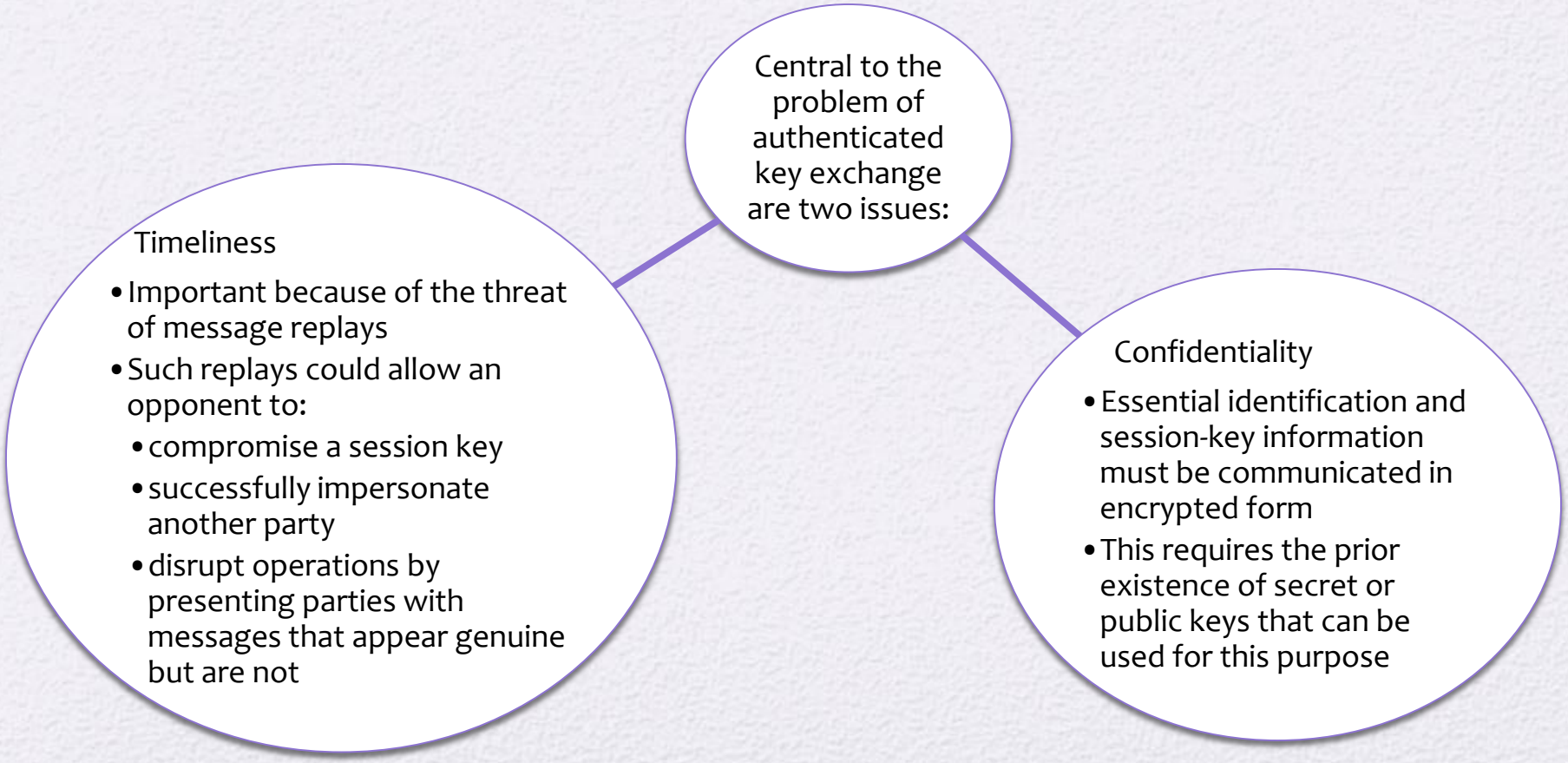
Mutual Authentication

- Protocols which enable communicating parties to satisfy themselves mutually about each other's identity and to exchange session keys



- To prevent masquerade and to prevent compromise of session keys, essential identification and session-key information must be communicated in encrypted form. This requires the prior existence of secret or public keys that can be used for this purpose.

Timeliness

- The second issue, timeliness, is important because of the threat of message replays. Such replays, at worst, could allow an opponent to compromise a session key or successfully impersonate another party.

Replay Attacks

1. The simplest replay attack is one in which the opponent simply copies a message and replays it later

2. An opponent can replay a timestamped message within the valid time window

3. An opponent can replay a timestamped message within the valid time window, but in addition, the opponent suppresses the original message; thus, the repetition cannot be detected

4. Another attack involves a backward replay without modification and is possible if symmetric encryption is used and the sender cannot easily recognize the difference between messages sent and messages received on the basis of content

Approaches to Coping With Replay Attacks

- Attach a sequence number to each message used in an authentication exchange
  - A new message is accepted only if its sequence number is in the proper order
  - Difficulty with this approach is that it requires each party to keep track of the last sequence number for each claimant it has dealt with
  - Generally not used for authentication and key exchange because of overhead
- Timestamps
  - Requires that clocks among the various participants be synchronized
  - Party A accepts a message as fresh only if the message contains a timestamp that, in A's judgment, is close enough to A's knowledge of current time
- Challenge/response
  - Party A, expecting a fresh message from B, first sends B a nonce (challenge) and requires that the subsequent message (response) received from B contain the correct nonce value

- Lamport Hash chain
- Therefore, any timestamp-based procedure must allow for a window of time sufficiently large to accommodate network delays yet sufficiently small to minimize the opportunity for attack

- The challenge-response approach is unsuitable for a connectionless type of application, because it requires the overhead of a handshake before any connectionless transmission, effectively negating the chief characteristic of a connectionless transaction.

One-Way Authentication

One application for which encryption is growing in popularity is electronic mail (e-mail)

- Header of the e-mail message must be in the clear so that the message can be handled by the store-and-forward e-mail protocol, such as SMTP or X.400
- The e-mail message should be encrypted such that the mail-handling system is not in possession of the decryption key

A second requirement is that of authentication

- The recipient wants some assurance that the message is from the alleged sender

Remote User-Authentication Using Symmetric Encryption

A two-level hierarchy of symmetric keys can be used to provide confidentiality for communication in a distributed environment

- Strategy involves the use of a trusted key distribution center (KDC)
- Each party shares a secret key, known as a master key, with the KDC
- KDC is responsible for generating keys to be used for a short time over a connection between two parties and for distributing those keys using the master keys to protect the distribution

- The use of a trusted key distribution center (KDC).
- Each party in the network shares a secret key, known as a master key, with the KDC. The KDC is responsible for generating keys to be used for a short time over a connection between two parties, known as session keys, and for distributing those keys using the master keys to protect the distribution.

Needham and Schroeder

1. A → KDC: $ID_A \| ID_B \| N_1$
2. KDC → A: $E(K_a, [K_s \| ID_B \| N_1 \| E(K_b, [K_s \| ID_A])])$
3. A → B: $E(K_b, [K_s \| ID_A])$
4. B → A: $E(K_s, N_2)$
5. A → B: $E(K_s, f(N_2))$

Secret keys K_a and K_b are shared between A and the KDC and B and the KDC, respectively. The purpose of the protocol is to distribute securely a session key K_s to A and B. A securely acquires a new session key in step 2. The message in step 3 can be decrypted, and hence understood, only by B. Step 4 reflects B's knowledge of K_s, and step 5 assures B of A's knowledge of K_s and assures B that this is

Modified protocol (Denning with timestamp)

1. $A \rightarrow KDC: \quad ID_A \| ID_B$
2. $KDC \rightarrow A: \quad E(K_a, [K_s \| ID_B \| T \| E(K_b, [K_s \| ID_A \| T])])$
3. $A \rightarrow B: \quad E(K_b, [K_s \| ID_A \| T])$
4. $B \rightarrow A: \quad E(K_s, N_1)$
5. $A \rightarrow B: \quad E(K_s, f(N_1))$

Improvement

1. $A \rightarrow B$: $ID_A \| N_a$
2. $B \rightarrow KDC$: $ID_B \| N_b \| E(K_b, [ID_A \| N_a \| T_b])$
3. $KDC \rightarrow A$: $E(K_a, [ID_B \| N_a \| K_s \| T_b]) \| E(K_b, [ID_A \| K_s \| T_b]) \| N_b$
4. $A \rightarrow B$: $E(K_b, [ID_A \| K_s \| T_b]) \| E(K_s, N_b)$

1. A initiates the authentication exchange by generating a nonce, N_a, and sending that plus its identifier to B in plaintext. This nonce will be returned to A in an encrypted message that includes the session key, assuring A of its timeliness.

1. A → B: $ID_A \| N_a$

B alerts the KDC that a session key is needed. Its message to the KDC includes its identifier and a nonce, N_b. This nonce will be returned to B in an encrypted message that includes the session key, assuring B of its timeliness.

2. B → KDC: $ID_B \| N_b \| E(K_b, [ID_A \| N_a \| T_b])$

The KDC passes on to A B's nonce and a block encrypted with the secret key that B shares with the KDC. The block serves as a "ticket" that can be used by A for subsequent authentications, as will be seen. The KDC also sends to A a block encrypted with the secret key shared by A and the KDC. This block verifies that B has received A's initial message (ID_B) and that this is a timely message and not a replay (N_a), and it provides A with a session key (K_s) and the time limit on its use (T_b).

3. KDC $\rightarrow$ A: $E(K_a, [ID_B \| N_a \| K_s \| T_b]) \| E(K_b, [ID_A \| K_s \| T_b]) \| N_b$

4. A transmits the ticket to B, together with the B's nonce, the latter encrypted with the session key. The ticket provides B with the secret key that is used to decrypt $E(K_s, N_b)$ to recover the nonce. The fact that B's nonce is encrypted with the session key authenticates that the message came from A and is not a replay.

4. $A \rightarrow B: \quad E(K_b, [ID_A \| K_s \| T_b]) \| E(K_s, N_b)$

Subsequent Authentication

1. $A \rightarrow B$: $E(K_b, [ID_A \| K_s \| T_b]) \| N'_a$
2. $B \rightarrow A$: $N'_b \| E(K_s, N'_a)$
3. $A \rightarrow B$: $E(K_s, N'_b)$

One Way Authentication (Email communication)

1. $A \rightarrow KDC: \quad ID_A \| ID_B \| N_1$
2. $KDC \rightarrow A: \quad E(K_a, [K_s \| ID_B \| N_1 \| E(K_b, [K_s \| ID_A])])$
3. $A \rightarrow B: \quad E(K_b, [K_s \| ID_A]) \| E(K_s, M)$

Suppress-Replay Attacks

- The Denning protocol requires reliance on clocks that are synchronized throughout the network
- A risk involved is based on the fact that the distributed clocks can become unsynchronized as a result of sabotage on or faults in the clocks or the synchronization mechanism
- The problem occurs when a sender's clock is ahead of the intended recipient's clock
  - An opponent can intercept a message from the sender and replay it later when the timestamp in the message becomes current at the recipient's site
  - Such attacks are referred to as *suppress-replay attacks*

Kerberos

- Authentication service developed as part of Project Athena at MIT
- A workstation cannot be trusted to identify its users correctly to network services
  - A user may gain access to a particular workstation and pretend to be another user operating from that workstation
  - A user may alter the network address of a workstation so that the requests sent from the altered workstation appear to come from the impersonated workstation
  - A user may eavesdrop on exchanges and use a replay attack to gain entrance to a server or to disrupt operations
- Kerberos provides a centralized authentication server whose function is to authenticate users to servers and servers to users
  - Relies exclusively on symmetric encryption, making no use of public-key encryption

Introduction

- Kerberos
- Developed at MIT
- Version 4/ Version 5 workswith TCP/IP networks
- A centralized network protocol that provides distributed authentication
- Trusted third-party authentication service

- Challenge
- A network environment where users at workstations need to access restricted distributed services (files, printers, etc)

- Services (symmetric key cryptography)
- Authentication
- Confidentiality
- Data integrity
- Authorization/Access control

Threats

- Unauthorized user gaining access to services/data
- Impersonation
- Pretend to be another user
- Address spoofing
- Eavesdropping
- Replay attack

Example

- If a set of users is provided with dedicated personal computers that have no network connections, then a user's resources and files can be protected by physically securing each personal computer.

- The time-sharing operating system must provide the security.
- The operating system can enforce access-control policies based on user identity and use the logon procedure to identify users.

- This distributed architecture consisting of dedicated user workstations (clients) and distributed or centralized servers.
- In this environment, three approaches to security can be envisioned.

Approach

- Rely on each individual client workstation to assure the identity of its user or users and rely on each server to enforce a security policy based on user identification (ID)

- Require that client systems authenticate themselves to servers, but trust the client system concerning the identity of its user.

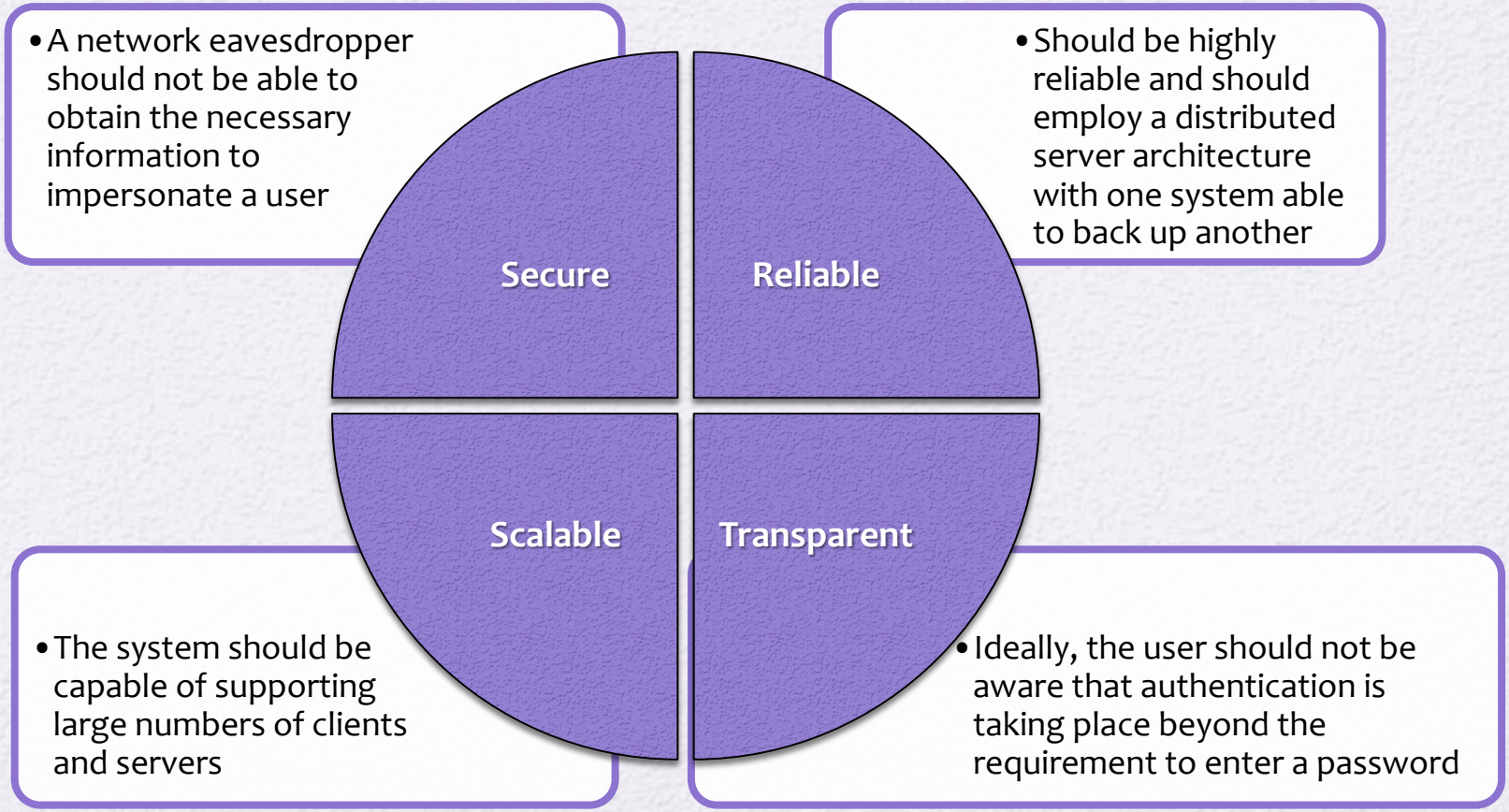
- Require the user to prove his or her identity for each service invoked. Also require that servers prove their identity to clients.
- This is followed in Kerberos.
- Kerberos assumes a distributed client/server architecture and employs one or more Kerberos servers to provide an authentication service.

Firewall vs. Kerberos?

- Firewalls make a risky assumption: that attackers are coming from the outside. In reality, attacks frequently come from within.
- Kerberos assumes that network connections (rather than servers and work stations) are the weak link in network security.

Kerberos Requirements

- The first published report on Kerberos listed the following requirements:



- Client (C)
  - Access services on servers throughout network
  - Logs in to workstation (userid/password)
- Kerberos
  - Authentication Server (AS)
    - Responsible for authenticating users (all passwords)
    - Shares a unique key with each server
    - Issues a ticket-granting ticket (TGT)
  - Ticking-granting Server (TGS)
    - Verifies users authenticated by AS
    - Grants users service-granting tickets for particular services
- Application Server (V)
  - Authenticates user using service-granting ticket
  - Provides a service (files, printers, emails, etc)

Kerberos Version 4

- Makes use of DES to provide the authentication service
- Authentication server (AS)
  - Knows the passwords of all users and stores these in a centralized database
  - Shares a unique secret key with each server
- Ticket
  - Created once the AS accepts the user as authentic; contains the user's ID and network address and the server's ID
  - Encrypted using the secret key shared by the AS and the server
- Ticket-granting server (TGS)
  - Issues tickets to users who have been authenticated to AS
  - Each time the user requires access to a new service the client applies to the TGS using the ticket to authenticate itself
  - The TGS then grants a ticket for the particular service
  - The client saves each service-granting ticket and uses it to authenticate its user to a server each time a particular service is requested

- It was designed for an environment where a user logs into a workstation by providing a user name and a password.
- These are used by the workstation to obtain information from the KDC that can be used by any process to access remote resources on behalf of the user.

Kerberos library

- telnet: a protocol for acting as a terminal on a remote system
- BSD UNIX that support remote login (rlogin), remote file copying (rcp), and remote command execution (rsh)
- NFS a utility that permits files on a remote node on the network to be accessed as though they were local files

Competition

SSL	Kerberos
Uses public key encryption	Uses private key encryption
Is certificate based (asynchronous)	Relies on a trusted third party (synchronous)
Ideal for the WWW	Ideal for networked environments
Key revocation requires Revocation Server to keep track of bad certificates	Key revocation can be accomplished by disabling a user at the Authentication Server
Certificates sit on a users hard drive (even if they are encrypted) where they are subject to being cracked.	Passwords reside in users' minds where they are usually not subject to secret attack.
Uses patented material, so the service is not free. Netscape has a profit motive in wide acceptance of the standard.	Kerberos has always been open source and freely available.

The Version 4 Authentication Dialogue

The lifetime associated with the ticket-granting ticket creates a problem:

- If the lifetime is very short (e.g., minutes), the user will be repeatedly asked for a password
- If the lifetime is long (e.g., hours), then an opponent has a greater opportunity for replay

A network service (the TGS or an application service) must be able to prove that the person using a ticket is the same person to whom that ticket was issued

Servers need to authenticate themselves to users

Authentication Server

- use an authentication server (AS) that knows the passwords of all users and stores these in a centralized database.
- In addition, the AS shares a unique secret key with each server. These keys have been distributed physically or in some other secure manner.

- Private Key: Each party uses the same secret key to encode and decode messages.
- Uses a trusted third party which can vouch for the identity of both parties in a transaction. Security of third party is imperative.

(1) C → AS: $ID_C \| P_C \| ID_V$

(2) AS → C: $Ticket$

(3) C → V: $ID_C \| Ticket$

$$Ticket = E(K_v, [ID_C \| AD_C \| ID_V])$$

where

C = client

AS = authentication server

V = server

ID_C = identifier of user on C

ID_V = identifier of V

P_C = password of user on C

AD_C = network address of C

K_v = secret encryption key shared by AS and V

Attack

- An opponent could capture the ticket transmitted in message (2), then use the name *ID* and transmit a message of form (3) from another workstation.
- The server would receive a valid ticket that matches the user ID and grant access to the user on that other workstation. To prevent this attack, the AS includes in the ticket the network address from which the original request came.

- we would like to minimize the number of times that a user has to enter a password. Suppose each ticket can be used only once. If user C logs on to a workstation in the morning and wishes to check his or her mail at a mail server, C must supply a password to get a ticket for the mail server.
- If C wishes to check the mail several times during the day, each attempt requires reentering the password. We can improve matters by saying that tickets are reusable.

- print server, a mail server, a file server, and so on
- **ticket-granting server**

Once per user logon session:

(1) $C \rightarrow AS$: $ID_C \| ID_{tgs}$

(2) $AS \rightarrow C$: $E(K_c, Ticket_{tgs})$

Once per type of service:

(3) $C \rightarrow TGS$: $ID_C \| ID_V \| Ticket_{tgs}$

(4) $TGS \rightarrow C$: $Ticket_v$

Once per service session:

(5) $C \rightarrow V$: $ID_C \| Ticket_v$

$$Ticket_{tgs} = E(K_{tgs}, [ID_C \| AD_C \| ID_{tgs} \| TS_1 \| Lifetime_1])$$

$$Ticket_v = E(K_v, [ID_C \| AD_C \| ID_v \| TS_2 \| Lifetime_2])$$

- The service-granting ticket has the same structure as the ticket-granting ticket.

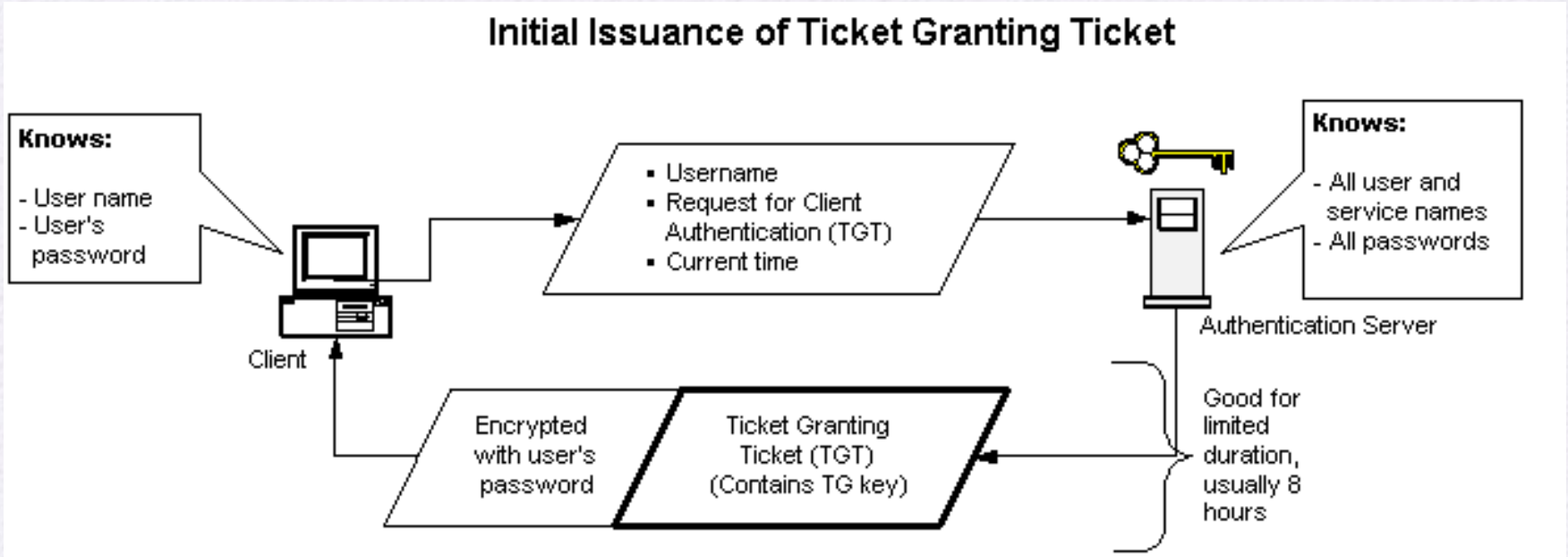
The Version 4 Authentication Dialogue

The lifetime associated with the ticket-granting ticket creates a problem:

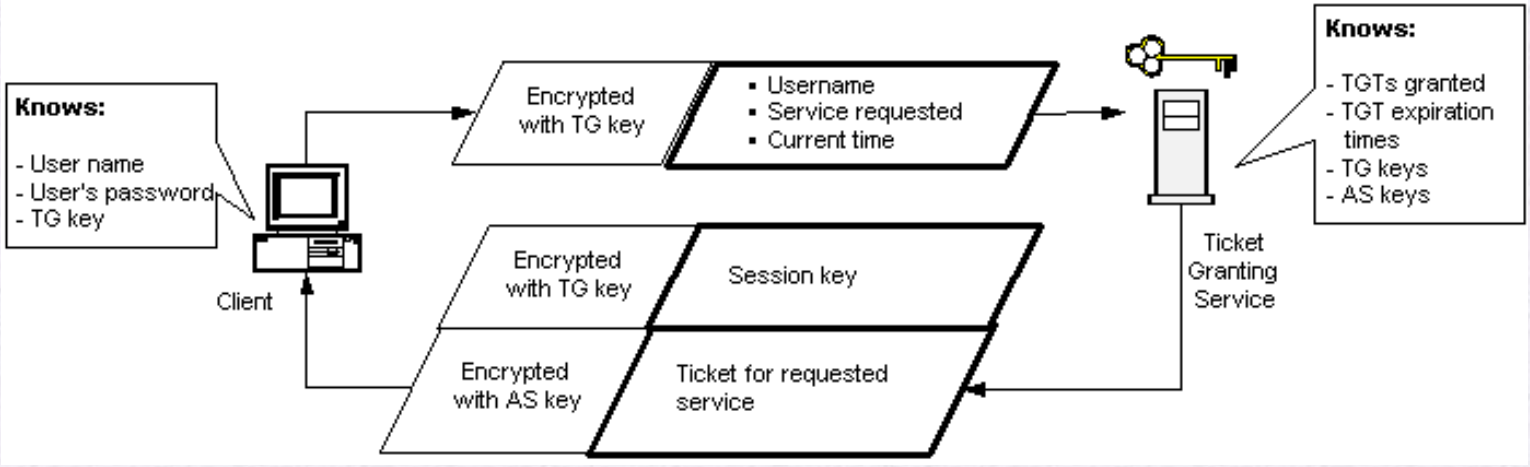
- If the lifetime is very short (e.g., minutes), the user will be repeatedly asked for a password
- If the lifetime is long (e.g., hours), then an opponent has a greater opportunity for replay

A network service (the TGS or an application service) must be able to prove that the person using a ticket is the same person to whom that ticket was issued

Servers need to authenticate themselves to users


Initial Issuance of Ticket Granting Ticket
Knows:
- User name
- User's password
Client
▪ Username
▪ Request for Client Authentication (TGT)
▪ Current time
Knows:
- All user and service names
- All passwords
Authentication Server
Encrypted with user's password
Ticket Granting Ticket (TGT) (Contains TG key)
Good for limited duration, usually 8 hours

Susequent Requests for Services from the Ticket Granting Service



Communication between the Client and the Application Server

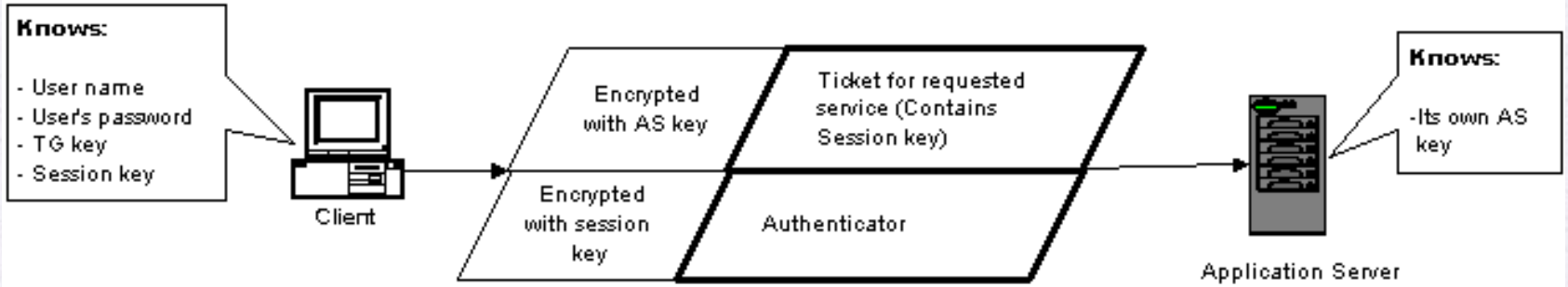


Table 15.1 (page 464 in textbook)

Summary of Kerberos Version 4 Message Exchanges

(1) C → AS $ID_c \| ID_{tgs} \| TS_1$

(2) AS → C $E(K_c, [K_{c,tgs} \| ID_{tgs} \| TS_2 \| Lifetime_2 \| Ticket_{tgs}])$

$Ticket_{tgs} = E(K_{tgs}, [K_{c,tgs} \| ID_C \| AD_C \| ID_{tgs} \| TS_2 \| Lifetime_2])$

(a) Authentication Service Exchange to obtain ticket-granting ticket

(3) C → TGS $ID_v \| Ticket_{tgs} \| Authenticator_c$

(4) TGS → C $E(K_{c,tgs}, [K_{c,v} \| ID_v \| TS_4 \| Ticket_v])$

$Ticket_{tgs} = E(K_{tgs}, [K_{c,tgs} \| ID_C \| AD_C \| ID_{tgs} \| TS_2 \| Lifetime_2])$

$Ticket_v = E(K_v, [K_{c,v} \| ID_C \| AD_C \| ID_v \| TS_4 \| Lifetime_4])$

$Authenticator_c = E(K_{c,tgs}, [ID_C \| AD_C \| TS_3])$

(b) Ticket-Granting Service Exchange to obtain service-granting ticket

(5) C → V $Ticket_v \| Authenticator_c$

(6) V → C $E(K_{c,v}, [TS_5 + 1])$ (for mutual authentication)

$Ticket_v = E(K_v, [K_{c,v} \| ID_C \| AD_C \| ID_v \| TS_4 \| Lifetime_4])$

$Authenticator_c = E(K_{c,v}, [ID_C \| AD_C \| TS_5])$

(c) Client/Server Authentication Exchange to obtain service

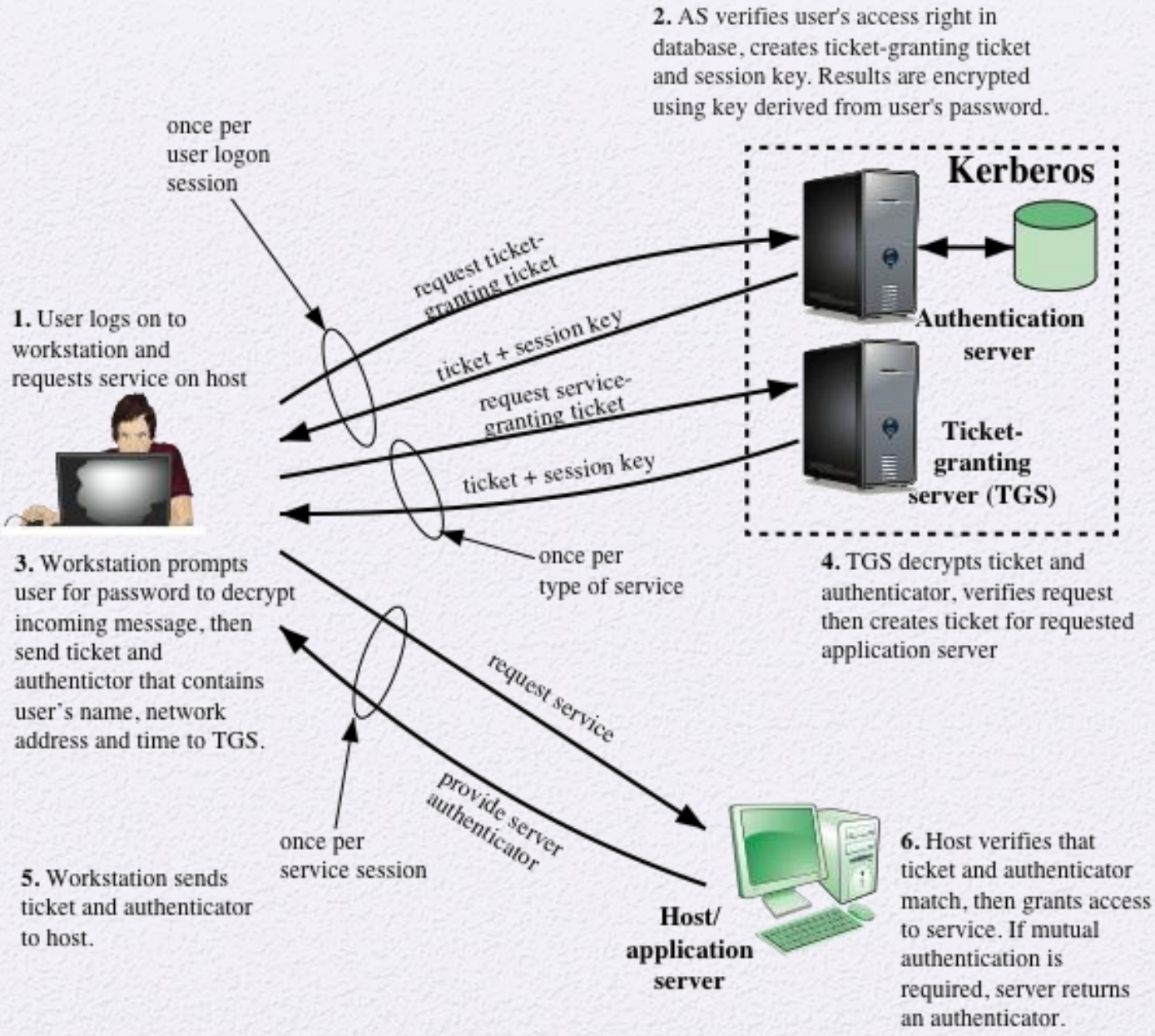


Figure 15.1 Overview of Kerberos

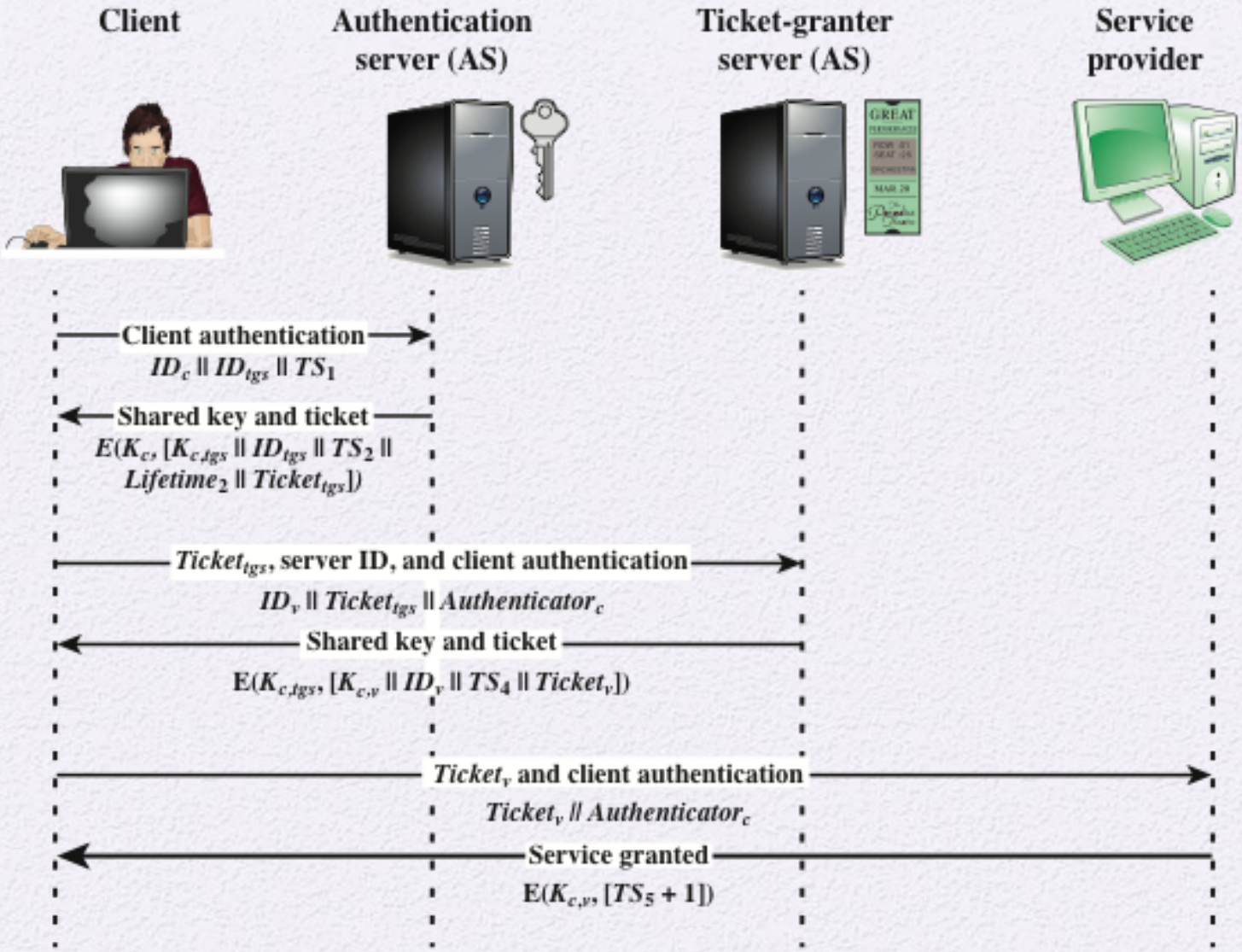


Figure 15.2 Kerberos Exchanges

Table 15.2 Rationale for the Elements of the Kerberos Version 4 Protocol
(page 1 of 3)

Message (1)	Client requests ticket-granting ticket.
ID_C	Tells AS identity of user from this client.
ID_{tgs}	Tells AS that user requests access to TGS.
TS_1	Allows AS to verify that client's clock is synchronized with that of AS.
Message (2)	AS returns ticket-granting ticket.
K_c	Encryption is based on user's password, enabling AS and client to verify password, and protecting contents of message (2).
$K_{c,tgs}$	Copy of session key accessible to client created by AS to permit secure exchange between client and TGS without requiring them to share a permanent key.
ID_{tgs}	Confirms that this ticket is for the TGS.
TS_2	Informs client of time this ticket was issued.
$Lifetime_2$	Informs client of the lifetime of this ticket.
$Ticket_{tgs}$	Ticket to be used by client to access TGS.

(This table can be found on pages 467 – 468 in the textbook)

Message (3)	Client requests service-granting ticket.
ID_V	Tells TGS that user requests access to server V.
$Ticket_{tgs}$	Assures TGS that this user has been authenticated by AS.
$Authenticator_c$	Generated by client to validate ticket .
Message (4)	TGS returns service-granting ticket.
$K_{c,tgs}$	Key shared only by C and TGS protects contents of message (4).
$K_{c,v}$	Copy of session key accessible to client created by TGS to permit secure exchange between client and server without requiring them to share a permanent key.
ID_V	Confirms that this ticket is for server V.
TS_4	Informs client of time this ticket was issued.
$Ticket_V$	Ticket to be used by client to access server V.
$Ticket_{tgs}$	Reusable so that user does not have to reenter password.
K_{tgs}	Ticket is encrypted with key known only to AS and TGS, to prevent Tampering.
$K_{c,tgs}$	Copy of session key accessible to TGS used to decrypt authenticator, thereby authenticating ticket.
ID_C	Indicates the rightful owner of this ticket.
AD_C	Prevents use of ticket from workstation other than one that initially requested the ticket.
ID_{tgs}	Assures server that it has decrypted ticket properly.
TS_2	Informs TGS of time this ticket was issued.
$Lifetime_2$	Prevents replay after ticket has expired.
$Authenticator_c$	Assures TGS that the ticket presenter is the same as the client for whom the ticket was issued has very short lifetime to prevent replay.
$K_{c,tgs}$	Authenticator is encrypted with key known only to client and TGS, to prevent tampering.
ID_C	Must match ID in ticket to authenticate ticket.
AD_C	Must match address in ticket to authenticate ticket.
TS_3	Informs TGS of time this authenticator was generated.

Message (5)	Client requests service.
$Ticket_V$	Assures server that this user has been authenticated by AS.
$Authenticator_C$	Generated by client to validate ticket.
Message (6)	Optional authentication of server to client.
$K_{C,V}$	Assures C that this message is from V.
$TS_5 + 1$	Assures C that this is not a replay of an old reply.
$Ticket_V$	Reusable so that client does not need to request a new ticket from TGS for each access to the same server.
K_V	Ticket is encrypted with key known only to TGS and server, to prevent Tampering.
$K_{C,V}$	Copy of session key accessible to client; used to decrypt authenticator, thereby authenticating ticket.
ID_C	Indicates the rightful owner of this ticket.
AD_C	Prevents use of ticket from workstation other than one that initially requested the ticket.
ID_V	Assures server that it has decrypted ticket properly.
TS_4	Informs server of time this ticket was issued.
$Lifetime_4$	Prevents replay after ticket has expired.
$Authenticator_C$	Assures server that the ticket presenter is the same as the client for whom the ticket was issued; has very short lifetime to prevent replay.
$K_{C,V}$	Authenticator is encrypted with key known only to client and server, to prevent tampering.
ID_C	Must match ID in ticket to authenticate ticket.
AD_C	Must match address in ticket to authenticate ticket.
TS_5	Informs server of time this authenticator was generated.

Weaknesses and Solutions

If TGT stolen, can be used to access network services.	Only a problem until ticket expires in a few hours.
Subject to dictionary attack.	Timestamps require hacker to guess in 5 minutes.
Very bad if Authentication Server compromised.	Physical protection for the server.

Kerberos Realm

- Networks of servers and clients under control of Authentication Server (administrative domain)
- Kerberos environment
  - Kerberos server
  - Clients
    - UID and hashed passwords registered
  - Application servers
    - Secret key registered
- Inter-realm authentication
  - Secret key shared between realms

Kerberos Realms and Multiple Kerberi

- A full-service Kerberos environment consisting of a Kerberos server, a number of clients, and a number of application servers requires that:
  - The Kerberos server must have the user ID and hashed passwords of all participating users in its database; all users are registered with the Kerberos server
  - The Kerberos server must share a secret key with each server; all servers are registered with the Kerberos server
  - The Kerberos server in each interoperating realm shares a secret key with the server in the other realm; the two Kerberos servers are registered with each other

Kerberos Realm

- A set of managed nodes that share the same Kerberos database
- The database resides on the Kerberos master computer system, which should be kept in a physically secure room
- A read-only copy of the Kerberos database might also reside on other Kerberos computer systems
- All changes to the database must be made on the master computer system
- Changing or accessing the contents of a Kerberos database requires the Kerberos master password