RAMAIAH
Institute of Technology

M.S. Ramaiah Institute of Technology
(Autonomous Institute, Affiliated to VTU)
Department of Computer Science and Engineering

Course Name: Cryptography and Network Security

Course Code – CSE555

Credits - 3:0:0

Term: Oct 2024 – Jan 2025

UNIT -5

Prepared by: Dr. Sangeetha. V
Associate Professor

RAMAIAH Institute of Technology

Textbooks

1. Behrouz A. Forouzan, Debdeep Mukhopadhyay: Cryptography and Network Security, Tata McGraw-Hill, 3rd Edition, 2015
2. William Stallings, Cryptography and Network Security, Pearson Education, 7th Edition, 2018

Reference Book:

1. Bernard Menezes: Cryptography, Network Security and Cyber Laws , Cengage Learning, First edition, 2018.
1. AtulKahate: Cryptography and Network Security", 4th Edition, Tata McGraw Hill, 2019.
2. William Stallings: Network Security Essentials: Applications and Standards by, 6th edition, Pearson Education, 2018.

OUTLINE - (Text 2)

Kerberos (Chapter 15.3)

Transport Level security: Web Security Considerations, Secure Sockets Layer, Transport level Security, HTTPS.

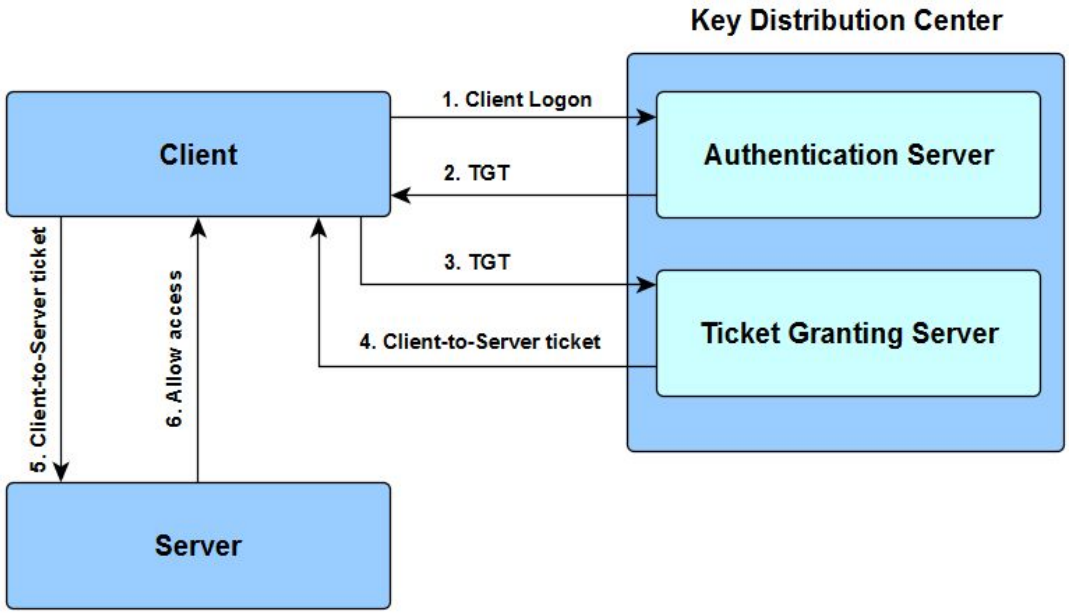
Electronic Mail security: S/MIME, Pretty Good Privacy

Kerberos

- Motivation
- Kerberos Version 4
- Kerberos Version 5

Kerberos

- Kerberos is an authentication service designed for use in a distributed environment.
- Kerberos provides a trusted third-party authentication service that enables clients and servers to establish authenticated communication.



Accessing the college Library

- **You (the student):** Represent the **client** trying to access the library services.

- **Library Administrator Desk:** Acts as the **Authentication Server (AS)** to verify your identity.
- **Permission Slip:** Acts as the **Ticket Granting Ticket (TGT)** you get from the administrator.
- **Section Supervisors:** Represent the **Ticket Granting Server (TGS)** for each specific section of the library (e.g., book lending, computer lab).

- **Specific Service (e.g., book shelf):** Represents the **Service Server (SS)** that grants you access to a resource.

Kerberos

- Kerberos provides a centralized authentication server whose function is to authenticate users to servers and servers to users.
- Kerberos relies exclusively on symmetric encryption.
- Two versions of Kerberos are in common use.
  - Version 4 implementations still exist.
  - Version 5 corrects some of the security deficiencies of version 4 and has been issued as a proposed Internet Standard (RFC 4120).

Kerberos

Motivation for the Kerberos approach

- 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.
- When these users instead are served by a centralized timesharing system, 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.
- Today, neither of these scenarios is typical.

Kerberos

Motivation for the Kerberos approach

Kerberos assumes a distributed client/server architecture and employs one or more Kerberos servers to provide an authentication service.

Kerberos

Motivation for the Kerberos approach

Kerberos listed the following requirements.

- **Secure**: Kerberos should be strong enough that a potential opponent does not find it to be the weak link.
- **Reliable:** Kerberos should be highly reliable and should employ a distributed server architecture with one system able to back up another.
- **Transparent**: Ideally, the user should not be aware that authentication is taking place beyond the requirement to enter a password.
- **Scalable:** The system should be capable of supporting large numbers of clients and servers. This suggests a modular, distributed architecture.

Kerberos

Kerberos version 4

- Version 4 of Kerberos makes use of DES to provide the authentication service.
- Therefore, we adopt a strategy used by Bill Bryant of Project Athena and build up to the full protocol by looking first at several hypothetical dialogues.
- Each successive dialogue adds additional complexity to counter security vulnerabilities revealed in the preceding dialogue.

Kerberos

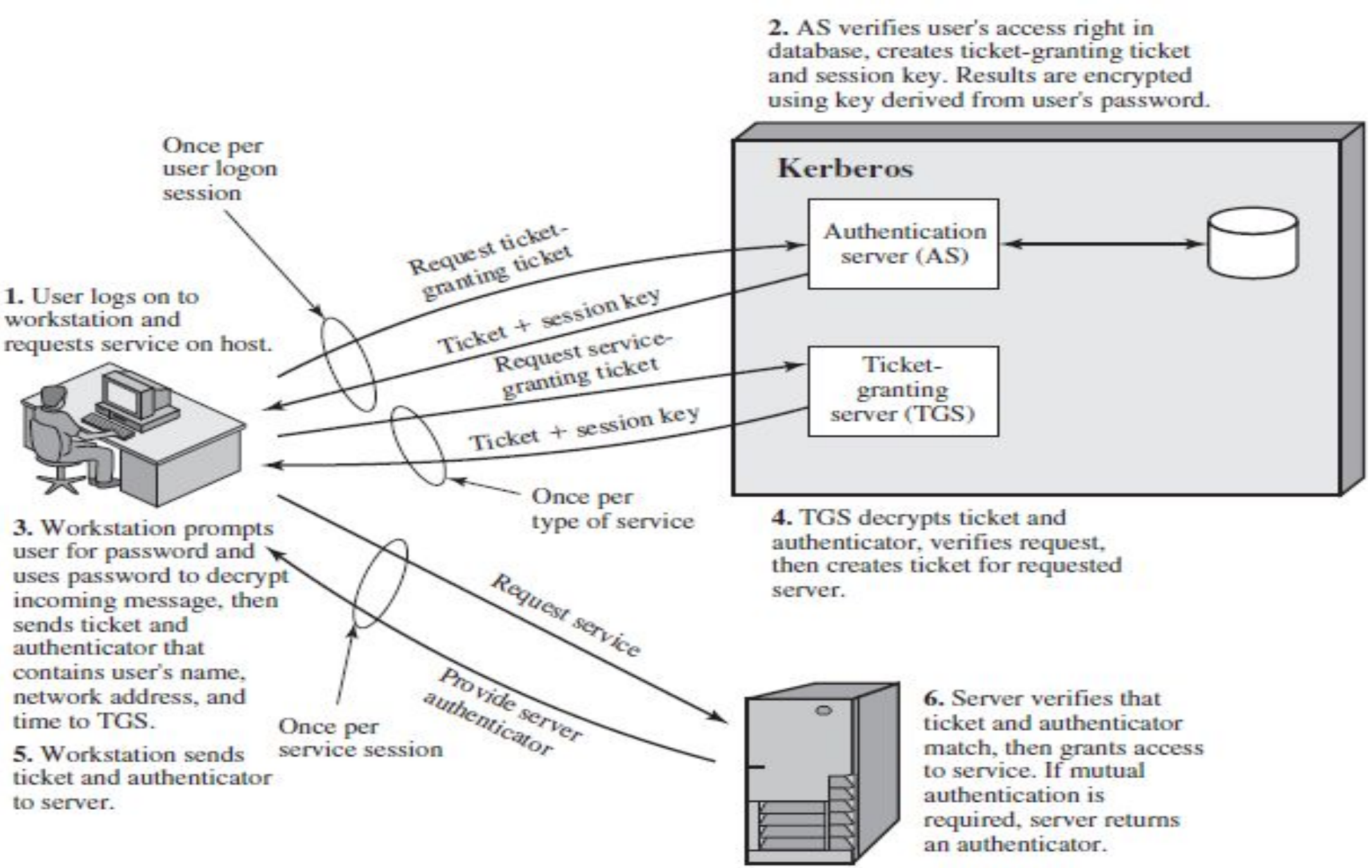
Kerberos version 4 - A SIMPLE AUTHENTICATION DIALOGUE

- In an unprotected network environment, any client can apply to any server for service.
- The obvious security risk is that of impersonation. An opponent can pretend to be another client and obtain unauthorized privileges on server machines.
- To counter this threat, servers must be able to confirm the identities of clients who request service.
- Each server can be required to undertake this task for each client/server interaction, but in an open environment, this places a substantial burden on each server.

Kerberos

Kerberos version 4 - A SIMPLE AUTHENTICATION DIALOGUE

- An alternative is to 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.



Overview of Kerberos

Kerberos

Step 1 - A user logs in to their workstation or client application and request service on host.

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

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

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

Kerberos

Step 5 - Workstation send ticket and authenticator to server

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

Kerberos

Kerberos version 4 - A SIMPLE AUTHENTICATION DIALOGUE

Consider the following hypothetical dialogue:

$$\textbf{(1)}\ \mathrm{C} \rightarrow \mathrm{AS}: \quad ID_C \| P_C \| ID_V$$

$$\textbf{(2)}\ \mathrm{AS} \rightarrow \mathrm{C}: \quad Ticket$$

$$\textbf{(3)}\ \mathrm{C} \rightarrow \mathrm{V}: \quad ID_C \| Ticket$$

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

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

Kerberos

Kerberos version 4 - A SIMPLE AUTHENTICATION DIALOGUE

User logs on to a workstation and requests access to server V.

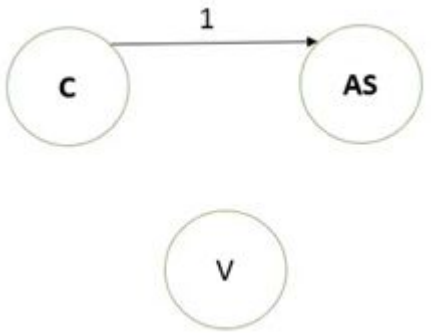
1. The client module C in the user's workstation requests the user's password and then sends a message to the AS that includes the user's ID, the server's ID, and the user's password.

$$\textbf{(1)}\ \mathrm{C} \rightarrow \mathrm{AS}: \quad ID_C \| P_C \| ID_V$$

$$\textbf{(2)}\ \mathrm{AS} \rightarrow \mathrm{C}: \quad Ticket$$

$$\textbf{(3)}\ \mathrm{C} \rightarrow \mathrm{V}: \quad ID_C \| Ticket$$

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



RAMAIAH Institute of Technology

Kerberos

Kerberos version 4 - A SIMPLE AUTHENTICATION DIALOGUE

2. The AS checks its database to see if the user has supplied the proper password for this user ID and whether this user is permitted access to server V.

If both tests are passed, the AS accepts the user as authentic and must now convince the server that this user is authentic.

To do so, the AS creates **a ticket** that contains the user's ID and network address and the server's ID. This ticket is encrypted using the secret key shared by the AS and this server. This ticket is then sent back to C.

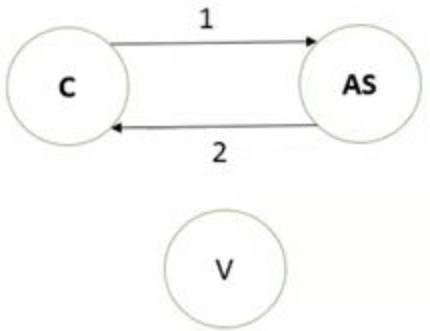
Because the ticket is encrypted, it cannot be altered by C or by an opponent.

(1) C → AS: $ID_C \| P_C \| ID_V$

(2) AS → C: $Ticket$

(3) C → V: $ID_C \| Ticket$

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



RAMAIAH
Institute of Technology

Kerberos

Kerberos version 4 - A SIMPLE AUTHENTICATION DIALOGUE

3. With this ticket, C can now apply to V for service.

C sends a message to V containing C's ID and the ticket.

V decrypts the ticket and verifies that the user ID in the ticket is the same as the unencrypted user ID in the message.

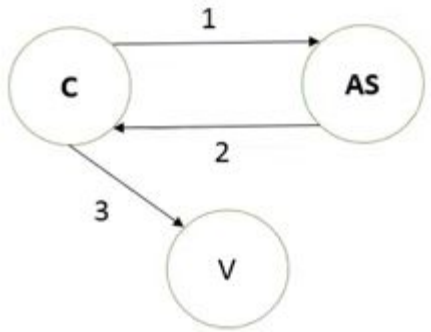
If these two match, the server considers the user authenticated and grants the requested service.

(1) $C \rightarrow AS: \quad ID_C \| P_C \| ID_V$

(2) $AS \rightarrow C: \quad Ticket$

(3) $C \rightarrow V: \quad ID_C \| Ticket$

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



Kerberos

Kerberos version 4 - A SIMPLE AUTHENTICATION DIALOGUE

Problem 1:

- 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.

Kerberos

Kerberos version 4 - A SIMPLE AUTHENTICATION DIALOGUE

Problem 2:

- The second problem is plaintext transmission of the password
- An eavesdropper could capture the password and use any service accessible to the victim.

(1) $C \rightarrow AS: \quad ID_C \| P_C \| ID_V$

(2) $AS \rightarrow C: \quad Ticket$

(3) $C \rightarrow V: \quad ID_C \| Ticket$

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

Kerberos

Kerberos version 4 - A MORE SIMPLE AUTHENTICATION DIALOGUE

- We introduce a scheme for avoiding plaintext passwords and a new server, known as the ticket-granting server (TGS).
- TGS issues tickets to users who have been authenticated to AS.
- The new (but still hypothetical) scenario is as follows.

RAMAIAH
Institute of Technology

Kerberos

Kerberos version 4 - A MORE SIMPLE AUTHENTICATION DIALOGUE

Once per user logon session:

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

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

Once per type of service:

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

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

Once per service session:

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

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

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

Kerberos

Kerberos version 4 - A MORE SIMPLE AUTHENTICATION DIALOGUE

Once per user logon session:

(1) C → AS: $ID_C \| ID_{tgs}$

(2) AS → C: $E(K_c, Ticket_{tgs})$

Once per type of service:

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

(4) TGS → C: $Ticket_v$

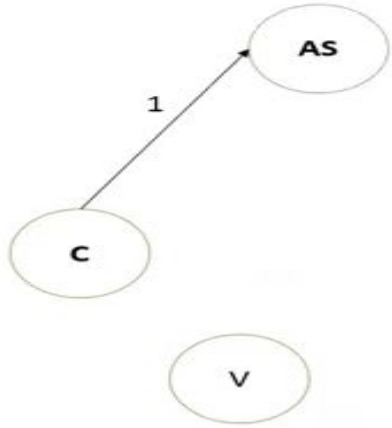
Once per service session:

(5) C → V: $ID_C \| Ticket_v$

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

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

1. The client requests a ticket-granting ticket on behalf of the user by sending its user's ID to the AS, together with the TGS ID, indicating a request to use the TGS service.



Kerberos

Kerberos version 4 - A MORE SIMPLE AUTHENTICATION DIALOGUE

Once per user logon session:

(1) C → AS: $ID_C \| ID_{tgs}$

(2) AS → C: $E(K_c, Ticket_{tgs})$

Once per type of service:

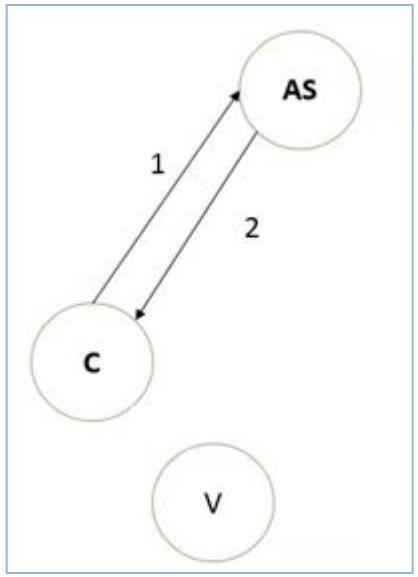
(3) C → TGS: $ID_C \| ID_V \| Ticket_{tgs}$

(4) TGS → C: $Ticket_v$

Once per service session:

(5) C → V: $ID_C \| Ticket_v$

$$Ticket_{tgs} = E(K_{tgs}, [ID_C \| AD_C \| ID_{tgs} \| TS_1 \| Lifetime_1])$$
$$Ticket_v = E(K_v, [ID_C \| AD_C \| ID_v \| TS_2 \| Lifetime_2])$$



2. The AS responds with a ticket that is encrypted with a key that is derived from the user's password (K_c), which is already stored at the AS.

When this response arrives at the client, the client prompts the user for his or her password, generates the key, and attempts to decrypt the incoming message.

If the correct password is supplied, the ticket is successfully recovered.

Kerberos

Kerberos version 4 - A MORE SIMPLE AUTHENTICATION DIALOGUE

Once per user logon session:

(1) C → AS: $ID_C \| ID_{tgs}$

(2) AS → C: $E(K_c, Ticket_{tgs})$

Once per type of service:

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

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

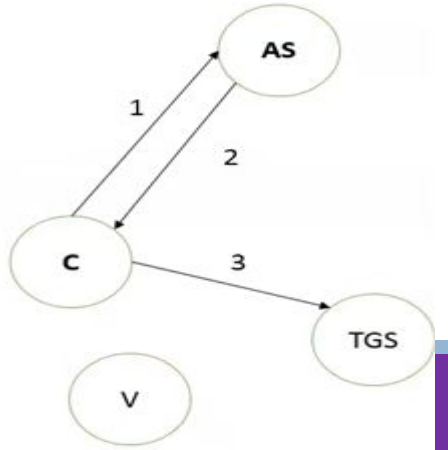
Once per service session:

(5) C → V: $ID_C \| Ticket_v$

$$Ticket_{tgs} = E(K_{tgs}, [ID_C \| AD_C \| ID_{tgs} \| TS_1 \| Lifetime_1])$$
$$Ticket_v = E(K_v, [ID_C \| AD_C \| ID_v \| TS_2 \| Lifetime_2])$$

3. The client requests a service-granting ticket on behalf of the user.

For this purpose, the client transmits a message to the TGS containing the user's ID, the ID of the desired service, and the ticket-granting ticket.



Kerberos

Kerberos version 4 - A MORE SIMPLE AUTHENTICATION DIALOGUE

Once per user logon session:

(1) C → AS: $ID_C \| ID_{tgs}$

(2) AS → C: $E(K_c, Ticket_{tgs})$

Once per type of service:

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

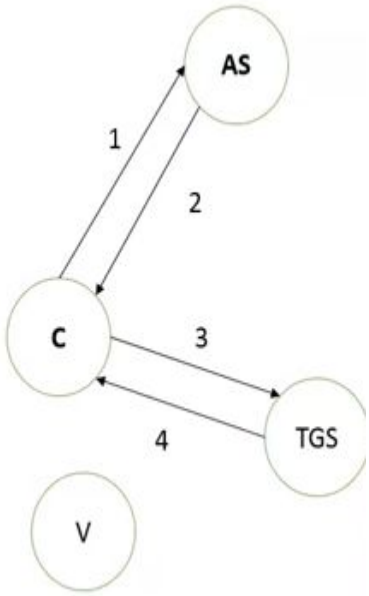
(4) TGS → C: $Ticket_v$

Once per service session:

(5) C → V: $ID_C \| Ticket_v$

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

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



4. The TGS decrypts the incoming ticket using a key shared only by the AS and the TGS (K_{tgs}) and verifies the success of the decryption by the presence of its ID.

It checks to make sure that the lifetime has not expired.

Then it compares the user ID and network address with the incoming information to authenticate the user.

If the user is permitted access to the server V, the TGS issues a ticket to grant access to the requested service.

Kerberos

Kerberos version 4 - A MORE SIMPLE AUTHENTICATION DIALOGUE

Once per user logon session:

(1) C → AS: $ID_C \| ID_{tgs}$

(2) AS → C: $E(K_c, Ticket_{tgs})$

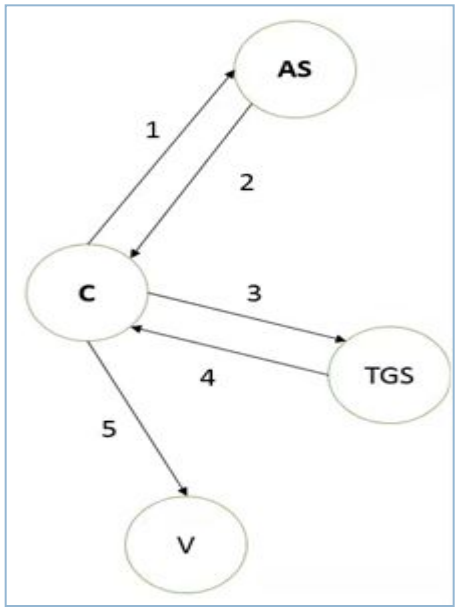
Once per type of service:

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

(4) TGS → C: $Ticket_v$

Once per service session:

(5) C → V: $ID_C \| Ticket_v$



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

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

5. The client requests access to a service on behalf of the user.

For this purpose, the client transmits a message to the server containing the user's ID and the service-granting ticket.

The server authenticates by using the contents of the ticket.

Kerberos

Kerberos version 4 - A MORE SIMPLE AUTHENTICATION DIALOGUE

Problem 1:

Lifetime associated with the ticket-granting ticket.

If this lifetime is very short (e.g., minutes), then 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.

Kerberos

Kerberos version 4 - A MORE SIMPLE AUTHENTICATION DIALOGUE

Problem 2:

The second problem is that there may be a requirement for servers to authenticate themselves to users.

Without such authentication, an opponent could sabotage the configuration so that messages to a server were directed to another location.

The false server would then be in a position to act as a real server and capture any information from the user and deny the true service to the user.

RAMAIAH
Institute of Technology

Kerberos - THE VERSION 4 AUTHENTICATION DIALOGUE

Summary of Kerberos Version 4 Message Exchanges

(1) $C \rightarrow AS \quad ID_c \| ID_{tgs} \| TS_1$

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

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

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

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

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

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

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

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

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

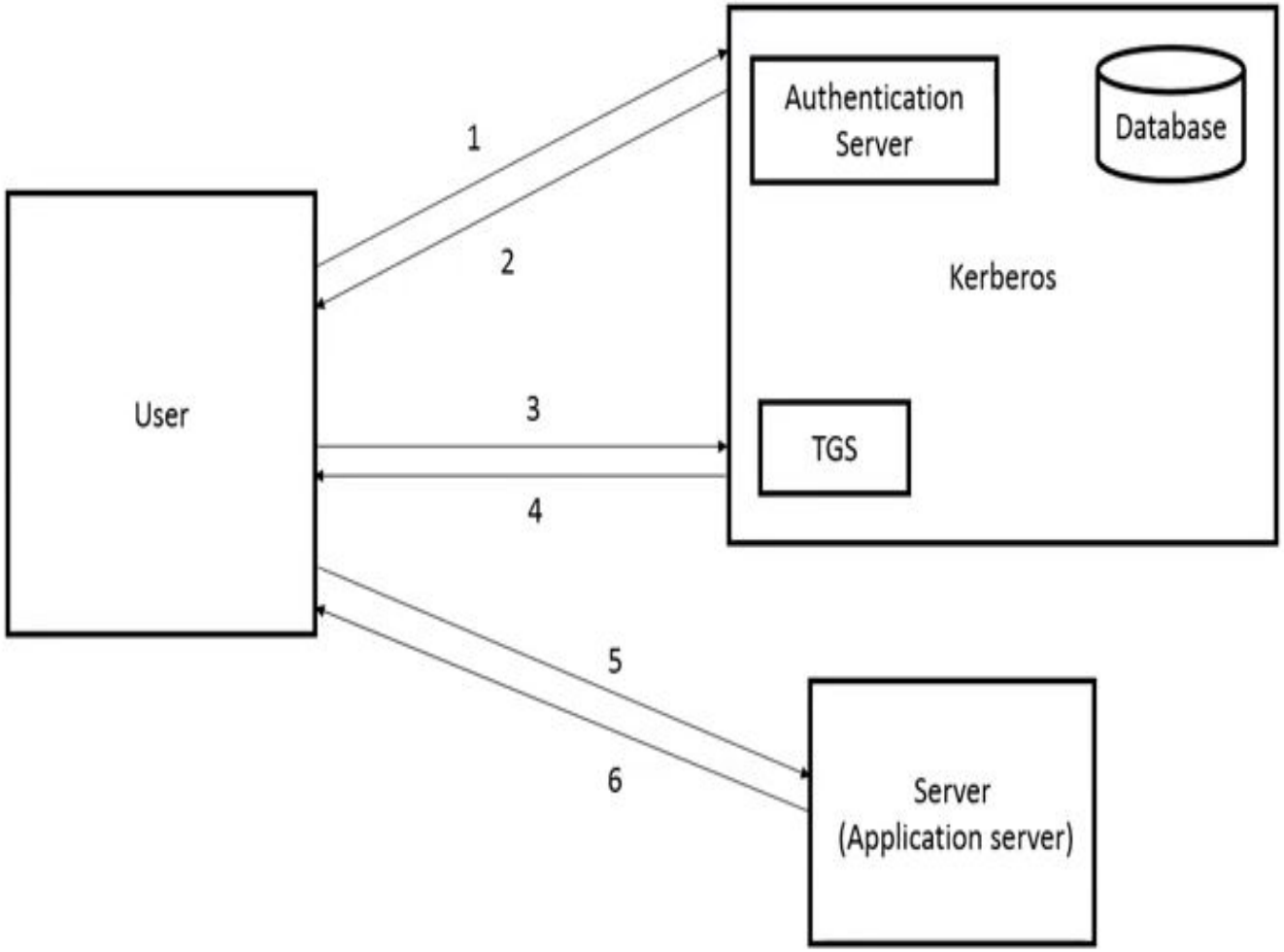
(5) $C \rightarrow V \quad Ticket_v \| Authenticator_c$

(6) $V \rightarrow C \quad E(K_{c,v}, [TS_5 + 1])$ (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



RAMAIAH
Institute of Technology

Kerberos - THE VERSION 4 AUTHENTICATION DIALOGUE

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

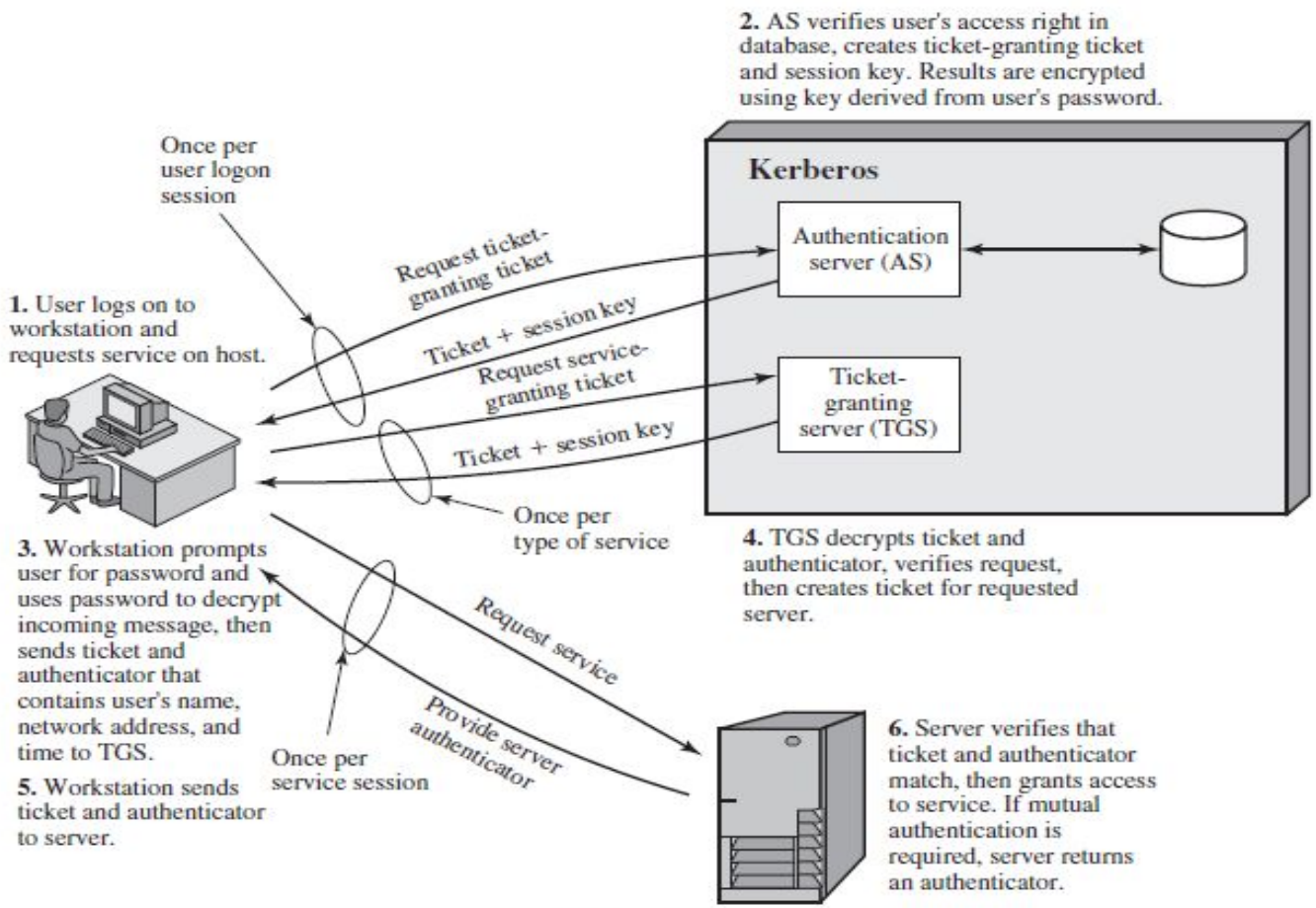
(5) C → V $Ticket_v \| Authenticator_c$

(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



Overview of Kerberos

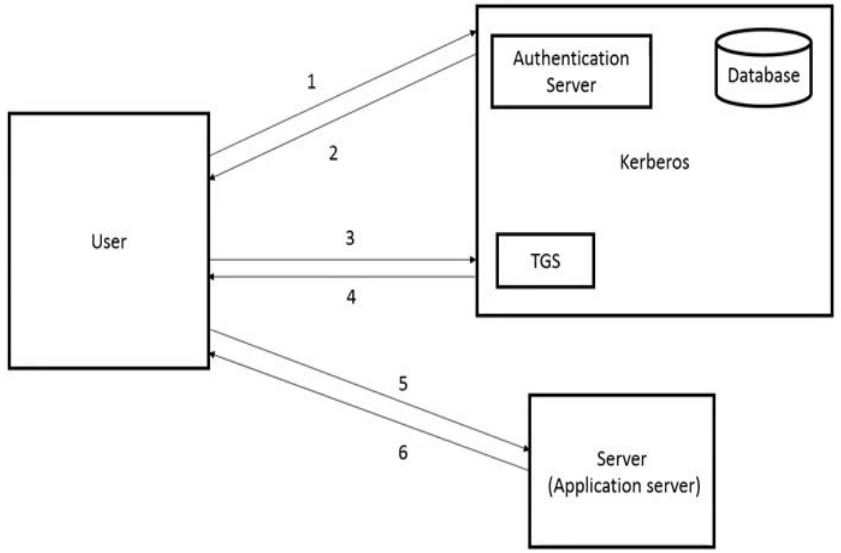
(1) C → AS $ID_c \| ID_{tgs} \| TS_1$

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

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

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.

(a) Authentication Service Exchange



K_c = key that is derived from user password

$K_{c,tgs}$ = session key for C and TGS

K_{tgs} = key shared only by the AS and the TGS

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

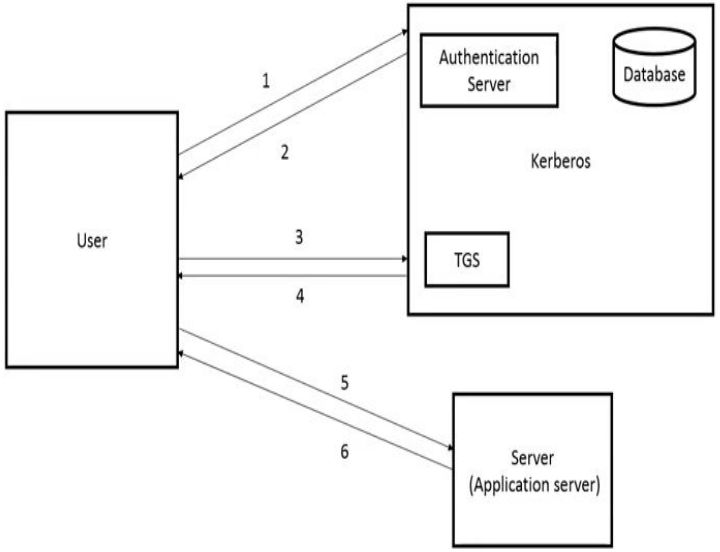
(4) $TGS \rightarrow C \quad E(K_{c,tgs}, [K_{c,v} \| ID_v \| TS_4 \| Ticket_v])$

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

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

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

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



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.

K_c = key that is derived from user password
$K_{c,tgs}$ = session key for C and TGS
K_{tgs} = key shared only by the AS and the TGS

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

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

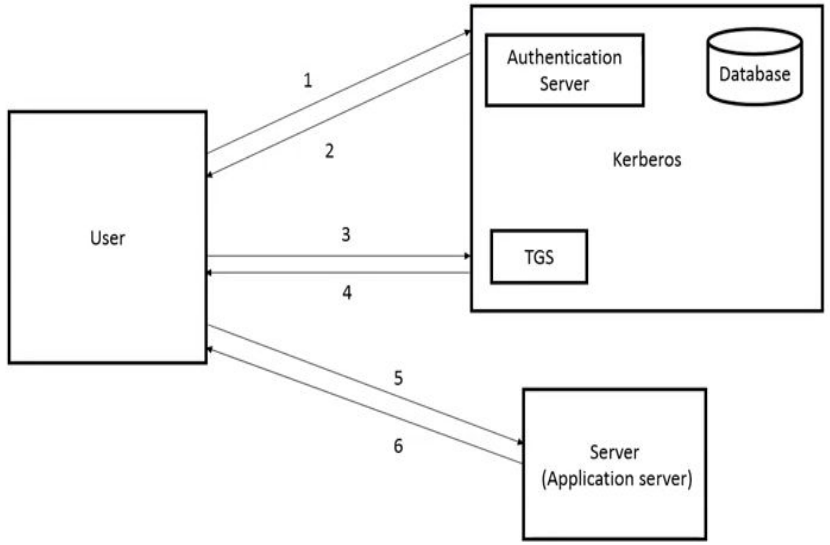
$$Ticket_{tgs} = E(K_{tgs}, [K_{c,tgs} \| ID_C \| AD_C \| ID_{tgs} \| TS_2 \| Lifetime_2])$$

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

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

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

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.
K_v	Ticket is encrypted with key known only to TGS and server, to prevent tampering.



Kerberos - THE VERSION 4 AUTHENTICATION DIALOGUE

Summary of Kerberos Version 4 Message Exchanges

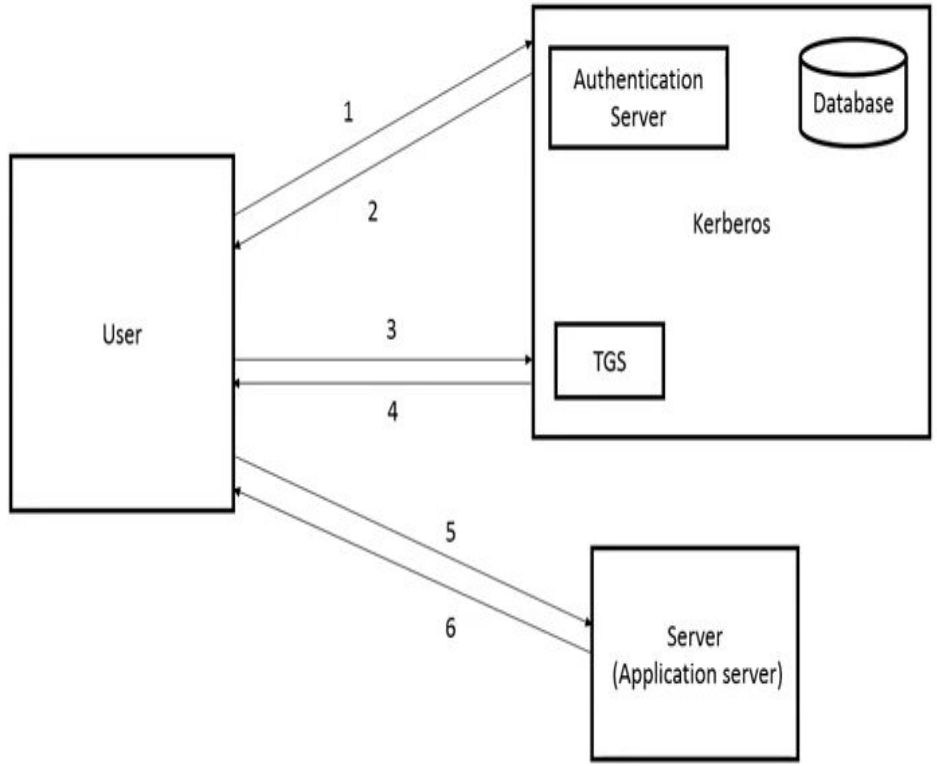
(5) $C \rightarrow V$ $Ticket_v \| Authenticator_c$

(6) $V \rightarrow C$ $E(K_{c,v}, [TS_5 + 1])$ (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



RAMAIAH
Institute of Technology

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

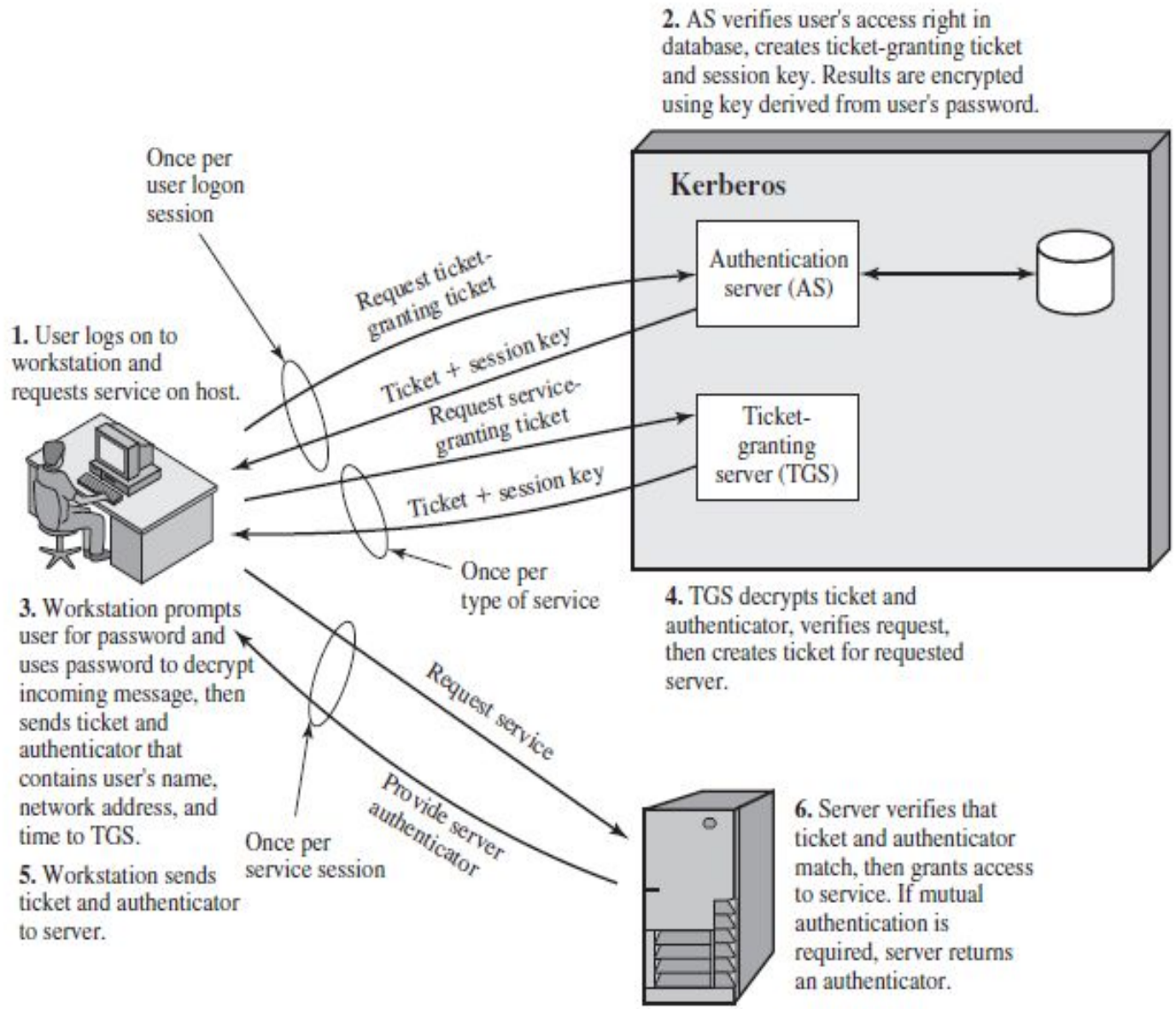
(5) C → V $Ticket_v \| Authenticator_c$

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

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

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

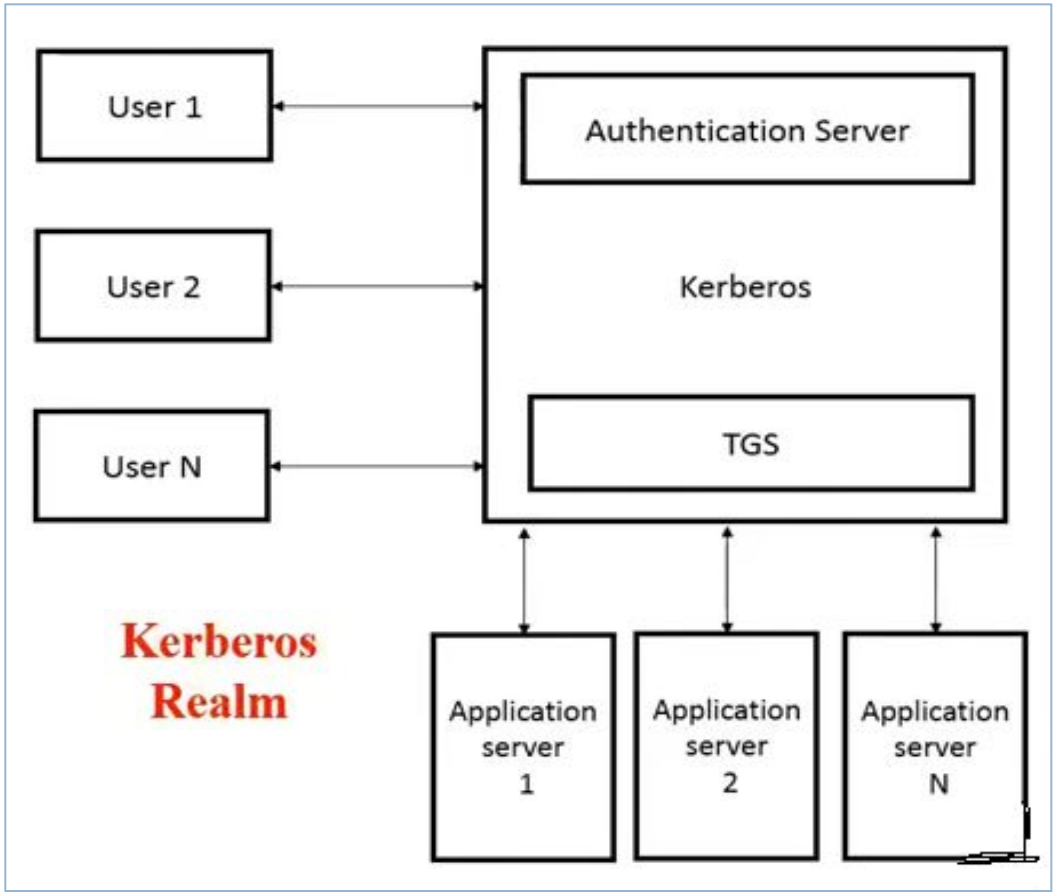
(c) Client/Server Authentication Exchange to obtain service



Overview of Kerberos

KERBEROS REALM

- A full-service Kerberos environment consists of
  - a Kerberos server
  - a number of clients, all are registered with Kerberos server
  - a number of application servers, all are sharing keys with Kerberos server
- Such an environment is referred to as a **Kerberos realm.**



RAMAIAH
Institute of Technology

Inter Realm Authentication

(1) $C \rightarrow AS: \quad ID_c \| ID_{tgs} \| TS_1$

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

(3) $C \rightarrow TGS: \quad ID_{tgsrem} \| Ticket_{tgs} \| Authenticator_c$

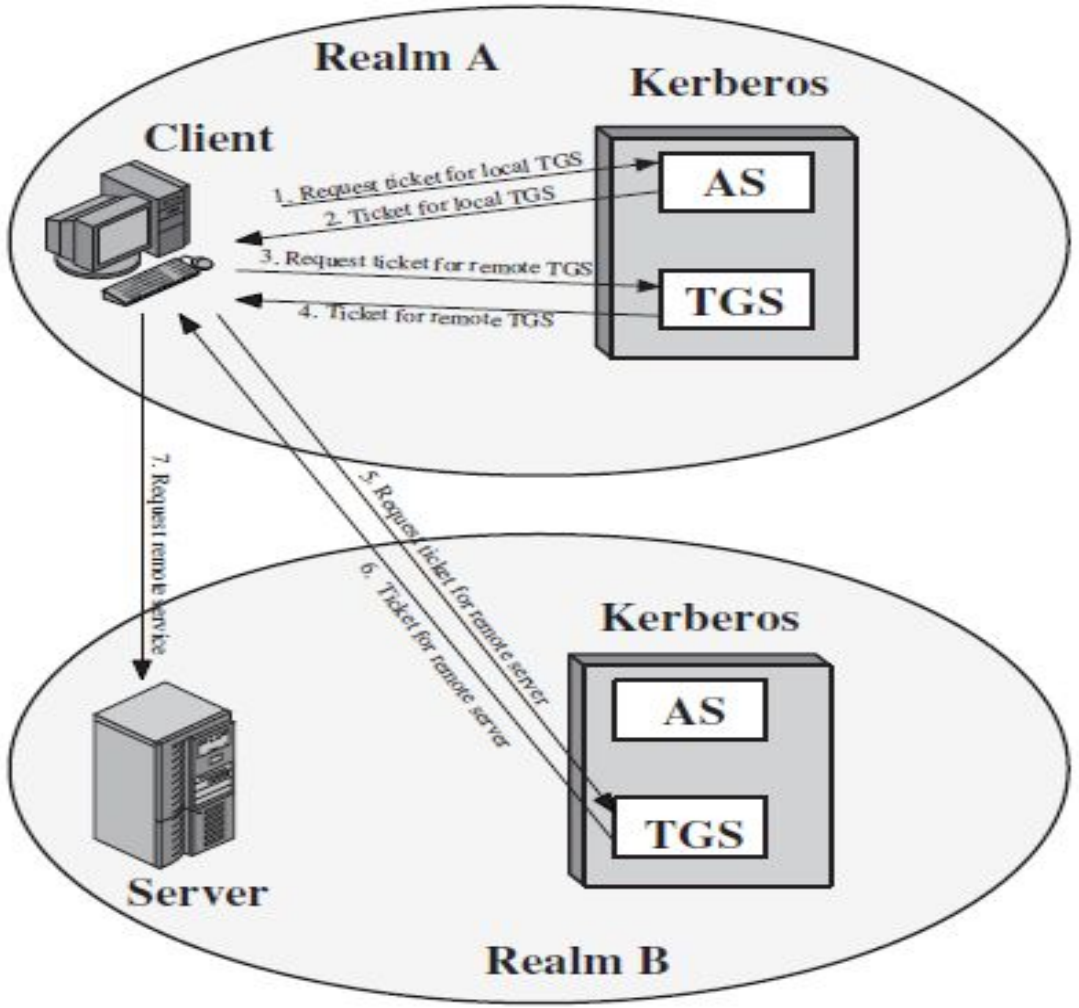
(4) $TGS \rightarrow C: \quad E(K_{c,tgs}, [K_{c,tgsrem} \| ID_{tgsrem} \| TS_4 \| Ticket_{tgsrem}])$

(5) $C \rightarrow TGS_{rem}: \quad ID_{vrem} \| Ticket_{tgsrem} \| Authenticator_c$

(6) $TGS_{rem} \rightarrow C: \quad E(K_{c,tgsrem}, [K_{c,vrem} \| ID_{vrem} \| TS_6 \| Ticket_{vrem}])$

(7) $C \rightarrow V_{rem}: \quad Ticket_{vrem} \| Authenticator_c$

1. C → AS : Request ticket for local TGS
2. AS → C : Ticket for local TGS
3. C → TGS : Request ticket for remote TGS
4. TGS → C : Ticket for remote TGS
5. C → TGS_{rem} : Request ticket for remote Server
6. TGS_{rem} → C : Ticket for remote Server
7. C → V_{rem} : Request for remote service



Request for Service in Another Realm

Kerberos Version 5

Kerberos version 5 is specified in RFC 4120 and provides a number of improvements over version 4.

- Differences between versions 4 and 5
- Version 5 protocol

Kerberos Version 5

Differences between versions 4 and 5

Version 5 is intended to address the limitations of version 4 in two areas:

1. Environmental shortcomings
2. Technical deficiencies

Kerberos Version 5

Environmental shortcomings

1. Encryption system dependence
2. Internet protocol dependence
3. Message byte ordering
4. Ticket lifetime
5. Authentication forwarding
6. Interrealm authentication

Kerberos Version 5

Environmental shortcomings

1.Encryption system dependence

- Version 4 ***use only DES (Data encryption Standards).***
- In version 5, ***any encryption techniques can be used.***

2.Internet protocol dependence

- Version 4 supports ***Internet Protocol (IP) addresses***, but cannot support ***ISO network address.***
- Version 5 supports ***any types of network addresses*** with variable length.

Kerberos Version 5

Environmental shortcomings

3. Message byte ordering

- In version 4, the sender of a message employs a byte ordering of its own choosing and tags the message to indicate least significant byte in lowest address or most significant byte in lowest address.
- In version 5, all message structures are defined using Abstract Syntax Notation One (ASN.1) and Basic Encoding Rules (BER), which provide an unambiguous byte ordering.

Kerberos Version 5

Environmental shortcomings

4. Ticket lifetime

- In version 4, the ticket lifetime has to be specified in units for a lifetime of 5 minutes. It encoded in an 8-bit quantity in units of five minutes.

- Thus, the maximum lifetime that can be expressed is

$2^8 \text{ X } 5 = 1280$ minutes $\Rightarrow$ 21 Hours

- In version 5, ticket one lifetime can allowing arbitrary lifetimes.

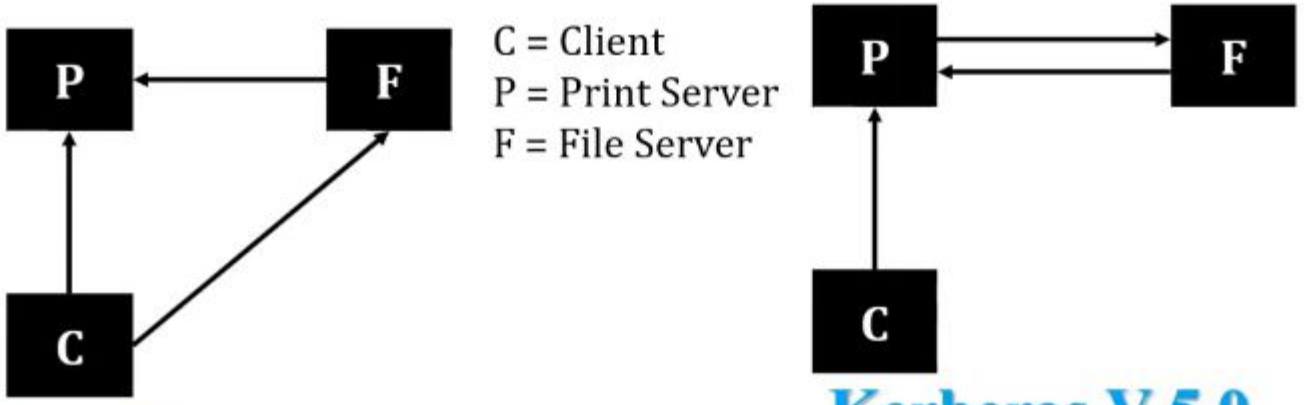
RAMAIAH
Institute of Technology

Kerberos Version 5

Environmental shortcomings

5.Authentication forwarding

- Version 4 does not allow credentials issued to one client to be forwarded to some other host and used by some other client.
- Version 5 provides authentication forwarding, it means client to access a server and have that server access another server on behalf of the client.

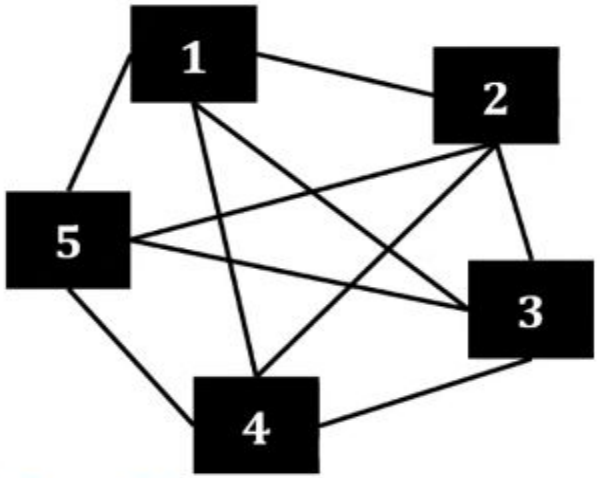


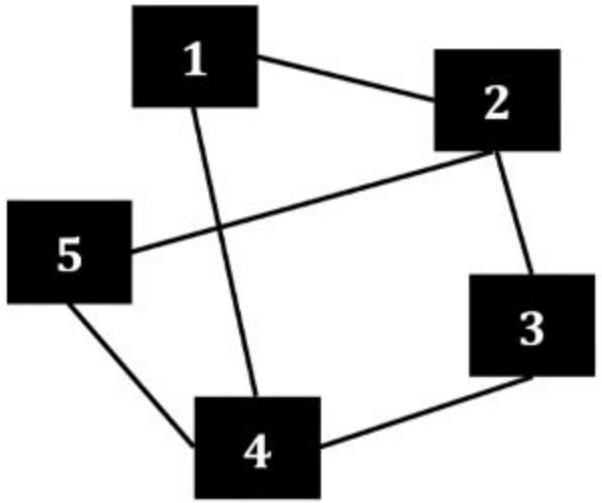
Kerberos Version 5

Environmental shortcomings

6.Interrealm authentication

- In version 4, interoperability among realms requires ***many Kerberos - to - Kerberos relationships***.
- In version 5 supports a method that requires ***fewer relationships***.





Kerberos Version 5

Technical deficiencies

1. Double encryption
2. PCBC encryption
3. Session keys
4. Password attacks

Kerberos Version 5

1. Double encryption

   In Kerberos Version4, tickets provided to clients are encrypted twice—once with the secret key of the target server and then again with a secret key known to the client.

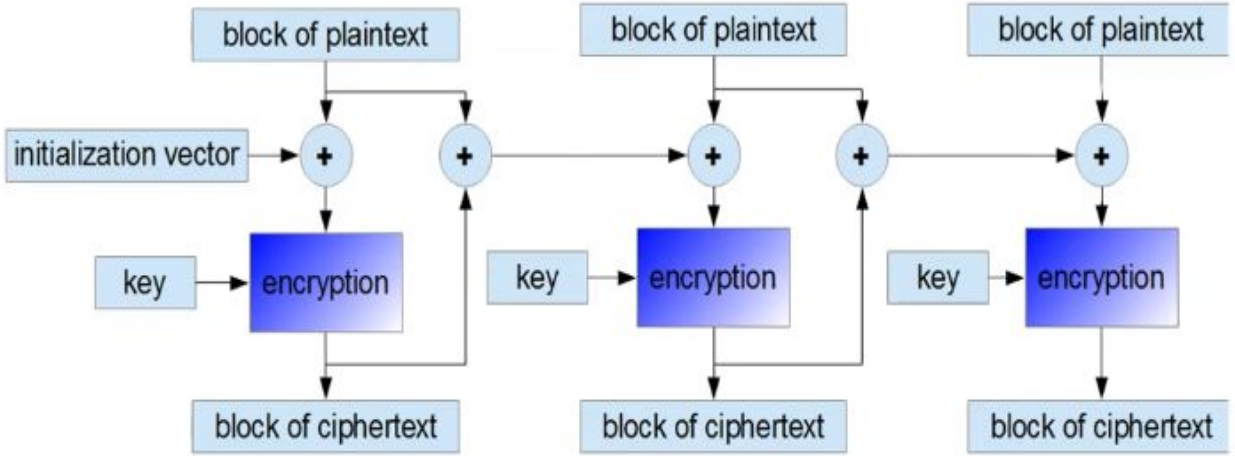
   The second encryption is not necessary and is computationally wasteful.

RAMAIAH
Institute of Technology

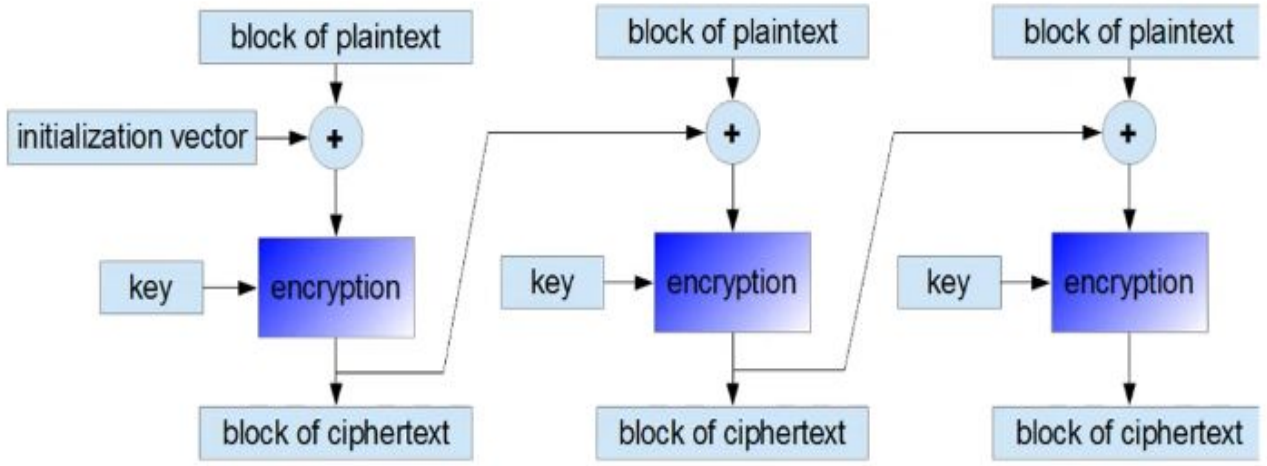
Kerberos Version 5

PCBC encryption

- Encryption in version 4 makes use of a non standard mode of DES known as ***propagating cipher block chaining (PCBC).*** This mode is vulnerable to an attack involving the ***interchange of ciphertext blocks.***
- Version 5 allows the ***standard CBC mode*** to be used for encryption.



PCBC Encryption

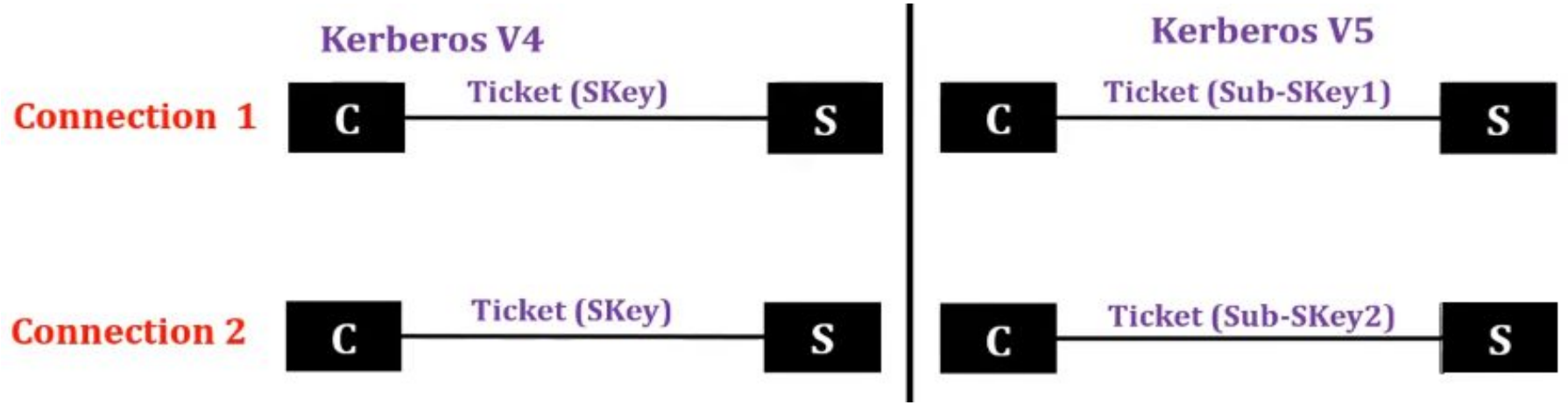


CBC Encryption

Kerberos Version 5

Session keys

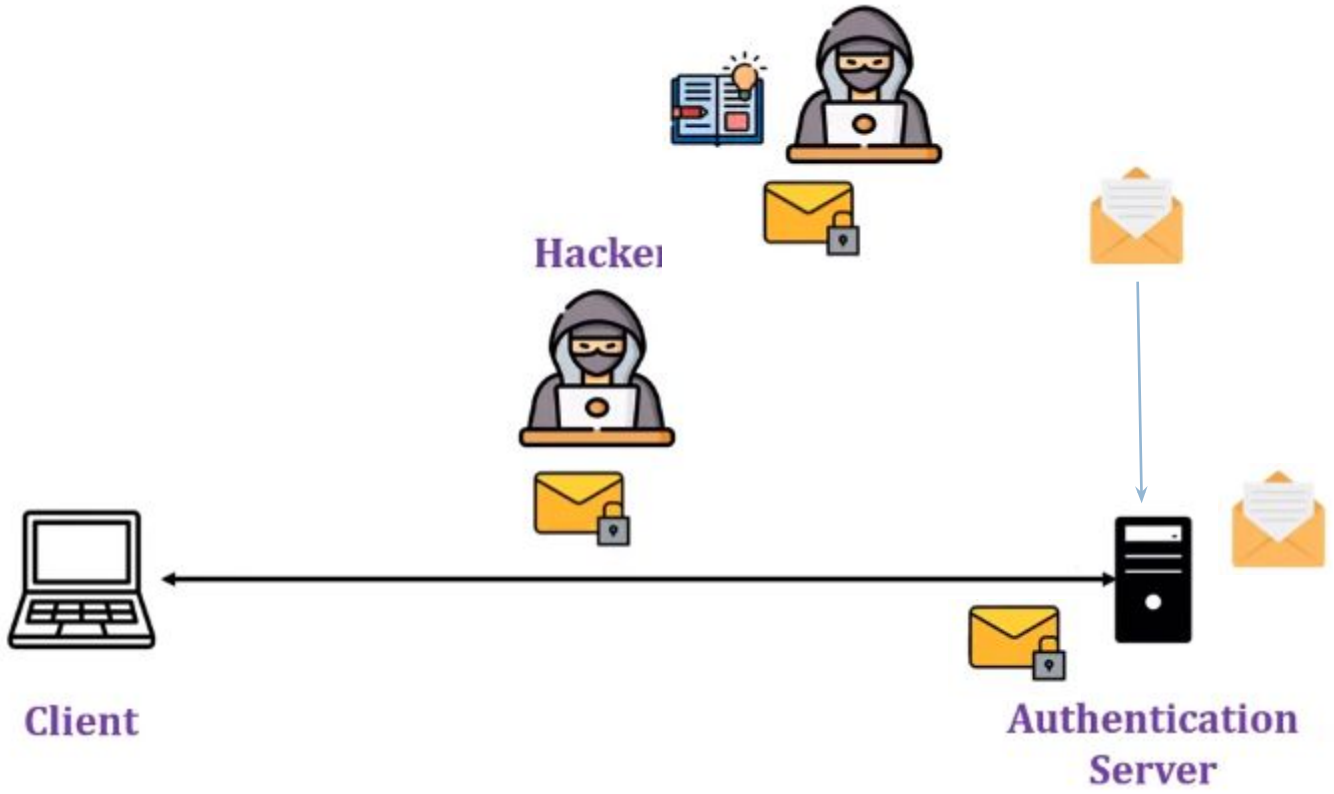
- Each ticket includes a ***session key used for encrypting messages.***
- However, because the same ticket may be used repeatedly, ***replay attack is possible.***
- In version 5, it is possible for a client and server to negotiate a ***sub-session key***, which is to be used only for that one connection.



Kerberos Version 5

Password attacks

Both versions are vulnerable to a password attack



RAMAIAH Institute of Technology

Kerberos Version 5

Summary of Kerberos Version 5 Message Exchanges

(1) C → AS Options || ID_c || $Realm_c$ || ID_{tgs} || $Times$ || $Nonce_1$

(2) AS → C $Realm_C$ || ID_C || $Ticket_{tgs}$ || $E(K_c, [K_{c,tgs} || Times || Nonce_1 || Realm_{tgs} || ID_{tgs}])$

$Ticket_{tgs} = E(K_{tgs}, [Flags || K_{c,tgs} || Realm_c || ID_C || AD_C || Times])$

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

(3) C → TGS Options || ID_v || $Times$ || $Nonce_2$ || $Ticket_{tgs}$ || $Authenticator_c$

(4) TGS → C $Realm_c || ID_C || Ticket_v || E(K_{c,tgs}, [K_{c,v} || Times || Nonce_2 || Realm_v || ID_v])$

$Ticket_{tgs} = E(K_{tgs}, [Flags || K_{c,tgs} || Realm_c || ID_C || AD_C || Times])$

$Ticket_v = E(K_v, [Flags || K_{c,v} || Realm_c || ID_C || AD_C || Times])$

$Authenticator_c = E(K_{c,tgs}, [ID_C || Realm_c || TS_1])$

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

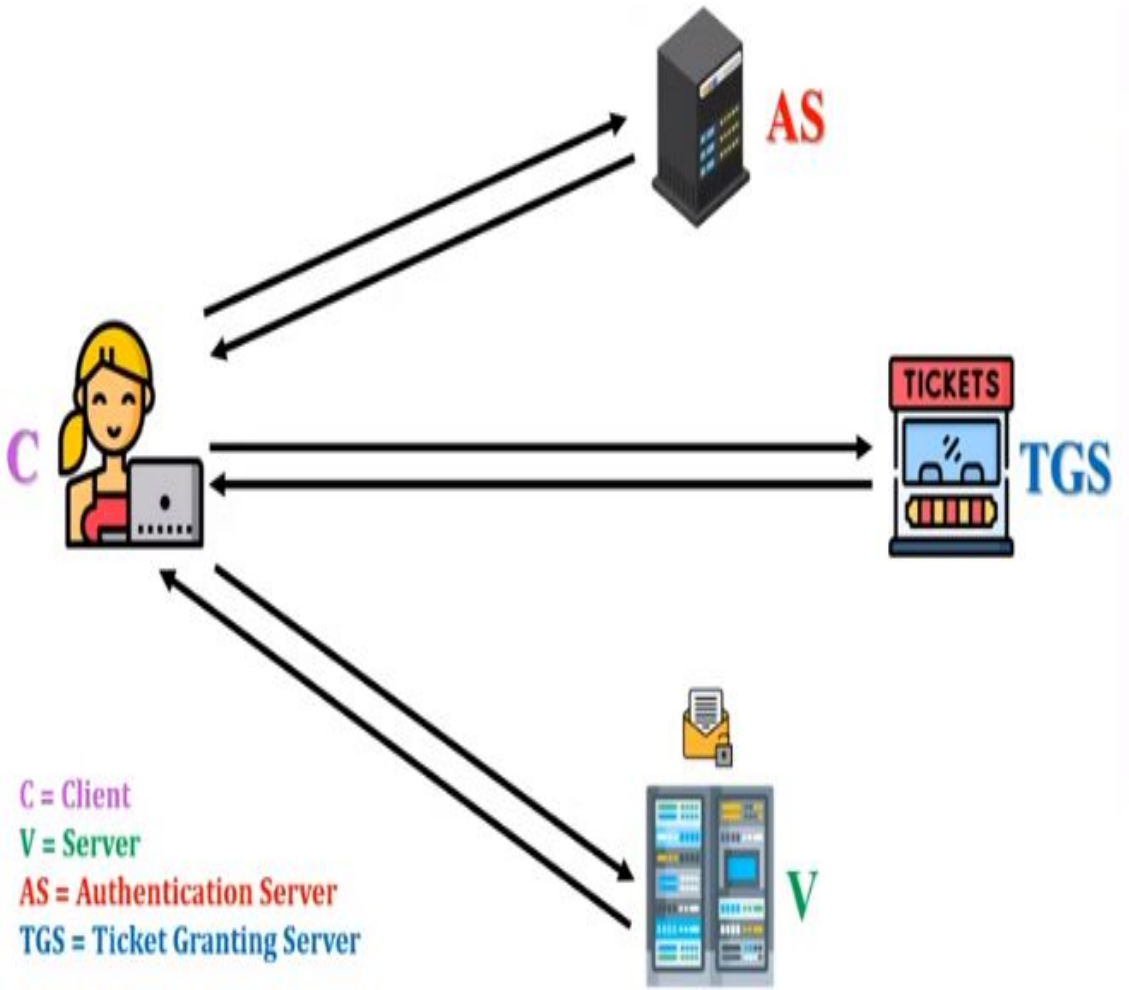
(5) C → V Options || $Ticket_v$ || $Authenticator_c$

(6) V → C $E_{K_{c,v}}[TS_2 || Subkey || Seq\#]$

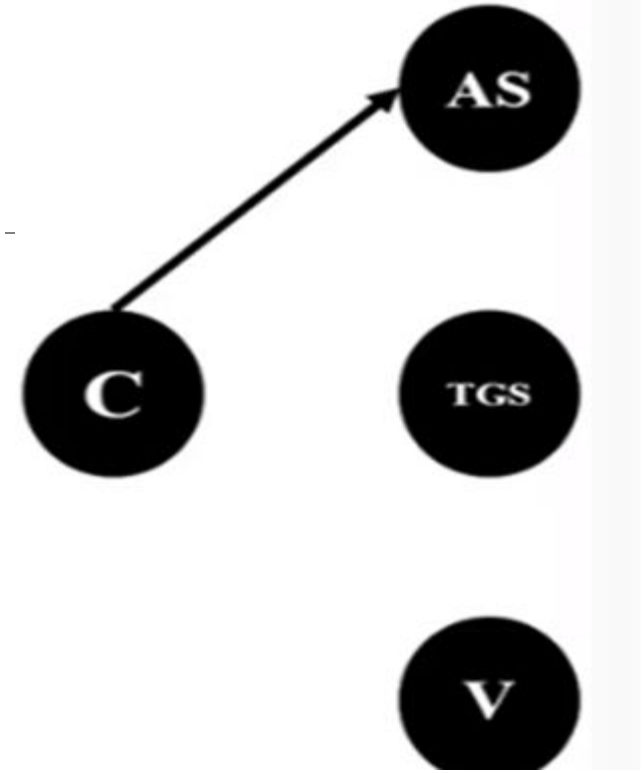
$Ticket_v = E(K_v, [Flag || K_{c,v} || Realm_c || ID_C || AD_C || Times])$

$Authenticator_c = E(K_{c,v}, [ID_C || Relam_c || TS_2 || Subkey || Seq\#])$

(c) Client/Server Authentication Exchange to obtain service



1) C ➔ AS: Options || ID_c || $Realm_c$ || ID_{tgs} || Times || $Nonce_1$



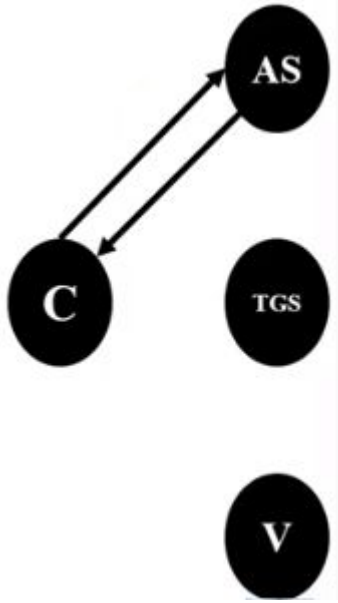
- **Options:** Used to request that certain flags be set in the returned ticket
- **Realm:** Indicates realm of user
- **Times:** Used by the client to request the following time settings in the ticket:
  - **from:** start time for the requested ticket
  - **till:** expiration time for the requested ticket
  - **rtime:** requested renew-till time
- **Nonce:** A random value to avoid replay attack

INITIAL	This ticket was issued using the AS protocol and not issued based on a ticket-granting ticket.
PRE-AUTHENT	During initial authentication, the client was authenticated by the KDC before a ticket was issued.
HW-AUTHENT	The protocol employed for initial authentication required the use of hardware expected to be possessed solely by the named client.
RENEWABLE	Tells TGS that this ticket can be used to obtain a replacement ticket that expires at a later date.
MAY-POSTDATE	Tells TGS that a postdated ticket may be issued based on this ticket-granting ticket.
POSTDATED	Indicates that this ticket has been postdated; the end server can check the authtime field to see when the original authentication occurred.
INVALID	This ticket is invalid and must be validated by the KDC before use.
PROXIABLE	Tells TGS that a new service-granting ticket with a different network address may be issued based on the presented ticket.
PROXY	Indicates that this ticket is a proxy.
FORWARDABLE	Tells TGS that a new ticket-granting ticket with a different network address

Kerberos Version 5

(2) $AS \rightarrow C \quad Realm_C \| ID_C \| Ticket_{tgs} \| E(K_c, [K_{c,tgs} \| Times \| Nonce_1 \| Realm_{tgs} \| ID_{tgs}])$

$Ticket_{tgs} = E(K_{tgs}, [Flags \| K_{c,tgs} \| Realm_c \| ID_C \| AD_C \| Times])$



K_c = Share between C & AS

$K_{c,tgs}$ = Shared between C & TGS

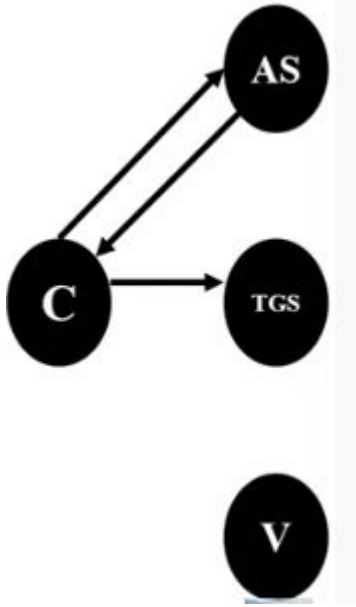
K_{tgs} = Shared between TGS & AS

Kerberos Version 5

3) C → TGS: $Options \| ID_v \| Times \| Nonce_2 \| Ticket_{tgs} \| Authenticator_c$

$Ticket_{tgs} = E(K_{tgs} [Flags \| K_{c,tgs} \| Realm_c \| ID_c \| AD_c \| Times])$

$Authenticator_c = E(K_{c,tgs} [ID_c \| Realm_c \| TS_1])$



K_{tgs} = Shared between TGS & AS

$K_{c,tgs}$ = Shared between C & TGS

Kerberos Version 5

(4) $TGS \rightarrow C \quad Realm_c \| ID_C \| Ticket_v \| E(K_{c,tgs}, [K_{c,v} \| Times \| Nonce_2 \| Realm_v \| ID_v])$

$$Ticket_{tgs} = E(K_{tgs}, [Flags \| K_{c,tgs} \| Realm_c \| ID_C \| AD_C \| Times])$$

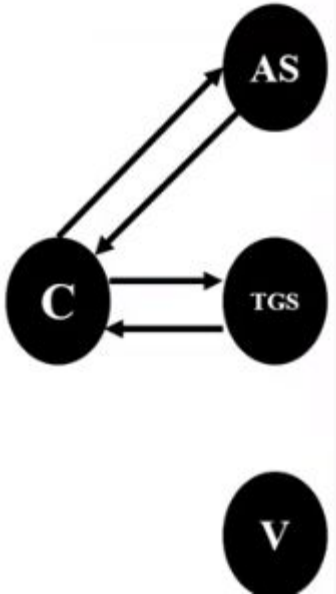
$$Ticket_v = E(K_v, [Flags \| K_{c,v} \| Realm_c \| ID_C \| AD_C \| Times])$$

$$Authenticator_c = E(K_{c,tgs}, [ID_C \| Realm_c \| TS_1])$$

K_{tgs} = Shared between TGS & AS

$K_{c,tgs}$ = Shared between C & TGS

K_v = Shared between V & TGS



Kerberos Version 5

5) C → V Options || $Ticket_v$ || $Authenticator_c$

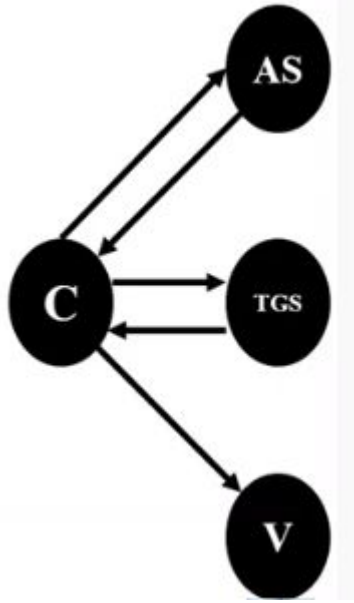
$Ticket_v = E(K_v, [Flags \| K_{c,v} \| Realm_c \| ID_c \| AD_c \| Times])$

$Authenticator_c = E(K_{c,v} [ID_c \| Realm_c \| TS_2 \| Subkey \| Seq\#])$

In message (5), The authenticator includes several new fields:

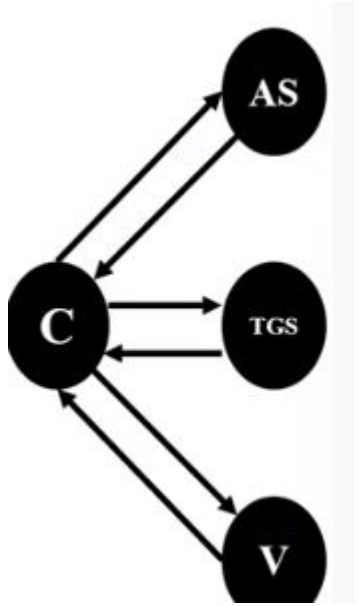
- **Subkey:** The client's choice for an encryption key to be used to protect this specific application session. If this field is omitted, the session key from the ticket ($K_{c,v}$) is used.
- **Sequence number:** Sequence number is used to detect replay attack.

K_v = Shared between V & TGS $K_{c,v}$ = Shared between C & V



Kerberos Version 5

(6) $V \rightarrow C \quad E_{K_{c,v}}[TS_2 \| Subkey \| Seq\neq]$



OUTLINE - (Text 2)

Kerberos (Chapter 15.3)

Transport Level security: Web Security Considerations, Secure Sockets Layer, Transport level Security, HTTPS.

Electronic Mail security: S/MIME, Pretty Good Privacy

THANK YOU