Kerberos

Overview

- authentication service designed for use in a **distributed** environment.
- makes use of a trusted **third-party authentication** service
  - enables clients and servers to establish authenticated communication.
- developed as part of Project Athena at MIT
- Addresses the following threats
  - user pretend to be another user operating from that workstation
  - user may alter the network address and impersonate the workstation
  - eavesdrop on exchanges and use a replay attack for gaining entry or disrupt
- provides a centralized authentication server to authenticate users to servers and servers to users
- relies exclusively on symmetric encryption, making no use of public-key encryption
- two versions in use 4 & 5

Motivation / Requirements (SRTS)

- **Secure**
  - A network eavesdropper should not be able to obtain the necessary information to impersonate a user.
  - strong enough such that a potential opponent does not find it to be the weak link.
- Reliable
  - should be highly reliable
  - should employ a distributed server architecture,
    - one system able to back up another.
- Transparent
  - user should not be aware that authentication is taking place
    - beyond the requirement to enter a password.
- Scalable
  - capable of supporting large numbers of clients and servers
    - modular, distributed architecture.

Kerberos Encryption Techniques

Simple Kerberos Dialogue

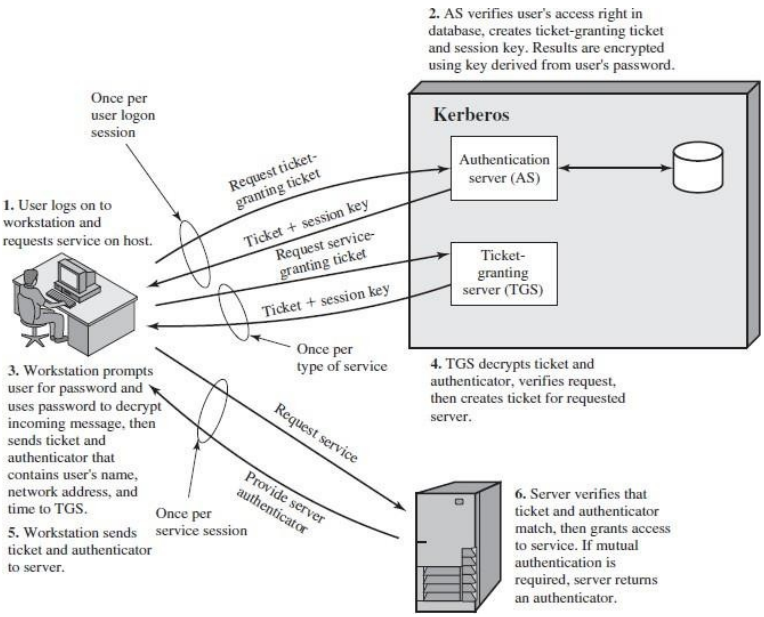
(1) C → AS: $ID_C \| P_C \| ID_V$
(2) AS → C: $Ticket$
(3) C → V: $ID_C \| Ticket$

$Ticket = E(K_v, [ID_C \| AD_C \| ID_V])$

K_v = secret encryption key shared by AS and V

C = client
AS = authentication server
V = server
ID_C = identifier of user on C

Kerberos Overview



Kerberos Version 4 Message Exchanges

i) Authentication Service Exchange to obtain ticket-granting ticket

1) Client requests ticket-granting ticket

- User logs on to workstation and requests service on host

$\mathbf{C \rightarrow AS}\quad ID_c \| ID_{tgs} \| TS_1$

ID_C	Tells AS identity of user from this client.
ID_{tgs}	Tells AS that user requests access to TGS.
TS_1	Allows AS to verify that client's clock is synchronized with that of AS.

2) AS returns ticket-granting ticket

- AS verifies user's access right in database, creates ticket-granting ticket and session key. Results are encrypted using key derived from user's password

$\mathbf{AS \rightarrow C}\quad \mathrm{E}(K_c, [K_{c,tgs} \| ID_{tgs} \| TS_2 \| Lifetime_2 \| Ticket_{tgs}])$

K_c	Encryption is based on user's password, enabling AS and client to verify password, and protecting contents of message (2).
$K_{c,tgs}$	Copy of session key accessible to client created by AS to permit secure exchange between client and TGS without requiring them to share a permanent key.
ID_{tgs}	Confirms that this ticket is for the TGS.
TS_2	Informs client of time this ticket was issued.
$Lifetime_2$	Informs client of the lifetime of this ticket.
$Ticket_{tgs}$	Ticket to be used by client to access TGS.

ii) Ticket-Granting Service Exchange to obtain service-granting ticket

3) Client requests service-granting ticket

- Workstation prompts user for password and uses password to decrypt incoming message, then sends ticket and authenticator that contains user's name, network address, and time to TGS

$$\mathbf{C \rightarrow TGS} \quad ID_v \| Ticket_{tgs} \| Authenticator_c$$

ID_V	Tells TGS that user requests access to server V.
$Ticket_{tgs}$	Assures TGS that this user has been authenticated by AS.
$Authenticator_c$	Generated by client to validate ticket.

4) TGS returns service-granting ticket

- TGS decrypts ticket and authenticator, verifies request, then creates ticket for requested server

$$\mathbf{TGS \rightarrow C} \quad E(K_{c,tgs}, [K_{c,v} \| ID_v \| TS_4 \| Ticket_v])$$

$$Ticket_{tgs} = E(K_{tgs}, [K_{c,tgs} \| ID_C \| AD_C \| ID_{tgs} \| TS_2 \| Lifetime_2])$$

$$Ticket_v = E(K_v, [K_{c,v} \| ID_C \| AD_C \| ID_v \| TS_4 \| Lifetime_4])$$

$$Authenticator_c = E(K_{c,tgs}, [ID_C \| AD_C \| TS_3])$$

iii) Client/Server Authentication Exchange to obtain service

5) Client requests service

- Workstation sends ticket and authenticator to server

$$\mathbf{C \rightarrow V} \quad Ticket_v \| Authenticator_c$$

$Ticket_V$	Assures server that this user has been authenticated by AS.
$Authenticator_c$	Generated by client to validate ticket.

6) Optional authentication of server to client

- Server verifies that ticket and authenticator match, then grants access to service. If mutual authentication is required, server returns an authenticator

$$\mathbf{V \rightarrow C} \quad E(K_{c,v}, [TS_5 + 1]) \text{ (for mutual authentication)}$$

$$Ticket_v = E(K_v, [K_{c,v} \| ID_C \| AD_C \| ID_v \| TS_4 \| Lifetime_4])$$

$$Authenticator_c = E(K_{c,v}, [ID_C \| AD_C \| TS_5])$$

Kerberos Realms And Multiple Kerberi

Kerberos realm

- a set of managed nodes that share the same Kerberos database.
- database resides on the Kerberos master computer system, kept in a physically secure room.
  - A read-only copy on other Kerberos computer systems.
- all changes to the database must be made on the master computer system.
  - requires the Kerberos master password

Kerberos principal

- a service or user that is known to the Kerberos system.
- Each Kerberos principal is identified by its principal name.
- Principal names consist of three parts: a service or user name, an instance name, and a realm name

Requirements

1. The Kerberos server must have the user ID and hashed passwords of all participating users in its database. All users are registered with the Kerberos server.

2. The Kerberos server must share a secret key with each server. All servers are registered with the Kerberos server

3. The Kerberos server in each interoperating realm shares a secret key with the server in the other realm. The two Kerberos servers are registered with each other

Exchanges

(1) $C \rightarrow AS$: $ID_c \| ID_{tgs} \| TS_1$

(2) $AS \rightarrow C$: $E(K_c, [K_{c,tgs} \| ID_{tgs} \| TS_2 \| Lifetime_2 \| Ticket_{tgs}])$

(3) $C \rightarrow TGS$: $ID_{tgsrem} \| Ticket_{tgs} \| Authenticator_c$

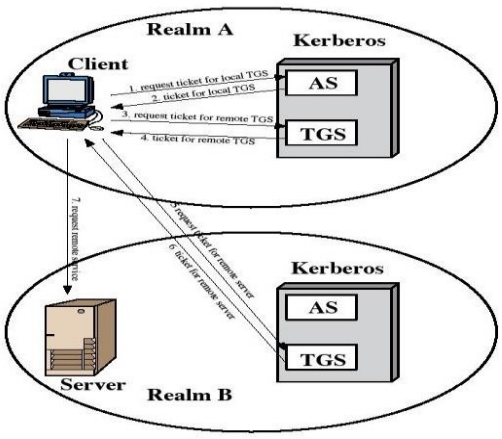
(4) $TGS \rightarrow C$: $E(K_{c,tgs}, [K_{c,tgsrem} \| ID_{tgsrem} \| TS_4 \| Ticket_{tgsrem}])$

(5) $C \rightarrow TGS_{rem}$: $ID_{vrem} \| Ticket_{tgsrem} \| Authenticator_c$

(6) $TGS_{rem} \rightarrow C$: $E(K_{c,tgsrem}, [K_{c,vrem} \| ID_{vrem} \| TS_6 \| Ticket_{vrem}])$

(7) $C \rightarrow V_{rem}$: $Ticket_{vrem} \| Authenticator_c$

Request for Service in another Realm



Environmental shortcomings of Kerberos version 4

Encryption system dependence

- Export restriction on DES as well as doubts about

the strength of DES In version 5,

- ciphertext is tagged with an encryption type identifier so that any encryption technique may be used
- Encryption keys are tagged with a type and a length,
  - allowing the same key to be used in different algorithms
  - allowing the specification of different variations on a given algorithm.

Internet protocol dependence:

- Version 4 requires the use of Internet Protocol (IP) addresses. Other address types, such as the ISO network address, are not accommodated.
- Version 5 network addresses are tagged with type and length, allowing any network address

Message byte ordering

- In version 4, the sender of a message employs a byte ordering of its Own
- In version 5, all message structures are defined using Abstract Syntax Notation One (ASN.1) and Basic Encoding Rules (BER), which provide an unambiguous byte ordering.

Ticket lifetime:

- Lifetime values in version 4 are encoded in an 8-bit quantity in units of five minutes.
  - 1280 minutes, or over 21 hours
- In version 5, tickets include an explicit start time and end time, allowing tickets with arbitrary lifetimes

Authentication forwarding:

- Version 4 does not allow credentials issued to one client to be forwarded to some other host and used by some other client
- Version 5 provides this capability

Interrealm authentication:

- In version 4, interoperability among N realms requires on the order of N2 Kerberos-to-Kerberos relationships
- Version 5 supports a method that requires fewer relationships

Technical deficiencies

Double encryption

- second encryption is not necessary and is computationally wasteful

PCBC encryption

- Version 4 uses nonstandard mode of DES known as propagating cipher block chaining (PCBC)
- Version 5 provides explicit integrity mechanisms, allowing the standard CBC mode to be used

- a checksum or hash code is attached to the

message prior to encryptionSession keys

Password attacks

- Both versions are vulnerable to a password attack
- Version 5 does provide a mechanism known as Preauthentication

Kerberos Version 5 Message Exchanges

Authentication Service Exchange to obtain ticket-granting ticket

$$
\begin{aligned}
&(1)\quad C \rightarrow AS \quad \text{Options} \| ID_c \| Realm_c \| ID_{tgs} \| Times \| Nonce_1 \\
&(2)\quad AS \rightarrow C \quad Realm_C \| ID_C \| Ticket_{tgs} \| E(K_c, [K_{c,tgs} \| Times \| Nonce_1 \| Realm_{tgs} \| ID_{tgs}]) \\
&\qquad\qquad Ticket_{tgs} = E(K_{tgs}, [Flags \| K_{c,tgs} \| Realm_c \| ID_C \| AD_C \| Times])
\end{aligned}
$$

Ticket-Granting Service Exchange to obtain service-granting ticket

$$
\begin{aligned}
&(3)\quad C \rightarrow TGS \quad \text{Options} \| ID_v \| Times \| Nonce_2 \| Ticket_{tgs} \| Authenticator_c \\
&(4)\quad TGS \rightarrow C \quad Realm_c \| ID_C \| Ticket_v \| E(K_{c,tgs}, [K_{c,v} \| Times \| Nonce_2 \| Realm_v \| ID_v]) \\
&\qquad\qquad Ticket_{tgs} = E(K_{tgs}, [Flags \| K_{c,tgs} \| Realm_c \| ID_C \| AD_C \| Times]) \\
&\qquad\qquad Ticket_v = E(K_v, [\text{Flags} \| K_{c,v} \| \text{Realm}_c \| \text{ID}_C \| \text{AD}_C \| \text{Times}]) \\
&\qquad\qquad Authenticator_c = E(K_{c,tgs}, [ID_C \| Realm_c \| TS_1])
\end{aligned}
$$

Client/Server Authentication Exchange to obtain service

$$
\begin{aligned}
&(5)\quad C \rightarrow V \quad \text{Options} \| \text{Ticket}_v \| \text{Authenticator}_c \\
&(6)\quad V \rightarrow C \quad E_{K_{c,v}}[TS_2 \| \text{Subkey} \| \text{Seq}\neq] \\
&\qquad\qquad \text{Ticket}_v = E(K_v, [\text{Flag} \| K_{c,v} \| \text{Realm}_c \| \text{ID}_C \| \text{AD}_C \| \text{Times}]) \\
&\qquad\qquad Authenticator_c = E(K_{c,v}, [ID_C \| Relam_c \| TS_2 \| Subkey \| Seq\neq])
\end{aligned}
$$

New Elements

- **Realm**: Indicates realm of user
- **Options**: Used to request that certain flags be set in the returned ticket
- **Times**: Used by the client to request the following time settings in the ticket:
  - **from**: the desired start time for the requested ticket
  - **till**: the requested expiration time for the requested ticket
  - **rtime**: requested renew-till time
- **Nonce**: A random value to be repeated in message (2) to assure that the response is fresh