Kerberos Authentication Protocol

Kerberos is a mutual authentication protocol which lets workstations to share network resources and communicate with each other in a secure manner.

Version 4 of Kerberos is found in most implementations but version 5 is also in use now.

Kerberos Authentication Protocol

Kerberos is a mutual authentication protocol which lets workstations to share network resources and communicate with each other in a secure manner.

Version 4 of Kerberos is found in most implementations but version 5 is also in use now.

Working

There are 4 parties involved in the Kerberos Protocol:

Alice: The Client Workstation

Authentication Server (AS): Authenticates the Client during Login

Ticket Granting Server (TGS): Issues tickets to certify proof of identity

Bob: The Server offering services

Kerberos Authentication Protocol

The job of the AS is to authenticate every user at the login time.

The AS Shares a unique secret password with every user.

The job of the TGS is to certify to the servers in the network that a user is really who he/she calims to be.

For proving this, the mechanism of tickets is used.

There are 3 primary steps in the Kerberos Protocol :

1) Login
2) Obtaining a Service Granting Ticket
3) Communicate

LOGIN

1. Alice uses her workstation and sends her name in plaintext to the AS.

LOGIN

1. Alice uses her workstation and sends her name in plaintext to the AS.

2. The AS first creates a package of the user name(ALICE) and a randomly generated session key (KS).

LOGIN

1. Alice uses her workstation and sends her name in plaintext to the AS.

2. The AS first creates a package of the user name(ALICE) and a randomly generated session key (KS).

3. The AS first creates a package of the user name(ALICE) and a randomly generated session key (KS).

LOGIN

1. Alice uses her workstation and sends her name in plaintext to the AS.

2. The AS first creates a package of the user name(ALICE) and a randomly generated session key (KS).

3. The AS first creates a package of the user name(ALICE) and a randomly generated session key (KS).

4. It encrypts this package with the symmetric key that the AS shares with the Ticket Granting Server(TGS)

LOGIN

1. Alice uses her workstation and sends her name in plaintext to the AS.

2. The AS first creates a package of the user name(ALICE) and a randomly generated session key (KS).

3. The AS first creates a package of the user name(ALICE) and a randomly generated session key (KS).

4. It encrypts this package with the symmetric key that the AS shares with the Ticket Granting Server(TGS)

5. The output of this step is called as Ticket Granting Ticket (TGT).

LOGIN

1. Alice uses her workstation and sends her name in plaintext to the AS.

2. The AS first creates a package of the user name(ALICE) and a randomly generated session key (KS).

3. The AS first creates a package of the user name(ALICE) and a randomly generated session key (KS).

4. It encrypts this package with the symmetric key that the AS shares with the Ticket Granting Server(TGS)

5. The output of this step is called as Ticket Granting Ticket (TGT).

6. The TGT can be opened only by the TGS.

LOGIN

1. Alice uses her workstation and sends her name in plaintext to the AS.

2. The AS first creates a package of the user name(ALICE) and a randomly generated session key (KS).

3. The AS first creates a package of the user name(ALICE) and a randomly generated session key (KS).

4. It encrypts this package with the symmetric key that the AS shares with the Ticket Granting Server(TGS)

5. The output of this step is called as Ticket Granting Ticket (TGT).

6. The TGT can be opened only by the TGS.

7. The AS then combines the TGT with the session key(KS) and encrypts the two together using a symmetric key derived from the password of Alice (KA).

Step 1:

LOGIN

1. Alice uses her workstation and sends her name in plaintext to the AS.

2. The AS first creates a package of the user name(ALICE) and a randomly generated session key (KS).

3. The AS first creates a package of the user name(ALICE) and a randomly generated session key (KS).

4. It encrypts this package with the symmetric key that the AS shares with the Ticket Granting Server(TGS)

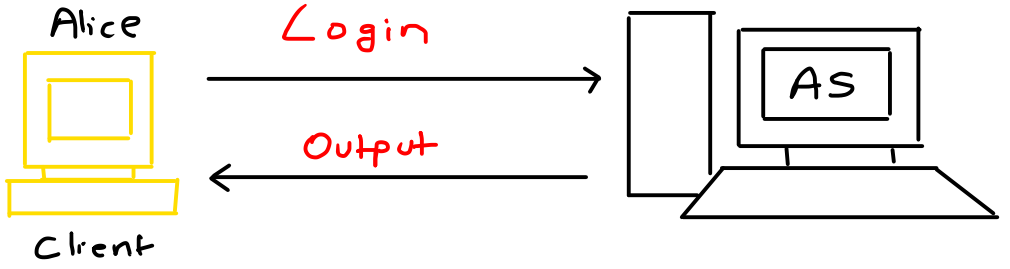
5. The output of this step is called as Ticket Granting Ticket (TGT).

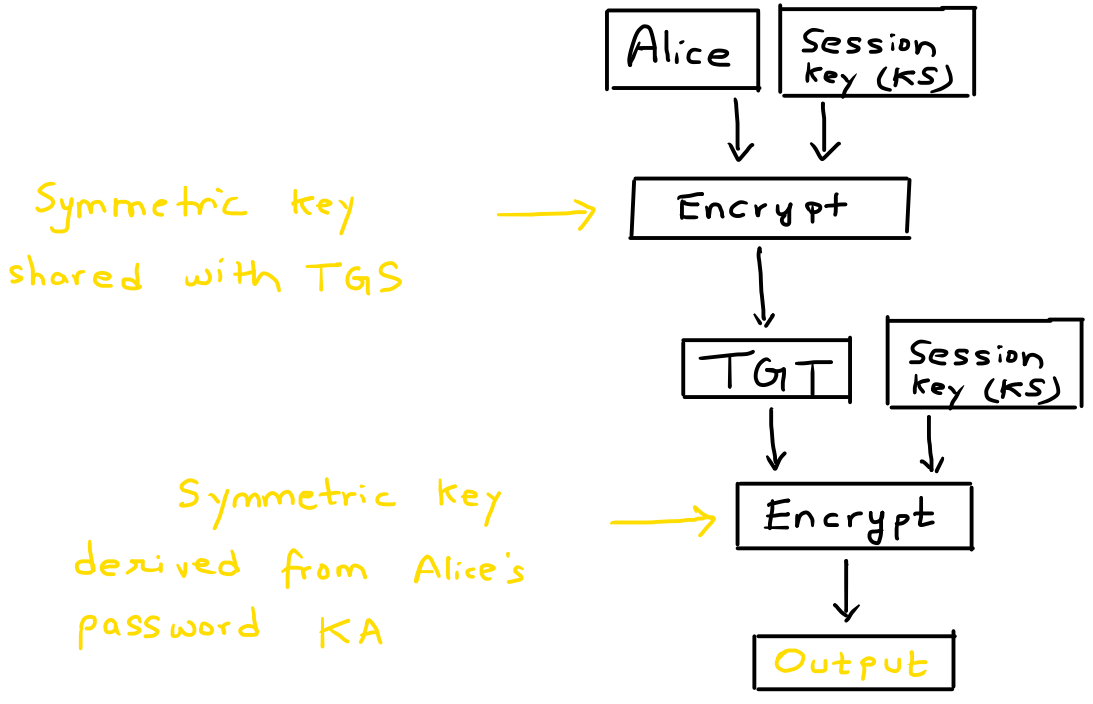
6. The TGT can be opened only by the TGS.

7. The AS then combines the TGT with the session key(KS) and encrypts the two together using a symmetric key derived from the password of Alice (KA).

8. The final output can therefore be only opened by Alice.

Step 1:


Alice
Login
Output
AS
Client

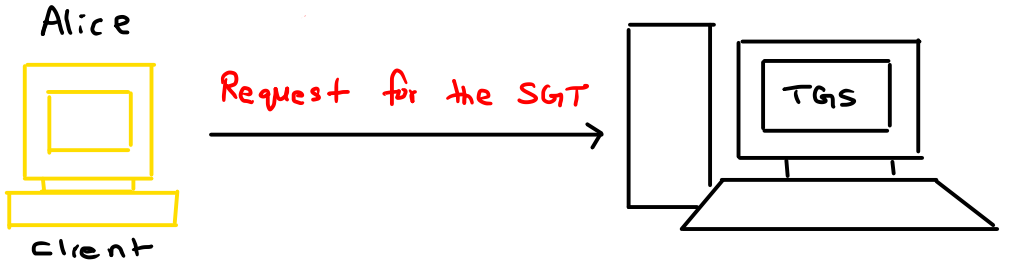

Alice
Session key (KS)
Symmetric key shared with TGS
Encrypt
TGT
Session key (KS)
Symmetric key derived from Alice's password KA
Encrypt
Output

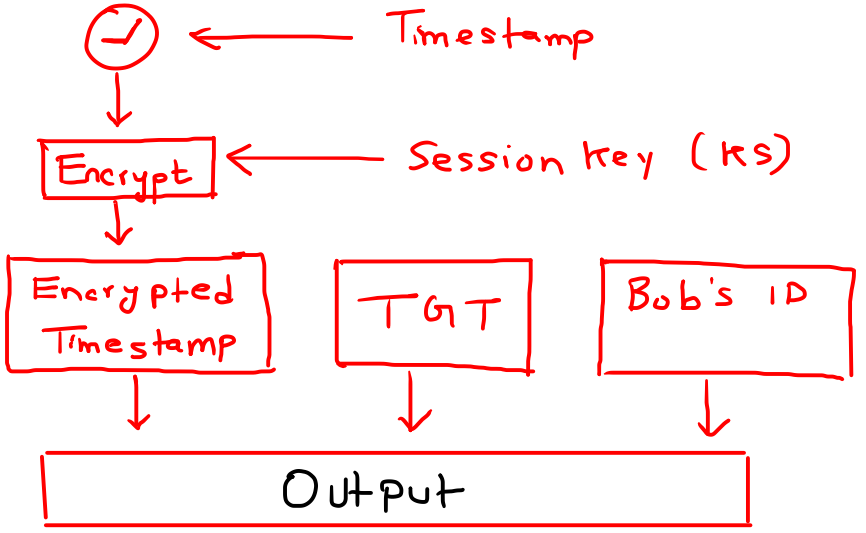
Step 2: Obtaining a Service Granting Ticket (SGT)

1.. Alice would now inform her workstation that she needs to contact the server (Bob), hence she needs a ticket.

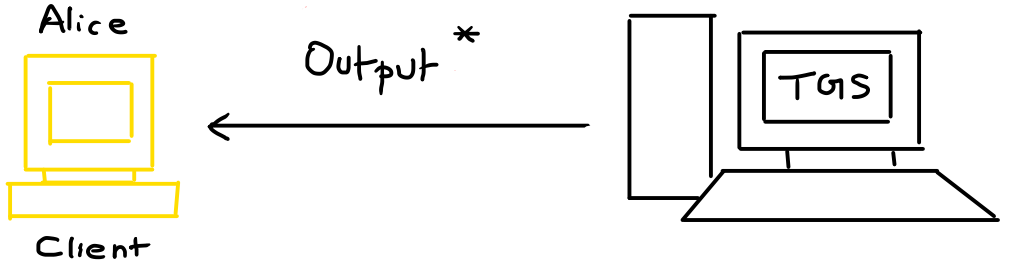
2.. Alice's workstation creates a message intended for the TGS which contains the following items.

a) The TGT obtained in Step 1
b) The id of the server (Bob) whose services she is interested in.
c) The current Timestamp, encrypted with the same session key KS





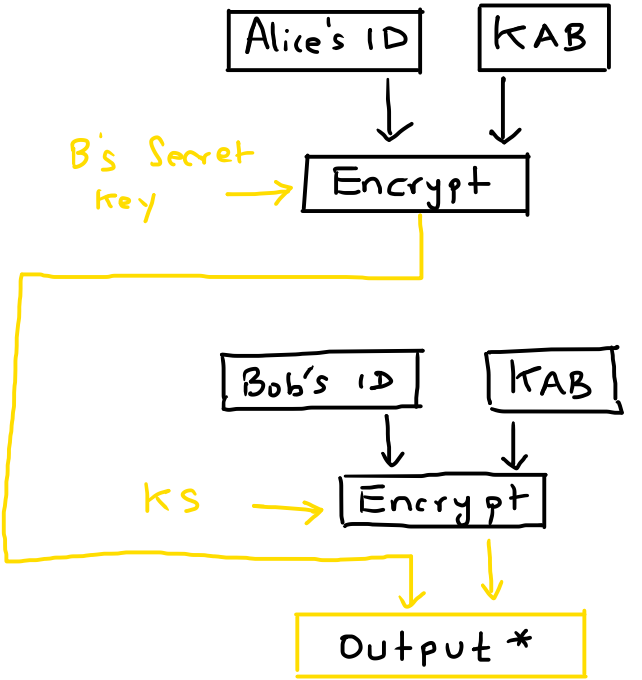
3. Once the TGS is satisfied of the credentials of Alice, the TGS creates a session key KAB, for Alice to have a secure communication with Bob.



4. TGS sends it twice to Alice:

a) once combined with Bob's id and encrypted with the session key KS, and second time

b) combined with Alice's id and encrypted with Bob's secret key.



Step 3: Client Contacts Server for Resources

1. Alice can now send KAB to Bob in order to enter into a session with him/her.

2. Alice simply forwards KAB encrypted with Bob's Secret key (received in the previous step) to Bob.

3. This will ensure that only Bob can access KAB.

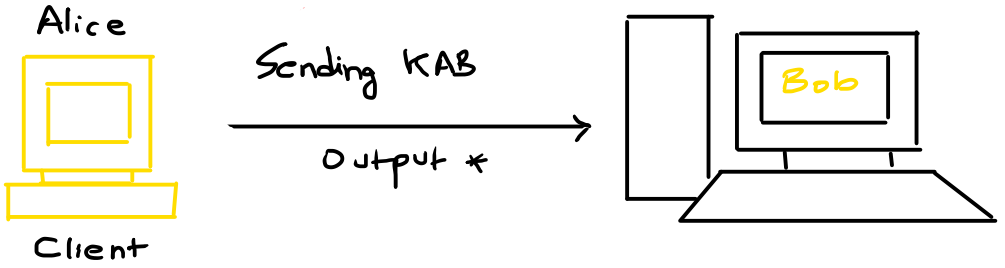
4. Furthermore Alice also attaches a timestamp encrypted with KAB to Bob, to guard against any Replay Attacks.

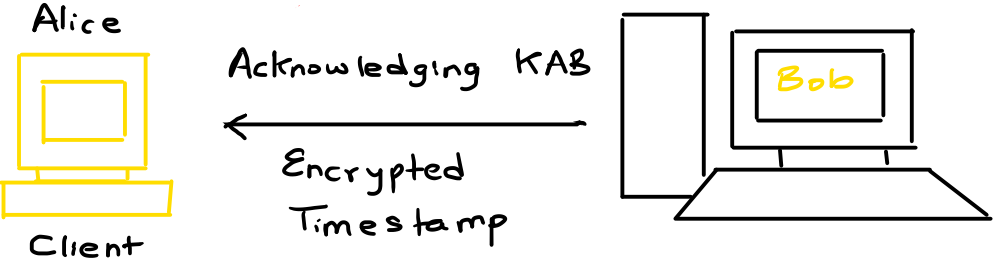
5. Since only Bob has his secret key, he uses it to first obtain the information (Alice's ID + KAB), which he uses to decrypt the encrypted time stamp value.

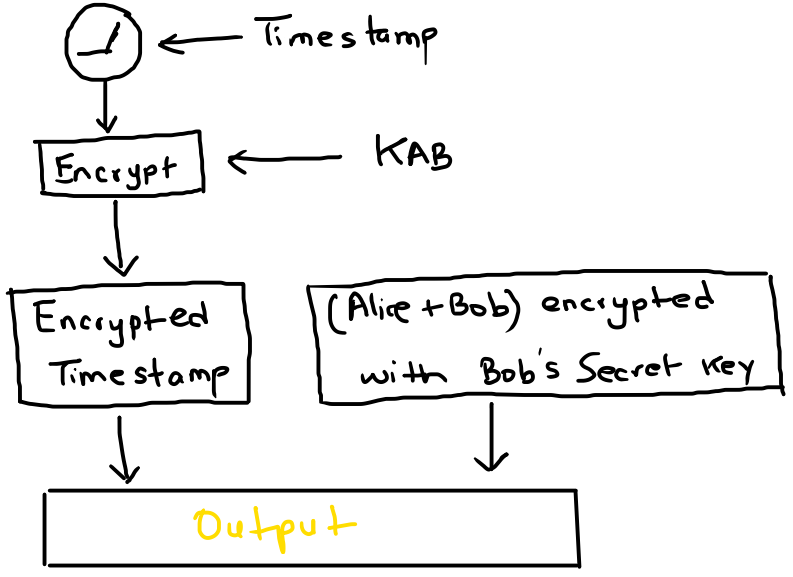
6. Now for Alice to know that Bob ha indeed received KAB correctly or not , Bob increments the timestamp value by 1, encrypts it with KAB and sends it back to Alice.

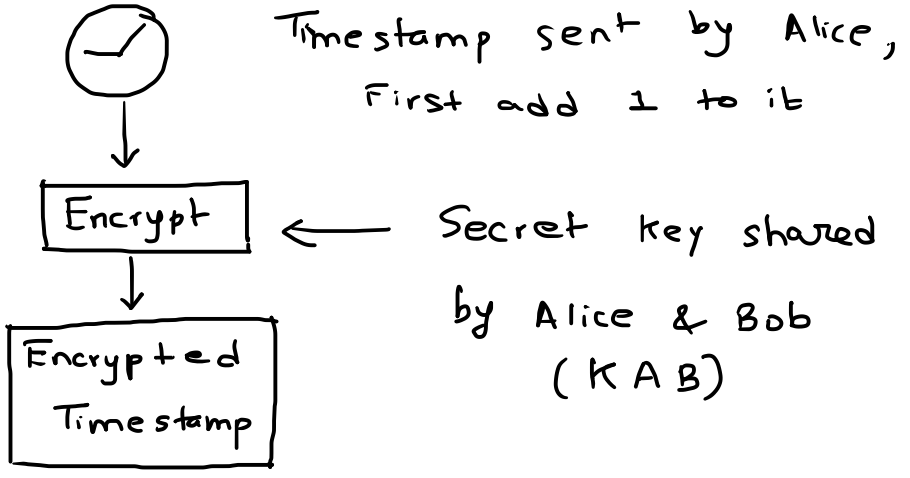
7. Since only Alice and Bob know KAB, Alic can open the packet and verify the incremented Timestamp value.

8. After successful verification, Alice and Bob can now communicate with each other securely using the session key , KAB (SGT)


Alice
Sending KAB
Output *
Bob
Client


Alice
Acknowledging KAB
Encrypted
Timestamp
Bob
Client


Timestamp
Encrypt
KAB
Encrypted
Timestamp
(Alice+Bob) encrypted
with Bob's Secret key
Output


Timestamp sent by Alice,
First add 1 to it
Encrypt
Secret key shared
by Alice & Bob
(KAB)
Encrypted
Timestamp