# Web Application Hacking Lesson XSRF part 2

**Lesson Objectives**

- Review lecture on the XSRF(below)
- Complete lab on Cross Site Request Forgery(below)

**Cross Site Request Forgery Lecture:**

Cross Site Request Forgery is very dangerous, and also quite common. OWASP describes Cross Site Request Forgery as:

Cross-Site Request Forgery (CSRF) is an attack that tricks the victim into loading a page that contains a malicious request. It is malicious in the sense that it inherits the identity and privileges of the victim to perform an undesired function on the victim's behalf, like change the victim's e-mail address, home address, or password, or purchase something. CSRF attacks generally target functions that cause a state change on the server but can also be used to access sensitive data.

For most sites, browsers will automatically include with such requests any credentials associated with the site, such as the user's session cookie, basic auth credentials, IP address, Windows domain credentials, etc. Therefore, if the user is currently authenticated to the site, the site will have no way to distinguish this from a legitimate user request.

In this way, the attacker can make the victim perform actions that they didn't intend to, such as logout, purchase item, change account information, retrieve account information, or any other function provided by the vulnerable website.

In other words, this is one reason why people tell you not to click on links (or open e-mails) from people you do not trust. Just by clicking a link they can steal your information or change your password without you knowing about it.

**Lab for Cross Site Request Forgery Introduction**

The Attack:

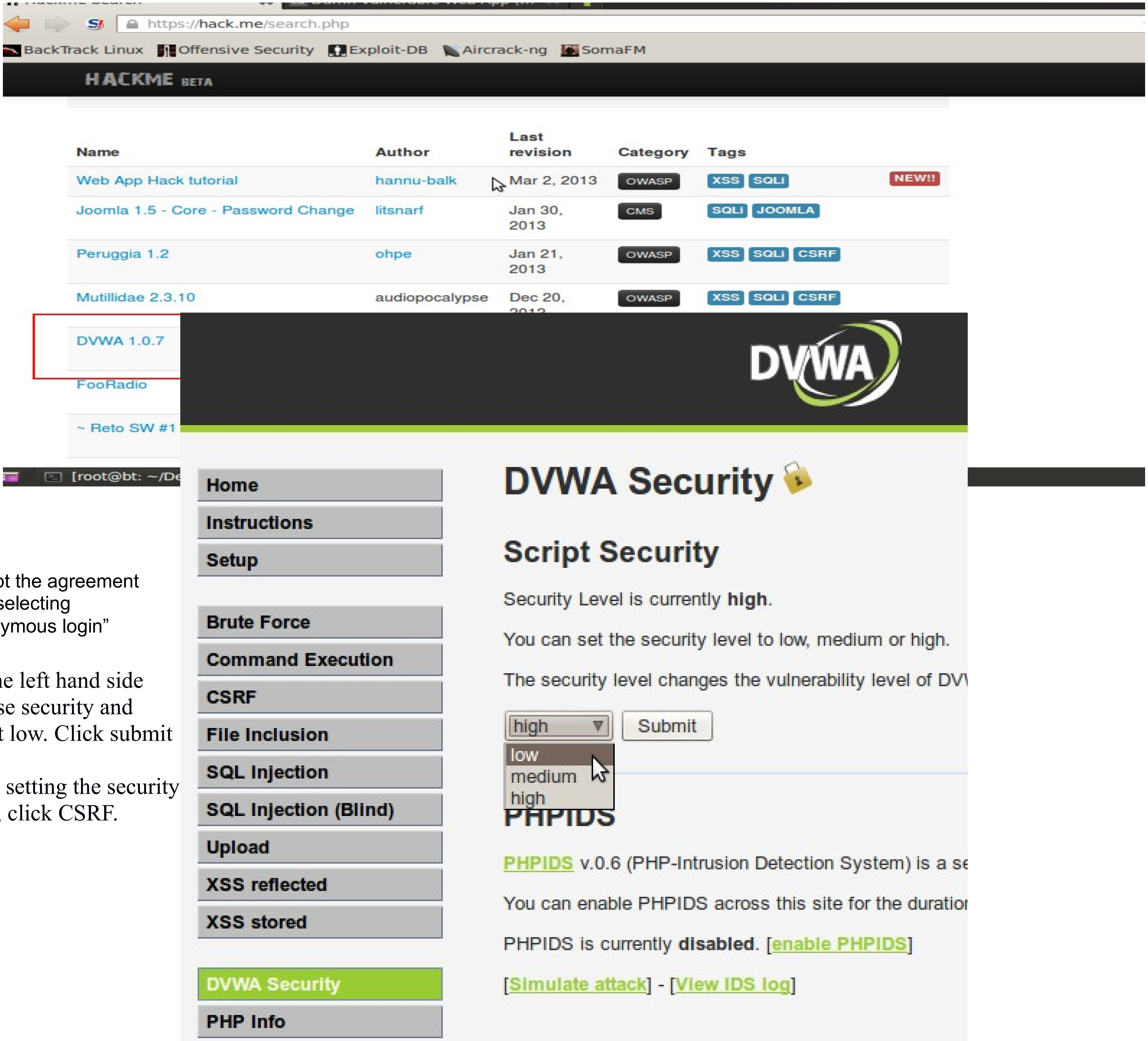
We are going to use Burp Suite to capture the HTTP request when we try to change a password, and from that we will create a "hidden" link to send to a victim (via e-mail, IM, etc.) that will change their password to whatever we want so we can access their account.

Lab Prep

Open the link to https://hack.me

Choose “Start a hackme”

Scroll down and select “DVWA 1.0.7”



https://hack.me/search.php

| Name | Author | Last revision | Category | Tags |
|---|---|---|---|---|
| Web App Hack tutorial | hannu-balk | Mar 2, 2013 | OWASP | XSS SQLI NEW!! |
| Joomla 1.5 - Core - Password Change | litsnarf | Jan 30, 2013 | CMS | SQLI JOOMLA |
| Peruggia 1.2 | ohpe | Jan 21, 2013 | OWASP | XSS SQLI CSRF |
| Mutillidae 2.3.10 | audiopocalypse | Dec 20, 2012 | OWASP | XSS SQLI CSRF |
| DVWA 1.0.7 | | | | |
| FooRadio | | | | |
| ~ Reto SW #1 | | | | |

[root@bt: ~/De

## DVWA Security

### Script Security

Security Level is currently **high**.

You can set the security level to low, medium or high.

The security level changes the vulnerability level of DV

high ▼ Submit

### PHPIDS

PHPIDS v.0.6 (PHP-Intrusion Detection System) is a se

You can enable PHPIDS across this site for the duratio

PHPIDS is currently **disabled**. [enable PHPIDS]

[Simulate attack] - [View IDS log]

Accept the agreement after selecting “anonymous login”

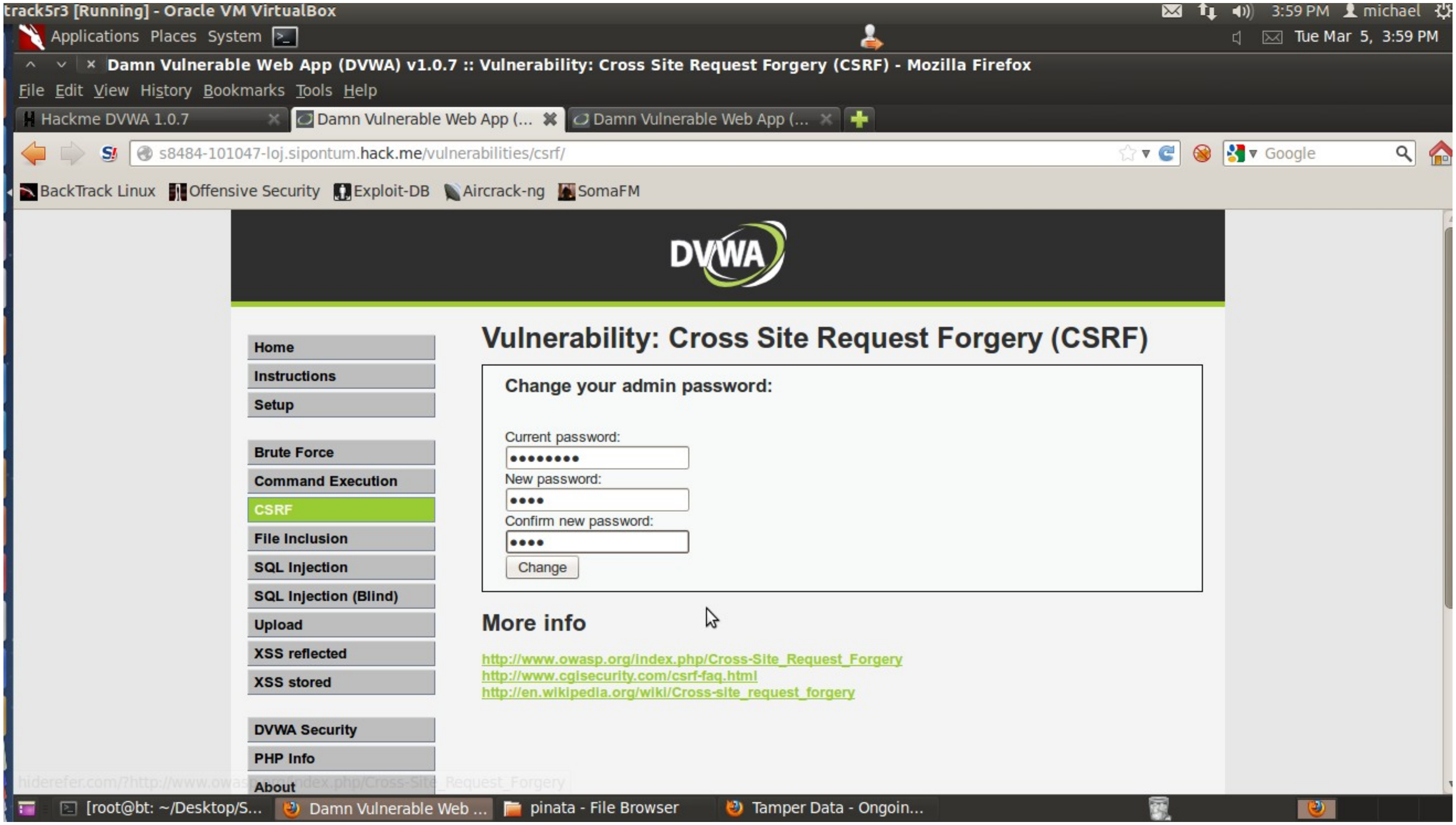
On the left hand side choose security and select low. Click submit

After setting the security level, click CSRF.

Lab 1– CSRF via GET requests

Launch ZAP 2.0

1. in the directory that holds the zap file, type:  java -jar zap.jar
2. Open firefox and configure it to use the ZAP 2.0 Proxy  http://www.youtube.com/watch?v=Xp_PBH7wjiw&list=PLEBitBW-Hlsv8cEIUntAO8st2UGhmrjUB&index=2
3. Refresh the site so that it ZAP 2.0 lists the site on the left hand panel
4. Go to the CSRF page in DVWA and Change your admin password by entering a password in the New password and Confirm new password fields and clicking the Change button.



track5r3 [Running] - Oracle VM VirtualBox

Damn Vulnerable Web App (DVWA) v1.0.7 :: Vulnerability: Cross Site Request Forgery (CSRF) - Mozilla Firefox

s8484-101047-loj.sipontum.hack.me/vulnerabilities/csrf/

Vulnerability: Cross Site Request Forgery (CSRF)

Change your admin password:

Current password: •••••••••

New password: ••••

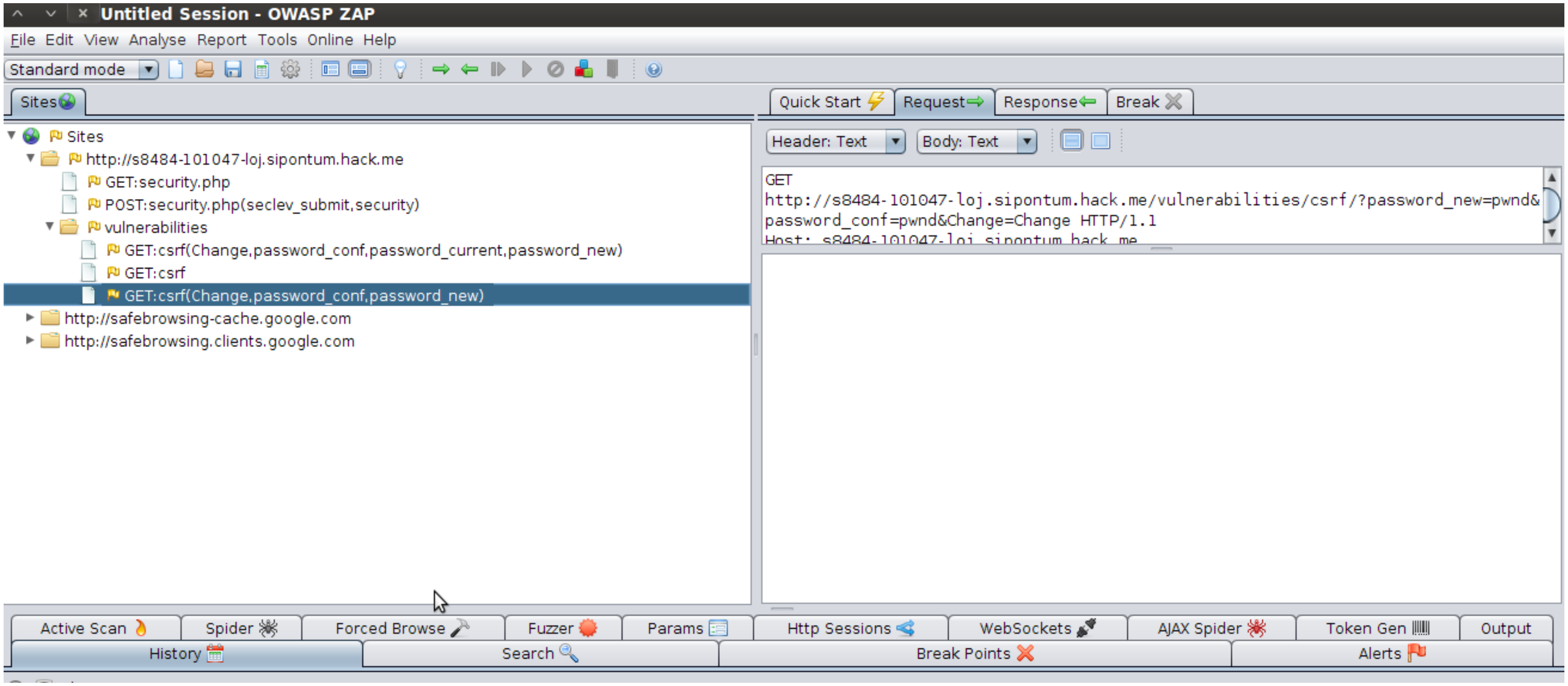
Confirm new password: ••••

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

hiderefer.com/?http://www.owasp.org/index.php/Cross-Site_Request_Forgery

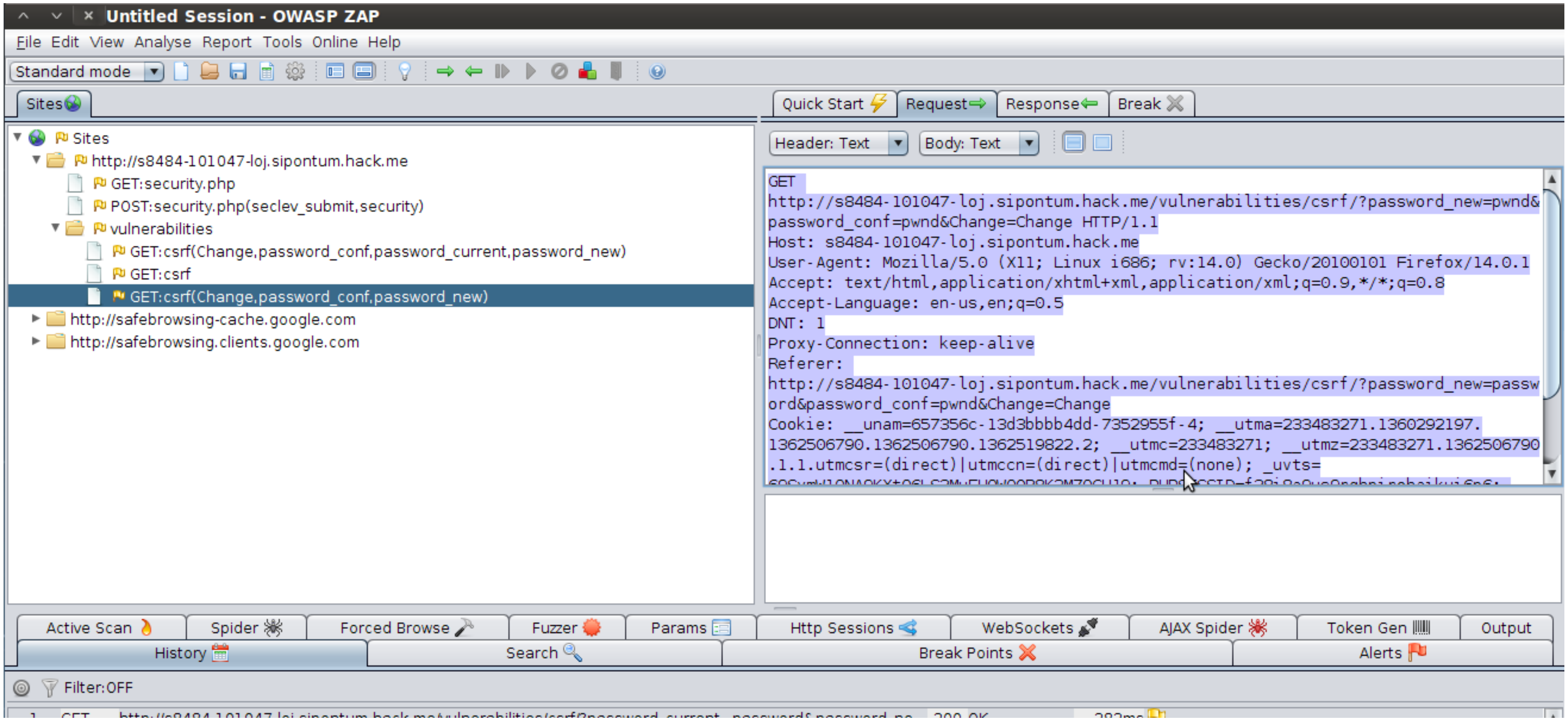
In ZAP, in the left pane, under sites expand the “vulnerabilities” pages.  Highlight the GET request as shown



```
Sites
  http://s8484-101047-loj.sipontum.hack.me
    GET:security.php
    POST:security.php(seclev_submit,security)
    vulnerabilities
      GET:csrf(Change,password_conf,password_current,password_new)
      GET:csrf
      GET:csrf(Change,password_conf,password_new)
  http://safebrowsing-cache.google.com
  http://safebrowsing.clients.google.com
```

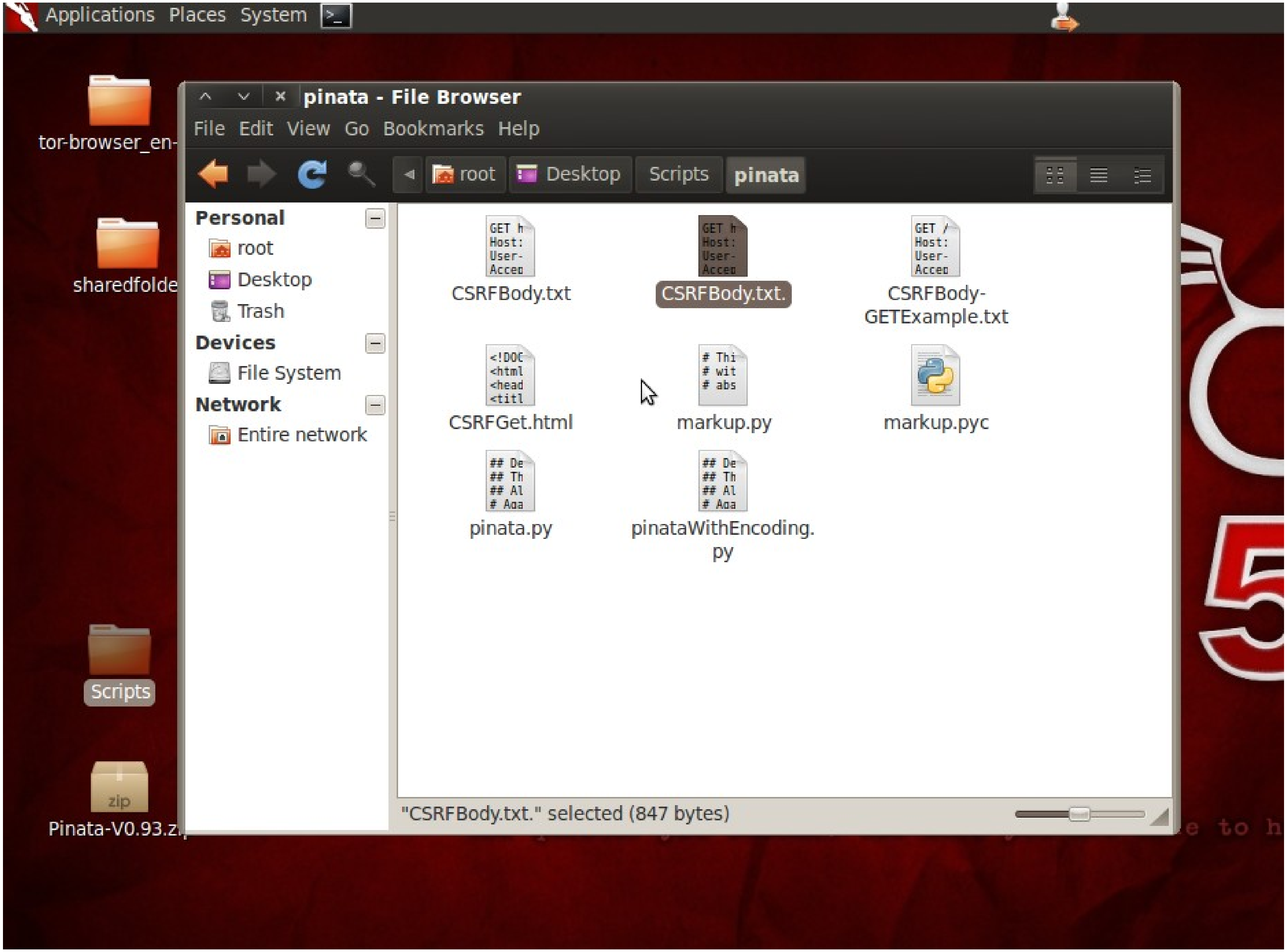
```
GET
http://s8484-101047-loj.sipontum.hack.me/vulnerabilities/csrf/?password_new=pwnd&
password_conf=pwnd&Change=Change HTTP/1.1
Host: s8484-101047-loj.sipontum.hack.me
```

On the right hand panel, select the entire text that is under the “REQUEST” tab.


Untitled Session - OWASP ZAP
File Edit View Analyse Report Tools Online Help
Standard mode
Sites
Sites
http://s8484-101047-loj.sipontum.hack.me
GET:security.php
POST:security.php(seclev_submit,security)
vulnerabilities
GET:csrf(Change,password_conf,password_current,password_new)
GET:csrf
GET:csrf(Change,password_conf,password_new)
http://safebrowsing-cache.google.com
http://safebrowsing.clients.google.com
Quick Start
Request
Response
Break
Header: Text
Body: Text
Active Scan
Spider
Forced Browse
Fuzzer
Params
Http Sessions
WebSockets
AJAX Spider
Token Gen
Output
History
Search
Break Points
Alerts
Filter:OFF

```
GET
http://s8484-101047-loj.sipontum.hack.me/vulnerabilities/csrf/?password_new=pwnd&
password_conf=pwnd&Change=Change HTTP/1.1
Host: s8484-101047-loj.sipontum.hack.me
User-Agent: Mozilla/5.0 (X11; Linux i686; rv:14.0) Gecko/20100101 Firefox/14.0.1
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-us,en;q=0.5
DNT: 1
Proxy-Connection: keep-alive
Referer:
http://s8484-101047-loj.sipontum.hack.me/vulnerabilities/csrf/?password_new=passw
ord&password_conf=pwnd&Change=Change
Cookie: __unam=657356c-13d3bbbb4dd-7352955f-4; __utma=233483271.1360292197.
1362506790.1362506790.1362519822.2; __utmc=233483271; __utmz=233483271.1362506790
.1.1.utmcsr=(direct)|utmccn=(direct)|utmcmd=(none); _uvts=
```

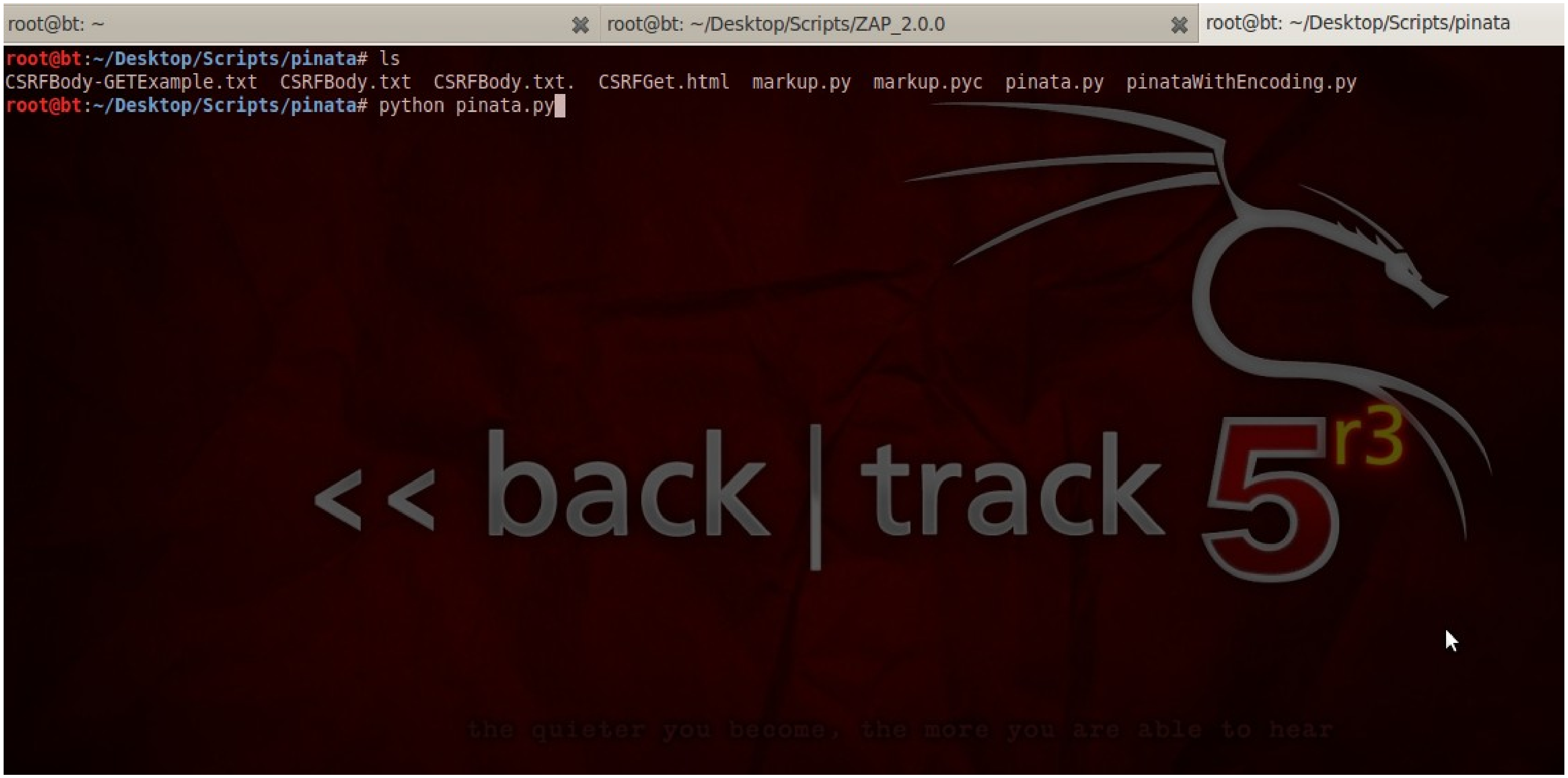
Right click the text and copy the text.



"CSRFBody.txt." selected (847 bytes)

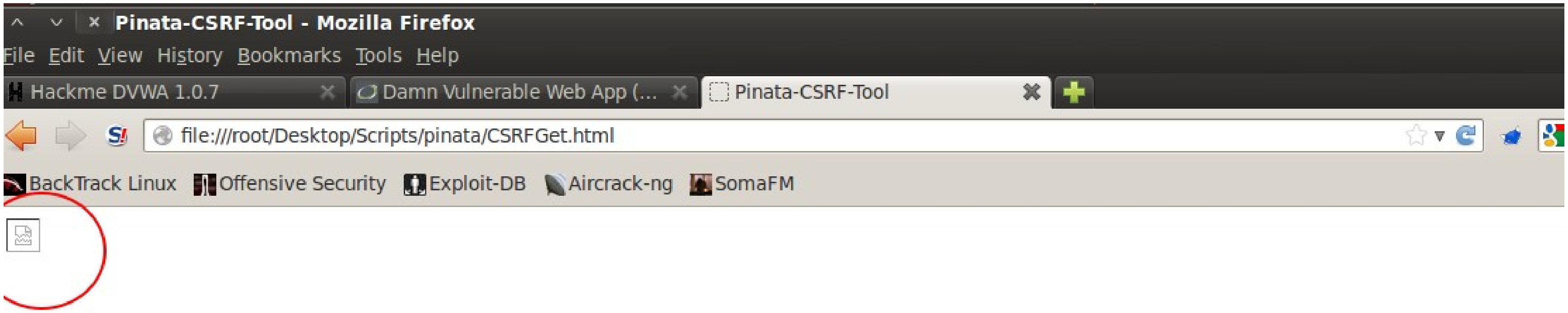
In the pinata directory open the CSRFBody.txt file with gedit (right click the file , open with) and paste this text in there.

From a shell, navigate to the pinata directory ,and run the python script pinata.py by typing python pinata.py


root@bt: ~
root@bt: ~/Desktop/Scripts/ZAP_2.0.0
root@bt: ~/Desktop/Scripts/pinata
<< back | track 5 r3
the quieter you become, the more you are able to hear

```
root@bt:~/Desktop/Scripts/pinata# ls
CSRFBody-GETExample.txt  CSRFBody.txt  CSRFBody.txt.  CSRFGet.html  markup.py  markup.pyc  pinata.py  pinataWithEncoding.py
root@bt:~/Desktop/Scripts/pinata# python pinata.py
```

Now, using FireFox , navigate and open the file CSRFGet.html file. Notice that there seems to be a broken image link on the left hand side


Pinata-CSRF-Tool - Mozilla Firefox
File Edit View History Bookmarks Tools Help
Hackme DVWA 1.0.7
Damn Vulnerable Web App (...
Pinata-CSRF-Tool
file:///root/Desktop/Scripts/pinata/CSRFGet.html
BackTrack Linux
Offensive Security
Exploit-DB
Aircrack-ng
SomaFM

Now log out of the app, and test the if the password was changed .
Login with user name of admin and the password of pwnd.

Note that you are now logged in with a password that you changed via new HTML code.

**Now imagine this on a bank website, where all you needed to do was get someone to click a link that would deposit X amount of money in Y account, or on Facebook where you could change peoples passwords and get into all the hidden stuff they locked you out of.**

## Proof of Lab

- 
- **Proof of Lab Instructions**:
    1. Do a <PrtScn> of all input, and results
    2. Paste into a word document
    3. Post to Teambox and email to me
-

Questions:

1. What XSRF?
2. What are some ways that an attacker could use XSRF in an attack?
3. Google some ways and list 2 that one could implement as protection vs. XSRF