OWASP OWTF
Summer Storm

Abraham Aranguren

OWASP OWTF Project Leader
@7a_ @owtfp
abraham.aranguren@owasp.org
Agenda

• GSoC Overview
• What is OWASP OWTF?
• Status update on OWTF GSoC projects
  • OWTF Reporting
  • OWTF Multiprocessing
  • OWTF MiTM Proxy
  • OWTF Testing Framework
• OWASP Testing Guide with OWTF
• Conclusion
Agenda

• GSoC Overview
• What is OWASP OWTF?
• Status update on OWTF GSoC projects
  • OWTF Reporting
  • OWTF Multiprocessing
  • OWTF MiTM Proxy
  • OWTF Testing Framework
• OWASP Testing Guide with OWTF
• Conclusion
Google Summer of Code (GSoC) Overview
GSoC Stats + Outcome

- OWASP got **11** slots from Google
- OWASP received **84** proposals
- **73** students (87%) could not be selected.
- Final slot breakdown:
  - **4** - OWASP ZAP
  - **4** - OWASP OWTF
  - **1** - OWASP Hackademic
  - **1** - OWASP ModSecurity
  - **1** - OWASP PHP Security Project

OWTF GSoC Overview

- **14** students showed interest (email)
- **11** (*79%*) students submitted a proposal
- **14** proposals were submitted (*16%* of 84)
- **5** OWTF proposals ended in the top 11
- **1** student was lost in de-duplication process (accepted by another org)
- **4** OWTF proposals were finally selected (*36%* of 11)

Why submit for OWTF?

OWTF GSoC student poll summary:

- “It’s python”
- “I like this project”
- “It’s a project I can do with my skills”
- “OWTF is the best project to learn about other tools/security”
- “Other mentors/org didn’t reply” (!)
- “Quick feedback/encouragement/advice”

Selected OWTF Proposals

- **Reporting**: Assem Chelli
- **Multiprocessing**: Ankush Jindal
- **MiTM Proxy**: Bharadwaj Machiraju
- **Testing Framework**: Alessandro Fanio González

Dedicated OWTF mentors

Without them 3 OWTF students would have been lost (GSoC 1 dedicated mentor x student rule):

Andrés Morales, Andrés Riancho, Azeddine Islam Mennouchi, Gareth Heyes, Hani Benhabiles, Javier Marcos de Prado, Johanna Curiel, Krzysztof Kotowicz, Martin Johns

THANK YOU for stepping up!
What is OWASP OWTF?
aka The Offensive (Web) Testing Framework
OWTF = Test/Exploit ASAP

- **Test Separation**
  - Start without permission

- **Automation**
  - Unite Tools, knowledge, Standards (OWASP, PTES)

- **Human analysis**
  - Information Management
OWTF’s Chess-like approach

Kasparov against Deep Blue - http://www.robotikka.com
OWTF Plugin Groups (-g)

- **web**: Try to cover the OWASP Testing Guide
  owtf.py http://demo.testfire.net (-g web: optional) ← **web only**
  owtf.py –l web ← List web plugins

- **net**: Somewhat like nmap scripts
  owtf.py demo.testfire.net (-g net: optional) ← **portscan + probe**
  NOTE: if a web service is found, web plugins will also run
  owtf.py –l net ← List net plugins

- **aux**: Somewhat like msfcli in metasploit
  owtf.py -f -o Targeted_Phishing
  SMTP_HOST=mail.pwnlabs.es SMTP_PORT=25
  SMTP_LOGIN=victim SMTP_PASS=victim EMAIL_FROM=sevena@pwnlabs.es
  EMAIL_PRIORITY=no EMAIL_SUBJECT='Test subject' EMAIL_BODY='test_body.txt'
  EMAIL_TARGET='victim@pwnlabs.es'
  ← **Phishing via SET**
  owtf.pl –l aux ← List aux plugins
Web Plugin Types (-t)

At least **50%** (32 out of 64) of the tests in the OWASP Testing guide can be legally* performed to some degree without permission

* Except in Spain, where visiting a page can be illegal 😊
* This is only my interpretation and not that of my employer + might not apply to your country!
OWTF Report = Chess-like Analysis

You need to understand this to use the OWTF report efficiently 😊

From Alexander Kotov - "Think like a Grandmaster":
1) Draw a list of candidate moves (3-4) ← 1st Sweep (!deep)
   1) Draw up a list of candidate paths of attack = rank what matters

2) Analyse each variation only once (!) ← 2nd Sweep (deep)
   2) Analyse [ tool output + other info ] once and only once

3) After step 1 and 2 make a move
   3) After 1) and 2) exploit the best path of attack

Ever analysed X in depth to only see “super-Y” later?
Demo 1: Admin interface

Pre-Engagement: No permission to test $\iff$ preparation

1) Run passive plugins $\iff$ legit + no traffic to target
   Sitefinity CMS found

2) Identify best path of attack:
   - Sitefinity default admin password
   - Public sitefinity shell upload exploits

Engagement: Permission to test $\iff$ exploitation

Try best path of attack first
Demo 1: Outcome

1 minute after getting permission …
Demo 1: Outcome

5 minutes after getting permission ...

El volumen de la unidad C no tiene etiqueta.
El número de serie del volumen es: 5CFC-2842

Directorio de c: \windows

14/02/2013 09:44

14/02/2013 09:44

03/03/2013 23:22
14/06/2002 18:46
17/07/2004 11:40
29/12/2006 00:31
24/08/2001 16:00
04/09/2007 17:14
24/08/2001 16:00
27/04/2008 18:13

0 0.log
19.274 000001_.tmp
19.528 002292_.tmp
19.569 002961_.tmp
17.336 A pescar.bmp
180.000 aaRemove.exe
26.680 Abanicos.bmp
Demo 2: Crossdomain

Attack preparation (pre-engagement safe) ← preparation
1) Run semi-passive plugins ← legit
   Missconfigured crossdomain, fingerprint wordpress version
2) Identify best path of attack:
   crossdomain + phishing + wordpress plugin upload + meterpreter
3) Replicate customer environment in lab
4) Prep attack: Adapt public payloads to target
5) Test in lab

Launching the attack ← exploitation
1) Tested attack works flawlessly on the first shot
2) Pivot
3) Show impact
OWTF Financials: Ideas plz 😊

Funding granted so far (THANK YOU Brucon + Google!):

- €5,000 – Brucon 5x5
  [http://blog.brucon.org/2013/02/the-5by5-race-is-on.html](http://blog.brucon.org/2013/02/the-5by5-race-is-on.html)
- $2,000 – GSoC ($500 x student)

What should we do with that money?
Status update on OWTF GSoC Projects
OWTF Reporting
by
Assem Chelli

Dedicated Mentor: Gareth Heyes (@garethheyes)
Co-mentors: Azeddine Islam Mennouchi, Hani Benhabiles, Johanna Curiel, Abraham Aranguren
Reporting Agenda

• Old report limitations
• Reporting goals
• Pre-implementation research
• Prototype voting/feedback
• Upcoming features
Old Report != Sexy 😞
Old report limitations

- Complicated + hard to understand
- Poor loading time of “big” reports (i.e. 30+ websites)
- Not cross-browser compatible (Firefox only)
- Inability to suit various screen sizes
- Not visually appealing :(
- Direct HTML generation from python code
Reporting Goals

• UI simplification + intuitiveness
• Better load time + responsiveness
• Cross-browser compatibility
• Improved screen size support (i.e. mobile users, etc)
• Improve visual appeal ← with community backing
• Build a skin system ← Users can choose/create skins
• Move HTML into template files:
  \texttt{!python = designer-friendly = more people can help us}
• Optimise click flow + mouse movement
Pre-implementation research

Twitter bootstrap gives us:

- Browser compatibility
- Pre-configured layouts
- Pre-defined styles
- Icon sets
- jQuery plugin integration
- Responsiveness + Simplicity
Pre-implementation research

**Jinja2** gives us:

- A python templating engine
- Python-like expressions
- Templates evaluated in a sandbox
Prototype Voting/Feedback

Demo 3: Online Survey Results

Want to vote? 😊 ← Shortcut: http://7-a.org + search “voting”
Survey:
https://docs.google.com/forms/d/1w613Y-rwPMw454k2oAd2M
Demo 4: Voted Prototype

WTF Summary Report v0.20

Summary  Home  Browse  Review  History  Targets (4)  Plugins (23)  Logs  Links  Skins

 Targets

- 195.251.127.254
- 209.85.148.141
- 209.85.148.84
- 65.81.137.117


General Info  OWASP Test Groups  Ports

...  Ports

content here

Upcoming Features (WIP)

- Implement skin system
- Implement chosen prototype
- Extraction of CSS/HTML into templates
- Sub-report loading via AJAX
- Default plugin vulnerability rankings
OWTF Multiprocessing

by

Ankush Jindal

Dedicated Mentor: Andrés Riancho (@w3af)
Co-mentor: Abraham Aranguren
Multiprocessing Agenda

- Multiprocessing goals
- Pre-implementation research
- Development challenges
- Net plugins demo
- Upcoming features
Multiprocessing Goals

- Reduce scanning time
- Port of OSCP scripts into OWTF $\leftarrow$ net plugins
- Scan multiple targets in parallel
- Rational usage of disk/RAM/CPU
- Stability + Reliability $= \neg$crash
- Identify + parallelise bottleneck components:
  
  Plugin execution, Reporting
Pre-Implementation Research

**Tested** candidate libraries:

<table>
<thead>
<tr>
<th>Library</th>
<th>Multiprocessing</th>
<th>Threading</th>
<th>gevent (distributed)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Shared Memory</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>

**Results:**
1. Shared memory led to incorrect results in legacy code
2. Multiprocessing performed better or approx. the same
3. Threading = GIL FUD on multiple-core machines 😊

**Conclusion:**
*Multiprocessing for plugins, Threading for smaller tasks*
Challenges during development:

- OWTF resets config on the fly via "Switch To Target"
  Solved via memory separation in multiprocessing

<table>
<thead>
<tr>
<th>Process 1</th>
<th>Process 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Config = Target 1</td>
<td>Config = Target 2</td>
</tr>
</tbody>
</table>

- Concurrent DB queries + no shared memory + File DB:
  Solved via dedicated DB process + messaging system + file locks for integrity
  (Processes perform DB reads+writes via messages)

- Implemented `ncurses` interface to stop OWTF

- Debugging unusual behaviour on concurrent processes 😊
Demo 5: Net Plugins

Port of the OSCP scripts into OWTF:
● Ping sweep + DNS zone transfers + port scanning
● Port scanning via nmap using “waves” (--portwaves)
  owtf.py --portwaves=10,100,1000 target.com
  First scan “top 10” ports, then “remaining until top 100”, ..
● Firing relevant **net** plugins depending on ports open

**Net plugins** implement:
● Vulnerability probing of network services (i.e. ftp, smtp,..)
Upcoming Features

- **Plugin profiling** for better resource usage:
  Monitor resources to determine “launchable” plugins depending on [load + expected resource consumption]

- **Reporter process**:
  To run in parallel + reduce report re-assembly iterations (i.e. instead of re-assemble once x plugin execution)

- **Identify + parallelise other bottleneck components**
OWTF MiTM Proxy
by
Bharadwaj Machiraju

Dedicated Mentor: Krzysztof Kotowicz (@kkotowicz)
Co-mentors: Javier Marcos de Prado, Martin Johns, Abraham Aranguren
MiTM Proxy Agenda

- MiTM Proxy Goals
- Pre-implementation research
- Development challenges
- Examples of working functionality 😊
- Performance benchmarks
- Upcoming features
MiTM Proxy Goals

- **Extended grep plugin coverage:**
  1) Data from manual browsing
  2) Data from proxified tools
- **Tool proxification** (if launched from OWTF)
- **SSL MiTM**
- **Proxy cache**: Avoid redundant requests
- **Request Throttling** based on target responsiveness
  (i.e. avoid unintended DoS)
- **Intelligent request retries**
  (i.e. ensure HTTP response retrieval where possible)
Pre-Implementation Research

● **Goal:**
   Select best python proxy framework ← best starting point

● **Test Cases:**
   Speed, HTTP Verb support, HTTP/1.1, HTTPS support, etc.

● **Frameworks:**
   Twisted, Mitmproxy, Tornado, Honeyproxy

● **Verdict:** *Tornado*
   Best [ performance + feature-set + reusability ]
Pre-Implementation Research
Development Challenges

- **Tornado**: Is a python web framework (!proxy)

<table>
<thead>
<tr>
<th>Pros</th>
<th>Cons</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scalability: Tens of thousands of connections</td>
<td>Not built to make proxy servers</td>
</tr>
<tr>
<td>Server + Client = Proxy</td>
<td>Client is more limited than server. <strong>Solution</strong>: Use tornado’s async curl client</td>
</tr>
</tbody>
</table>

- **SSL MiTM**: on-the-fly certificate generation, etc.
- **Proxy cache**: Race condition handling
- **Tool Proxification**: Not all tools could be proxified

*BUT Tool Proxification for tools with proxy CLI options IS working 😊*
Proxy SSL MiTM is working 😋
Proxy Cache is working 😊

(1) Request

INBOUND PROXY

(2) Request searched in cache

FILE CACHE /tmp/owtf-cache/

CACHE HIT

(3) Request present in cache
(4) Request fetched from cache
(5) Response returned from cache

CACHE MISS

(3) Request not present in cache
(4) A lock is acquired for writing the new cache file
(5) The new response is fetched, written and file lock released
(6) Response returned
Race-condition handling is working

- **Thread II**
  - Waiting for lock

- **Thread I**
  - Acquire lock
  - Cached file exists? No
    - Write tmp file.
    - Rename tmp file to cache file
    - Lock released & Response returned
  - Yes
    - Acquire lock
    - Lock released & Response returned

Cache file for the request exists from this point as it is created by thread-1.
Performance Benchmarks

Average request rate (requests/sec)

Proxies:
- Mitmproxy
- Twisted Proxy
- OWTF Proxy
- OWTF Multi-instance Proxy

Websites:
- facebook.com
- webscantest.com
- testfire.net
- vulnweb.com
- SSL test (facebook.com)
Upcoming features

- **Improved grep plugins**: Run on all transactions
- **Request Throttling** based on target responsiveness (i.e. avoid unintended DoS)
- **Intelligent request retries** (i.e. ensure HTTP response retrieval where possible)
- **Cookie based authentication**
  At proxy level = Ability to scan authenticated portions of a website.
- **Plug-n-Hack support**: Upcoming Mozilla standard
OWTF Testing Framework
by
Alessandro Fanio González

Dedicated Mentor: Andrés Morales Zamudio (@andresmz)
Co-mentor: Abraham Aranguren
Testing Framework Agenda

- Importance of testing
- Testing framework goals
- Pre-implementation research
- Development challenges
- Initial focus: Unit testing
- New focus: Functional testing
- Upcoming features
Importance of testing

• Improve code quality
• Ensure everything works as expected
• Prevent unintentional bugs:
  While developing new features or fixing other bugs
• Provide stability to the project
Testing Framework Goals

- Writing OWTF tests = As easy as possible
- Ensure OWTF integrity after code changes:

  1. Automated tests to verify OWTF modules behave as expected (unit tests)
  2. Automated tests to verify OWTF security test output is as expected (functional tests)
Pre-implementation research

Goals: ← Determine best starting point
1. Select best testing/mocking library ← for unit tests
2. Select best mock web server ← for functional tests

Tests:
1. Feature-set comparison among many mocking libraries
2. Reuse of Bharadwaj’s research (for mock web server)

Results:
1. Best mock library for OWTF = Flexmock
2. Best mock web server for OWTF = Tornado
Development Challenges

- Understand internal OWTF components
- Extend the testing library to complete features
- Make the testing framework easy to use:
  
  * Generate classes and methods dynamically, using metaclasses and introspection
- Fix broken tests due to fast-moving codebase

*Due to initial unit testing focus*
Initial focus: Unit testing

Important metric for unit testing = code coverage

Test coverage:
Number of executed lines of code after running all tests

When we run the entire test suite:
1. An HTML code coverage report is generated
2. Lines executed x file can be viewed in the report

Current OWTF code coverage = 58%
New focus: Functional testing

<table>
<thead>
<tr>
<th><strong>Unit test</strong> approach</th>
<th><strong>Functional test</strong> approach</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Pro</strong>: Fast</td>
<td><strong>Con</strong>: Slower</td>
</tr>
<tr>
<td><strong>Pro</strong>: Isolated</td>
<td><strong>Con</strong>: Not isolated</td>
</tr>
<tr>
<td><strong>Pro</strong>: Code coverage metrics (i.e. are we at 100% or not?)</td>
<td><strong>Con</strong>: No code coverage metrics</td>
</tr>
<tr>
<td><strong>Con</strong>: Harder to write (i.e. you kinda have to love/10et TDD 😊)</td>
<td><strong>Pro</strong>: Easier to write (i.e. closer to command-line usage)</td>
</tr>
<tr>
<td><strong>Con</strong>: Code dependent (i.e. refactoring = broken test)</td>
<td><strong>Pro</strong>: Code independent (i.e. refactoring != broken test)</td>
</tr>
<tr>
<td><strong>Con</strong>: Difficult to create tests for security edge cases (i.e. unusual web server behaviour)</td>
<td><strong>Pro</strong>: Easier to create tests for security edge cases (i.e. unusual web server behaviour)</td>
</tr>
<tr>
<td><strong>Con</strong>: Can’t find bugs due to third-party tools/incompatibilities</td>
<td><strong>Pro</strong>: Will find bugs due to third-party tools/incompatibilities</td>
</tr>
</tbody>
</table>
Demo 6: A testing example

Functional testing:

- Set the web server to return a custom robots.txt file, and start the server
- Write tests (almost) as if you were using OWTF from the command line: run the Spiders_Robots_and_Crawlers plugin
- Assert that the URLs contained in robots.txt are in the OWTF output

Unit testing:

- Show code coverage report from initial project focus
Upcoming features

**Functional tests for:**

1. **web** plugins: *OWASP Testing Guide coverage*
2. **net** and **aux** plugins: *PTES coverage*

- **Automated Continuous Integration:**
  
  *Run tests automatically after each commit*
Questions?

owtf.org
OWASP Testing Guide with OWASP OWTF
Context consideration:

**Case 1** → robots.txt Not Found

...should Google index a site like this?

Or should robots.txt exist and be like this?

**User-agent:** *

**Disallow:** /
Case 1 → robots.txt Not Found - Semi passive

- Direct request for robots.txt
- Without visiting entries

Spiders, Robots And Crawlers - SEMI PASSIVE

**HTTP TRANSACTIONS**

<table>
<thead>
<tr>
<th>REQUEST</th>
<th>RESPONSE</th>
</tr>
</thead>
<tbody>
<tr>
<td>GET /robots.txt HTTP/1.1</td>
<td>404 Not Found</td>
</tr>
</tbody>
</table>

See Transaction 3 (0s, 863ms)

```
GET /robots.txt HTTP/1.1
Accept-Encoding: identity
Host: demo.testfire.net
Connection: close
User-Agent: Mozilla/5.0 (X11; Linux 1686; rv:6.0) Gecko/20100101 Firefox/6.0

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN" "http://www.w3.org/TR/html4/strict.dtd">
<html>
<head>
<title>The page cannot be found</title>
<meta http-equiv="Content-Type" content="text/html; charset=Windows-1252">
<style type="text/css">
  BODY { font: 8pt/12pt verdana }
</style>
</head>
<body>
</body>
</html>
```
Case 2 → robots.txt Found - **Passive**
- **Indirect** Stats, Downloaded txt file for review, “Open All in Tabs”

---

**Spiders, Robots, and Crawlers - PASSIVE**

<table>
<thead>
<tr>
<th>PLUGIN</th>
<th>START</th>
<th>END</th>
<th>RUNTIME</th>
<th>OUTPUT FILES</th>
</tr>
</thead>
<tbody>
<tr>
<td>passive/Spiders_Robots_and_Crawlers@OWASP-IG-001.py</td>
<td>08/02/2012-13:37</td>
<td>08/02/2012-13:37</td>
<td>2s, 384ms</td>
<td>Browse</td>
</tr>
</tbody>
</table>

---

**NOTES**

---

**Passive Analysis Results:**
- Analysis via tool.motoricerca.info

**Online Resources:**
- Analysis via tool.motoricerca.info
- robots.txt via anonymous.org

---

**Raw regexp processing:**
robots.txt was found. 16 lines: 0 Allowed, 14 Disallowed, 0 Sitemap.
Saved to: owtf_review/195.251.127.254/80/http_hackademic1.teilar.gr/partial/Spiders_Robots_And_Crawlers/passive/robots1.txt

---

**Disallow Entries:**
- /administrator/
- /cache/
- /components/

---

**Open All In Tabs**
**OWTF HTML Filter challenge:** Embedding of untrusted third party HTML

**Defence layers:**
1) HTML Filter: Open source challenge
Filter 6 unchallenged since 04/02/2012, Can you hack it? 😊
http://blog.7-a.org/2012/01/embedding-untrusted-html-xss-challenge.html
2) HTML 5 sanboxed iframe
3) Storage in another directory = cannot access OWTF Review in localStorage

New Robots.txt Syntax Checker: a validator for robots.txt files

**Analyzing file** http://hackademic1.teilar.gr/robots.txt

**No errors found in this robots.txt file**

The following block of code DISALLOWS the crawling of the following files and directories: /administrator/
/cache/ /components/ /images/ /includes/ /installation/ /language/ /libraries/ /media/ /modules/ /plugins/
/templates/ /tmp/ /xmlrpc/ to all spiders/robots.

Line 5: Disallow: /images/
Start reporting!: Take your notes with fancy formatting

**Step 1** - Click the “Edit” link

**Step 2** - Start documenting findings + Ensure preview is ok

<table>
<thead>
<tr>
<th>Finding</th>
<th>URL</th>
<th>Detail</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fingerprint on not found error messages</td>
<td><a href="http://hackademic1.teilar.gr/installation/">http://hackademic1.teilar.gr/installation/</a></td>
<td>The requested URL /installation/ was not found on this server.</td>
</tr>
<tr>
<td>Joomla Login page</td>
<td><a href="http://hackademic1.teilar.gr/administrator/">http://hackademic1.teilar.gr/administrator/</a></td>
<td>Apache/2.2.17 (Fedora) Server at hackademic1.teilar.gr Port 80</td>
</tr>
</tbody>
</table>
Start reporting!: Paste PoC screenshots

Joomla Login page: http://hackademic1.teilar.gr/administrator/
The magic bar ;) – Useful to generate the human report later

Report for target: http://hackacademic1.teilar.gr

Findings

1. High Severity
   ○ Cross Site Flashing (OWASP-DV-004) - High Severity
     No notes found for any plugin under this category

2. Medium Severity
   ○ Spiders, Robots, and Crawlers (OWASP-IG-001) - Medium Severity
     A joomla administrator login URL was found at: http://hackacademic1.teilar.gr/administrator/
Passive Plugin

**Step 1** - Browse output files to review the full raw tool output:

<table>
<thead>
<tr>
<th>START</th>
<th>END</th>
<th>RUNTIME</th>
<th>OUTPUT FILES</th>
</tr>
</thead>
<tbody>
<tr>
<td>08/02/2012-13:37</td>
<td>08/02/2012-13:37</td>
<td>2s, 384ms</td>
<td>Browse</td>
</tr>
</tbody>
</table>

**Step 2** - Review tools run by the passive Search engine discovery plugin:

- MetaSploit_search_email_collector.txt 4 KB 08/02/2012 13:40:02
- TheHarvester.txt 6 KB 08/02/2012 13:39:04
- goohost_Google_search_Email.txt 08/02/2012 13:40:07
- goohost_Google_search_Host.txt 08/02/2012 13:40:05
- goohost_Google_search_IP.txt 1 KB 08/02/2012 13:40:06
- goohost_email_check.txt 08/02/2012 13:40:06
- goohost_host_check.txt 08/02/2012 13:40:03
- metasploit_emails.txt 1 KB 08/02/2012 13:40:02

Was your favourite tool not run?
Tell OWTF to run your tools on: **owtf_dir/profiles/resources/default.cfg** (backup first)
Tool output can also be reviewed via clicking through the OWTF report directly:

TEST COMMAND


THEHARVESTER OUTPUT (EXECUTION TIME: 1M, 20S, 906MS)

**************************************************************************
*TheHarvester Ver. 2.0 (reborn)  *
*Coded by Christian Martorella  *
*Edge-Security Research  *
*csmith@edge-security.com  *
**************************************************************************

Full harvest..
[-] Searching in Google..
    Searching 0 results...
    Searching 100 results...
    Searching 200 results...
    Searching 300 results...
    Searching 400 results...
    Searching 500 results...
    Searching 600 results...
    Searching 700 results...
    Searching 800 results...
    Searching 900 results...
    Searching 1000 results...
    Searching 1100 results...
    Searching 1200 results...
    Searching 1300 results...

NOTE: Output longer than 25 lines, Click here to see all output!
The Harvester:
- Emails
- Employee Names
- Subdomains
- Hostnames

http://www.edge-security.com/theHarvester.php
Metadata analysis:
- TODO: Integration with FOCA when CLI callable via wine (/cc @chemaalonso 😊)
- Implemented: Integration with Metagoofil

Search engine discovery/reconnaissance (OWASP-IG-002)

Search Engine Discovery Reconnaissance - SEMI PASSIVE

<table>
<thead>
<tr>
<th>PLUGIN</th>
<th>START</th>
<th>END</th>
</tr>
</thead>
<tbody>
<tr>
<td>semi_passive/Search_engine_discovery_reconnaissance@OWASP-IG-002.py</td>
<td>08/02/2012-13:44</td>
<td>08/02/2012-13:47</td>
</tr>
</tbody>
</table>

TEST COMMAND

```bash
```

METAGOOFIL OUTPUT (EXECUTION TIME: 2M, 49S, 581MS)

```
***************
* Metagoofil Ver 2.1 *
* Christian Martorella *
* Edge-Security.com *
* cmartorella_at_edge-security.com *
* Blackhat Arsenal Edition *
***************

[-] Starting online search...
[-] Searching for pdf files with a limit of 1500
```

http://www.edge-security.com/metagoofil.php
Inbound proxy not stable yet but all this happens automatically:
- robots.txt entries added to "Potential URLs"
- URLs found by tools are scraped + added to "Potential URLs"
During Active testing (later):
- "Potential URLs" visited + added to "Verified URLs" + Transaction log

**Verify URLs**

**Potential URLs**
- All URLs
- File URLs
- Fuzzable URLs
- Image URLs
- Error URLs
- External URLs

**Verified URLs**
- All URLs
- File URLs
- Fuzzable URLs
- Image URLs
- Error URLs
- External URLs
### All HTTP transactions logged by target in transaction log

**Step 1** - Click on “Transaction Log”

**Step 2** – Review transaction entries

<table>
<thead>
<tr>
<th>SCOPE</th>
<th>LINKS</th>
<th>ID</th>
<th>SECONDS</th>
<th>TIME</th>
<th>STATUS</th>
<th>METHOD</th>
<th>URL</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>F</td>
<td>3</td>
<td>0.4128510952</td>
<td>0s, 412ms</td>
<td>200 OK</td>
<td>GET</td>
<td><a href="http://hackademic1.teilar.gr/robots.txt">http://hackademic1.teilar.gr/robots.txt</a></td>
</tr>
<tr>
<td>T</td>
<td>F</td>
<td>4</td>
<td>0.542858839035</td>
<td>0s, 542ms</td>
<td>200 OK</td>
<td>OPTIONS</td>
<td><a href="http://hackademic1.teilar.gr">http://hackademic1.teilar.gr</a></td>
</tr>
</tbody>
</table>
Step 3 - Review raw transaction information (if desired)

HTTP URL
http://hackademic1.teilar.gr/robots.txt

HTTP Request
GET /robots.txt HTTP/1.1
Accept-Encoding: identity
Host: hackademic1.teilar.gr
Connection: close
User-Agent: Mozilla/5.0 (X11; Linux i686; rv:6.0) Gecko/20100101 Firefox/6.0

HTTP Response Headers
200 OK
Date: Wed, 08 Feb 2012 12:45:07 GMT
Server: Apache/2.2.17 (Fedora)
ETag: "2610a3-130-49e3c7fe84f00"
Accept-Ranges: bytes
Content-Length: 304
Connection: close
Content-Type: text/plain; charset=UTF-8

HTTP Response Body
User-agent: *
Disallow: /administrator/
Disallow: /cache/
Disallow: /components/
Disallow: /images/
Disallow: /includes/
Step 1 - Make all direct OWTF requests go through Outbound Proxy:
Passes all entry points to the tactical fuzzer for analysis later

root@bt:/tmp# /root/owtf/owtf.py -f -x 127.0.0.1:8080 -t semi_passive http://crackme.cenzic.com

Step 2 - Entry points can then also be analysed via tactical fuzzer:
Goal: What is that server running?

Manually verify request for fingerprint:

<table>
<thead>
<tr>
<th>HTTP TRANSACTIONS</th>
<th>RESPONSE</th>
</tr>
</thead>
<tbody>
<tr>
<td>REQUEST</td>
<td></td>
</tr>
<tr>
<td>See Transaction 7</td>
<td></td>
</tr>
<tr>
<td>0s, 446ms</td>
<td></td>
</tr>
<tr>
<td>Site</td>
<td></td>
</tr>
<tr>
<td>F</td>
<td></td>
</tr>
<tr>
<td>R</td>
<td></td>
</tr>
<tr>
<td>H</td>
<td></td>
</tr>
<tr>
<td>GET / HTTP/1.1</td>
<td>200 OK</td>
</tr>
<tr>
<td>Accept-Encoding:</td>
<td>Date: Wed, 08 Feb 2012 12:45:15 GMT</td>
</tr>
<tr>
<td>identity</td>
<td>Server: Apache/2.2.17 (Fedora)</td>
</tr>
<tr>
<td>Host: hackademi1.teilar.gr</td>
<td>X-Powered-By: PHP/5.3.8</td>
</tr>
<tr>
<td>Connection: close</td>
<td>Set-Cookie: 26238b056396bb02ea2977b17de46c4c=pcar7hv2fejn92v14nfo</td>
</tr>
<tr>
<td>User-Agent: Mozilla/5.0 (X11; Linux i686; rv:6.0) Gecko/20100118</td>
<td>P3P: CP=&quot;NOI ADM DEV PSAI COM NAV OUR OTRo STP IND DEM&quot;</td>
</tr>
<tr>
<td></td>
<td>Expires: Mon, 1 Jan 2001 00:00:00 GMT</td>
</tr>
<tr>
<td></td>
<td>Last-Modified: Wed, 08 Feb 2012 12:45:15 GMT</td>
</tr>
<tr>
<td></td>
<td>Cache-Control: no-store, no-cache, must-revalidate, post-check=0, pre-check=0</td>
</tr>
<tr>
<td></td>
<td>Pragma: no-cache</td>
</tr>
<tr>
<td></td>
<td>Content-Length: 7490</td>
</tr>
<tr>
<td></td>
<td>Connection: close</td>
</tr>
<tr>
<td></td>
<td>Content-Type: text/html; charset=utf-8</td>
</tr>
</tbody>
</table>
Whatweb integration with non-aggressive parameter (semi passive detection):

TEST COMMAND

cd owtf_review/195.251.127.254/80/http_hackademic1.teilar.gr/partial/Web_Application_Fingerprint
/semi_passive/; . /root/owtf_dev/scripts/setrubyenv.sh 1.8; /root/owtf_dev/tools/whatweb/whatweb-0.4.7/whatweb
--user-agent 'Mozilla/5.0 (X11; Linux i686; rv:6.0) Gecko/20100101 Firefox/6.0' --color=never --aggression 1
http://hackademic1.teilar.gr | sed "s/\|/\n/g"

WHATWEB SEMIPASSIVE CHECK (1 REQUEST) OUTPUT (EXECUTION TIME: 65, 749MS)

1.8
There are 2 choices for the alternative ruby (providing /usr/bin/ruby).

<table>
<thead>
<tr>
<th>Selection</th>
<th>Path</th>
<th>Priority</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>/usr/bin/ruby1.8</td>
<td>500</td>
<td>auto mode</td>
</tr>
<tr>
<td>* 1</td>
<td>/usr/bin/ruby1.8</td>
<td>500</td>
<td>manual mode</td>
</tr>
<tr>
<td>2</td>
<td>/usr/bin/ruby1.9.2</td>
<td>400</td>
<td>manual mode</td>
</tr>
</tbody>
</table>

Press enter to keep the current choice[*] or type selection number: http://hackademic1.teilar.gr [200] PasswordField[passwd]

MetaGenerator[Joomla! 1.5 - Open Source Content Management]
HTTPServer[Fedora Linux][Apache/2.2.17 (Fedora)]
Apache[2.2.17]
IP[195.251.127.254]
PHP[5.3.8]
X-Powered-By[PHP/5.3.8]
Joomla[1.5][com_content, com_user]
Cookies[26238b056396bb02ea2977b17de46c4c]
Title[Hackademic]
probably Mambo[com_content, com_user]
Country[GREECE][GR]

https://github.com/urbanadventurer/WhatWeb
Fingerprint header analysis: Match stats

Web Application Fingerprint - SEMI PASSIVE

<table>
<thead>
<tr>
<th>PLUGIN</th>
<th>START</th>
<th>END</th>
<th>RUNTIME</th>
</tr>
</thead>
<tbody>
<tr>
<td>semi_passive/Web_Application_Fingerprint@OWASP-IG-004.py</td>
<td>08/02/2012-13:44</td>
<td>08/02/2012-13:44</td>
<td>7s, 679ms</td>
</tr>
</tbody>
</table>

NOTES

Header Analysis Summary

<table>
<thead>
<tr>
<th>LOG</th>
<th>See log</th>
</tr>
</thead>
</table>

HTTP TRANSACTION STATS

5 out of 5 (100.0%) matched

ANALYSIS COMMAND

Convenient vulnerability search box (1 box per header found 😊): 
**Search All** → **Open all site searches in tabs**

## Header Value Analysis

NOTE: Only unique values per header are shown with a link to an example transaction

<table>
<thead>
<tr>
<th>HEADER</th>
<th>VALUES</th>
</tr>
</thead>
<tbody>
<tr>
<td>Server</td>
<td>Apache/2.2.17 (Fedora)</td>
</tr>
<tr>
<td>X-Powered-By</td>
<td>PHP/5.3.8</td>
</tr>
<tr>
<td>X-AspNet-Version</td>
<td>Not Found</td>
</tr>
<tr>
<td>X-Runtime</td>
<td>Not Found</td>
</tr>
<tr>
<td>X-Version</td>
<td>Not Found</td>
</tr>
<tr>
<td>MicrosoftSharePointTeamServices</td>
<td>Not Found</td>
</tr>
</tbody>
</table>

**SEARCH FOR VULNERABILITIES:** Apache/2.2.17 (Fedora)  
**SEARCH ALL**

NVD (High)  | OSVDB (High)  | BugTraq  | ExploitDB  | ExploitSearch (Exploits Only)  | ExploitSearch (All) |

**SEARCH FOR VULNERABILITIES:** PHP/5.3.8  
**SEARCH ALL**

NVD (High)  | OSVDB (High)  | BugTraq  | ExploitDB  | ExploitSearch (Exploits Only)  | ExploitSearch (All) |
<table>
<thead>
<tr>
<th>Date</th>
<th>D</th>
<th>A</th>
<th>V</th>
<th>Description</th>
<th>Plat.</th>
<th>Author</th>
</tr>
</thead>
<tbody>
<tr>
<td>2010-10-01</td>
<td></td>
<td></td>
<td>✓</td>
<td>Microsoft IIS 6.0 ASP Stack Overflow (Stack Exhaustion) Denial of Service (MS10-065)</td>
<td>8951</td>
<td>Kingcope</td>
</tr>
<tr>
<td>2010-08-14</td>
<td></td>
<td></td>
<td>✓</td>
<td>Sports Accelerator Suite v2.0 (news_id) Remote SQL Injection Vulnerability</td>
<td>1789</td>
<td>LiquidWorm</td>
</tr>
<tr>
<td>2009-01-29</td>
<td></td>
<td></td>
<td>✓</td>
<td>Full MSSQL Injection PWNage</td>
<td>1851</td>
<td>CWH Underground</td>
</tr>
<tr>
<td>2009-05-26</td>
<td></td>
<td></td>
<td>✓</td>
<td>Microsoft IIS 6.0 WebDAV Remote Authentication Bypass Exploit (pl)</td>
<td>5209</td>
<td>ka0x</td>
</tr>
<tr>
<td>2009-05-15</td>
<td></td>
<td></td>
<td>✓</td>
<td>Microsoft IIS 6.0 WebDAV Remote Authentication Bypass Vulnerability</td>
<td>3153</td>
<td>Kingcope</td>
</tr>
<tr>
<td>2008-11-28</td>
<td></td>
<td></td>
<td>✓</td>
<td>Web Calendar System &lt;= 3.40 (XSS/SQL) Multiple Remote Vulnerabilities</td>
<td>339</td>
<td>Bl@ckbe@rD</td>
</tr>
<tr>
<td>2008-11-17</td>
<td></td>
<td></td>
<td>✓</td>
<td>Q-Shop 3.0 Remote XSS/SQL Injection Vulnerabilities</td>
<td>455</td>
<td>Bl@ckbe@rD</td>
</tr>
</tbody>
</table>
Web Application Fingerprint (OWASP-IG-004)

NVD - http://web.nvd.nist.gov - CVSS Score = High

Search Results (Refine Search)

There are 8 matching records. Displaying matches 1 through 8.

**CVE-2010-1256**

**TA10-159B**

**Summary:** Unspecified vulnerability in Microsoft IIS 6.0, 7.0, and 7.5, when Extended Protection for Authentication is enabled, allows remote authenticated users to execute arbitrary code via unknown vectors related to "token checking" that trigger memory corruption, aka "IIS Authentication Memory Corruption Vulnerability."

**Published:** 06/08/2010

**CVSS Severity:** 8.5 (HIGH)

**CVE-2009-3023**

**TA09-286A VU#276653**

**Summary:** Buffer overflow in the FTP Service in Microsoft Internet Information Services (IIS) 5.0 through 6.0 allows remote authenticated users to execute arbitrary code via a crafted NLST (NAME LIST) command that uses wildcards, leading to memory corruption, aka "IIS FTP Service RCE and DoS Vulnerability."

**Published:** 08/31/2009

**CVSS Severity:** 9.3 (HIGH)

**CVE-2009-1535**

NVD is the U.S. government repository of standards based vulnerability management data. This data enables automation of vulnerability management, security measurement, and compliance (e.g. FISMA).
OSVDB - [http://osvdb.org](http://osvdb.org) - CVSS Score = High
Microsoft IIS Unicode Requests to WebDAV Multiple Authentication

www.securityfocus.com/bid/34993

Microsoft IIS ASP Remote Code Execution Vulnerability

www.securityfocus.com/bid/18858
ENTRY [METASPLOIT modules/exploits/windows/iis/ms01_026_dbldecode.rb] match

Microsoft IIS/PWS CGI Filename Double Decode Command Execution

This module will execute an arbitrary payload on a Microsoft IIS installation that is vulnerable to the CGI double-decode vulnerability of 2001. NOTE: This module will leave a metasploit payload in the IIS scripts directory.

Exploits
- METASPLOIT modules/exploits/windows/iis/ms01_026_dbldecode.rb - [Search]

References
- BID 2708 - [Search]
- CVE-2001-0333 - [Search]
- MS01-026 - [Search]
- OSVDB 556 - [Search]

Initial Date Seen [2011-07-15 15:33:35]
Last Date Updated [2011-07-15 15:33:35]
Passive Fingerprint analysis

<table>
<thead>
<tr>
<th>Plugin</th>
<th>Start</th>
<th>End</th>
<th>Runtime</th>
<th>Output File</th>
</tr>
</thead>
<tbody>
<tr>
<td>passive/Web_Application_Fingerprint@OWASP-IG-004.py</td>
<td>08/02/2012-13:37</td>
<td>08/02/2012-13:37</td>
<td>0s, 19ms</td>
<td>Browse</td>
</tr>
</tbody>
</table>

Search for Vulnerabilities:

- NVD (High)
- OSVDB (High)
- BugTraq
- ExploitDB
- ExploitSearch (Exploits Only)
- ExploitSearch (All)
- NVD (All)
- OSVDB (All)

Online Resources: [Open All In Tabs]

- centralops.net TCP Query
- netcraft.com General
- netcraft.com Uptime
- whois.webhosting.info Banner
- www.shodanhq.com
- builtwith.com
## Site report for zero.webappsecurity.com

<table>
<thead>
<tr>
<th>Site</th>
<th><a href="http://zero.webappsecurity.com">http://zero.webappsecurity.com</a></th>
</tr>
</thead>
<tbody>
<tr>
<td>Last reboot</td>
<td>unknown</td>
</tr>
<tr>
<td>Domain</td>
<td>webappsecurity.com</td>
</tr>
<tr>
<td>Netblock owner</td>
<td>Hewlett-Packard Company</td>
</tr>
<tr>
<td>IP address</td>
<td>15.216.12.12</td>
</tr>
<tr>
<td>Site rank</td>
<td>143078</td>
</tr>
<tr>
<td>Country</td>
<td>US</td>
</tr>
<tr>
<td>Nameserver</td>
<td>ns1.inflow.net</td>
</tr>
<tr>
<td>Date first seen</td>
<td>April 2004</td>
</tr>
<tr>
<td>DNS admin</td>
<td><a href="mailto:dnsadmin@inflow.net">dnsadmin@inflow.net</a></td>
</tr>
<tr>
<td>Domain Registrar</td>
<td>markmonitor.com</td>
</tr>
<tr>
<td>Reverse DNS</td>
<td>zero-g1w2555g.austin.hp.com</td>
</tr>
<tr>
<td>Organisation</td>
<td>Hewlett-Packard Company, 3000 Hanover St., United States</td>
</tr>
<tr>
<td>Nameserver Organisation</td>
<td>SunGard Data Systems Inc., PO Box 459ATTN INFLOW.NET, care of Network Solutions, Drums, Panama</td>
</tr>
</tbody>
</table>

**Hosting History**

<table>
<thead>
<tr>
<th>Netblock Owner</th>
<th>IP address</th>
<th>OS</th>
<th>Web Server</th>
<th>Last changed</th>
</tr>
</thead>
</table>
Content Management Systems

Blogger
Blogger Usage Statistics - Websites using Blogger
Google Blogger Software.

JavaScript Libraries

Google JS Api
Google JS Api Usage Statistics - Websites using Google JS Api
Google Mashup Editor (GME) includes a JavaScript API that gives you direct access to the document object model (DOM) via JavaScript. This API lets you use JavaScript to perform operations that duplicate and go beyond the features available in the GME tags. The API is useful when you want to access an object in the application from a JavaScript expression. You can also use the API to perform CRUD operations (create, read, update, delete) on entries in a data feed.

Widgets

Google Plus One
Google Plus One Usage Statistics - Websites using Google Plus One
Google’s answer to Facebook Like.

Lightbox
Lightbox Usage Statistics - Websites using Lightbox
Lightbox JS is a simple, unobtrusive script used to overlay images on the current page. It’s a snap to setup and works on all modern browsers.

http://builtwith.com
Search in the headers without touching the site:

```
HTTP/1.0 302
Date: Tue, 11 Oct 2011 02:04:01 GMT
Server: Apache
X-Powered-By: PHP/4.4.9
Location: http://www.google.com
Transfer-Encoding: chunked
Content-Type: text/html
```

http://www.shodanhq.com/
Passive suggestions
- Prepare your test in a terminal window to hit “Enter” on “permission minute 1”

CMS Fingerprint - Potentially useful commands

**WPSCAN PLUGIN ENUMERATION (WORDPRESS)**

```
```

**CMS EXPLORER PLUGIN ENUMERATION (WORDPRESS)**

```
```

**DIRBUSTER WORDPRESS ALL**

```
```

**DIRBUSTER WORDPRESS PLUGINS**

```
```

**DIRBUSTER WORDPRESS THEMES**

```
```
What else can be done with a fingerprint?
Environment replication
Download it .. Sometimes from project page 😊

Also check http://www.oldapps.com/, Google, etc.
Static Analysis, Fuzz, Try exploits, ..

RIPS for PHP: http://rips-scanner.sourceforge.net/
Yasca for most other (also PHP): http://www.scovetta.com/yasca.html
Questions?

owtf.org
<table>
<thead>
<tr>
<th>Plugin</th>
<th>Start</th>
<th>End</th>
<th>Runtime</th>
<th>Output Files</th>
</tr>
</thead>
<tbody>
<tr>
<td>passive/Application_Discovery@OWASP-IG-005.py</td>
<td>08/02/2012-13:37</td>
<td>08/02/2012-13:37</td>
<td>0s, 15ms</td>
<td>Browse</td>
</tr>
</tbody>
</table>

**Online Resources:**

- Hurricane Electric TOP Domain DNS records
- Hurricane Electric Host Name DNS records
- whois.webhosting.info (Virtual Hosts)
- intodns.com
- www.robtext.com
- centralops.net TCP Query
- centralops.net Domain Dossier
- centralops.net AutoWhois
- centralops.net Ping
- centralops.net NsLookup
- dnsgoodies.com SMTP Open Relay
- dnsgoodies.com Spam DB Check
- dnsgoodies.com Abuse Lookup
zero.webappsecurity.com

cName

zero-pro.austin.hp.com

A

15.216.12.12

A

zerog1w2555g.austin.hp.com

A

15.216.0.0/16

www.robtex.com - Passive DNS Discovery
Reverse Whois: "Domain Administrator" owns about 416,674 other domains
Email Search: hp.domains@hp.com is associated with about 3,108 domains
hostmaster@hp.com is associated with about 1,414 domains

Registrar History: 2 registrars
NS History: 5 changes on 2 unique name servers over 9 years.
IP History: 5 changes on 4 unique IP addresses over 7 years.
Whois History: 45 records have been archived since 2004-04-01.

Log In or Create a FREE account to start monitoring this domain name

Registrant:
Domain Administrator
Hewlett-Packard Company
3000 Hanover St.
Palo Alto CA 94304
US
hp.domains@hp.com +1.8005247638 Fax: +1.6508522936

http://whois.domaintools.com
<table>
<thead>
<tr>
<th>Service</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>FTP - 21</td>
<td>Error: TimedOut</td>
</tr>
<tr>
<td>SMTP - 25</td>
<td>Error: TimedOut</td>
</tr>
</tbody>
</table>
| HTTP - 80 | HTTP/1.1 302 Object moved  
Connection: close  
Date: Tue, 15 Nov 2011 08:57:10 GMT  
Server: Microsoft-IIS/6.0  
X-Powered-By: ASP.NET  
Location: banklogin.asp?serviceName=FreebankCaastAd  
AD_REFERRING_URL=http://www.Freebank.com  
Content-Length: 263  
Content-Type: text/html  
Set-Cookie: ASPSESSIONIDAATCACC=LMEDKIAEPKAGOFAMO  
Cache-control: private |
| POP3 - 110 | Error: TimedOut |
| IMAP - 143 | Error: TimedOut |
Has Google found error messages for you?

### Testing For Error Code - PASSIVE

<table>
<thead>
<tr>
<th>PLUGIN</th>
<th>START</th>
</tr>
</thead>
<tbody>
<tr>
<td>passive/Testing_for_Error_Code@OWASP-IG-006.py</td>
<td>08/02/2012</td>
</tr>
</tbody>
</table>

### Online Resources:
- [hexillion.com For Passive Verification Queries](#)
- [Google Search (Errors in title)](#)
- [Google Search (Errors in body)](#)

---

**Testing for Error Code (OWASP-IG-006)**
Invalid Data Please try again.
zero.webappsecurity.com/rootlogin.asp
Invalid Data Please try again.
Invalid Data Please try again.
zero.webappsecurity.com/rootlogin.asp?txtPassPhrase...
Invalid Data Please try again.
The Test Page
zero.webappsecurity.com/test/test.html
LOGIC CHECKS WORKED. The welcome page · Error logs.

Check errors via Google Cache
### Testing For Ssl-Tls - PASSIVE

<table>
<thead>
<tr>
<th>PLUGIN</th>
<th>START</th>
<th>END</th>
</tr>
</thead>
<tbody>
<tr>
<td>passive/Testing_for_SSL-TLS@OWASP-CM-001.py</td>
<td>08/02/2012-13:37</td>
<td>08/02/2012-13:37</td>
</tr>
</tbody>
</table>

**Online Resources:**

[www.ssllabs.com](http://www.ssllabs.com)
The link is generated with OWTF with that box ticked: Important!

SSL Server Test

This free online service performs a deep analysis of the configuration of any SSL web server on the public Internet. Please note that the information you submit here is used only to provide you the service. We don't use the domain name you submit, we don't store the test results, and we never will.

Domain name: 

Do not show the results on the boards

https://www.ssllabs.com/ssldb/analyze.html
Pretty graphs to copy-paste to your OWTF report 😊

https://www.ssllabs.com/ssldb/analyze.html
Do not forget about **Strict-Transport-Security**!

sslstrip chances decrease dramatically:

**Only 1st time user visits the site!**

---

This plugin looks for server-side protection headers to enforce SSL

### Header Analysis Summary

<table>
<thead>
<tr>
<th>HTTP TRANSACTION STATS</th>
<th>0 out of 197 (0.0%) matched</th>
</tr>
</thead>
</table>
Not found example:

Header Value Analysis

NOTE: Only unique values per header are shown with a link to an example transaction

<table>
<thead>
<tr>
<th>HEADER</th>
<th>VALUES</th>
</tr>
</thead>
<tbody>
<tr>
<td>Strict-Transport-Security</td>
<td>Not Found</td>
</tr>
</tbody>
</table>

Found example:

Header Analysis Summary

<table>
<thead>
<tr>
<th>LOG</th>
<th>See log</th>
</tr>
</thead>
<tbody>
<tr>
<td>HTTP TRANSACTION STATS</td>
<td>2 out of 5 (40.0%) matched</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ANALYSIS COMMAND</th>
</tr>
</thead>
<tbody>
<tr>
<td>grep -IH &quot;(Strict-Transport-Security): &quot; owtf_review/173.194.65.84 /443/https_accounts.google.com /transactions/response_headers /scope_*</td>
</tr>
</tbody>
</table>

Header Value Analysis

NOTE: Only unique values per header are shown with a link to an example transaction

<table>
<thead>
<tr>
<th>HEADER</th>
<th>VALUES</th>
</tr>
</thead>
<tbody>
<tr>
<td>Strict-Transport-Security</td>
<td>max-age=2592000; includeSubDomains</td>
</tr>
</tbody>
</table>
**HTML content analysis: HTML Comments**

<table>
<thead>
<tr>
<th>PLUGIN</th>
<th>START</th>
<th>END</th>
<th>RUNTIME</th>
</tr>
</thead>
<tbody>
<tr>
<td>grep/Application_Configuration_Management@OWASP-CM-004.py</td>
<td>02/03/2012-08:24</td>
<td>02/03/2012-08:24</td>
<td>0s, 874ms</td>
</tr>
</tbody>
</table>

**NOTES**

### HTML Comments

**STATS**
- 17 Unique HTML Comments found
- 52 out of 197 (26.0%) transactions matched

**HTML COMMENTS**
- Unique as TEXT
- Unique as HTML
- All as HTML

**COMMAND**
grep -IHIE "<!--"

**LOG**
[See log]
Efficient HTML content matches analysis

Step 1 - Click Unique as TEXT

Step 2 - Human Review of Unique matches
Efficient HTML content matches analysis

**Step 1 - Click** Unique as HTML

**Step 2 - Review Unique matches (click on links for sample match info)**

<table>
<thead>
<tr>
<th>ID</th>
<th>Links</th>
<th>Match</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>&lt;span&gt;Site F&lt;/span&gt;</td>
<td>&lt;!--[if lt IE 7.]&gt; &lt;link href=&quot;/templates/blackbearpro/css/ie6.css&quot; rel=&quot;stylesheet&quot; type=&quot;text/css&quot;/&gt; --&gt;</td>
</tr>
<tr>
<td>10</td>
<td>&lt;span&gt;Site F&lt;/span&gt;</td>
<td>&lt;!-- #content { padding-left:0px; width: 600px; } #container { background-image: url(/images/body.png); } --&gt;</td>
</tr>
<tr>
<td>186</td>
<td>&lt;span&gt;Site F&lt;/span&gt;</td>
<td>&lt;!--[if IE 7]&gt; &lt;link href=&quot;/templates/khepri/css/ie7.css&quot; rel=&quot;stylesheet&quot; type=&quot;text/css&quot;/&gt; --&gt;</td>
</tr>
<tr>
<td>186</td>
<td>&lt;span&gt;Site F&lt;/span&gt;</td>
<td>&lt;!--[if lte IE 6]&gt; &lt;link href=&quot;/templates/khepri/css/ie6.css&quot; rel=&quot;stylesheet&quot; type=&quot;text/css&quot;/&gt; --&gt;</td>
</tr>
<tr>
<td>192</td>
<td>&lt;span&gt;Site F&lt;/span&gt;</td>
<td>&lt;!--[if lt IE 7.]&gt; &lt;link href=&quot;/gr/templates/blackbearpro/css/ie6.css&quot; rel=&quot;stylesheet&quot; type=&quot;text/css&quot;/&gt; &lt;!--[endif]--&gt;</td>
</tr>
<tr>
<td>192</td>
<td>&lt;span&gt;Site F&lt;/span&gt;</td>
<td>&lt;!-- #content { padding-left:0px; width: 600px; } #container { background-image: url(/images/body.png); } --&gt;</td>
</tr>
</tbody>
</table>

Want to see all? then click All as HTML
**HTML content analysis:** CSS and JavaScript Comments /* */

<table>
<thead>
<tr>
<th>CSS/JS Comments</th>
<th></th>
</tr>
</thead>
</table>
| **STATS**       | - 12 Unique CSS/JS Comments found  
|                 | - 3 out of 197 (1.0%) transactions matched  |

<table>
<thead>
<tr>
<th>CSS/JS COMMENTS</th>
<th></th>
</tr>
</thead>
</table>
|                 | - Unique as TEXT  
|                 | - Unique as HTML  
|                 | - All as HTML  |

<table>
<thead>
<tr>
<th>COMMAND</th>
<th></th>
</tr>
</thead>
</table>

| LOG | See log |
HTML content analysis: Single line JavaScript Comments (///)

<table>
<thead>
<tr>
<th>STATS</th>
</tr>
</thead>
<tbody>
<tr>
<td>• 0 Unique Single Line JS Comments found</td>
</tr>
<tr>
<td>• 0 out of 197 (0.0%) transactions matched</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SINGLE LINE JS COMMENTS</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Unique as TEXT</td>
</tr>
<tr>
<td>• Unique as HTML</td>
</tr>
<tr>
<td>• All as HTML</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>COMMAND</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>LOG</th>
</tr>
</thead>
<tbody>
<tr>
<td>See log</td>
</tr>
</tbody>
</table>
**HTML content analysis:** PHP source code

| STATS              | 0 Unique Potential PHP source code found  
|                   | 0 out of 197 (0.0%) transactions matched |
| POTENTIAL PHP SOURCE CODE | Unique as TEXT  
|                   | Unique as HTML  
|                   | All as HTML     |
| LOG               | See log         |
**HTML content analysis:** ASP source code

| STATS                          | • 0 Unique Potential ASP source code found  
|                               | • 0 out of 197 (0.0%) transactions matched |
| POTENTIAL ASP SOURCE CODE     | • Unique as TEXT  
|                               | • Unique as HTML  
|                               | • All as HTML     |
| LOG                           | See log          |
## Old Backup And Unreferenced Files - PASSIVE

<table>
<thead>
<tr>
<th>Plugin</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>passive/Old_Backup_and_Unreferenced_Files@OWASP-CM-006.py</td>
<td></td>
</tr>
</tbody>
</table>

### Online Resources:

- Google Search (Logs, Passwords, Juicy stuff)
- Google Search (Email files)
- Google Search (Source code, DB Dumps, Other)
- Google Search (Obscure extensions)
- Google Search (Directory Indexing)
**Old Backup And Unreferenced Files - GREP**

<table>
<thead>
<tr>
<th>PLUGIN</th>
<th>START</th>
<th>END</th>
</tr>
</thead>
<tbody>
<tr>
<td>grep/Old_Backup_and_Unreferenced_Files@OWASP-CM-006.py</td>
<td>09/02/2012-08:32</td>
<td>09/02/2012-08:32</td>
</tr>
</tbody>
</table>

**NOTES**

This plugin shows all URLs classified as 'Files' for review, there could be cool stuff here :)

All known File URLs in Scope:  

- [http://demo.testfire.net/admin/clients.xls](http://demo.testfire.net/admin/clients.xls)
- [http://demo.testfire.net/pr/communityannualreport.pdf](http://demo.testfire.net/pr/communityannualreport.pdf)
Testing For Admin Interfaces - *PASSIVE*

<table>
<thead>
<tr>
<th>PLUGIN</th>
<th>START</th>
</tr>
</thead>
<tbody>
<tr>
<td>passive/Testing_for_AdminInterfaces@OWASP-CM-007.py</td>
<td>08/02/2012-13:3</td>
</tr>
</tbody>
</table>

Online Resources: [Open All In Tabs](#)

- Google Search *(phpmyadmin,admin,backend,private,secret,login,logon)*
- Google Search *(username,login,password)*
If you find an admin interface don’t forget to..
Google for default passwords:

```
sitefinity "by default" password
```

About 5,770 results (0.20 seconds)

**Sitefinity Watch > How to secure Sitefinity’s Administrative UI**

4 Mar 2010 – Users are then required to provide a valid username & password to gain entry to Sitefinity. By default, Sitefinity’s administrative username is set to **admin**.

How to secure Sitefinity’s Administrative UI

Thursday, March 04, 2010

Sitefinity’s Administrative Web Interface is accessed by adding `/Sitefinity` to the web site’s URL. Users are then required to provide a valid username & password to gain entry to Sitefinity. By default, Sitefinity’s administrative username is set to **admin**.

A few customers have expressed concern that this does not offer enough protection from malicious users or bots. If an attacker knows a web site is using Sitefinity then they also know the login URL and the **admin** username. The only thing that remains is the **admin password**.
Disclaimer: Permission is required for this
<table>
<thead>
<tr>
<th>PLUGIN</th>
<th>START</th>
<th>END</th>
<th>RUNTIME</th>
<th>OUTPUT</th>
</tr>
</thead>
<tbody>
<tr>
<td>semi_passive/HTTP_Methods_and_XST@OWASP-CM-008.py</td>
<td>08/02/2012-13:44</td>
<td>08/02/2012-13:44</td>
<td>1s, 230ms</td>
<td>Bro</td>
</tr>
</tbody>
</table>

**HTTP TRANSACTIONS**

<table>
<thead>
<tr>
<th>REQUEST</th>
<th>RESPONSE</th>
</tr>
</thead>
<tbody>
<tr>
<td>OPTIONS / HTTP/1.1</td>
<td>200 OK</td>
</tr>
<tr>
<td>See Transaction 4 (0s, 403ms)</td>
<td>Allow: OPTIONS, TRACE, GET, HEAD</td>
</tr>
<tr>
<td>Site</td>
<td>Content-Length: 0</td>
</tr>
<tr>
<td>F</td>
<td>Server: Microsoft-IIS/6.0</td>
</tr>
<tr>
<td>R</td>
<td>Public: OPTIONS, TRACE, GET, HEAD, POST</td>
</tr>
<tr>
<td>H</td>
<td>X-Powered-By: ASP.NET</td>
</tr>
<tr>
<td>B</td>
<td>Date: Wed, 08 Feb 2012 14:26:09 GMT</td>
</tr>
<tr>
<td>User-Agent: Mozilla/5.0 (X11; Linux i686; rv:6.0) Gecko/20100101</td>
<td>Connection: close</td>
</tr>
<tr>
<td>PLUGIN</td>
<td>START</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>passive/HTTP_Methods_and_XST@OWASP-CM-008.py</td>
<td>08/02/2011</td>
</tr>
</tbody>
</table>

**NOTES**

Online Resources:  
- [hexillion.com OPTIONS check](#)  
- [hexillion.com TRACE check](#)
Querying zero.webappsecurity.com [15.216.12.12]...

[begin response]

HTTP/1.1 200 OK
Content-Length: 111
Content-Type: message/http
Server: Microsoft-IIS/6.0
X-Powered-By: ASP.NET
Date: Tue, 15 Nov 2011 08:36:26 GMT
Connection: close

TRACE / HTTP/1.0
Host: zero.webappsecurity.com
User-Agent: AspTcpQuery sample (http://www.hexillion.com/)

[end response]
Is the login page on “http” instead of “https”?

This plugin looks for password fields and then checks the URL (i.e. http vs. https) uniqueness in this case is performed via URL + password field
Total insecure matches: 53

**Password fields**

<table>
<thead>
<tr>
<th>STATS</th>
</tr>
</thead>
<tbody>
<tr>
<td>• 47 Unique Password fields found</td>
</tr>
<tr>
<td>• 52 out of 197 (26.0%) transactions matched</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>PASSWORD FIELDS</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Unique as TEXT</td>
</tr>
<tr>
<td>• Unique as HTML</td>
</tr>
<tr>
<td>• All as HTML</td>
</tr>
</tbody>
</table>
Pro Tip: When browsing the site manually...
... **look** carefully at pop-ups like this:

Consider (i.e. **prep the attack**):

**Firesheep**: [http://codebutler.github.com/firesheep/](http://codebutler.github.com/firesheep/)

**SSLStrip**: [https://github.com/moxie0/sslstrip](https://github.com/moxie0/sslstrip)
Mario was going to report a bug to Mozilla and found another!
Abuse user/member public search functions:

- Search for "" (nothing) or "a", then "b", ..
- Download all the data using 1) + pagination (if any)
- Merge the results into a CSV-like format
- Import + save as a spreadsheet
- Show the spreadsheet to your customer
Analyse the username(s) they gave you to test:

• Username based on numbers?
  USER12345
• Username based on public info? (i.e. names, surnames, ..)
  name.surname
• Default CMS user/pass?
Part 1 - Remember Password: Autocomplete

<table>
<thead>
<tr>
<th>Good</th>
<th>Bad</th>
</tr>
</thead>
<tbody>
<tr>
<td>Via 1) <code>&lt;form ... autocomplete=&quot;off&quot;&gt;</code></td>
<td><code>&lt;form action=&quot;/user/login&quot; method=&quot;post&quot;&gt;</code></td>
</tr>
<tr>
<td>Or Via 2) <code>&lt;input ... autocomplete=&quot;off&quot;&gt;</code></td>
<td><code>&lt;input type=&quot;password&quot; name=&quot;pass&quot; /&gt;</code></td>
</tr>
</tbody>
</table>

This plugin looks for password and form tags to review the autocomplete attribute

**Autocomplete fields**

- 12 Unique Autocomplete fields found
- 52 out of 197 (26.0%) transactions matched

**Stats**

**Autocomplete fields**

- Unique as TEXT
- Unique as HTML
- All as HTML

```
grep -IHie "type=.password" owtf_review/195.251.127.254
```
Manual verification for password autocomplete (i.e. for the customer)

Easy “your grandma can do it” test:
1. Login
2. Logout
3. Click the browser Back button twice*
4. Can you login again –without typing the login or password- by re-

Can the user re-submit the login form via the back button?
* Until the login form submission

Other sensitive fields: Pentester manual verification
- Credit card fields
- Password hint fields
- Other
Part 2 - Password Reset forms

Manually look at the questions / fields in the password reset form

- Does it let you specify your email address?
- Is it based on public info? (name, surname, etc)
- Does it send an email to a potentially dead email address you can register? (i.e. hotmail.com)
Goal: Is Caching of sensitive info allowed?

Manual verification steps: “your grandma can do it” 😊 (need login):
1. Login
2. Logout
3. Click the browser Back button
4. Do you see logged in content or a this page has expired error / the login page?

Manual analysis tools:
• Commands: curl -i http://target.com
• Proxy: Burp, ZAP, WebScarab, etc
• Browser Plugins:

### HTTP/1.1 headers

<table>
<thead>
<tr>
<th>Good</th>
<th>Bad</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cache-Control: no-cache</td>
<td>Cache-control: private</td>
</tr>
</tbody>
</table>

### HTTP/1.0 headers

<table>
<thead>
<tr>
<th>Good</th>
<th>Bad</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pragma: no-cache</td>
<td>Pragma: private</td>
</tr>
<tr>
<td>Expires: &lt;past date or illegal (e.g. 0)&gt;</td>
<td>Expires: &lt;way too far in the future&gt;</td>
</tr>
</tbody>
</table>

### The world

<table>
<thead>
<tr>
<th>Good</th>
<th>Bad</th>
</tr>
</thead>
<tbody>
<tr>
<td><a href="https://accounts.google.com">https://accounts.google.com</a></td>
<td>No caching headers = caching allowed</td>
</tr>
<tr>
<td>Cache-control: no-cache, no-store</td>
<td>HTTP/1.1 200 OK</td>
</tr>
<tr>
<td>Pragma: no-cache</td>
<td>Date: Tue, 09 Aug 2011 13:38:43 GMT</td>
</tr>
<tr>
<td>Expires: Mon, 01-Jan-1990 00:00:00 GMT</td>
<td>Server: .....</td>
</tr>
<tr>
<td></td>
<td>X-Powered-By: .....</td>
</tr>
<tr>
<td></td>
<td>Connection: close</td>
</tr>
<tr>
<td></td>
<td>Content-Type: text/html; charset=UTF-8</td>
</tr>
</tbody>
</table>
This plugin looks for server-side protection headers and tags against cache snooping.

**Header Analysis Summary**

**Log**

See log

**HTTP Transaction Stats**

53 out of 197 (26.0%) matched

**Analysis Command**

grep -HiE "(Cache-Control|Pragma|Expires): "
owtf_review/195.251.127.254
/80/http_hackademic1.teilar.gr
/transactions/response_headers
/scope * | sed -e
's|owtf_review/195.251.127.254||g'
-e s|response_headers//|/|g'

**Header Value Analysis**

NOTE: Only unique values per header are shown with a link to an example transaction

<table>
<thead>
<tr>
<th>HEADER</th>
<th>VALUES</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cache-Control</td>
<td>no-store, no-cache, must-revalidate, post-check=0, pre-check=0</td>
</tr>
<tr>
<td>Pragma</td>
<td>no-cache</td>
</tr>
<tr>
<td>Expires</td>
<td>Mon, 1 Jan 2001 00:00:00 GMT</td>
</tr>
</tbody>
</table>
### Cache Control Meta Tags

<table>
<thead>
<tr>
<th>Good</th>
<th>Bad</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;META HTTP-EQUIV=&quot;Cache-Control&quot; CONTENT=&quot;no-cache&quot;&gt;</code></td>
<td><code>&lt;META HTTP-EQUIV=&quot;Cache-Control&quot; CONTENT=&quot;private&quot;</code></td>
</tr>
</tbody>
</table>

#### Stats
- 0 Unique Cache Control Meta Tags found
- 0 out of 197 (0.0%) transactions matched

#### Cache Control Meta Tags
- Unique as TEXT
- Unique as HTML
- All as HTML

#### Command
```
grep -iH "<META.*?HTTP-EQUIV" owtf_review/195.251.127.254/80/http_hackademic1.teilar.gr/transactions/response_bodies/scope_*/ | cut -f1 -d:\|sort -u
```
Step 1 - Find CAPTCHAs: Passive search

<table>
<thead>
<tr>
<th>PLUGIN</th>
<th>START</th>
<th>END</th>
</tr>
</thead>
<tbody>
<tr>
<td>passive/Testing_for_Captcha@OWASP-AT-008.py</td>
<td>08/02/2012-13:37</td>
<td></td>
</tr>
</tbody>
</table>

Online Resources:

- Google Search (captcha, security code)
**Offline Manual analysis:**
- Download image and try to break it
- Are CAPTCHAs reused?
- Is a hash or token passed? (Good algorithm? Predictable?)
- Look for vulns on CAPTCHA version

**CAPTCHA breaking tools**
PWNtcha - captcha decoder - [http://caca.zoy.org/wiki/PWNtcha](http://caca.zoy.org/wiki/PWNtcha)
Captcha Breaker - [http://churchturing.org/captcha-dist/](http://churchturing.org/captcha-dist/)
### Manually Examine cookies for weaknesses offline

<table>
<thead>
<tr>
<th>Base64 Encoding (=! Encryption 😊)</th>
<th>Decoded value</th>
</tr>
</thead>
<tbody>
<tr>
<td>MTkyLjE2OC4xMDAuMTpvd2FzcHVzZXI6cGFzc3dvcmQ6MTU6NTg=</td>
<td>owaspuser:192.168.100.1:a7656fafe94dae72b1e1487670148412</td>
</tr>
</tbody>
</table>

### Session Management Schema - EXTERNAL

<table>
<thead>
<tr>
<th>PLUGIN</th>
<th>START</th>
<th>END</th>
<th>RU</th>
</tr>
</thead>
<tbody>
<tr>
<td>external/Session_Management_Schema@OWASP-SM-001.py</td>
<td>03/03/2012-07:15</td>
<td>03/03/2012-07:15</td>
<td>0s</td>
</tr>
</tbody>
</table>

### Online Resources:
- [Open All In Tabs](#)
- [Gareth Hayes' HackVertor](#)
- [Raul Siles' (Taddong) F5 BIG IP Cookie Decoder](#)
Questions?

owtf.org
Natural language conversion
Convert this to hex then octal

You are not logged in. You can still view everyone's public tags but you need to register.

Input 100 100

$http://hackvertor.co.uk/public$
Lots of decode options, including:

- auto_decode
- auto_decode_repeat
- d_base64
- etc.

Output:

```
192.168.100.1:owaspuser:password:15:58
```

http://hackvertor.co.uk/public
F5 BIG-IP Cookie decoder:

```
root@bt:~# ./BIG-IP_cookie_decoder.py 1677787402.36895.0000
[*] String to decode: 1677787402.36895.0000
[*] Decoded IP: 10.1.1.100
[*] Decoded port: 8080
```

Cookies Attributes (OWASP-SM-002)

- **Secure**: not set = session cookie leaked = pwned
- **HttpOnly**: not set = cookies stealable via JS
- **Domain**: set properly
- **Expires**: set reasonably
- **Path**: set to the right /sub-application
- **1 session cookie that works is enough ..**
## Cookies Attributes (OWASP-SM-002)

### Header Value Analysis

NOTE: Only unique values per header are shown with a link to an example transaction

<table>
<thead>
<tr>
<th>HEADER</th>
<th>VALUES</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>7bf9911fab0c9735a81838a8466b569d=na02mmgho6p9jisslen9v3t6o5; path=/</td>
</tr>
<tr>
<td></td>
<td>26238b056396bb02ea2977b17de46c4c=3h20bvblbinmrfti751kgmf94; path=/</td>
</tr>
<tr>
<td></td>
<td>26238b056396bb02ea2977b17de46c4c=e5to3mpc56qdgfj61o9rlghf3g3; path=/</td>
</tr>
<tr>
<td></td>
<td>26238b056396bb02ea2977b17de46c4c=i4t79up0lp1kl4oihpa0n3uf20; path=/</td>
</tr>
<tr>
<td></td>
<td>74d4eed8cbb936df5ee62291facacd8c=4k03b9r77mdrvhp7ukr23s0td5; path=/</td>
</tr>
<tr>
<td></td>
<td>26238b056396bb02ea2977b17de46c4c=p9hf1fu9069pq9j56dci465ra2; path=/</td>
</tr>
</tbody>
</table>

### Cookie Attribute Analysis

<table>
<thead>
<tr>
<th>COOKIE: 7BF9911FAB0C9735A81838A8466B569D</th>
</tr>
</thead>
<tbody>
<tr>
<td>ATTRIBUTE</td>
</tr>
<tr>
<td>Value</td>
</tr>
<tr>
<td>secure</td>
</tr>
<tr>
<td>HttpOnly</td>
</tr>
<tr>
<td>domain</td>
</tr>
<tr>
<td>path</td>
</tr>
<tr>
<td>expires</td>
</tr>
</tbody>
</table>
**Session Fixation (OWASP-SM-003)**

**Manually check when verifying credentials during pre-engagement:**
Login and analyse the Session ID cookie (i.e. PHPSESSID)

<table>
<thead>
<tr>
<th>Good</th>
<th>Bad (normal + by default)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Before: 10a966616e8ed63f7a9b741f80e65e3c</td>
<td>Before: 10a966616e8ed63f7a9b741f80e65e3c</td>
</tr>
<tr>
<td>After: Nao2mxgho6p9jisslen9v3t6o5f943h</td>
<td>After: 10a966616e8ed63f7a9b741f80e65e3c</td>
</tr>
</tbody>
</table>

**IMPORTANT:** You can also set the session ID via JavaScript (i.e. XSS)
Exposed Session Variables (OWASP-SM-004)

Session ID:
• In URL
• In POST
• In HTML

Example from the field:
http://target.com/xxx/xyz.function?session_num=7785

Bypassing Authorization Schema (OWASP-AZ-002)

Look at unauthenticated cross-site requests:

http://other-site.com/user=3&report=4
Referer: site.com

Change ids in application: (ids you have permission for!)
http://site.com/view_doc=4
Headers Enabling/Disabling Client-Side XSS filters:

- **X-XSS-Protection** (IE-Only)
- **X-Content-Security-Policy** (FF >= 4.0 + Chrome >= 13)
Review JavaScript code on the page:

```html
<script>
document.write("Site is at: " + document.location.href + ").");
</script>

Sometimes active testing possible in your browser (no trip to server = not an attack = not logged):
http://target.com/...#vulnerable_param=xss

Did Google find SQLi for you?

sql OR syntax OR error site:zero.webappsecurity.com

7 results (0.11 seconds)

LSWEB General Access Error Log
zero.webappsecurity.com/errors/errors.log
File Format: Unrecognized - View as HTML

My ERROR - zero.webappsecurity.com (HP)
zero.webappsecurity.com/error.html
Error Diagnostic Information The welcome page.
<!--#exec cmd="/bin/ls /" -->
<!--#INCLUDE VIRTUAL="/web.config"-->
1. Browse Site
2. Time requests
3. Get top X slowest requests
4. Slowest = Best DoS target
Google searches: inurl:wsdl site:example.com

Public services search:
http://seekda.com/
http://www.wsindex.org/
http://www.soapclient.com/

Online Resources: Open All In Tabs

Google Search (Web Services)
wsindex.org
www.soapclient.com
www.xmethods.net
WSDL analysis

Sensitive methods in WSDL?
i.e. Download DB, Test DB, Get CC, etc.
http://www.example.com/ws/FindIP.asmx?WSDL

<wsdl:operation name="getCreditCard" parameterOrder="id">
  <wsdl:input message="impl:getCreditCardRequest"
            name="getCreditCardRequest"/>
  <wsdl:output message="impl:getCreditCardResponse"
              name="getCreditCardResponse"/>
</wsdl:operation>
1. Domain A’s page can send a request to Domain B’s page from Browser
2. BUT Domain A’s page cannot read Domain B’s page from Browser
Testing for CSRF (OWASP-SM-005)

- Request == Predictable → Pwned → “..can send a request to Domain B” (SOP)

**CSRF Protection 101:**
- Require long random token (99% hidden anti-CSRF token) → Not predictable
- Attacker cannot read the token from Domain B (SOP) → Domain B ignores request

<table>
<thead>
<tr>
<th>Potentially Good</th>
<th>Bad</th>
</tr>
</thead>
<tbody>
<tr>
<td>Anti-CSRF token present: Verify with permission</td>
<td>No anti-CSRF token</td>
</tr>
</tbody>
</table>

**Testing For Csrfd - GREP**

<table>
<thead>
<tr>
<th>PLUGIN</th>
<th>START</th>
<th>END</th>
<th>RUNTIME</th>
<th>OUTPUT</th>
</tr>
</thead>
<tbody>
<tr>
<td>grep/Testing_for_CSRF@OWASP-SM-005.py</td>
<td>02/03/2012-10:46</td>
<td>02/03/2012-10:46</td>
<td>0s, 397ms</td>
<td>Browse</td>
</tr>
</tbody>
</table>

**Hidden fields**

**STATS**

- 99 Unique Hidden fields found
- 52 out of 197 (26.0%) transactions matched

**HIDDEN FIELDS**

- Unique as TEXT
- Unique as HTML
- All as HTML
Similar to CSRF:
Is there an anti-replay token in the request?

<table>
<thead>
<tr>
<th>Potentially Good</th>
<th>Bad</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Anti-CSRF token present</strong>: Verify <strong>with</strong> permission</td>
<td><strong>No anti-CSRF token</strong></td>
</tr>
</tbody>
</table>
1) Passive search for Flash/Silverlight files + policies:

**Testing For Cross Site Flashing - PASSIVE**

<table>
<thead>
<tr>
<th>PLUGIN</th>
<th>START</th>
<th>END</th>
<th>R UI</th>
</tr>
</thead>
<tbody>
<tr>
<td>passive/Testing_for_Cross_site_flashed@OWASP-DV-004.py</td>
<td>08/02/2012-13:37</td>
<td>08/02/2012-13:37</td>
<td>0s</td>
</tr>
</tbody>
</table>

**NOTES**

Online Resources: **Open All In Tabs**

- Google Search (SWF Files)
- Google Search (Silverlight Files)
- Google Search (crossdomain.xml, clientaccesspolicy.xml Files)

Flash file search:

`filetype:swf site:adobe.com`

About 12,300 results (0.13 seconds)

Silverlight file search:

`filetype:xap OR filetype:scr site:microsoft.com`

2 results (0.19 seconds)

**Communications:** Standby Continuous Replication in Exchange ...
Static analysis: Download + decompile Flash files

$ flare hello.swf

```javascript
onClipEvent (enterFrame) {
    if (this._y > -254) {
        this._y += -3;
    }
    if (this._y > -254 and this._y < -175.3) {
        this._yscale -= 0;
        this._xscale -= 0;
    } else {
        if (this._y <= -157.7) {
            this. vscale -= 2;
        }
    }
}
```

Flare: http://www.nowrap.de/flare.html
Flasm (timelines, etc): http://www.nowrap.de/flasm.html
Static analysis tools

Adobe SWF Investigator
http://labs.adobe.com/technologies/swfinvestigator

SWFScan

Summary

Examination of the ActionScript revealed that it does not make use of any browser communication APIs. When embedding this SWF in an HTML page one should set the AllowNetworkingAccess flag to internal. This will implicitly disable the SWF applications communication ability to the browser.

When a SWF is embedded within HTML, there are several flags which inform the Flash player if the SWF file should have access to content from the browser or from the network. The AllowNetworkingAccess flag tells the Flash player to disallow the SWF...
Active testing 😊
1) Trip to server = need permission
   http://target.com/test.swf?xss=foo&xss2=bar

2) But ... your browser is yours:
   No trip to server = no permission needed
   http://target.com/test.swf#?xss=foo&xss2=bar

Good news: Unlike DOM XSS, the # trick will always work for Flash Files
Some technologies allow settings that relax SOP:
• Adobe Flash (via policy file)
• Microsoft Silverlight (via policy file)
• HTML 5 Cross Origin Resource Sharing (via HTTP headers)

Cheating: Reading the policy file or HTTP headers ≠ attack

http://www.adobe.com/devnet/flashplayer/articles/fplayer9_security.html
Policy file retrieval for analysis

Testing For Cross Site Flashing - SEMI PASSIVE

<table>
<thead>
<tr>
<th>PLUGIN</th>
<th>START</th>
<th>END</th>
</tr>
</thead>
<tbody>
<tr>
<td>semi_passive/Testing_for_Cross_site_flashin...</td>
<td>08/02/2012-13:44</td>
<td>08/02/2012-13:44</td>
</tr>
</tbody>
</table>

NOTES

HTTP://HACKADEMIC1.TEILAR.GR/CROSSDOMAIN.XML

Not Found

HTTP://HACKADEMIC1.TEILAR.GR/CLIENTACCESSPOLICY.XML

Not Found

HTTP TRANSACTIONS

REQUEST

See Transaction 5 (0s, 245ms) Site F

R H B

GET /crossdomain.xml HTTP/1.1
Accept-Encoding: identity
Host: hackademic1.teilar.gr
Connection: close
User-Agent: Mozilla/5.0 (X11; Linux i686; rv:6.0) Gecko/20

RESPONSE

404 Not Found
Date: Wed, 08 Feb 2012 12:45:13 GMT
Server: Apache/2.2.17 (Fedora)
Content-Length: 300
Connection: close
Content-Type: text/html; charset=iso-8859-1

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 2.0//EN">
<html>
<head>
<title>404 Not Found</title>
</head>
<body>
Cross Site Flashing (OWASP-DV-004)

CSRF by design → read tokens = attacker WIN

Flash / Silverlight - crossdomain.xml

```xml
<cross-domain-policy>
  <allow-access-from domain="*"/>
</cross-domain-policy>
```

Bad defence example: restrict pushing headers accepted by Flash:
All headers from any domain accepted

```xml
<allow-http-request-headers-from domain="*" headers="*"/>
```

Flash: http://kb2.adobe.com/cps/403/kb403185.html
CSRF by design → read tokens = attacker WIN

Silverlight - clientaccesspolicy.xml

```xml
<?xml version="1.0" encoding="utf-8"?>
<access-policy>
  <cross-domain-access>
    <policy>
      <allow-from http-request-headers="SOAPAction">
        <domain uri="*"/>
      </allow-from>
      <grant-to><resource path="/" include-subpaths="true"/></grant-to>
    </policy>
  </cross-domain-access>
</access-policy>
```

Silverlight: http://msdn.microsoft.com/en-us/library/cc197955%28v=vs.95%29.aspx
Cross Site Flashing (OWASP-DV-004)

Testing For Cross Site Flashing - EXTERNAL

<table>
<thead>
<tr>
<th>PLUGIN</th>
<th>START</th>
<th>END</th>
</tr>
</thead>
<tbody>
<tr>
<td>external/Testing_for_Cross_site_flash@OWASP-DV-004.py</td>
<td>08/02/2012-13:37</td>
<td></td>
</tr>
</tbody>
</table>

NOTES

Online Resources: [Open All In Tabs]

- Krzysztof Kotowicz's CORS proxy browser
- Erlend Oftedal's MalaRIA proxy for crossdomain.xml + clientaccesspolicy.xml
- Julien Couvreur's PoC via URL
- Craft Flash file for Free via Haxe
- Mario Heiderich's sample Haxe file
- Silverlight's clientaccesspolicy.xml info
- crossdomain.xml explained
- fscommand to call JavaScript from Flash
Workshop exercise

1) **Install swftools:**
   wget http://www.swftools.org/swftools-0.9.2.tar.gz
   tar xvfz swftools-0.9.2.tar.gz
   cd swftools-0.9.2
   sh ./configure
   make
   make install
   whereis swfdump ➞ Check that we have swfdump installed now
   swfdump: /usr/local/bin/swfdump
Workshop exercise (continued)

2) Analyse vulnerable file:

wget http://demo.testfire.net/vulnerable.swf ← Download vulnerable file
swfdump -a vulnerable.swf > vulnerable.txt ← Disassemble flash file
grep -B1 GetVariable vulnerable.txt|tr " " "\n"|grep '('|sort -u ← Get FlashVars
("empty_mc")
("externallInterfaceVar")
("flash")
("font")
("fontTxtFieldExists")
("fontVar")
("getUrlBlankVar")
("getUrlJSParam")
("getUrlParentVar") ← Used in this example
...
Workshop exercise (continued)

3) Verify using the “#” trick (payload not sent to target):

http://demo.testfire.net/vulnerable.swf#

getURLParentVar=javascript:alert('pwned!')

And you get:

XSS 😊
Cross Origin Resource Sharing (CORS) (OWTF-WGP-002)

This plugin looks for HTML 5 Cross Origin Resource Sharing (CORS) headers

**Header Analysis Summary**

<table>
<thead>
<tr>
<th>LOG</th>
<th>See log</th>
</tr>
</thead>
<tbody>
<tr>
<td>HTTP TRANSACTION STATS</td>
<td>0 out of 74 (0.0%) matched</td>
</tr>
<tr>
<td>ANALYSIS COMMAND</td>
<td>grep -IH &quot;(Access-Control-Allow-Origin</td>
</tr>
</tbody>
</table>

**Header Value Analysis**

NOTE: Only unique values per header are shown with a link to an example transaction

<table>
<thead>
<tr>
<th>HEADER</th>
<th>VALUES</th>
</tr>
</thead>
<tbody>
<tr>
<td>Access-Control-Allow-Origin</td>
<td>Not Found</td>
</tr>
<tr>
<td>Access-Control-Allow-Credentials</td>
<td>Not Found</td>
</tr>
</tbody>
</table>
UI Redressing protections:

- **X-Frame-Options** *(best)*
- **X-Content-Security-Policy** *(FF >= 4.0 + Chrome >= 13)*
- **JavaScript Frame busting** *(bypassable sometimes)*

<table>
<thead>
<tr>
<th>Good</th>
<th>Bad</th>
</tr>
</thead>
<tbody>
<tr>
<td>X-Frame-Options: Deny</td>
<td></td>
</tr>
</tbody>
</table>

This plugin looks for server-side protection headers against Clickjacking (TODO: Add rudimentary search for frame busting)

**Header Analysis Summary**

<table>
<thead>
<tr>
<th>LOG</th>
<th>See log</th>
</tr>
</thead>
<tbody>
<tr>
<td>HTTP TRANSACTION STATS</td>
<td>0 out of 74 (0.0%) matched</td>
</tr>
</tbody>
</table>
Andrew Horton’s “Clickjacking for Shells”:
http://www.morningstarsecurity.com/research/clickjacking-wordpress

Krzysztof Kotowicz’s “Something Wicked this way comes”:
http://www.slideshare.net/kkotowicz/html5-something-wicked-this-way-comes-hackpra
https://connect.ruhr-uni-bochum.de/p3g2butmrt4/

Marcus Niemietz’s “UI Redressing and Clickjacking”:
Special thanks to

Adi Mutu (@an_animal), Alessandro Fanio González, Anant Shrivastava, Andrés Morales, Andrés Riancho (@w3af), Ankush Jindal, Assem Chelli, Azeddine Islam Mennouchi, Bharadwaj Machiraju, Chris John Riley, Gareth Heyes (@garethheyes), Hani Benhabiles, Javier Marcos de Prado, Johanna Curiel, Krzysztof Kotowicz (@kkotowicz), Marc Wickenden (@marcwickenden), Marcus Niemietz (@mniemietz), Mario Heiderich (@0x6D6172696F), Martin Johns, Michael Kohl (@citizen428), Nicolas Grégoire (@Agarri_FR), Sandro Gauci (@sandrogauci), OWASP Testing Guide contributors

All those OWTF students that tried to participate in the GSoC even if they couldn’t make it this time 😞

Finux Tech Weekly – Episode 17 – mins 31-49
Finux Tech Weekly – Episode 12 – mins 33-38
Exotic Liability – Episode 83 – mins 49-53
http://exoticliability.libsyn.com/exotic-liability-83-oh-yeah
Q & A

Contact/Links:
http://owtf.org
@7a_ @owtfp
abraham.aranguren@owasp.org