HTTP Fuzzing: Using JBroFuzz to fuzz the web away

Matt Tesauro
OWASP Live CD Project Lead
OWASP Global Projects Committee
OWASP Board Member
matt.tesauro@owasp.org

OWASP
Alabama
Feb. 18 2010

Copyright © The OWASP Foundation
Permission is granted to copy, distribute and/or modify this document under the terms of the OWASP License.
Presentation Overview

- Fuzzing in general
- Fuzzing in the web world
- HTTP Fuzzing with JBrofuzz
- Other fuzzing options
- Conclusions and such
About Matt

Varied IT Background

- Developer, DBA, Sys Admin, Pen Tester, Application Security, CISSP, CEH, RHCE, Linux+

Long history with Linux & Open Source

- First Linux install ~1998
- DBA and Sys Admin was all open source
- Contributor to many projects, leader of one

A bit of OWASP too.
Fun pics of me - just so Brad's happy
I clean up really well
Nobody's safe
A fuzz by any other name...

- 1913 Websters: “To make drunk.”

- WordNet 2.0: “uncomplimentary terms for a policeman
  “the first beard of an adolescent boy”

- For today: “a method to discover software flaws by providing unexpected inputs”
Where did fuzzing start?

- Similar to Boundary value analysis

- 1989 Professor Barton Miller
  - Early fuzzer of Unix applications
  - Pure black box approach with random strings
  - Code quality and reliability were drivers

- Next protocol specifications, network-enabled applications, browser rendering, file format fuzzing, ...
Developing your web fuzz

- Identify Target(s)
- Identify Inputs
- Create Fuzz Data
- Send/Submit Fuzz Data
- Monitor for Problems or Changes
- Verify Exploitability
Details for getting your web fuzz on

- Identify Target(s)
  - Scope of engagement determines
  - Look at components of the application
    - Libraries, AJAX Frameworks, ...
  - Size requires focus on soft spots/sensitive areas

- Identify Inputs
  - You've done IG-003. right?
    - OWASP Testing Guide, Information Gathering Section
  - Look for those inputs “you can't change”
    - Buttons, cookies, referer, hidden fields
Details for getting your web fuzz on

- Create Fuzz Data
  - Sometimes auto-generated by the tool
  - Fuzz lists
  - Tailored vs Brute

- Send/Submit Fuzz Data
  - GET vs POST
  - Other methods
    - SOAP, RESTful Services, WebDAV, ...
  - Very painful if not automated
Details for getting your web fuzz on

- Monitor for Problems or Changes
  - HTTP Status Codes
    - HTTP 500
  - Response page size
  - Response timing

- Verify Exploitability
  - Error !≠ Vulnerable
  - Manually verify and refine testing
  - Engagement scope determines
Fuzzing fail

■ Stateful testing
  ‣ Especially authorization testing
  ‣ Typically blind to roles and privileges

■ Logic errors or poor design
  ‣ Too close to see higher level issues

■ Incubated or multi-step vulnerabilities
  ‣ Focus is too narrow for this much context
Fuzzing Fail continued

- Hidden functionality
  - Orphaned pages or functions
  - Backdoors
    - e.g. hard coded passwords

- Server side errors
  - Memory errors
  - Stalled threads (short of DOS)
  - Depends on how 'crystal' your box is
Types o'Fuzz

- Mutation-based fuzzing
  - Use existing valid data
  - Mangle valid data to create test cases

- Generation-based fuzzing
  - Create test cases from nothing
  - Model existing target's data to create test cases
Fuzzing Sub-categories

- Pre-generated test cases
  - Create standard test cases and apply consistently
    - Results between tests are easily compared
    - Complete coverage = lots of test cases = work++
  - No random elements
    - limited to quality of the initially created test cases

- Random
  - Quick and dirty approach
    - Lacks targeting, longer test runs, inefficient
Fuzzing Sub-categories

- Manual Manipulation
  - Tester is the random element
  - Good as the testers knowledge & experience
  - Works well for custom situations

- Mutation or Brute Force Testing
  - Start with good data and continually make small modifications
    - Very little setup or domain knowledge required
    - Problems similar to random
Automatic Protocol Generation Testing

» Create a grammar which describes what is being tested
» Templates describe generalized test
» Only portions of the template are fuzzed, others are static
» Crucial to pick the right portions to fuzz
» Optimized to the likely vulnerable areas
Creating your own mutations

- Using Spreadsheets for payloads
  - Select and drag feature in popular spreadsheet software makes this easy
    - abc124user
    - bcd123user
Creating your own mutations

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
<th>G</th>
<th>H</th>
<th>I</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>abc</td>
<td>123</td>
<td>user</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>abc</td>
<td>124</td>
<td>user</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>abc</td>
<td>125</td>
<td>user</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>abc</td>
<td>126</td>
<td>user</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>abc</td>
<td>127</td>
<td>user</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>abc</td>
<td>128</td>
<td>user</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>abc</td>
<td>129</td>
<td>user</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>abc</td>
<td>130</td>
<td>user</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>abc</td>
<td>131</td>
<td>user</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>abc</td>
<td>132</td>
<td>user</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>abc</td>
<td>133</td>
<td>user</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>abc</td>
<td>134</td>
<td>user</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>a</td>
<td>b</td>
<td>c</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>123</td>
</tr>
<tr>
<td>14</td>
<td>b</td>
<td>c</td>
<td>e</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>123</td>
</tr>
<tr>
<td>15</td>
<td>c</td>
<td>d</td>
<td>f</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>124</td>
</tr>
<tr>
<td>16</td>
<td>a</td>
<td>b</td>
<td>c</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>125</td>
</tr>
<tr>
<td>17</td>
<td>b</td>
<td>c</td>
<td>e</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>126</td>
</tr>
<tr>
<td>18</td>
<td>a</td>
<td>b</td>
<td>c</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>127</td>
</tr>
<tr>
<td>19</td>
<td>b</td>
<td>c</td>
<td>e</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>128</td>
</tr>
<tr>
<td>20</td>
<td>c</td>
<td>d</td>
<td>f</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>129</td>
</tr>
<tr>
<td>21</td>
<td>a</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>22</td>
<td>b</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>23</td>
<td>c</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>24</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>25</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Say hello to JBroFuzz

JBroFuzz

“Web application fuzzer for requests made over HTTP or HTTPS. Its purpose is to provide a single, portable application that offers stable web protocol fuzzing capabilities.”
JBroFuzz features

- HTTP proxy support
- Encoder/Hash window
  - Base64, MD5, SHA-1, SHA-256, SHA-384, SHA-512 and URL (UTF-8)
- Very large selection of injection payloads
- Many built in user-agent strings
- Handles HTTP 100 Continue
- Search mechanism built in
- Syntax coloration
Its Demo time!

DEMO AHEAD

Watch out for explosions and demo gremlins
Other ways to fuzz HTTP

- OWASP WebScarab (Fuzzer tab)
  - Allows for fuzzing parameter(s) by priority
    - payloads: text files or generated

- Burp Proxy Suite (Intruder tab)
  - Allows for fuzzing the HTTP request in full
    - multiple positions and attack types
      - sniper, battering ram, pitchfork, cluster bomb

- WSFuzzer
  - Web Services Fuzzer
    - Command line, tons of options
Learn More

- OWASP Site:  
  or Google “OWASP JBroFuzz”


- Fuzzing:  
  Brute Force Vulnerability Discovery  
  ISBN: 0321446119
Try it before you buy it

- All the tools mentioned today are on the OWASP Live CD
  - A subproject of OWASP Web Testing Environment

- OWASP Site:

- Download & Community Site:
  http://AppSecLive.org

- Original site:  http://mtesauro.com/livecd/
What's next?

Using Selenium to hold state for web application penetration testing
By Yiannis Pavlosoglou

Presented at London chapter on January 14th

PDF of slides available:

which is a uselessly long URL so search for “Selenium” in the search box on http://www.owasp.org
Questions?

THE EXPLOSION OF BOTNETS HAS MANDATED
A NEW WARNING LABEL:

CLICK HERE TO REQUEST EXPLOIT
ON PORT 80