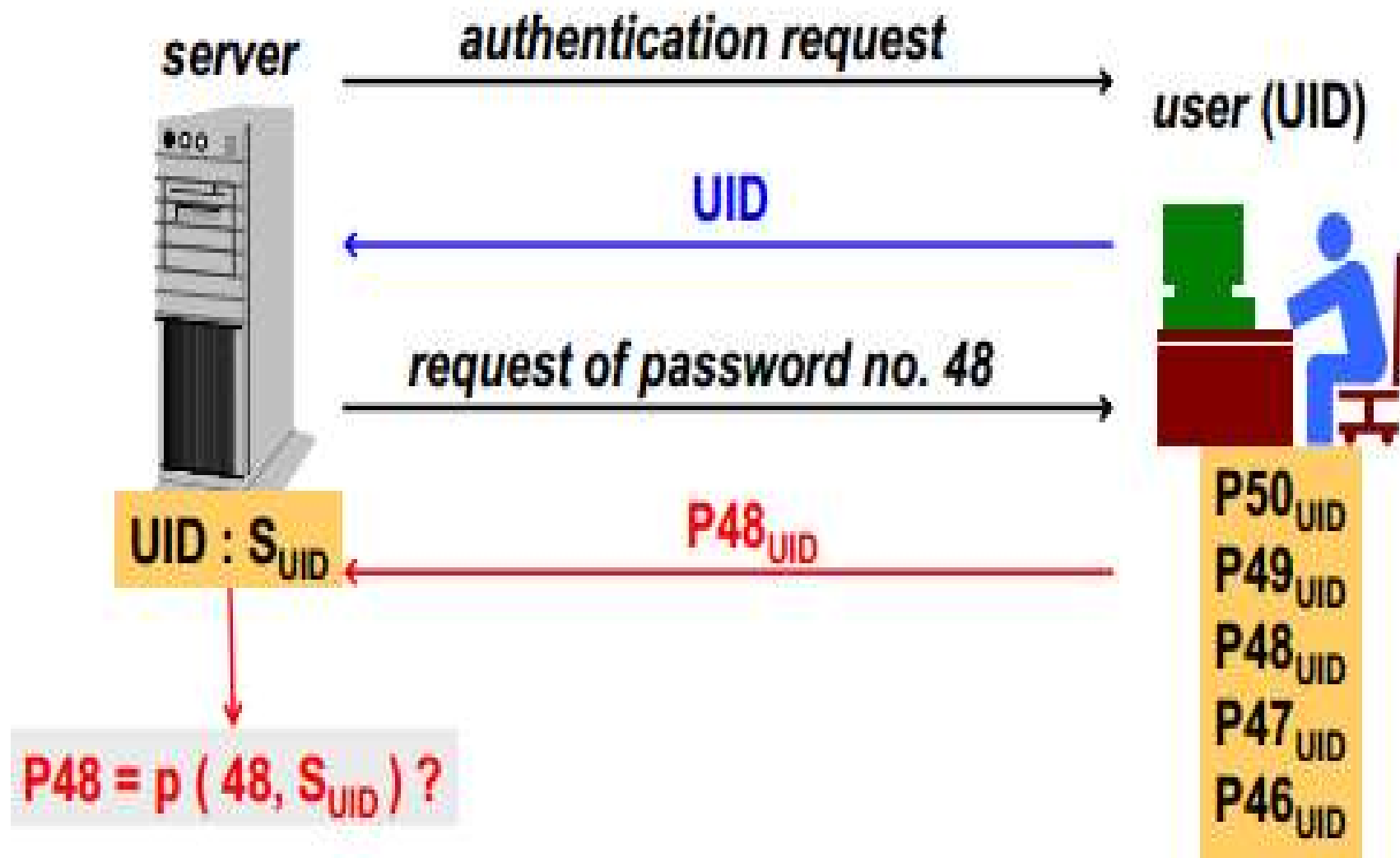
One-Time Passwords

Passwords (one-time token based)

server → user (UID): authentication request

user → server: UID

server → user: request of the **11:07** password

user → server: $P1107_{UID}$

$UID : S_{UID}$

$P1107 = p\,(\,11{:}07,\,S_{UID}\,)\,?$
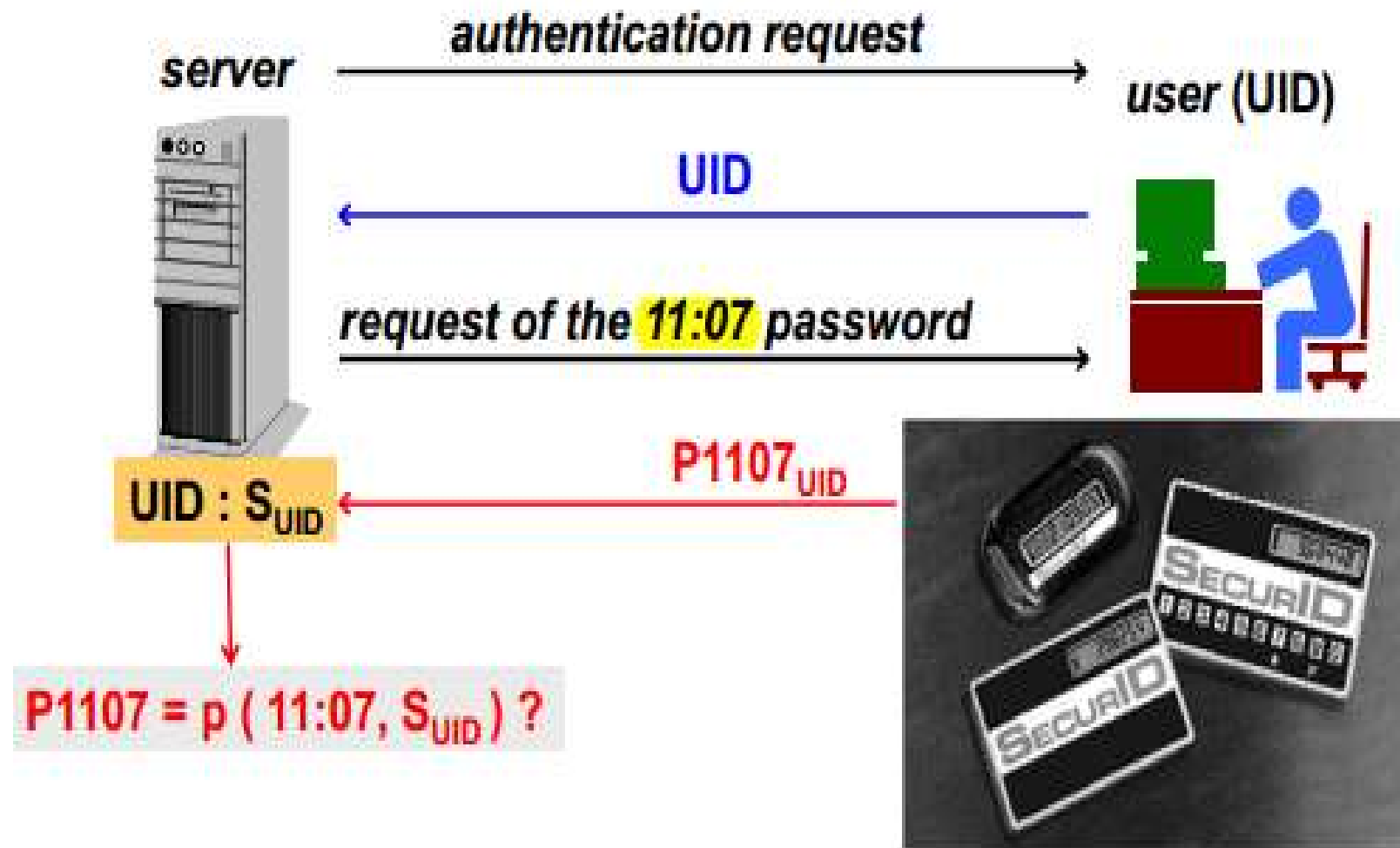
Secret keys for N-system network
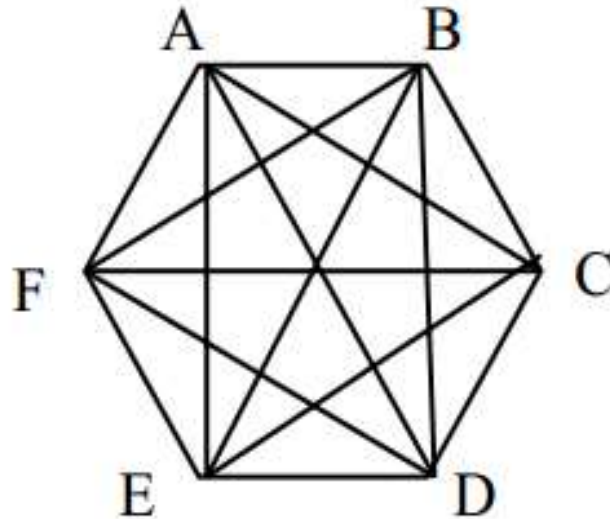
- n system need n(n-1) pairs of secret keys
- Each system remembers n-1 keys.
- If a new system comes in n new key are generated.
- If a system leaves, n-1 keys are removed.

Centralized Authentication

Authentication and/or data integrity
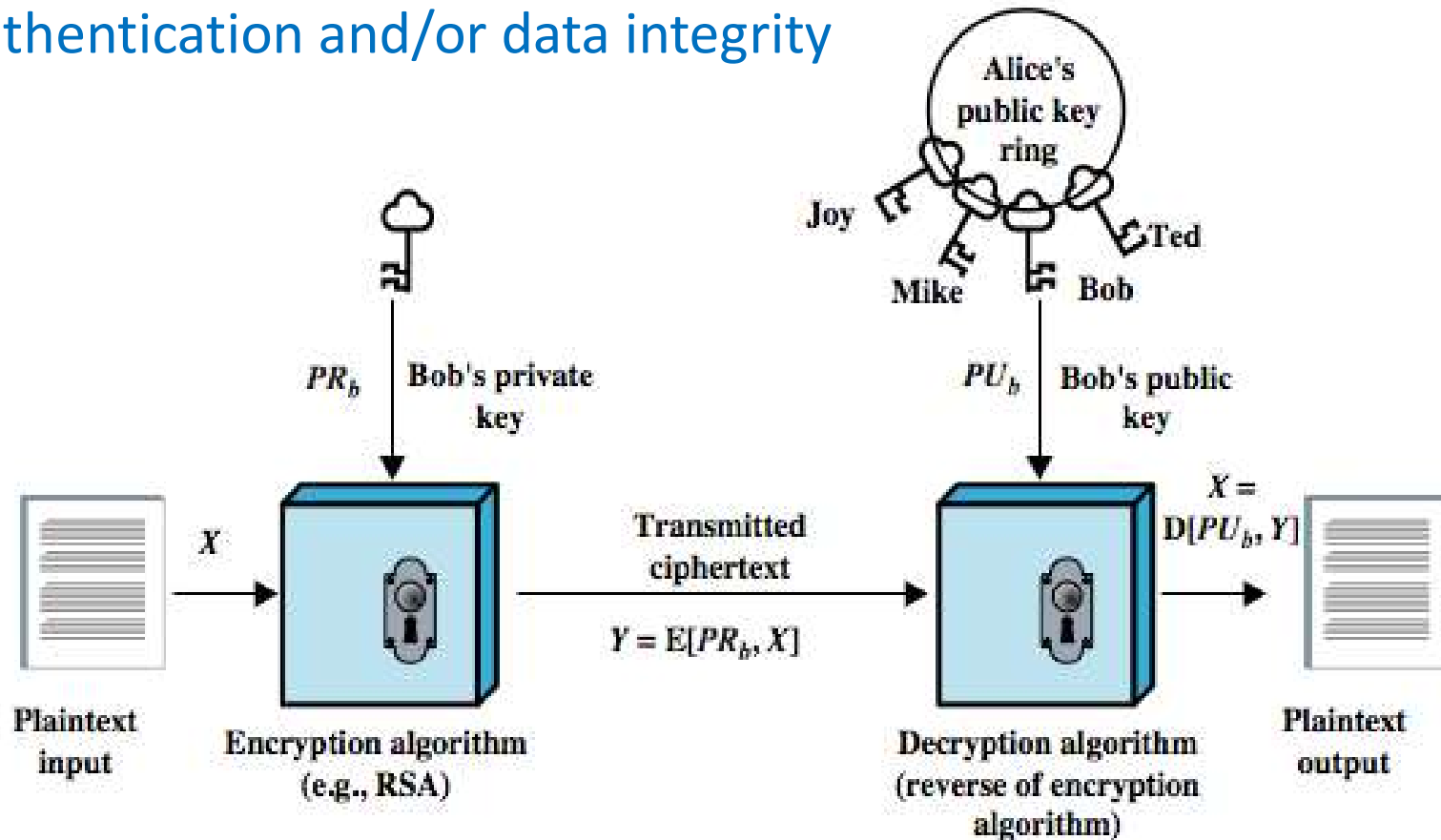
(b) Authentication

Key Distribution Center (KDC)
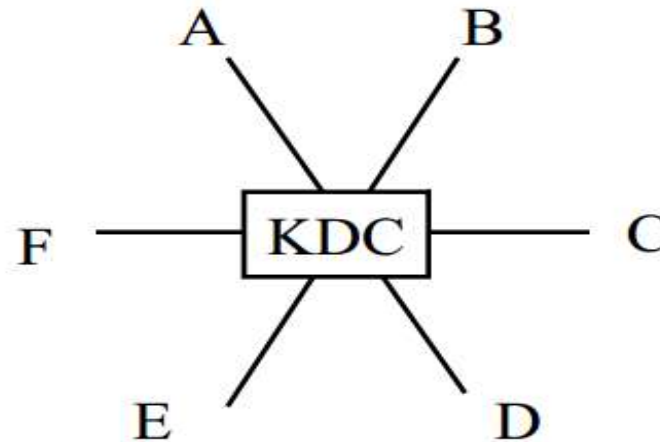
- Each node is configured with KDC's key.
- KDC has all the keys.
- A $\leftrightarrow$ B communication? KDC sends a key K_{AB} encrypted with A's key to A and B's key to B.

- 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

2. AS verifies user's access right in database, creates ticket-granting ticket and session key. Results are encrypted using key derived from user's password.

once per user logon session

1. User logs on to workstation and requests service on host.

request ticket-granting ticket

ticket + session key

request service-granting ticket

ticket + session key

once per type of service

Kerberos

Authentication Server (AS)

Ticket-granting Server (TGS)

4. TGS decrypts ticket and authenticator, verifies request, then creates ticket for requested server.

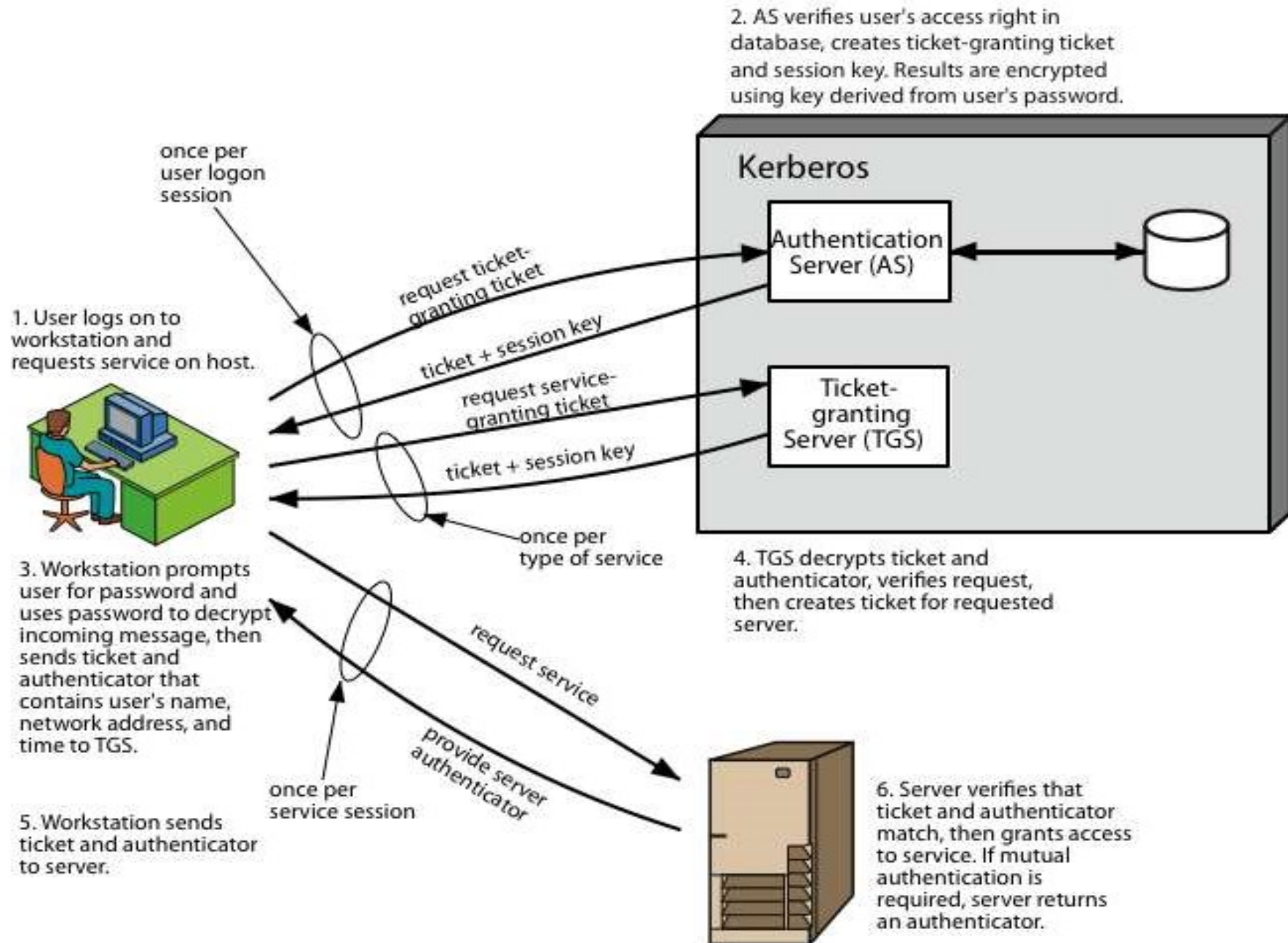3. Workstation prompts user for password and uses password to decrypt incoming message, then sends ticket and authenticator that contains user's name, network address, and time to TGS.

5. Workstation sends ticket and authenticator to server.

once per service session

request service

provide server authenticator

6. Server verifies that ticket and authenticator match, then grants access to service. If mutual authentication is required, server returns an authenticator.

Kerberos Exchanges

Client   **Authentication server (AS)**   **Ticket-granting server (TGS)**   **Service provider**

Client authentication →
$ID_c \parallel ID_{tgs} \parallel TS_1$

← Shared key and ticket
$E(K_c, [K_{c,tgs} \parallel ID_{tgs} \parallel TS_2 \parallel Lifetime_2 \parallel Ticket_{tgs}])$

$Ticket_{tgs}$, server ID, and client authentication →
$ID_v \parallel Ticket_{tgs} \parallel Authenticator_c$

← Shared key and ticket
$E(K_{c,tgs}, [K_{c,v} \parallel ID_v \parallel TS_4 \parallel Ticket_v])$

$Ticket_v$ and client authentication →
$Ticket_v \parallel Authenticator_c$

← Service granted
$E(K_{c,v}, [TS_5 + 1])$

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