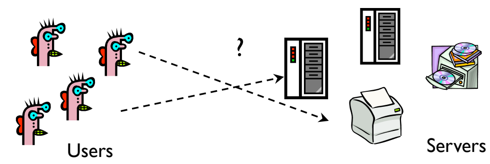
CS 4803
Computer and Network Security

Alexandra (Sasha) Boldyreva

Kerberos

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: compromise of one server is enough to compromise all users
- Inefficient: to change his password, user must contact every server

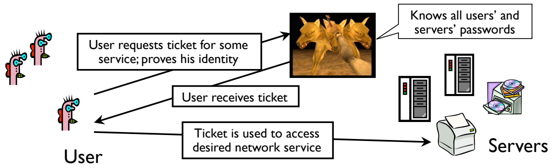
Requirements

- Security
  - Against attacks by passive eavesdroppers and actively malicious users
- Reliability
- Transparency
  - Users shouldn't be aware of authentication taking place
  - Entering password is OK, if done rarely
- Scalability
  - Large number of users and servers

Recall the threats

- User impersonation
  - Malicious user with access to a workstation pretends to be another user from the same workstation
- Network address impersonation
  - Malicious user changes network address of his workstation to impersonate another workstation
- Eavesdropping, tampering and replay
  - Malicious user eavesdrops on, tampers with or replays other users' conversations to gain unauthorized access

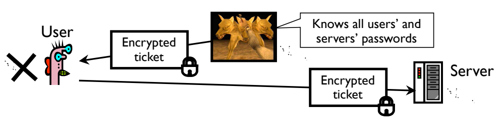
Solution: Trusted Third Party



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

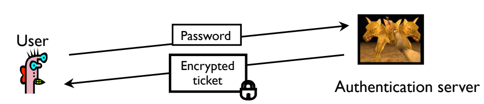
What is a ticket for?

What Should a Ticket Include?



- User name
- Server name
- Address of user's workstation
  - Otherwise, a user on another workstation can steal the ticket and use it to gain access to the server
- Ticket lifetime
- A few other things (e.g., session key)

How Is Authentication Done?



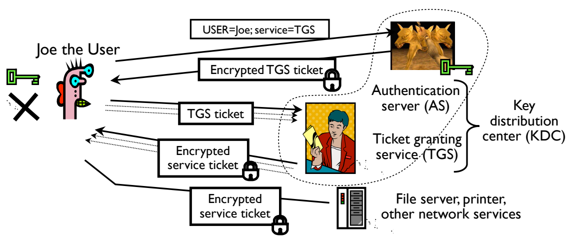
- Insecure: passwords are sent in plaintext
  - Eavesdropper can steal the password and later impersonate the user to the authentication server
- Inconvenient: need to send the password each time to obtain the ticket for any network service
  - Separate authentication for email, printing, etc.

Solution: Two-Step Authentication

Prove identity **once** to obtain special TGS ticket

Instead of password, use key derived from password

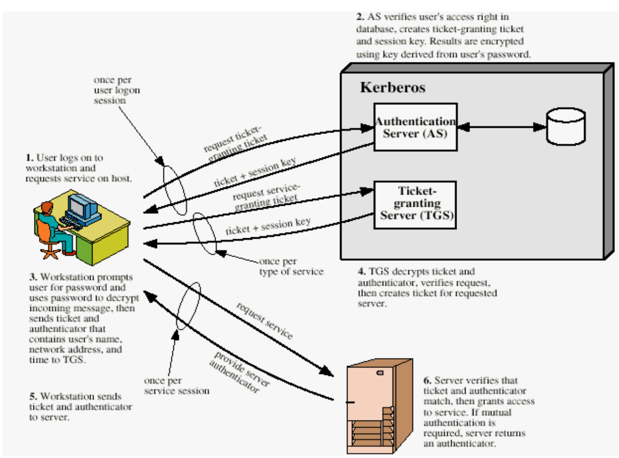
Use TGS to get tickets for many network services



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 be able to verify that the user who is presenting the ticket is the same user to whom the ticket was issued
- No server authentication
  - Attacker may misconfigure the network so that he receives messages addressed to a legitimate server
    - Capture private information from users and/or deny service
  - Servers must prove their identity to users

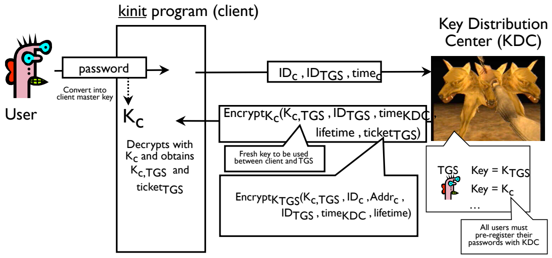
Summary of Kerberos



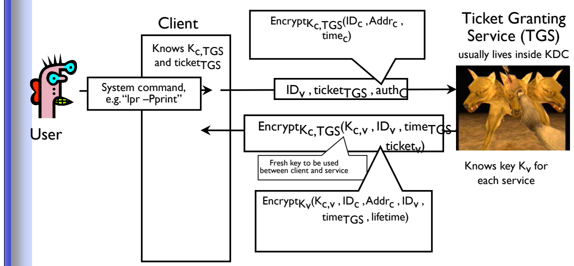
Symmetric Keys in Kerberos

- K_C is long-term key of client C
  - Derived from user's password
  - Known to client and key distribution center KDC
- K_{TGS} is long-term key of ticket granting service TGS
  - Known to KDC and TGS
- K_V is long-term key of network service V
  - Known to V and TGS; separate key for each service
- $K_{C,TGS}$ is short-term key between C and TGS
  - Created by KDC, known to C and TGS
- $K_{C,V}$ is short-term key betwen C and V
  - Created by TGS, known to C and V

"Single Logon" Authentication

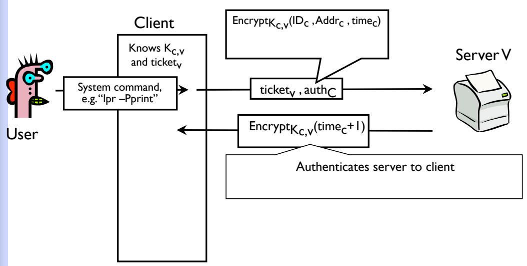


Obtaining A Service Ticket



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

Obtaining Service



- For each service request, client uses the short-term key for that service and the ticket he received from TGS

Kerberos in Large Networks

- One KDC isn't enough for large networks (why?)
- Network is divided into realms
  - KDCs in different realms have different key databases
- To access a service in another realm, users 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 key exchanges for full N-realm interoperation

Important Ideas in Kerberos

- Use of short-term session keys
  - Minimize distribution and use of long-term secrets; use them only to derive short-term session keys
  - Separate short-term key for each user-server pair
    - But multiple user-server sessions reuse the same key!
- Proofs of identity are based on authenticators
  - Client encrypts his identity, address and current time using a short-term session key
    - Also prevents replays (if clocks are globally synchronized)
  - Server learns this key separately (via encrypted ticket that client can't decrypt) and verifies user's identity

Problematic Issues

- Password dictionary attacks on client master keys
- Replay of authenticators
  - 5-minute lifetimes long enough for replay
  - Timestamps assume global, secure synchronized clocks
  - Challenge-response would be better
  - Encryption is used for authentication
- Same user-server key used for all sessions
- Homebrewed PCBC mode of encryption
  - Tries to combine integrity checking with encryption
- Extraneous double encryption of tickets
- No ticket delegation
  - Printer can't fetch email from server on your behalf

Kerberos Version 5

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

Practical Uses of Kerberos

- Email, FTP, SSH, 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