# H.....t.....t....p....p....o....s....t

**Wong Onn Chee**
**OWASP Singapore Lead**
ocwong@usa.net

**Tom Brennan**
**OWASP Foundation**
tomb@owasp.org

## OWASP AppSec DC 2010

11 Nov 2010

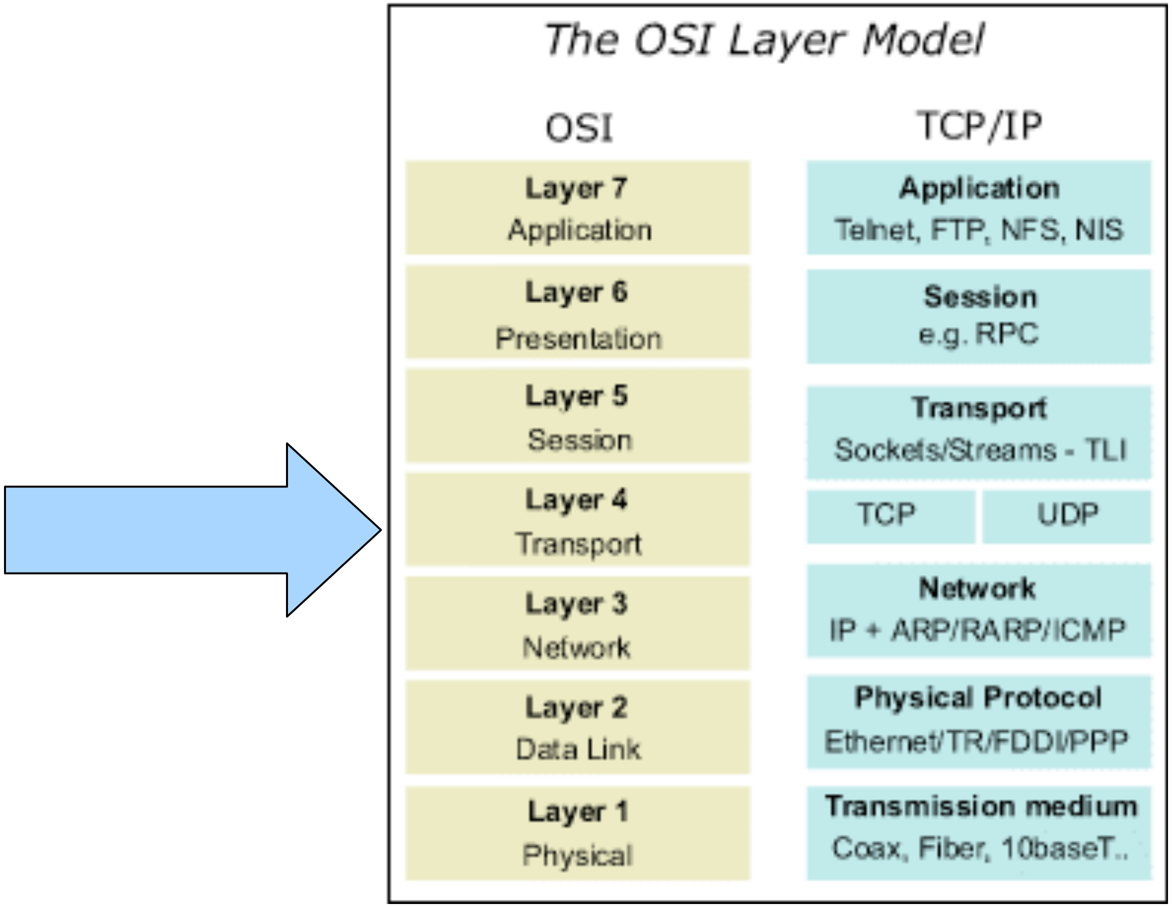
**The OWASP Foundation**
http://www.owasp.org

# Agenda

- **Introduction to Layer 7 DDOS attacks**
- Different types of Layer 7 DDOS web attacks
- Analysis of HTTP POST DDOS attack
- Demo

# First, there was Layer 4 DDOS......

- Past DDOS attacks were mainly Layer 4 (TCP) attacks.



# Layer 4 DDOS attacks

- Reach **bandwidth or connection limits** of **hosts or networking equipment**.
- Fortunately, current anti-DDOS solutions are effective in handling Layer 4 DDOS attacks.

# Then, there were Layer 7 DDOS attacks

- Operates at the application protocol level (OSI Layer 7).
- Eg. HTTP(S), SMTP, FTP and etc.

# Effectiveness of Layer 7 DDOS attacks

- Legitimate TCP or UDP connections. Difficult to differentiate from legitimate users => higher obscurity.

- Requires lesser number of connections => higher efficiency.

- Reach **resource limits** of **services**.
  Can deny services regardless of hardware capabilities of host => higher lethality.

# Agenda

- Introduction to Layer 7 DDOS attacks
- **Different types of Layer 7 DDOS web attacks**
- Analysis of HTTP POST DDOS attack
- Demo

# Types of Layer 7 DDOS web attacks

- Excludes causes related to stupid or inefficient codes. (Yes! You can DOS yourself)

- We will focus on protocol weaknesses of HTTP or HTTPS.

    - HTTP GET => Michal Zalewski, Adrian Ilarion Ciobanu, RSnake (Slowloris)
    - HTTP POST => Wong Onn Chee

# HTTP GET DDOS attack

- First highlighted by Michal Zalewski and Adrian Ilarion Ciobanu in 2007
  http://www.securityfocus.com/archive/1/456339/30/0/threaded
- Popularized in 2009 by Rsnake with the free tool, Slowloris.
- Slowloris used time-delayed HTTP headers to hold on to HTTP connections and exhaust web server threads or resources.
- Can evade Layer 4 DDOS protection systems. More info can be found at
  http://ha.ckers.org/blog/20090617/slowloris-http-dos/

# HTTP GET DDOS attack

- Apache Foundation disagreed this is a bug and had no plans to “fix it”. To AF, waiting for the HTTP headers to complete sending is a basic and inherent behavior of web servers.
- Microsoft IIS imposes a timeout for HTTP headers to be sent. Any HTTP connection which exceeds the headers timeout will be closed, hence rendering HTTP GET attacks ineffective against IIS web servers.

# Limitations of HTTP GET DDOS attack

- Does not work on IIS web servers or web servers with timeout limits for HTTP headers.

- Easily defensible using popular load balancers, such as F5 and Cisco, reverse proxies and certain Apache modules, such as mod_antiloris.

- Anti-DDOS systems may use “delayed binding”/“TCP Splicing” to defend against HTTP GET attacks.

# Agenda

- Introduction to Layer 7 DDOS attacks
- Different types of Layer 7 DDOS web attacks
- **Analysis of HTTP POST DDOS attack**
- Demo

# HTTP POST DDOS attack

- First discovered in Sep 2009 by Wong Onn Chee and his team.
- Escalated to Microsoft and AF in Q1 2010. Both interpreted this to be a protocol bug.
- Apache: “What you described is a known attribute (read: flaw) of the HTTP protocol over TCP/IP.  The Apache HTTP project declines to treat this expected use-case as a vulnerability in the software.”
- MS: “While we recognize this is an issue, this issue does not meet our bar for the release of a security update. We will continue to track this issue and the changes I mentioned above for release in a future service pack.”

# How HTTP POST DDOS attack works (HTTP/1.0)

- Uses HTTP POST requests, instead of HTTP GET which is used by Slowloris.
- “A POST request includes a message body in addition to a URL used to specify information for the action being performed. This body can use any encoding, but when webpages send POST requests from an HTML form element the Internet media type is "application/x-www-form-urlencoded". (source: Wikipedia - “POST (HTTP)”)”

# How HTTP POST DDOS attack works (HTTP/1.0) (cont'd)

- The field “Content-Length” in the HTTP Header tells the web server how large the message body is, for e.g., “Content-Length = 1000”

- The HTTP Header portion is complete and sent in full to the web server, hence bypassing IIS inherent protection.

# How HTTP POST DDOS attack works (HTTP/1.0) (cont'd)

- For e.g., Content-Length = 1000 (bytes)
- The HTTP message body is properly URL-encoded, but ......
- **.....is sent at, again for e.g., 1 byte per 110 seconds.**
- Multiply such connections by 20,000 and your IIS web server will be DDOS.
- Most web servers can accept up to 2GB worth of content in a single HTTP POST request.

# Sample code to simulate HTTP POST DDOS attack (HTTP/1.0)

```
private String getRequestHeader() {
String requestHeader = "";
requestHeader += param.getMethod() + " " + param.getUrl() + " HTTP/1.1\r\n";

requestHeader +=
  "Host: " + param.getHost() + "\r\n"
+ "User-Agent: " + httpUserAgent + "\r\n"
+ "Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8\r\n"
+ "Accept-Language: en-us,en;q=0.5\r\n"
+ "Accept-Encoding: gzip,deflate\r\n"
+ "Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7\r\n"

if (param.getContentLength() > 0) {
requestHeader += "Connection: keep-alive\r\n";
requestHeader += "Keep-Alive: 900\r\n";
requestHeader += "Content-Length: " + param.getContentLength() + "\r\n";
requestHeader += "\r\n";
}
return requestHeader;
}
```

Construction of legitimate headers

Random values for Content-Length header

# Sample code to simulate HTTP POST DDOS attack (HTTP/1.0)

Get random data -->

```
public static byte getRandomByte() {
int character = gen.nextInt();
return (byte) character;
}
```

Byte randomness

Send random data -->

```
public void sendXHeader() throws IOException {
StringBuffer header1 = new StringBuffer();
StringBuffer header2 = new StringBuffer();
int lengthOfXA = param.getRandomLengthOfXA();
int lengthOfXB = param.getRandomLengthOfXB();
for (int i=0 ; i<lengthOfXA ; i++) {
header1.append(Misc.getRandomByte());
}
```

Time interval randomness

# Sample code to simulate HTTP POST DDOS attack (HTTP/1.0)

```
for (int i=0 ; i<lengthOfXB ; i++) {
header2.append(Misc.getRandomByte());
}
socket.getOutputStream().write(("X-" + header1.toString() + ": " + header2.toString() + "\r\n").getBytes());
socket.getOutputStream().flush();
}

public void sendPOSTBodyRandomByte() throws IOException {
socket.getOutputStream().write(Misc.getRandomByte());
socket.getOutputStream().flush();
}
```

Sends the payload

## Why HTTP POST DDOS attack works

- Being “kind” folks (like all of you), web servers will “obey” the “Content-Length” field to wait for the remaining message body to be sent.
- By waiting for the complete message body to be sent, web servers can support users with slow or intermittent connections.
- Hence, any website which has forms, i.e. accepts HTTP POST requests, is susceptible to such attacks.
- Common uses of HTTP POST requests: login, uploading photo/video, sending webmail / attachments, submitting feedback and etc.

## Why HTTP POST DDOS attack works

- This attack can evade Layer 4 detection techniques as there is no malformed TCP, just like Slowloris.
- Unlike Slowloris, there is no delay in sending HTTP Header, hence nullifying IIS built-in defense, making IIS vulnerable too.
- Size, character sets and time intervals can be randomised to foil any recognition of Layer 7 traffic patterns by DDOS protection systems.
- Difficult to differentiate from legit connections which are slow.

## Interesting findings

- IIS 6.0 (W2K3) web server is vulnerable to this attack even when there is no form. Apache, IIS 7 or later require presence of forms for this attack to work.
- Apache requires lesser number of connections due to mandatory client or thread limit in httpd.conf.
- Besides its “unlimited connections” settings, a default IIS configuration will go down with 20,000 HTTP POST DDOS connections, regardless of hardware capabilities. This is due to the rapid fail protection sandbox feature in IIS.

## Interesting findings

- IIS with 8 cores and 16GB RAM = IIS with 2 cores and 2GB RAM
  Only 20k HTTP POST connections to DDOS either IIS!
- In HTTP/1.1 where chunked encoding is supported and there is no “Content-Length” HTTP header, the lethality is amplified.

  The web server does not even know up front from the headers how large is the POST request!

## Interesting findings

- Botnet operators had begun their “3G upgrade” to include Layer 7 DDOS techniques. Some may have completed their upgrade to include HTTP POST.

- We believe Layer 7 attacks may supersede Layer 4 attacks as the modus operandi of DDOS botnets in this new decade.

# Potential countermeasures

- Apache
  - (experimental) mod_reqtimeout
  - LimitRequestBody directive
- IIS
  - No reply from Microsoft on the availability of the proposed controls in the latest service pack for IIS.

# Potential countermeasures

- General
  - Limit the size of the request to each form's requirements.
    For e.g. a login form with a 20-char username field and a 20-char password field should not accept a 1KB POST message body
  - Identify the 95% or 99% percentile of normal access speed range to your website. Establish a speed floor for the outliers.
  - With the speed floor and maximum allowable body size for each form, establish a request timeout for each form (= Tedious! Good news for infosec folks?)

# Weaknesses of countermeasures

- Hackers can “sense” the speed floor and execute attacks just above the speed floor.
- Most (broadband) home users have uplink speed of at least 256 kbps. But we cannot set speed floors at 256 kbps.
- Speed floors = not friendly to overseas customers/visitors or local ones using mobile devices.
- HTTPS will be a challenge for front appliance-based defensive systems.

# Future “exploits”? - WebSockets

- WebSockets in HTML5 (draft expires February 17, 2011) http://www.whatwg.org/specs/web-socket-protocol/
- “Conceptually, WebSocket is really just a layer on top of TCP that adds a Web "origin"-based security model for browsers; adds an addressing and subprotocol naming mechanism to support multiple services on one port and multiple host names on one IP address; layers a framing mechanism on top of TCP to get back to the IP packet mechanism that TCP is built on, but **without length limits**; and reimplements the closing handshake in-band....”

# Future “exploits”? - WebSockets

- 6.3.  Data framing

The server must run through the following steps to process the bytes sent by the client.  If at any point during these steps a read is attempted but fails because the WebSocket connection is closed, then abort.

1. Try to read a byte from the client.  Let /frame type/ be that byte.
2. Try to read eight more bytes from the client.  Let **/frame length/ be the result of interpreting those eight bytes as a big-endian 64 bit unsigned integer**.
   (e.g. 99,999,999)

…….

- 99,999,999 / 1 byte per 110 secs
  = 10,999,999,890 secs
  = 127,315 days
  = 349 years

# Agenda

- Introduction to Layer 7 DDOS attacks
- Different types of Layer 7 DDOS web attacks
- Analysis of HTTP POST DDOS attack
- **Demo**

# Demo

- Old = you may already know about the components.
- New = New trend of “weaponized” online games which are web-based or client-based.
- Desktop firewalls do not block outgoing Port 80 connections once the process is whitelisted. (Need to be whitelisted, else game will not run)
- The one we are showing is a simple game using a self-signed Java applet. (good old Java sandbox bypass)