CS 203 / NetSys 240

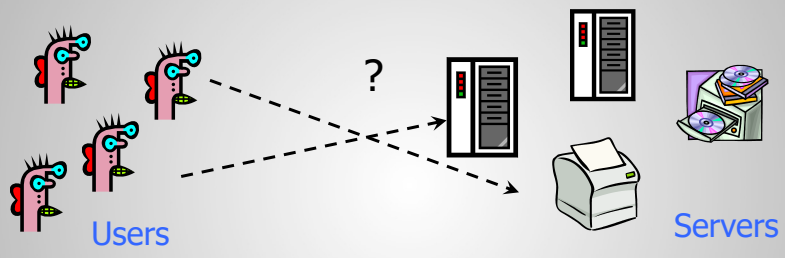
Single Sign-On (Kerberos)

1

Kerberos

2

Many-to-Many Authentication



How do users prove their identities when requesting services from machines on the network?

Naïve solution: every server knows every user's password

- Insecure: break into one server ⇒ compromise all users
- Inefficient: to change password, user must contact every server

slide 3

3

Requirements

- Security
  - Must be secure against attacks by passive eavesdroppers and actively malicious attackers (including rogue users)
- Reliability
  - Must be always available
- Transparency
  - Users should not notice authentication taking place
  - Entering password is OK, if done rarely enough
- Scalability
  - Must handle large numbers of users and servers

slide 4

4

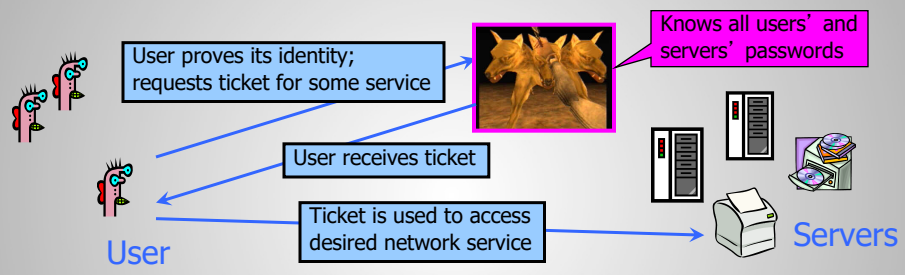
Threats

- User impersonation
  - Malicious user with access to a workstation pretends to be another user from the same workstation
    - Can't trust workstations to verify users' identities
- Network address impersonation
  - Malicious user changes network address of his workstation to impersonate another workstation
    - Can't trust network addresses
- Eavesdropping, tampering and replay
  - Malicious user eavesdrops on, tampers with, or replays, other users' conversations to gain unauthorized access

slide 5

5

Solution: Trusted Third Party

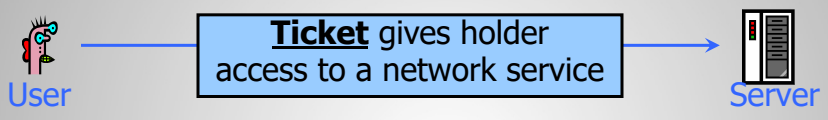


- Trusted authentication service
  - Knows all passwords, can grant access to any server
  - Convenient, but also single point of failure
  - Requires high level of physical security

slide 6

6

What Should a Ticket Look Like?

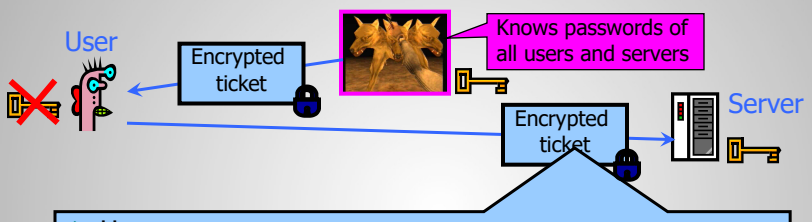


- Ticket cannot include server's password
  - Otherwise, next time user will access server directly without proving its identity to authentication service
- Solution: encrypt some information with a key known to the server, but not to the user!
  - Server can decrypt ticket and verify information
  - User does not learn server key

slide 7

7

What Should a Ticket Include?

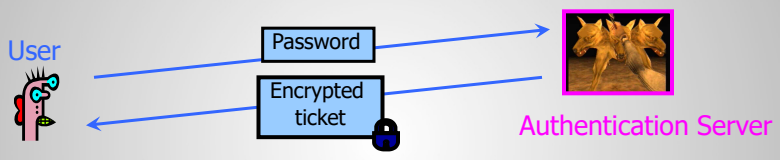


- User name
- Server name
- Address of user's workstation (pc/laptop/etc)
  - Otherwise, a different user on another workstation can steal the ticket and use it to gain access to the server
  - But same user moving to a different workstation would need a new ticket for same server
- Ticket lifetime
- A few other things (e.g., session key)

slide 8

8

How to authenticate initially?



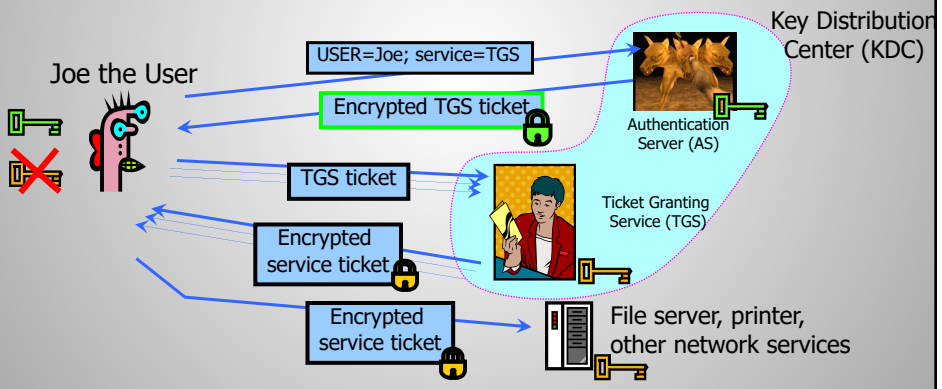
- Insecure: passwords are sent in plaintext
  - Eavesdropper can steal password and impersonate user
- Inconvenient: need to send the password each time to obtain the ticket for any network service
  - Separate authentication for email, printing, etc.

slide 9

9

Two-Step Authentication

- Prove identity **once** to obtain special TGT: TGS ticket
- Use TGT to get tickets for any network service



slide 10

10

Still Not Good Enough

- Ticket hijacking
  - Malicious user may steal the service ticket of another user on the same workstation and use it
    - IP address verification does not help
  - Servers must verify that the user who is presenting the ticket is the same user to whom the ticket was issued
- No server authentication
  - Attacker may mis-configure the network so that it receives messages addressed to a legitimate server
    - Capture private information from users and/or deny service
  - Servers must prove their identity to users

slide 11

11

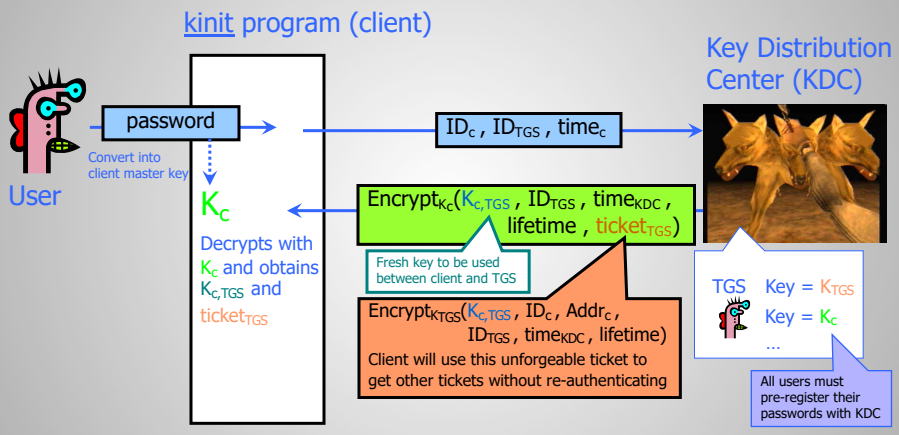
Symmetric Keys in Kerberos

- K_C is long-term key of client C
  - Derived from user C's password
    - user = human, client = Kerberos client-side sw
  - Known to client and key distribution center (KDC)
- K_{TGS} is long-term key of TGS
  - Known to KDC and ticket granting service (TGS)
- K_V is long-term (strong) key of network service V
  - Known to V and TGS; separate key for each service
- $K_{C,TGS}$ is short-term key shared between C and TGS
  - Created by KDC, known to C and TGS
  - Conveyed in TGT
- $K_{C,V}$ is shorter-term key shared between C and V
  - Created by TGS, known to C and V

slide 12

12

Single Sign-On (SSO) Authentication

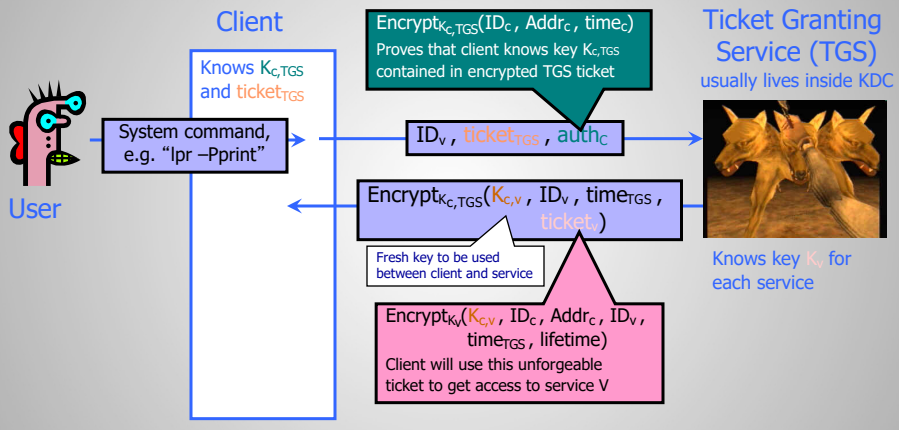


- Client only needs to obtain a TGT (TGS ticket) **once** per login session, e.g., daily = every morning
  - Ticket is encrypted; client cannot forge it or tamper with it

slide 13

13

Obtaining a Service Ticket

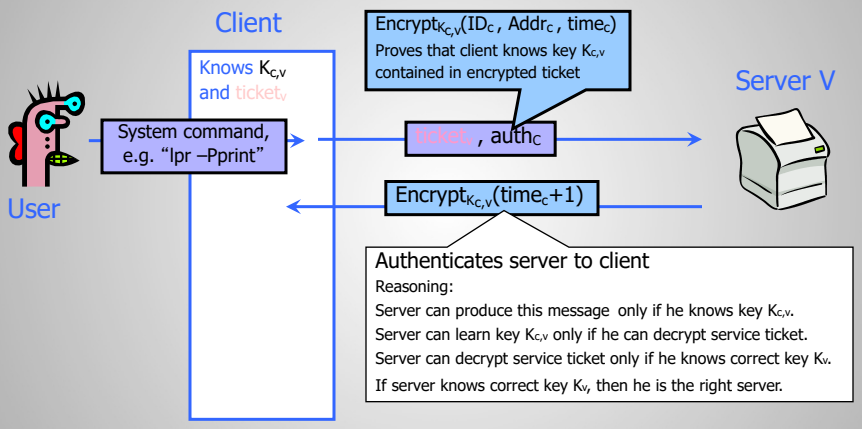


- Client uses TGS ticket to obtain a service ticket and a short-term key for each network service
  - Encrypted, unforgeable ticket per service (printer, email, etc.)

slide 14

14

Obtaining Service

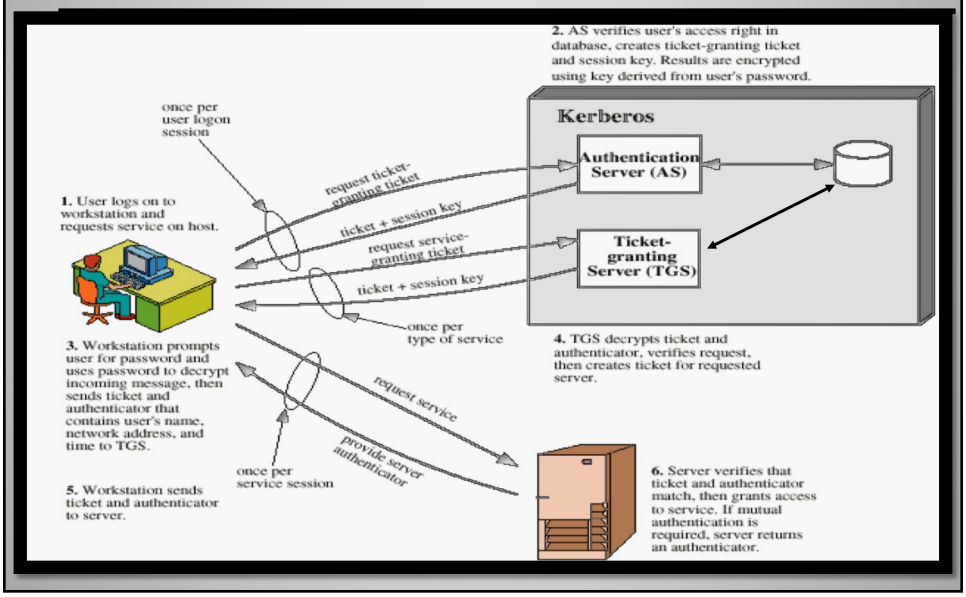


- For each service request, client uses the short-term key for that service – $K_{C,V}$ – and the ticket it received from TGS

slide 15

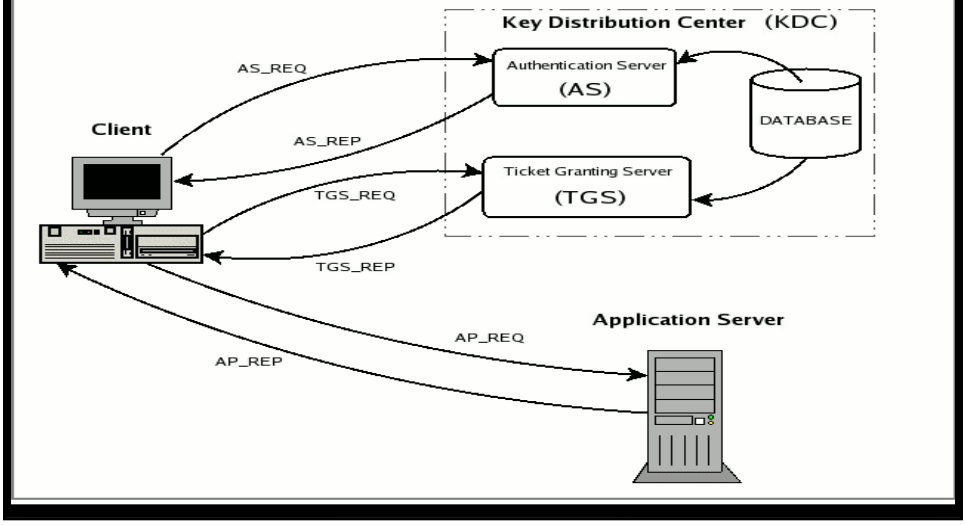
15

Summary of Kerberos



16

Summary of Kerberos



17

Kerberos Message Formats (v4)

AS_REQ = ($Principal_C$, krbtgt/REALM@REALM , IP_list , Lifetime)

TGT = ($Principal_C$, krbtgt/REALM@REALM , IP_list , $Timestamp_{AS}$, Lifetime , $K_{C,TGS}$)

AS_REP = {krbtgt/REALM@REALM , $Timestamp_{AS}$, Lifetime , $K_{C,TGS}$ } K_C , { TGT } K_{TGS}

Auth1 = { $Principal_C$, $Timestamp_C$, Checksum1 } $K_{C,TGS}$

TGS_REQ = ($Principal_V$, Lifetime , Auth1), { TGT } K_{TGS}

T_V = ($Principal_C$, $Principal_V$, IP_list , $Timestamp_{TGT}$, Lifetime , $K_{C,V}$)

TGS_REP = { $Principal_V$, $Timestamp_{TGT}$, Lifetime , $K_{C,V}$ } $K_{C,TGS}$, { T_V }K_V

Auth2 = { $Principal_C$, $Timestamp'_C$, Checksum2 } $K_{C,V}$

AP_REQ = Auth2, { T_V } K_V

AS_REP = optional... {$Timestamp'_C$ + 1 } $K_{C,V}$

slide 18

18

Kerberos in Large Networks

- One KDC isn't enough for large networks (why?)
- Network is divided into Kerberos realms
  - KDCs in different realms have different key databases
- To access a service in another realm, user must…
  - Get ticket for home-realm TGS from home-realm KDC
  - Get ticket for remote-realm TGS from home-realm TGS
    - As if remote-realm TGS were just another network service
  - Get ticket for remote service from that realm's TGS
  - Use remote-realm ticket to access service
  - N(N-1)/2 keys needed for full N-realm interoperation
    - Each KDC shares a key with every other KDC

slide 19

19

Important Ideas in Kerberos

- Short-term session keys
  - Long-term secrets used only to derive short-term keys
  - Separate session key for each user-server pair
- Proofs of identity are based on authenticators
  - Client encrypts his identity, address and current time using a short-term session key shared with server
    - Also prevents replays (if clocks are globally synchronized)
  - Server learns this key separately (by decrypting an encrypted ticket that client can't decrypt) and verifies client identity
- Symmetric cryptography only

slide 20

20

Kerberos Version 5

Current: Release 1.21.3, 06/26/24

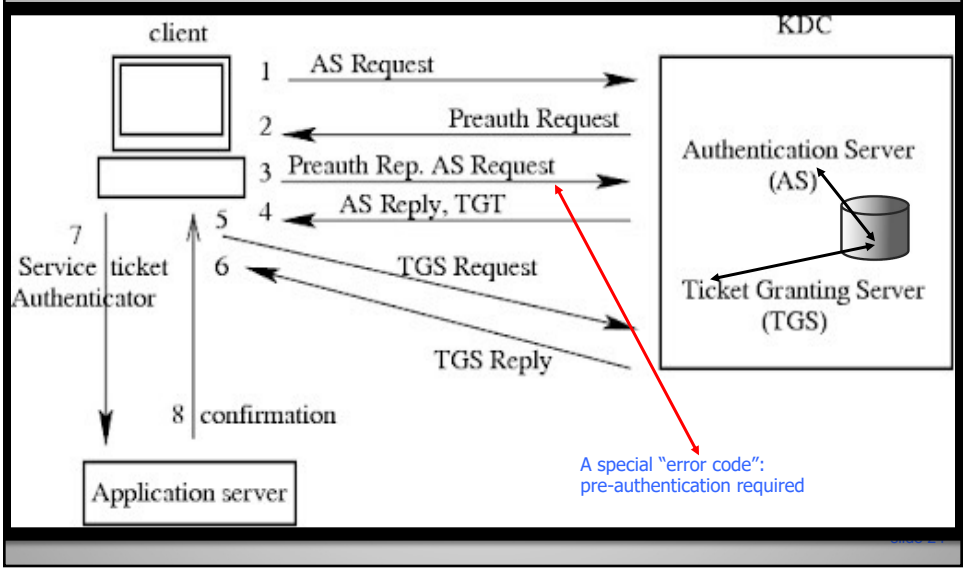
- Preauthentication in initial AS-REQ message
- Client-Server authentication
  - Separate subkey for each client-server session instead of re-using the session key contained in the ticket
  - Authentication via subkeys
- Authentication forwarding
  - Servers can access other servers on client's behalf
- Realm hierarchies for inter-realm authentication
- Richer ticket functionality
- Explicit integrity checking + standard CBC mode
- Multiple encryption schemes, not just DES

Pre-Authentication

- In Kerberos v4, anyone can generate AS_REQs for any user/client and obtain many AS_REQ/AS_REP pairs of known plaintext/ciphertext
- Thus, off-line password guessing attacks are easy!
- Pre-Authentication (Kerberos v5):

- PADATA = $\{ \text{Timestamp}_C \}K_c$ is required in AS_REQ message
- AS_REP is sent only if AS can decrypt PADATA and validate $\text{Timestamp}'_C$
- Trade-offs? Pre-authentication v. no pre-authentication?

Summary of Kerberos (with pre-authentication)



24

Kerberos Extensions

- Pre-authentication
- PKInit
- PKCross

slide 25

25

Practical Uses of Kerberos

- Email, FTP, network file systems and many other applications have been kerberized
  - Use of Kerberos is transparent for the end user
  - Transparency is important for usability!
- Local authentication
  - login and su in OpenBSD
- Authentication for network protocols
  - rlogin, rsh, telnet
- Secure windowing systems
  - xdm, kx