What is Kerberos?

- Kerberos is an authentication protocol that provides authentication for client-server applications through symmetric encryption.
- Authentication is mediated by an Authentication Server (AS) which has to be a trusted party.
- Ticket: specifies that a particular client has been authenticated.
- Realm: network under the control of an AS.
- Named after the Greek mythological character Kerberos (or Cerberus), known in Greek mythology as being the monstrous three-headed guard dog of Hades.

Kerberos Overview

- Designed originally for Project Athena at M.I.T.
- Implementation freely available from M.I.T.
- V5 is proposed as an Internet Standard (RFC 4120).
- Windows 2000/XP/Server 2003/Vista use Kerberos as their default authentication mechanism.
- Apple’s Mac OS X clients and servers also use Kerberos.
- Protects against eavesdropping and replay attacks.
- Uses symmetric encryption.
- First 3 versions are no longer in use.
- V5 is a generalization of V4 with several problems fixed and additional features.

Motivations for Kerberos

- Mutual authentication of two parties using symmetric key does not scale.
- Generalizing it to \( m \) users and \( n \) services requires a priori distribution of \( nXm \) keys.
- A possible improvement:
  - Use a trusted third party, with which each user and service shares a secret key; in this case we only need \( n + m \) keys.
Mediated Authentication

- The trusted third AS mediates the authentication process
- Each user and service share a secret key with AS
- AS generates a session key and securely distributes it to the communicating parties
- Communicating parties prove to each other that they know the session key

**Basic Authentication Protocol**

User performs logon on the client machine C; the client performs a one-way function on the key (hash usually) and this becomes the key $K_c$ shared between client and AS

$C \rightarrow AS: ID_c \| ID_S \| TS_c$

Note: the client does not need to send the password to AS nor $K_c$. AS generates the secret key by hashing the user password found in the AS DB (for example Active Directory in Windows Server)

$AS \rightarrow C: E_{K_c} (Ticket)$

Note: C is able to obtain the ticket only if C knows $K_c$

$C \rightarrow S: ID_c \| Ticket$

Ticket = $E_{K_s}[ ID_c \| ID_S \| TS_c]$

- $ID_c$ and $ID_S$ represent the identifiers of client and the service respectively
- $K_c$ is the key of client known to AS; the key is generated from the user password; the password has to be pre-registered with AS
- $E$ denotes encryption
- $K_s$ is the key of service S known to AS

Note: Ticket is encrypted; client cannot forge it or tamper with it

**Improved Authentication Protocol**

- The previous approach is still inconvenient
- If the user wants to access multiple servers, the user has to enter his/her password each time it needs accessing a network services – which has low usability
- An alternative would be to store key $K_c$ at the user machine, which is risky
- How to address this problem:
  - Introduce a new service – the ticket-granting service
  - Use two type of tickets with two different lifetimes:
    - One ticket grants to right to ask for service; it is generated once per login session Ticket_{tgs}
    - For each type of service, use a ticket that grants the right to use that particular service Ticket_{S}
  - Every time that service is needed, used the ticket Ticket_{S}
- Mark time when tickets are issued and also lifetime of tickets.
Improved Authentication Protocol - Steps

• When a user logs on, the client requests a ticket for the AS just as it would request a ticket for any other service.
• The AS responds by creating a logon session key and a ticket for a special server, the ticket-granting service (TGS).
• One copy of the logon session key is embedded in the ticket, and the ticket is encrypted with the TGS master key.
• Another copy of the logon session key is encrypted with the user’s master key derived from the user’s logon password. Both the ticket and the encrypted session key are sent to the client.
• When the client gets the AS’s reply, it decrypts the logon session key with the user’s master key derived from the user’s password. The client no longer needs the key derived from the user’s password because the client will now use the logon session key to decrypt its copy of any server session key it gets from the AS. The client stores the logon session key in its ticket cache along with its ticket for the full ticket-granting service.
• The ticket for the full ticket-granting service is called a ticket-granting ticket (TGT).
• When the client asks Kerberos for a ticket to a server, it presents credentials in the form of an authenticator message and a ticket — in this case a TGT — just as it would present credentials to any other service.

TGT = EKtgs
[ Kc,tgs || IDtgs || TS2 || Lifetime2 || Tickettgs ]

IDtgs denotes the identifier of the Ticket Granting Server (TGS)
TS2 and Lifetime2 are timestamps
Kc,tgs is the key shared by the TGS and client C (called the session key)
Tickettgs is the ticket

V4: Authentication Service Exchange

Goal: Obtain Ticket-Granting Ticket

C → AS: IDc || IDtgs || TS1
AS → C: EKc
[ Kc,tgs || IDtgs || TS2 || Lifetime2 || Tickettgs ]

Tickettgs = EKtgs
[ Kc,tgs || IDtgs || TS2 || Lifetime2 || Tickettgs ]

IDtgs denotes the identifier of the Ticket Granting Server (TGS)
TS1 and TS2 are timestamps
Kc is the key of client C known to AS
Kc,tgs is the key shared by the TGS and client C
Tickettgs is the ticket

V4: Ticket-Granting Service Exchange

Goal: Obtain Service-Granting Ticket

C → TGS: IDc || Tickettgs || AuthenticatorC
TGS → C: EKc,tgs
[ KS || IDc || TicketS || TS4 ]

TicketS = EKS
[ KS || IDc || TS4 || Lifetime4 || TicketS ]

AuthenticatorC = EKc,tgs
[ IDc || TS3 ]

KS is the key shared by the TGS and server S
IDc represents the identifier of the service
V4: Client-Server Authentication Exchange

Goal: Obtain Service

C $\rightarrow$ S: $\text{Tickets} || \text{Authenticator}_C$

S $\rightarrow$ C: $E_{K_{c,tgs}} [TS_5 + 1]$

$\text{Tickets} = E_{K_{c,tgs}} [K_{C,S} || \text{ID}_C || \text{AD}_C || \text{ID}_S || TS_4 || \text{Lifetime}_S]$

$\text{Authenticator}_C = E_{K_{c,tgs}} [\text{ID}_C || \text{AD}_C || TS_5]$

Summary of Symmetric Keys in Kerberos

- $K_c$ is long-term key of client C
  - Derived from user's password
  - Known to client and AS

- $K_{tgs}$ is long-term key of TGS
  - Known to AS and ticket granting service (TGS)

- $K_s$ is long-term key of service S
  - Known to S and TGS; separate key for each service

- $K_{C,tgs}$ is short-term key between client C and TGS
  - Created by AS, known to C and TGS

- $K_{C,s}$ is short-term key between client C and service S
  - Created by TGS, known to C and S

Request for Service in Another Realm

- Authenticate to local AS and obtain ticket to local TGS
- Ask local TGS for ticket for remote TGS, obtain ticket for remote TGS
- Ask remote TGS for ticket for remote server S, obtain ticket for remote server S
- Ask for service from remote server S