Kerberos

George Louthan

April 14, 2010

Challenge: How to prove identity...

- over the network,
- bidirectionally,
- using shared secrets (passwords),
- when the link is not secure,
- without divulging your passwords,
- in an environment with dozens of servers and hundreds of clients (or more)?
- The answer is, perhaps unsurprisingly, Kerberos.

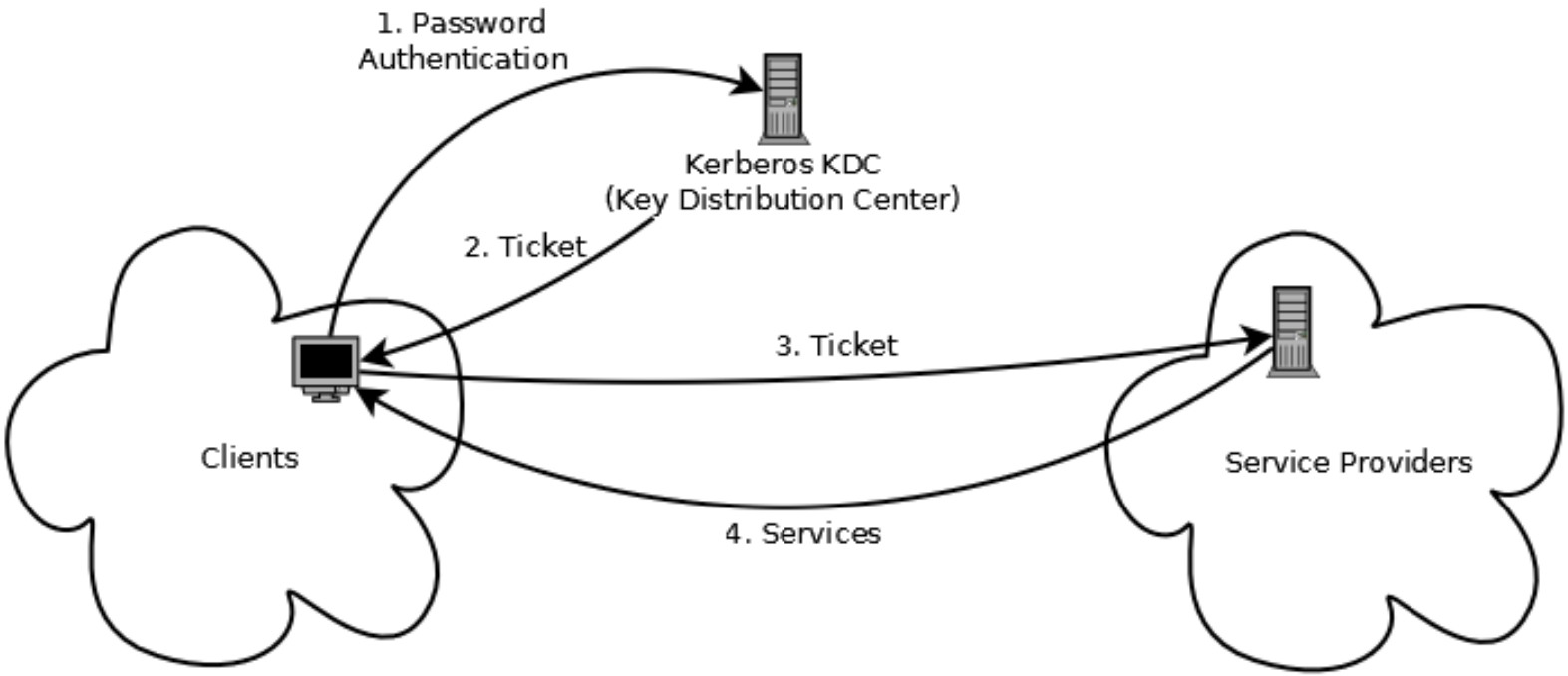
Kerberos Properties

- Provides mutual (bidirectional) authentication
- Uses shared secrets
- Secure even over insecure links (no replay or eavesdropping)
- Provides an authentication infrastructure (single sign-on)
- Requires a trusted third party
- Requires time synchronization
- Classified as a munition until about 10 years ago

Kerberos Infrastructure (Simplified)

- The goal: use a central server to authenticate a client to various services
- Introducing a new credential: the ticket
- Use password to authenticate to a central server - receive ticket
- Use tickets to authenticate to service providers
- Never use shared secret to authenticate to services
- Only the intended service can read the ticket

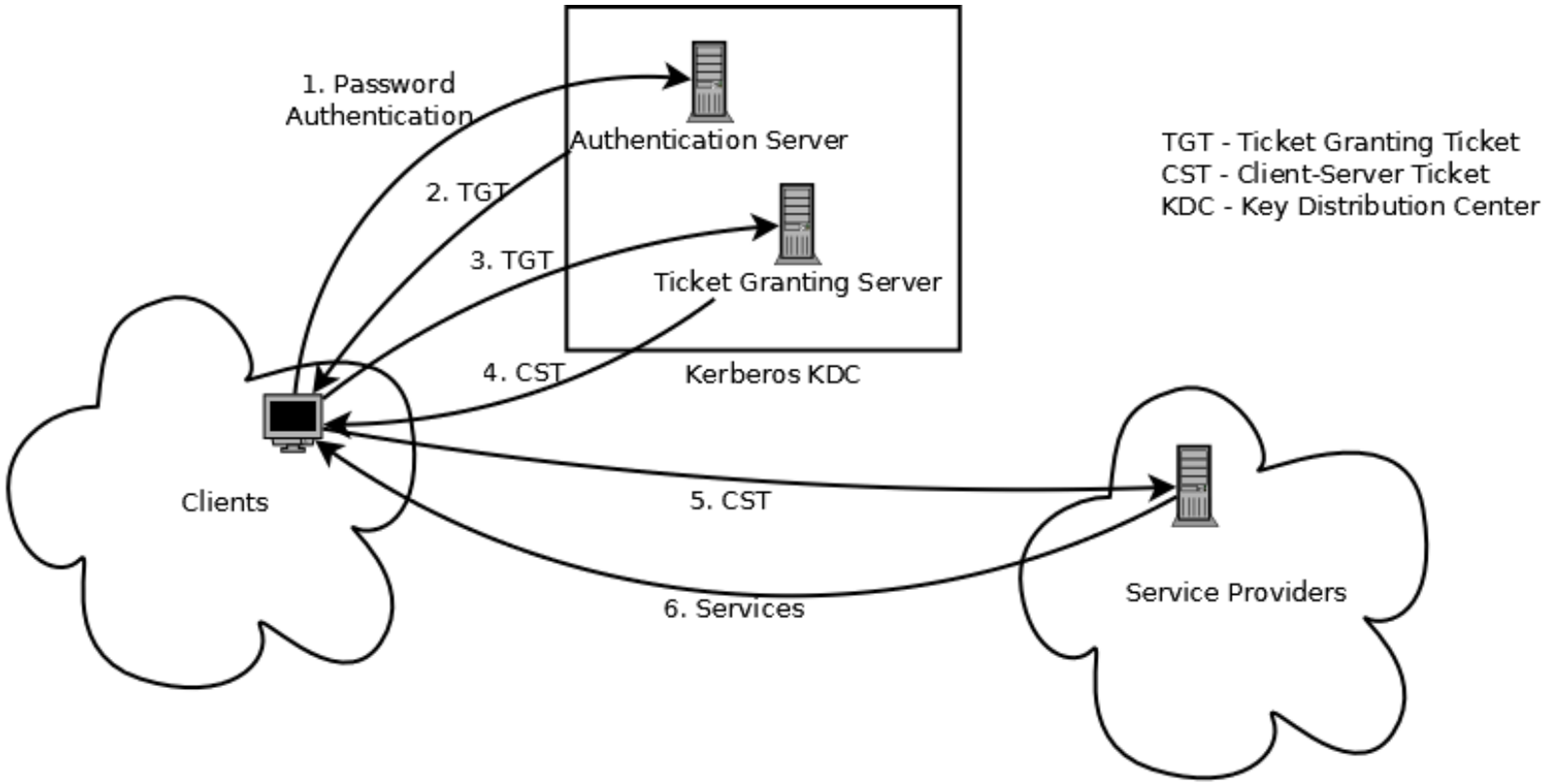
Kerberos Infrastructure (Simplified)



Tickets

- Used to delegate authentication from the KDC to the client/server pair
- Tickets for services expire quickly, one ticket at a time per client per service
- To prevent having to enter a password constantly, the architecture is somewhat more nuanced
  - The KDC has two parts: Authentication Server (AS) and Ticket Granting Server (TGS)
  - Ticket Granting is treated as just another service, requiring a ticket to be accessed
  - The AS grants and the TGS accepts a Ticket Granting Ticket (TGT), which takes a long time to expire
  - The TGS grants and all other servers accept client-server tickets.

Kerberos Infrastructure (Revised)



Authentication Mechanism for Kerberos: Basis

- Kerberos's cryptographic authentication mechanism is based on the symmetric Needham-Schroeder protocol
- Needham-Schroeder protocol uses shared keys (between clients and a third party) to negotiate a session key (between the clients)
- NS is insecure, but Kerberos fixes this with timestamps

Needham-Schroeder Protocol

A: Alice

B: Bob

S: Trusted third party / server

Known keys: K_{AS}, K_{BS}; Nonces: N_A, N_B. New key: K_{AB}.

$A \rightarrow S : A, B, N_A$

$S \rightarrow A : \{N_A, K_{AB}, B, \{K_{AB}, A\}_{K_{BS}}\}_{K_{AS}}$

$\mathbf{A \rightarrow B : \{K_{AB}, A\}_{K_{BS}}}$

$B \rightarrow A : \{N_B\}_{K_A B}$

$A \rightarrow B : \{N_B - 1\}_{K_A B}$

Needhan-Schroeder is Vulnerable

- No way to recover if K_{AB} is compromised
- Need a way to ensure that keys are fresh
- Kerberos uses timestamps

Using Needham-Schroeder to get a TGT (simplified)

A: User
TGS: Kerberos Ticket Granting Server
AS: Kerberos Authentication Server
AS knows K_{AS} (generated from A's password), K_{TGS} (TGS secret key), K_{SK} (a session key for A and TGS)

$A \rightarrow S : A, B, N_A$
$S \rightarrow A : \{N_A, K_{SK}, B, \{\mathbf{K_{SK}}, \mathbf{A}, \mathbf{validtime}\}_{\mathbf{K_{TGS}}}\}_{K_{AS}}$

Where do the secret keys come from?

- The protocol involves shared keys betweek users and servers
- These come from the user's password hashes
  - AS uses a known method to generate a key from the client user's password
  - AS uses this key to encrypt messages to the client
  - Client tries the same method with the password, uses the resulting key to decrypt
  - Password never sent across the network

Kerberos Deployment

- Kerberos is used in infrastructures such as Microsoft Active Directory
- More complicated deployments than described above are possible
- Each KDC designates a *realm* (Correspond to AD domains)
- Cross-realm authentication is possible if the KDCs share a key
- Generally a central time server is required
- Security of the Kerberos protocol depends upon security of the time server
- No administration (e.g. password setting or changing) protocol specified

References

- Bill Bryant and Theodore Ts'o. "Designing an Authentication System: a Dialogue in Four Scenes". http://www.mit.edu/ kerberos/dialogue.html
- Ross Anderson and Roger Needham. "Programming Satan's Computer".
- IETF RFC 4120. "The Kerberos Network Authentication Service (V5)" http://tools.ietf.org/html/rfc4120