Distributed Systems Security

Authentication Practice - 1

Prof. Steve Wilbur
s.wilbur@cs.ucl.ac.uk

Lecture Objectives

- Identify authentication needs for client-server systems
- Using variants of principles develop suitable protocols
- Examine Kerberos protocol
Scenario

Let us consider a computing environment where each user accesses services via a workstation on their desk.

A number of servers provide services such as filing, computation, printing and mail.

Servers need to protect their resources against unauthorised use, so users need to be authenticated.

Approaches

We can use one of three broad approaches:

- Let client workstation authenticate user and server trusts this as basis for access control and resource management.
- As above, but require the client machine to authenticate itself with the server.
- Require the user to prove their identity to each service used - might also expect server to prove its identity to clients.

Particularly as scale of system grows, the third approach, including partitioning of user space, will be needed.
Kerberos

The need for such authentication arose in Project Athena at MIT in the 1980s, and Kerberos was developed.

Principal requirements were:

- **Secure**: Kerberos should not provide an easy entry to the system for network eavesdroppers.
- **Reliable**: Kerberos mediates all server access so a robust system is needed allowing cover when a Kerberos server fails.
- **Transparent**: Other than initial password entry, users should not be aware of the authentication exchanges.
- **Scalable**: System must be capable of supporting large numbers of clients and servers.

Notation

We need to revise our notation to be more specific.

In particular, we need to distinguish between the client machine (workstation) that the user is logged into, the user (principal) and their names and addresses.

Thus:

- \( C \) is the client’s machine to/from which messages are transferred.
- \( \text{ADc} \) is the network address of \( C \).
- \( \text{IDc} \) is the name of the principal using \( C \).
  
  \( \text{IDc} \) can be \( \text{IDc} = \text{charles@xyz.com} \).
Notation - 2

To avoid confusion we will use the notation in Stallings p326

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>C</td>
<td>Client machine (workstation)</td>
</tr>
<tr>
<td>AS</td>
<td>Authentication server</td>
</tr>
<tr>
<td>V</td>
<td>Application server (e.g., file server)</td>
</tr>
<tr>
<td>Idc</td>
<td>Name of user on C</td>
</tr>
<tr>
<td>Pc</td>
<td>User’s password</td>
</tr>
<tr>
<td>ADc</td>
<td>Network address of machine C</td>
</tr>
<tr>
<td>IDv</td>
<td>Name of service</td>
</tr>
<tr>
<td>Kv</td>
<td>Personal encryption key (SKC) of V, only known to AS and V</td>
</tr>
</tbody>
</table>

Kerberos
Simple Authentication

- AS checks Principal’s credentials after step 1
- If satisfied, a ticket for V is generated
- Ticket is only valid for specific user and server and from a specific client machine
- Ticket is encrypted to prevent forgery and IDc is used in step 3 so V can verify correct decryption

Authentication
1. C->AS: IDc, Pc, IDv
2. AS->C: Ticket = [IDc, ADc, IDv]Kv
3. C->V: IDc, Ticket
Kerberos

Simple Authentication Issues

- Password needs to be re-entered each time a different server is accessed and AS has to be contacted to issue a ticket
- May be assumed that ticket is for “one use only”, but nothing in ticket to enforce that
- Nothing to prevent ticket being used for years!
- Password is transmitted in clear text over the network

- Introduce new Ticket Granting Service (TGS)

Kerberos

Better Authentication

Per login
1. C->AS: IDc, IDtgs
2. AS->C: {TICKETtgs}Kc

Per Service
3. C->TGS: IDc, IDv, TICKETtgs
4. TGS->C: TICKETv

Per Service Session
5. C->V: IDc, TICKETv

TICKET_{tgs} = [IDc, ADc, ID_{tgs}, TS1, Life1]_{K_{tgs}}
TICKET_v = [IDc, ADc, IDv, TS2, Life2]_{K_v}
Kerberos
Better Authentication - 2

- User asked for password to unwrap TICKETtgs which is held in client machine
- ∴ If C ends session before expiry of ticket captured message 2 poses no threat
- Also C cannot extend validity of ticket because encrypted with TGS’s key
- TGS needs to hold private keys of servers
- Neither V nor TGS will do business if ticket is out of date or client network address is wrong

Kerberos
Better Authentication - 3

- BUT, weakness in that if E captures message 3 and fakes ADc once C ends session, new service tickets could be generated by TGS
- OR, E could capture a service ticket at step 4 and use it in similar way
- Also, we may need to authenticate servers (show they are not Trojan Horses)
Kerberos
Version 4 Protocol

Per login
1. C→AS: IDc, IDtgs, TS1
2. AS→C: {Kctgs, IDtgs, TS2, Life2, TICKET(tgs, 2)}Kc

Per Service
3. C→TGS: IDv, TICKET(tgs, 2), AUTH(tgs, 3)
4. TGS→C: {Kcv, IDv, TS4, TICKET(v, 4)} Kctgs

Per Service Session
5. C→V: TICKET(v, 4), AUTH(v, 5)
6. V→C: [TS5+1]Kcv

TICKET(s, i)={Kcs, IDc, ADc, IDs, TSi, Lifei}Ks
AUTH(s, I) = {Idc, Adc, Tsi}Kcs

Session keys Kctgs and Kcv are introduced and securely conveyed to client machine from AS or TGS

Authenticator relies on only C being able to decrypt messages with Kc

Replay thwarted because T3 and T5 indicate current time so replay should be noticeable

Steps 4 and 6 provide proof that servers are legitimate ie. they possess Ktgs and Kv resp.
Kerberos Realms

- Within a given administration, eg. department:
  - AS has UID and (hashed) password of all users and key of TGS
  - TGS must have secret keys of all servers
- This is known as a Realm
- Users may need to access servers in remote realms, so:
  - TGS shares keys for trusted TGS in other realms

Kerberos Realms - 2

- Version 4 Protocol easily extended by local TGS issuing ticket for remote TGS (dialogue 2)
- Client then contacts remote realm for server ticket (dialogue 3)
Kerberos
Version 5 Protocol

- Improvements over version 4 to deal with:
  - Environmental shortcomings
  - Technical deficiencies

Kerberos
Version 5 Protocol - 2

- Environmental
  - Removes dependence on DES, allowing different algorithms and key lengths to be used
  - Removes IP dependency so can be used eg. with ISO addresses
  - Byte ordering dependency removed by using ASN.1 Basic Encoding Rules
  - Ticket lifetime specified by start+end time not up to 256 x 5 minutes
  - Authentication forwarding to allow delegated access to servers eg. print server delegated to access file server
  - Inter-realm improvement so fewer than \(O(N^2)\) inter-realm relationships needed, ie. scales much better
Kerberos
Version 5 Protocol

Technical
- Removes double encryption of tickets in steps 2 and 4 of V4 protocol
- Uses standard CBC encryption mode instead of Propagating CBC which has weaknesses
- Client and server can negotiate a sub-session key rather than use the session key offered by TGS
- Some measure to make password attacks more difficult (pre-authentication)

See Stallings pp 335-340 for more details
See RFC 1510 for full details (90pp!)

Per login
1. C->AS: Options, IDc, REALMc, IDtgs, Times, Nonce1
2. AS->C: REALMc, IDc, TICKET(tgs), {Kctgs, Times, Nonce1, REALMtgs, IDtgs}Kc

Per Service
3. C->TGS: Options, IDv, Times, Nonce2, TICKET(tgs), AUTH1
4. TGS->C: REALMc, Idc, TICKET(v), {Kcv, Times,Nonce2, REALMv, IDv} Kctgs

Per Service Session
5. C->V: Options, TICKET(v), AUTH2
6. V->C: {TS2, Subkey, Seq}Kcv

TICKET(s) = [Flags, Kcs, REALMc, IDc, Adc, Times]Ks
AUTH1 = [Idc, REALMc, TS1]Kctgs
AUTH2 = [Idc, REALMc, TS2, Subkey, Seq]Kcv
Further Reading

  - Kerberos: pp 323-340

  - Kerberos: pp 412-415

