Searching and Analyzing HTTP Data with the WASE Framework

Thomas Patzke
German OWASP Day 2016
Content

1. Why?

2. Introduction to WASE and the toolchain

3. Usage Examples
   • Security Testing of web applications
   • Mass-Scanning the Alexa Top 1M

4. Future Development & Ideas
What is the Problem?

Bad Performance!

Unflexible!
Try to do one of the following with your tool of choice:

- Search all POST requests that don’t contain a CSRF token
- List all values of a parameter or cookie that encountered while a web application security test
- List all values of a security header with its corresponding URL
- List all URLs where inferred content type is HTML while the server tells something different about its content type
- Show all HTML responses without a doctype definition
- Find all external script references
- Discover unsafe or nonse HTTP security header values

Bonus points: try it without a coffee break :)
Elasticsearch, Kibana, WASE

- Elasticsearch: a search and analytics engine for textual data
- Kibana: web frontend for Elasticsearch
- WASE: Web Audit Search Engine
  - Definition of a data structure for HTTP requests/responses for Elasticsearch
  - Toolchain: ElasticBurp, WASEProxy, WASEQuery
WASE Framework

- Burp Suite → ElasticBurp
- WASEProxy
- Arbitrary HTTP Client with Proxy Support
- DocHTTPRequestResponse
- Elasticsearch
- Kibana
- WASEQuery
Usage Examples

- Complex searches and analytics in web application security tests
- Mass Scans of web sites
- Malware analysis (someones master thesis)
Searches

- **All POSTs without CSRF-Token:**
  \[\text{request.method:POST} \ - \text{request.parameternames.raw:"csrftoken"}\]

- **2xx Responses recognised as HTML without <!DOCTYPE ...>:**
  \[\text{response.status:[200 TO 299]} \ \text{AND} \ \text{response.inferred_content_type:html \ -doctype}\]

- **HTML Responses not declared as such ones:**
  \[\text{response.inferred_content_type:html} \ \text{-response.content_type:html}\]

- **Responses without XFO:**
  \[\text{NOT response.headernames:"X-Frame-Options"}\]
Searching with Kibana
Searching with WASEQuery

- Kibana doesn’t like nested data structures and doesn’t expose many ElasticSearch features
- WASEQuery: collection of few useful queries

Example: List of all CSPs that contain the word unsafe
Mass-Scanning the Alexa Top 1m

- Scanning from AWS EC2 instances
  - 1 x t2.micro as scan controller (misused a bit as worker)
  - 4 x m4.large spot instances as scan workers
  - 4 x t2.micro.elasticsearch
  - 2h x 3 x r3.xlarge.elasticsearch for final analysis (required much RAM for some complex queries)
  - 1 day for scanning complete 1m list
  - ~1,50€ scanning costs, ~3€ analysis

- Tools:
  - GNU Parallel
  - curl
  - WASEProxy

- No response bodys!
- 35,6 GB ES Indexes
- 15.311.855 ES Docs

Diagram:
- Controller:
  - GNU Parallel
  - Domain List

- Worker:
  - curl
  - WASEProxy
  - Stores via HTTP
  - HTTP

- Elasticsearch

Internet
Results

Requests per Time Unit

- Count: 1,643,741
- Unique count of host: 1,316,382
- Unique count of host (2xx Responses): 840,262

Proxies
- Hits: 11.14% (13.89%)

Filters
- X-Frame-Options: 66,753
- X-Content-Type-Options: 63,977
- X-XSS-Protection: 50,453
- Strict-Transport-Security: 28,597
- Content-Security-Policy: 3,450
- Public-Key-Pins: 359
- Number of Domains: 777,704
## Results: Popular Embedded Objects

<table>
<thead>
<tr>
<th>Top 20 response.objects.raw Q</th>
<th>Unique count of host</th>
</tr>
</thead>
<tbody>
<tr>
<td>core.RE</td>
<td>204</td>
</tr>
<tr>
<td><a href="http://www.xatech.com/web_gear/chat/chat.swf">http://www.xatech.com/web_gear/chat/chat.swf</a></td>
<td>125</td>
</tr>
<tr>
<td>//www.youtube.com/get_player</td>
<td>104</td>
</tr>
<tr>
<td>index.swf</td>
<td>77</td>
</tr>
<tr>
<td>banner.swf</td>
<td>69</td>
</tr>
<tr>
<td><a href="http://swf.yowindow.com/yowidget3.swf">http://swf.yowindow.com/yowidget3.swf</a></td>
<td>67</td>
</tr>
<tr>
<td><a href="https://res.egtmgs.com/release/core/EmptySwf.swf">https://res.egtmgs.com/release/core/EmptySwf.swf</a></td>
<td>56</td>
</tr>
<tr>
<td>images/banner.swf</td>
<td>46</td>
</tr>
<tr>
<td>images/logo.swf</td>
<td>45</td>
</tr>
<tr>
<td><a href="http://releases.flowplayer.org/swf/flowplayer-3.2.1.swf">http://releases.flowplayer.org/swf/flowplayer-3.2.1.swf</a></td>
<td>36</td>
</tr>
<tr>
<td>//flash/mjup4li.swf?123</td>
<td>35</td>
</tr>
<tr>
<td>//video.limelight.com/player/loader.swf</td>
<td>34</td>
</tr>
<tr>
<td>main.swf</td>
<td>34</td>
</tr>
<tr>
<td>source/plugin/study_nge/images/clock.swf</td>
<td>32</td>
</tr>
<tr>
<td>data:application/x-silverlight-2</td>
<td>31</td>
</tr>
<tr>
<td>FLVPlayer_Progressive.swf</td>
<td>30</td>
</tr>
<tr>
<td>header.swf</td>
<td>30</td>
</tr>
<tr>
<td>images/top.swf</td>
<td>30</td>
</tr>
<tr>
<td>top.swf</td>
<td>29</td>
</tr>
</tbody>
</table>

### Filters:
- Scanned Domains: 777,704
- With JS References: 709,235
- With JS refs but no external: 109,716
# Results: DOCTYPE Declarations

<table>
<thead>
<tr>
<th>DOCTYPE html PUBLIC &quot;-//W3C//DTD XHTML 1.0 Transitional//EN&quot; &quot;<a href="http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd</a>&quot;</th>
<th>433,569</th>
</tr>
</thead>
<tbody>
<tr>
<td>DOCTYPE html PUBLIC &quot;-//W3C//DTD XHTML 1.0 Transitional//EN&quot; &quot;<a href="http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd</a>&quot;</td>
<td>123,066</td>
</tr>
<tr>
<td>doctype html</td>
<td>63,853</td>
</tr>
<tr>
<td>DOCTYPE html PUBLIC &quot;-//W3C//DTD HTML 4.01 Transitional//EN&quot; &quot;<a href="http://www.w3.org/TR/html4/loose.dtd">http://www.w3.org/TR/html4/loose.dtd</a>&quot;</td>
<td>11,113</td>
</tr>
<tr>
<td>DOCTYPE HTML</td>
<td>29,659</td>
</tr>
<tr>
<td>DOCTYPE HTML PUBLIC &quot;-//W3C//DTD HTML 4.01 Transitional//EN&quot; &quot;<a href="http://www.w3.org/TR/html4/strict.dtd">http://www.w3.org/TR/html4/strict.dtd</a>&quot;</td>
<td>7,701</td>
</tr>
<tr>
<td>doctype html</td>
<td>6,853</td>
</tr>
<tr>
<td>DOCTYPE html PUBLIC &quot;-//W3C//DTD XHTML 1.1//EN&quot; &quot;<a href="http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd</a>&quot;</td>
<td>4,992</td>
</tr>
<tr>
<td>DOCTYPE HTML PUBLIC &quot;-//W3C//DTD HTML 4.0 Transitional//EN&quot;</td>
<td>3,832</td>
</tr>
<tr>
<td>DOCTYPE HTML PUBLIC &quot;-//W3C//DTD HTML 4.01//EN&quot; &quot;<a href="http://www.w3.org/TR/html4/strict.dtd">http://www.w3.org/TR/html4/strict.dtd</a>&quot;</td>
<td>3,777</td>
</tr>
<tr>
<td>DOCTYPE html PUBLIC &quot;-//W3C//DTD XHTML+RDFa 1.0//EN&quot; &quot;<a href="http://www.w3.org/MarkUp/DTD/xhtml-rdf-1.dtd">http://www.w3.org/MarkUp/DTD/xhtml-rdf-1.dtd</a>&quot;</td>
<td>3,244</td>
</tr>
<tr>
<td>DOCTYPE html PUBLIC &quot;-//W3C//DTD HTML 4.01 Transitional//EN&quot; &quot;<a href="http://www.w3.org/TR/html4/loose.dtd">http://www.w3.org/TR/html4/loose.dtd</a>&quot;</td>
<td>1,882</td>
</tr>
<tr>
<td>DOCTYPE HTML PUBLIC &quot;-//W3C//DTD HTML 4.01 Transitional//EN&quot; &quot;<a href="http://www.w3.org/TR/html4/strict.dtd">http://www.w3.org/TR/html4/strict.dtd</a>&quot;</td>
<td>1,456</td>
</tr>
<tr>
<td>doctype html</td>
<td>1,364</td>
</tr>
<tr>
<td>DOCTYPE html PUBLIC &quot;-//W3C//DTD XHTML 1.0 Transitional//EN&quot; &quot;<a href="http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd</a>&quot;</td>
<td>1,336</td>
</tr>
<tr>
<td>DOCTYPE html PUBLIC &quot;-//W3C//DTD XHTML+RDFa 1.0//EN&quot; &quot;<a href="http://www.w3.org/MarkUp/DTD/xhtml-rdf-1.dtd">http://www.w3.org/MarkUp/DTD/xhtml-rdf-1.dtd</a>&quot;</td>
<td>1,152</td>
</tr>
<tr>
<td>DOCTYPE html PUBLIC &quot;-//W3C//DTD HTML 4.01//EN&quot; &quot;<a href="http://www.w3.org/TR/html4/strict.dtd">http://www.w3.org/TR/html4/strict.dtd</a>&quot;</td>
<td>1,112</td>
</tr>
<tr>
<td>DOCTYPE HTML PUBLIC &quot;-//W3C//DTD HTML 3.2 Final//EN&quot;</td>
<td>1,053</td>
</tr>
<tr>
<td>DOCTYPE html PUBLIC &quot;-//W3C//DTD HTML 4.01 Transitional//EN&quot;</td>
<td>1,041</td>
</tr>
</tbody>
</table>
What’s Next – Future Development

- Documentation and Automation of Mass-Scan Setup
- Extraction of further attributes
- Further Input Frontends:
  - PCAP
  - Raw Text Files
  - OWASP ZAP
- Development of a fancy query language
- Search interface in Burp Extension
That‘s it!

Get it on GitHub: https://github.com/thomaspatzke/WASE
...Pull Requests are Welcome! :)

Live Demo: http://wase-demo.patzke.org

Questions?
Mail: thomas@patzke.org
Twitter: @blubbfiction