

$A \rightarrow TGS: A, B$
 $TGS \rightarrow A: \{T_{TGS}, L, K_{AB}, B, \{T_{TGS}, L, K_{AB}, A\}K_{B,TGS}\}K_{A,TGS}$
 $A \rightarrow B: \{T_{TGS}, L, K_{AB}, A\}K_{B,TGS}, \{A, T_A\}K_{AB}$
 $B \rightarrow A: \{T_{A+1}\}K_{AB}$

Kerberos Protocol V 5

What is Kerberos?

Kerberos is an authentication protocol used in Client/Server and Peer-to-Peer Network Architectures. In an unprotected network environment a client can request any service from any server. This creates vulnerabilities for *Modification and Fabrication Security Attacks*. Servers need to authenticate and confirm the identity of each client. This process creates a great load on the server.

Kerberos protocol uses an authentication server (AS) that stores passwords of all users in a centralized database. In Microsoft's case, this would be the Domain Controller. (**need to check that**, its either DC or Active Directory). Authentication Server shares a *secret key* with each server. These keys are usually distributed using Diffie-Hellman key exchange protocol. The process of validation is described in the next paragraph.

A client types in the password in the beginning of the logon session to be validated. This password is sent to Authentication Server along with workstation ID and a server ID. When the password is matched correctly with the centralized database on the Authentication Server and the user's permissions have been checked to see his access rights to the server, a ticket is created containing user's ID, user's network address, and server's ID. This ticket is then encrypted using (3DES, Blowfish) a symmetric encryption scheme with a *secret key* that is shared between AS and the server. The ticket is then sent back to the client and now the client can apply for service to the server. The client sends the ticket with the client's ID to the server and if validated the service is started between

A→TGS: A, B
TGS→A: {T_{TGS}, L, K_{AB}, B, {T_{TGS}, L, K_{AB}, A}K_{B,TGS}}}K_{A,TGS}
A→B: {T_{TGS}, L, K_{AB}, A}K_{B,TGS}, {A, T_A}K_{AB}
B→A: {T_A+1}K_{AB}

the client and the service. In order not to make the user type in the password every time he wants to use the service, the tickets must be reusable.

We introduce a concept of a Ticket Granting Server (TGS) that will allow us to reuse the tickets and to get rid of *plaintext* transmission of the password to the Authentication Server. The eavesdropper can capture the password and use any service accessible to the client.

The current protocol is described quite well in the RFC1510 and could be found on this site <http://www.ietf.org/rfc/rfc1510.txt>.