Overview

• SSL/TLS
  – TLS Protocol Architecture
  – TLS Handshake Protocol
  – TLS Record Protocol

• HTTPS

• SSH
  – SSH Protocol Architecture
  – SSH Transport Protocol
  – SSH User Authentication Protocol
  – SSH Connection Protocol
SSL/TLS

Secure Sockets Layer
• Provides privacy and data integrity
• Netscape, around 1995
• SSLv3
• Key size: 128 bit / 40 bit for export

TLS: Transport Layer Security
• Based upon SSLv3
• IETF
• Two versions
  – TLS 1.0 (RFC 2246 & RFC 3546)
  – TLS 1.1 (RFC 4346 & RFC 4366)
TLS Goals

Defined in RFC 4346:

1. Cryptographic security
2. Interoperability
3. Extensibility
4. Relative efficiency
   – Session caching
# TLS Protocol Architecture

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Record Protocol</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TCP*</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**RFC 4346**

- Client / Server authentication
- Algorithm and key negotiation

- Confidentiality
- Message Integrity

**UDP: RFC 4347**
TLS Phases

Peer negotiation
- Determine algorithm support

Key exchange & Authentication
- Key exchange
- Authentication (certificate-based)
  - Using public key encryption

Data exchange
- Using symmetric encryption
TLS Handshake Protocol – Peer negotiation

Client hello
- SSL version
- Session ID
- Random
- Cipher suite
- Compression method

Server hello
- SSL version
- Session ID
- Random
- Cipher suite
- Compression method
TLS Handshake Protocol – Key exchange & Authentication

If requested by client, server sends its authentication data plus keys.

If necessary, clients sends its authentication data plus keys.
TLS Change Cipher Spec Protocol

- Change cipher spec activates new algorithms
- Finished verifies new algorithms
- After server sends Finished, actual data exchange starts
Ciphers in practice

- RSA_WITH_RC4_128_MD5 (!)
- RSA_WITH_AES_128_CBC_SHA
- DHE_RSA_WITH_AES_256_CBC_SHA
- RSA_WITH_AES_256_CBC_SHA
- RSA_WITH_RC4_128_SHA
- RSA_WITH_3DES_EDE_CBC_SHA (!)
- RSA_WITH_NULL_SHA
- DHE_RSA_WITH_CAMELLIA_256_CBC_SHA
- RSA_WITH_NULL_MD5 (!)
- DHE_RSA_WITH_AES_128_CBC_SHA
- others

See: The SSL Landscape - A Thorough Analysis of the X.509 PKI Using Active and Passive Measurements - Ralph Holz - IMC2011
TLS Record Protocol

Application data

Fragment

Compress

Add MAC

Encrypt

Append header
Certificates

- Create a certificate for your website
- Ask a CA to sign the certificate
- Use the certificate if clients access your website
Certificates in practice

- Chain valid
- 10: Expired
- 18: Self-signed end-host certificate
- 19: Root certificate of chain not in root store
- 20: No root certificate found for chain at all
- 32: Incorrect use of certificate for signing

% of all certificates

<table>
<thead>
<tr>
<th></th>
<th>Tue. Nov'09</th>
<th>TUM. Apr'11</th>
<th>Shanghai</th>
<th>EFF</th>
<th>MON1</th>
<th>MON2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chain valid</td>
<td>90%</td>
<td>90%</td>
<td>90%</td>
<td>90%</td>
<td>90%</td>
<td>90%</td>
</tr>
<tr>
<td>10: Expired</td>
<td>5%</td>
<td>5%</td>
<td>5%</td>
<td>5%</td>
<td>5%</td>
<td>5%</td>
</tr>
<tr>
<td>18: Self-signed</td>
<td>5%</td>
<td>5%</td>
<td>5%</td>
<td>5%</td>
<td>5%</td>
<td>5%</td>
</tr>
<tr>
<td>19: Root</td>
<td>5%</td>
<td>5%</td>
<td>5%</td>
<td>5%</td>
<td>5%</td>
<td>5%</td>
</tr>
<tr>
<td>20: No root</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
</tr>
<tr>
<td>32: Incorrect</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
</tr>
</tbody>
</table>
Certificate creation

Generate public-private key pair

Generate CSR:
Certificate Signing Request
- public key
- certificate
- signed with private key
Certificate signed by a CA

Send CSR

Check CSR integrity
- using public key

Verify identity website

Generate certificate
- signed with CAs private key

Send Certificate

Website

CA
Type of certificates

1. Domain Validated Certificate
   - DV-SSL
   - CA sends validation mail to web master

2. Organisation Validation Certificate
   - OV-SSL
   - CA additionally checks existence of organisation

3. Extended Validation Certificate
   - EV-SSL
   - CA additionally checks if requester is authorized
   - Expensive (> 100 Euro)
Certificate types in browsers

Browsers show for all certificates the “locked” sign

1. Domain Validated Certificate
   • Firefox: blue
   • Opera: yellow

2. Extended Validation Certificate
   • Firefox: green
   • IE: green
   • Chrome: green
   • Safari: green
Certificates

SSL connect

Send signed certificate

Check certificate
- CA trusted?
- On certificate revocation list (CRL)?
- Check using Online Certificate Status Protocol (OSCP)
HTTPS

- Introduced by Netscape
- Application on top of SSL/TLS
- Same as normal HTTP protocol
- RFC 2812 (Informational)
- Uses port 443
- Server requires certificate
- Clients may also hold their own certificates
SSH

• *NOT* an application on top of SSL
• Replacement for TELNET, rlogin, rsh
• 1995: version 1, 1996: version 2
• SSH Communications Security
• OpenSSH
• 2006: IETF secsh WG, many RFCs
SSH Protocol Architecture

- **SSH-CONNECT**
  - SSH Connection Protocol
    - RFC 4254

- **SSH-USERSAUTH**
  - SSH User Authentication Protocol
    - RFC 4252

- **SSH-TRANS**
  - SSH Transport Protocol
    - RFC 4253

TCP

**RFC 4251**
- Logical channels
- Client authentication
- Server authentication
- Confidentiality
- Integrity
- Compression (option)
SSH Transport Protocol

• Initial key exchange / Key re-exchange
  – Diffie-Hellman

• Encryption:
  – Recommended: aes128-cbc
  – Required: 3des-cbc

• Server authentication
  – Recommended: hmac-sha1-96
  – Required: hmac-sha1
  – Optional: hmac-md5, none

• Compression:
  – Required: None
  – Optional: zlib (lz77)
SSH Transport Protocol - Timeline

TCP connection setup

SSH version string exchange

SSH key exchange / algorithm negotiation

SSH data exchange

TCP connection release

key / algorithm re-negotiation

time
SSH Transport Protocol - Binary Packet Protocol

packet length

padding length

payload
(may be compressed)

random padding

MAC

max: 32768 octets

max: 35000 octets
SSH Transport Protocol - Server authentication

• Based on the server’s host key
• The client must check this key
• Models
  – the client has a local database that associates each host name with the corresponding public host key
  – the host name – to – key association is certified by a trusted CA and the server provides the necessary certificates or the client obtains them from elsewhere
  – check fingerprint of the key over an external channel (e.g., phone)
  – best effort:
    • accept host key without check when connecting the first time to the server
    • save the host key in the local database, and
    • check against the saved key on all future connections to the same server

source: www.hit.bme.hu/~buttyan/courses/BMEVIHI9367/SSH.ppt
SSH User Authentication Protocol

Methods
• public key
• password
• host based
• keyboard-interactive (RFC 4256)

Time Out period
• 10 minutes

Too many attempts
• 20 attempts
SSH Connection Protocol

Multiplexing protocol

- Opens and closes “channels”
- Multiple “channels” are mapped on a single SSH transport connection
- Each application uses one or more dedicated channels
- Allows for out-of-band control:
  - change size of a terminal window
  - application flow control
  - exit code of a server-side process.

- Each channel is locally identified by “channel number”

- Standard channel types include:
  - shell
  - direct-tcpip
  - forwarded-tcpip
SSH Applications

- SSH: Secure Shell
- SCP: Secure Copy
- SFTP: Secure FTP
- SSHFS: Secure SHell FileSystem
- ISMS: Integrated Security Model for SNMP
- Port forwarding (tunneling)
Port forwarding (Tunneling)

client
A
Application

server
B
Application

client
A
Application

firewall

C
SSH Server

22

28

server
B
Application

y

x

x

28

firewall

SSH Client

y

22

A

B

C
SSH References

• Wikipedia
  http://en.wikipedia.org/wiki/SSH

• IETF SECSH WG
  http://www.ietf.org/html.charters/secsh-charter.html

• Levente Buttyán, BME-HIT
  www.hit.bme.hu/~buttyan/courses/BMEVIHI4372/ssh.pdf

• SSH: The Definitive Guide
  http://www.snailbook.com/
SSH versus SSL/TLS

**SSH:**
- Server authentication mandatory
- Client authentication: many options
- Authentication based on SSH keys
- Provides logical data channel multiplexing

**SSL/TLS:**
- Server authentication is optional (although generally chosen)
- Client authentication: only public keys
- Authentication based on X.509 certificates
- Does not provide logical data channel multiplexing