Kerberos

- Authentication is the process of proving the identity of a person or machine. The most common means of authentication are user names and passwords.
- Kerberos is an authentication system; it was described originally in RFC 1510. Its current RFC is 4120, although there are a number of RFC's that relate to Kerberos.
- Kerberos provides distributed authentication services.
- It was created at MIT during the mid-1980's in order to provide authentication in a distributed server environment.
- The current version is V, the design of which began in 1989.
- Kerberos uses DES for encryption.
- The basic infrastructure is a central Authentication Server (AS) and one or more Ticket Granting Servers (TGS).
- This infrastructure is used by users in order to authenticate themselves to servers that supply services.

Procedure

1. To begin a request for service, the user starts a client program that connects to the AS to request a Ticket Granting Ticket (TGT). The client sends the user name in cleartext to the AS, which looks it up in a database.
2. If the user name is found, the AS obtains the user's hashed password from the database. This step implies that the user's hashed password have been entered previously into the AS's database. Kerberos does not specify how this transfer is made.
3. The AS creates a pseudorandom client/TGS session key.
4. The AS creates a TGT containing the following:
    a) user ID
    b) client host network address
    c) ticket validity period
    d) client/TGS session key
5. The AS encrypts the client/TGS session key using the hashed password.
6. It then encrypts the TGT using the a secret key that the AS shares with the TGS.
7. The AS sends both the encrypted client/TGS session key and the encrypted TGT to the client.
8. The client then asks the user to enter a password. The password is hashed and is then used to decrypt the client/TGS session key. This step authenticates the user on the premise that the password entered was correct.

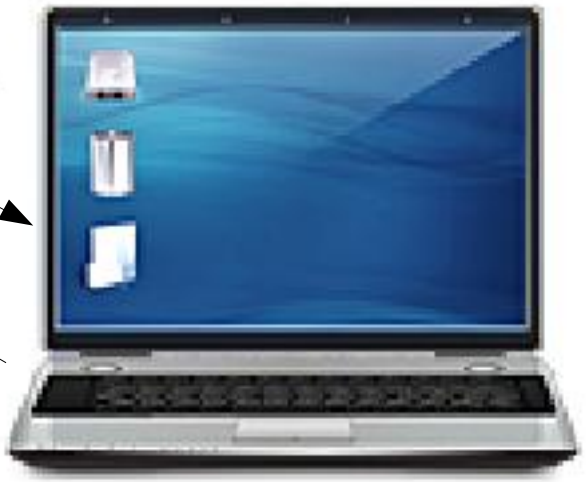
Step 1. Obtain a Ticket Granting Ticket (TGT)

AS

2

- TGT (encrypted using AS/TGS secret key)
  - user name
  - client IP address
  - validity period
  - client/TGS session key
- Client/TGS session key (encrypted using user password hash)

1 User name



Hash of user password is used to decrypt the client/TGS session key

9. In order to request a service, the client first creates an authenticator by concatenating the user ID with a timestamp and encrypting the message using the client/TGS session key.
10. The client then sends the following to the TGS:
    a) the TGT
    b) the ID of the requested service
    c) the encrypted authenticator
11. The TGS then decrypts the TGT using the secret key, thereby obtaining the client/TGS session key.
12. The TGS then decrypts the authenticator and checks whether the user ID's match and the timestamp is not stale.
13. If the request is valid, the TGS selects a pseudorandom client/server session key.
14. The TGS constructs a client/server ticket, which includes:
    a) user ID
    b) client host network address
    c) ticket validity period
    d) client/server session key
15. The TGS encrypts the client/server ticket with a secret key that it shares with the service.
16. The TGS encrypts the client/server session key using the client/TGS session key.
17. The TGS sends the encrypted ticket and session key back to the client.

Step 2. Obtain a client/server ticket

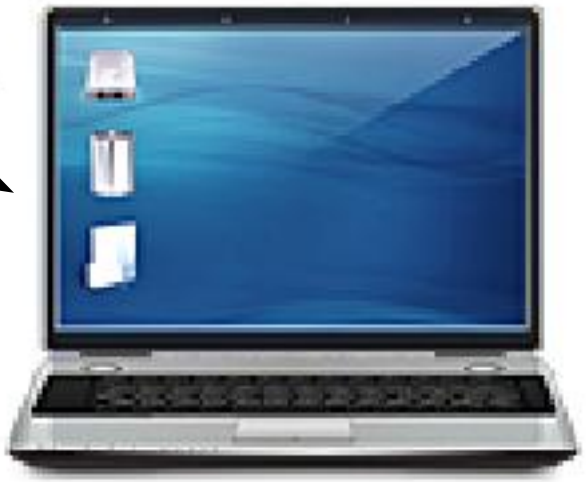
TGS

4

- Client/server ticket (encrypted using TGS/service server secret key)
  - user name
  - client IP address
  - validity period
  - client/server session key
- Client/server session key (encrypted using client/TGS session key)

3

- Authenticator (encrypted using client/TGS session key)
- TGT
- ID of requested service



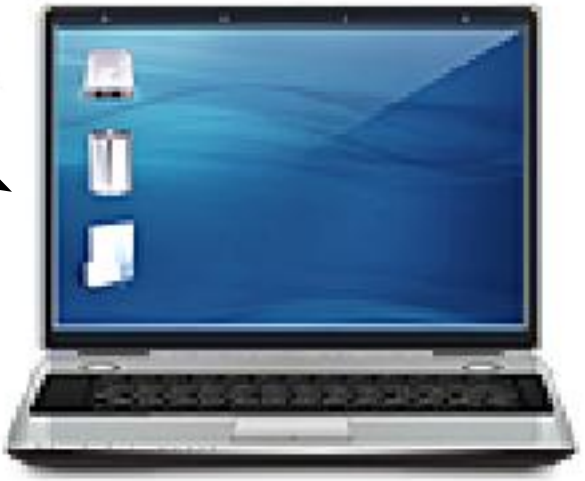
18. The client then decrypts the client/server session key and uses it to encrypt a new authenticator.
19. The client sends the client/server ticket and the encrypted authenticator to the service server.
20. The service server decrypts the ticket using its secret key and retrieves the client/server session key.
21. The service server then decrypts the authenticator and checks its validity.
22. If valid, the service server adds 1 to the timestamp found in the authenticator and then encrypts it using the client/server session key and sends it back to the client. This last step authenticates the service server to the client.
23. The client can now request services from the service server.

Step 3. Authenticate to obtain service

Service Server

6
- Revised timestamp (encrypted using client/server session key)

5
- Authenticator (encrypted using client/server session key)
- Client/server ticket



Further reading

MIT Kerberos Distribution Page

Kerberos (protocol) on Wikipedia

Description of Kerberos in the form of a play

Kerberizing a website