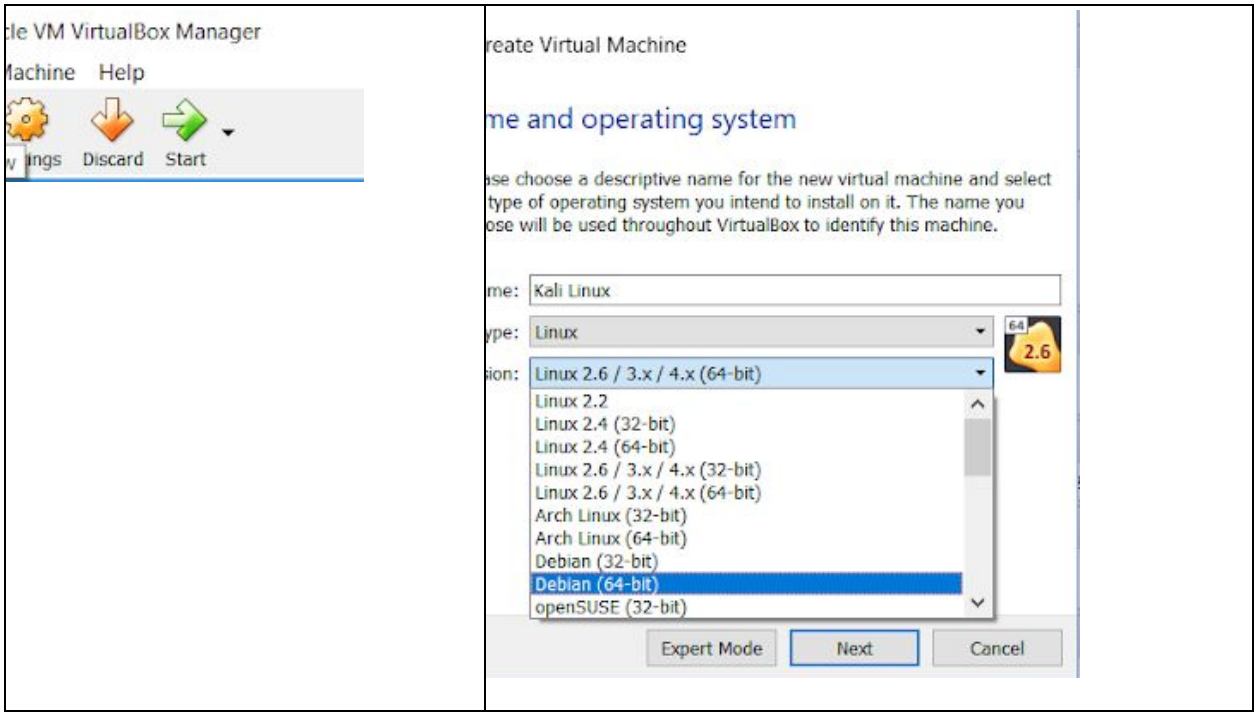
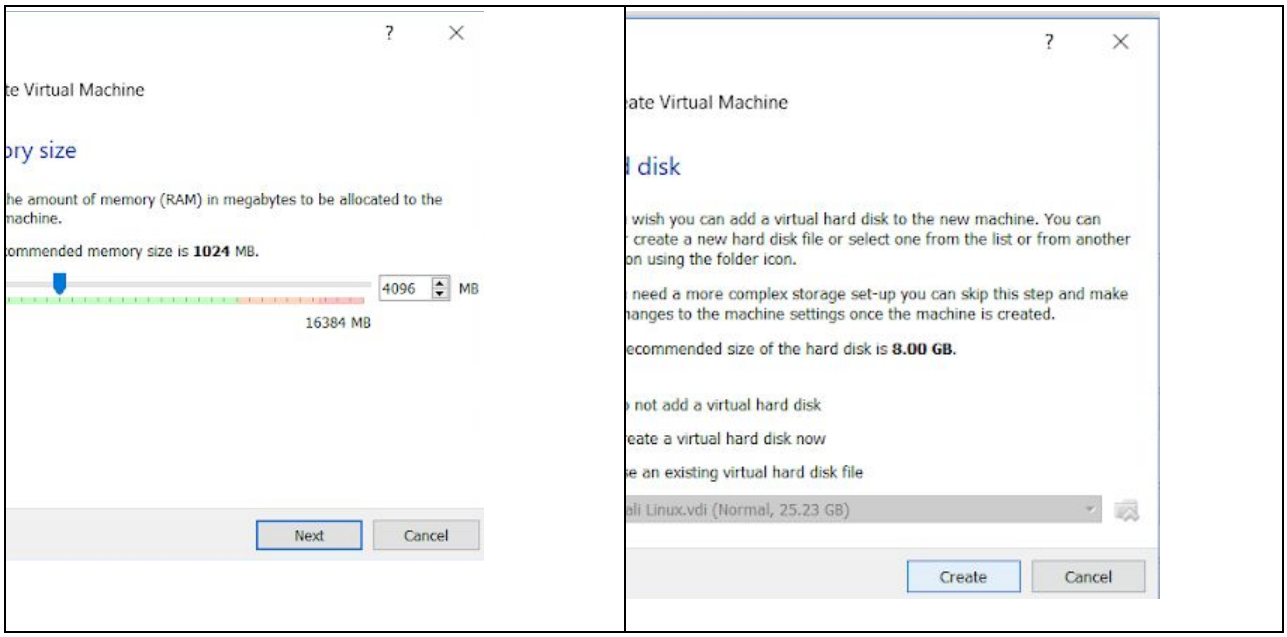
To exploit the vulnerabilities in the websites hosted in linux based Metasploitable OS, install Oracle VirtualBox VM and setup two linux environments:

Environment Setup:

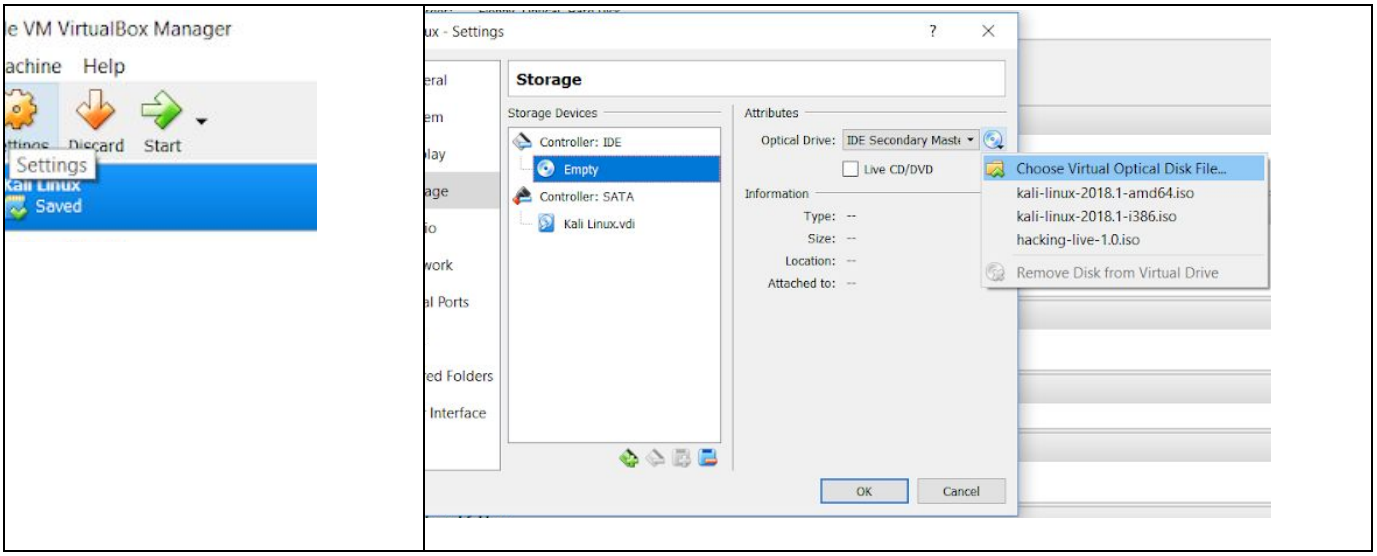
1. Kali Linux : install the ISO file from https://www.kali.org/downloads/ website.
    a. After downloading the ISO file follow below steps to create the VM:
    b. Click the new button and on the 'create virtual machine' pop-up, give the name of the virtual machine, select the type of OS as 'Linux' and version as 'Debian (64-bit)' and click Next.



    c. Give the memory size as 4096 MB and click next, in the next pop-up, select 'create virtual hard disk' and then select create, to create the virtual machine.



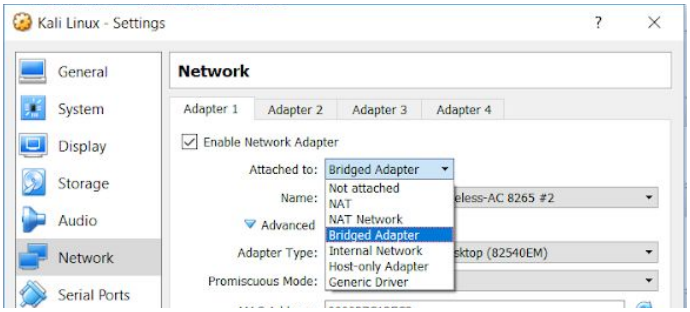
d. To add the ISO file to the Kali Linux VM, select the VM and click on settings as shown below:



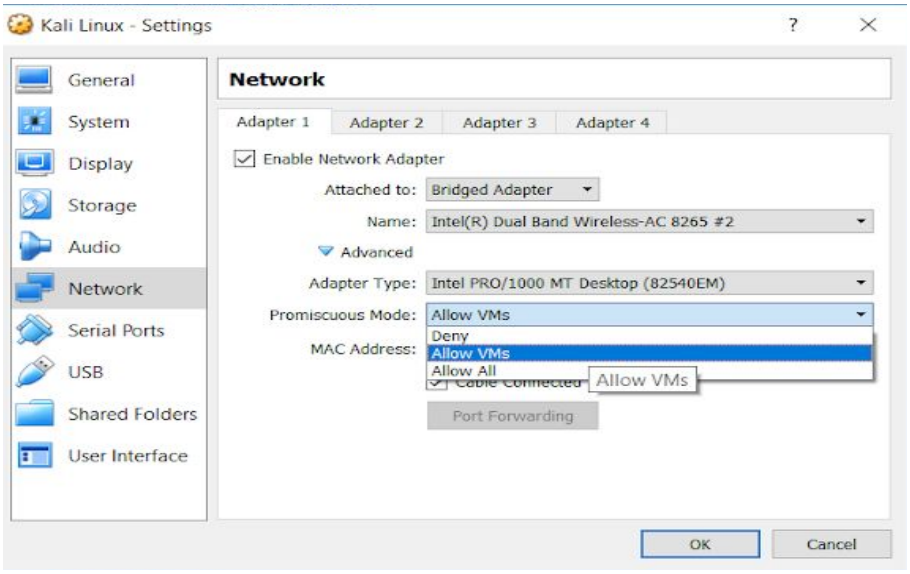
In the settings window, go to Storage, select click on the 'Empty' settings under the Controller, then click on the Disc image in the attributes section to choose the ISO file.

e. If you were to setup the VM with 64-bit ISO file and the option to create a 64-bit Debian Linux VM is not showing up in the VirtualBox, change the BIOS setup of your machine as explained in this article here: https://superuser.com/questions/866962/why-does-virtualbox-only-have-32-bit-option-no-64-bit-option-on-windows-7

f. Once you have attached the ISO file to the VM, click on Network in the Settings window and change the "Attached to" attribute from NAT to Bridged Adapter.



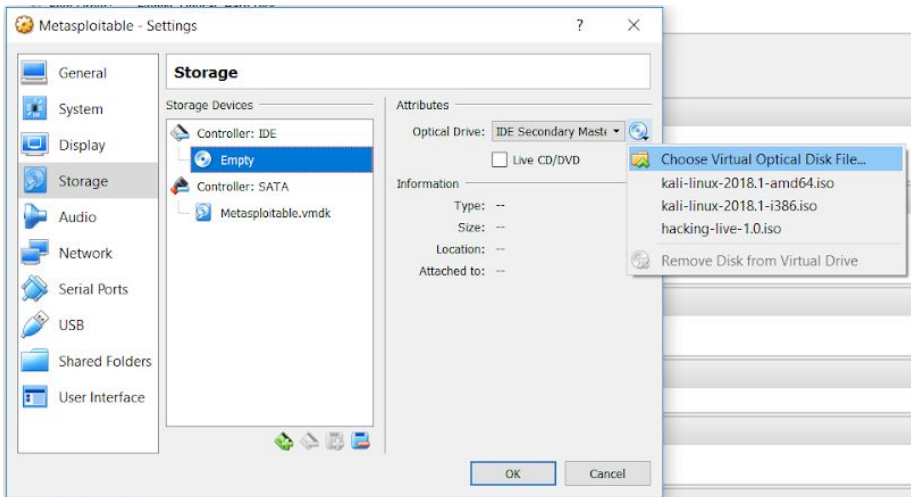
Then click on 'Advanced' attribute as shown below to change the Promiscuous mode from "Deny" to "Allow VMs". These settings will let the Kali Linux to interact with other VMs.



g. Click OK and start the VM to get started. Once you start the machine, choose appropriate settings to boot up the VM.

2. Similar to the step 1 above, click on 'New' button on VirtualBox to create Metasploitable linux VM. Make sure to download the Metasploitable VM image from the below link: http://sourceforge.net/projects/metasploitable/files/Metasploitable2/

Attach the .vmdk file to the VM in the storage section of the VM settings and start the machine:



3. Once you start the metasploitable, use msfadmin as login username and password to login into the system. Type in "ifconfig" command as shown below to know the IP address of the VM, which will be used to exploit the websites hosted in the metasploitable VM from Kali Linux:

```
To access official Ubuntu documentation, please visit:
http://help.ubuntu.com/
No mail.
msfadmin@metasploitable:~$ ifconfig
eth0      Link encap:Ethernet  HWaddr 08:00:27:08:e7:a8
          inet addr:11.219.148.114  Bcast:11.219.255.255  Mask:255.255.128.0
          inet6 addr: fe80::a00:27ff:fe08:e7a8/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:40 errors:0 dropped:0 overruns:0 frame:0
          TX packets:70 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:12775 (12.4 KB)  TX bytes:7864 (7.6 KB)
          Base address:0xd010 Memory:f0000000-f0020000

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:96 errors:0 dropped:0 overruns:0 frame:0
          TX packets:96 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:21437 (20.9 KB)  TX bytes:21437 (20.9 KB)

msfadmin@metasploitable:~$
msfadmin@metasploitable:~$
```

We would be using this highlighted IP address to exploit the websites.

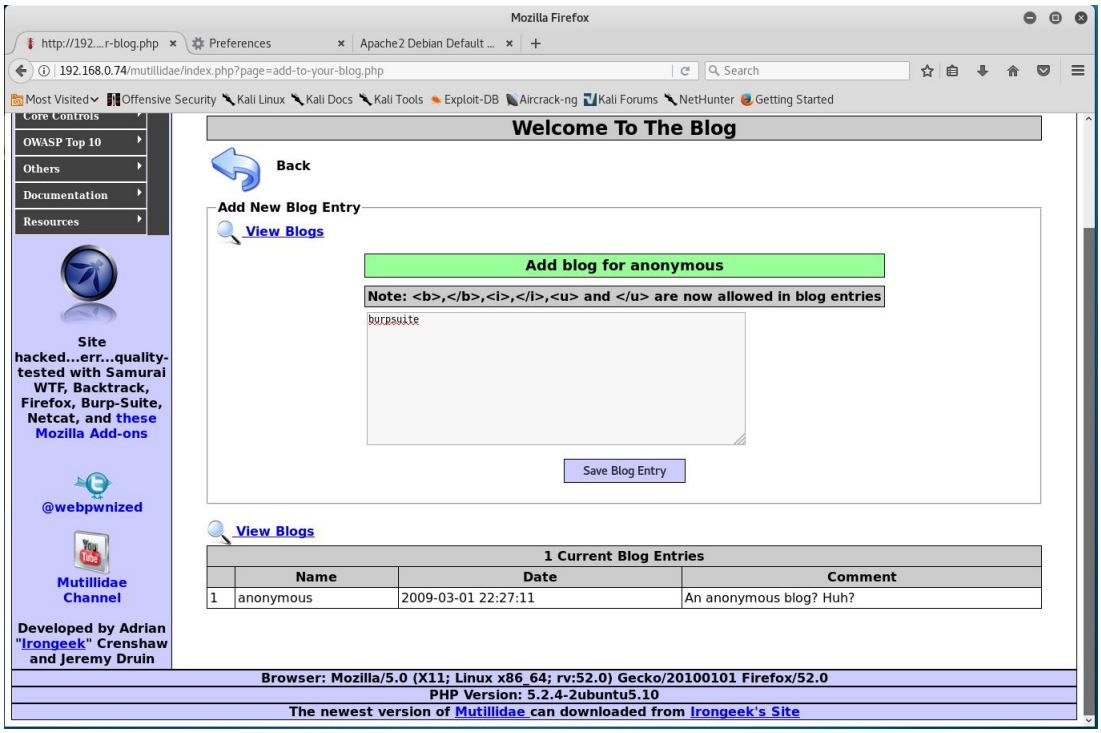
4. Once you start the Kali Linux VM, hit this IP address in a browser to list the websites hosted on this VM:

We would be exploiting Mutillidae website. Below are the list of exploits and their possible fixes:

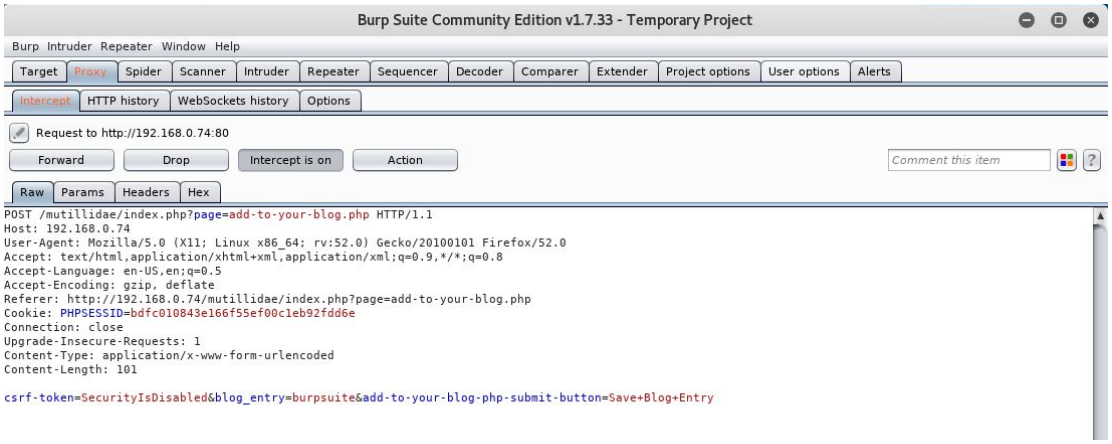
Exploits & Fixes:

1. Cross Site Request Forgery - CSRF

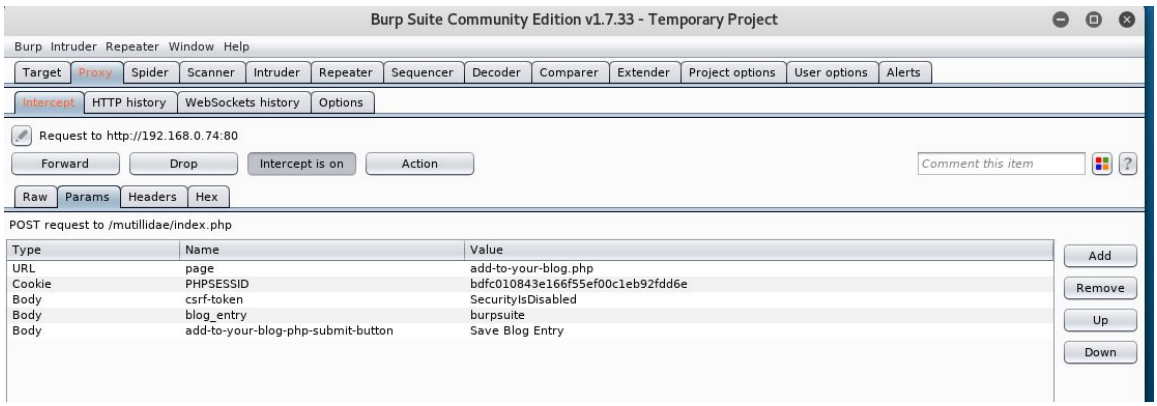
Go to add-to-your-blog.php and turn on the intercept of burp suite.



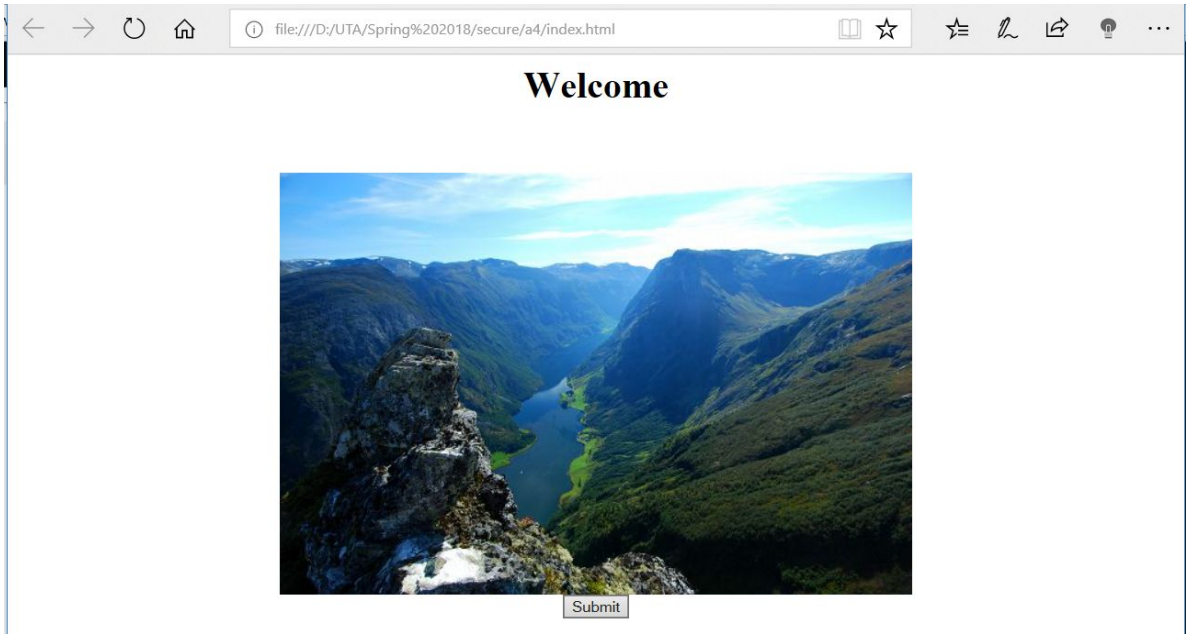
Once the blog entry is submitted, we will see the POST request in burp suite.



In “Params” tab we will get the 3 body parameters name and value.



Create an html file with hidden input fields.

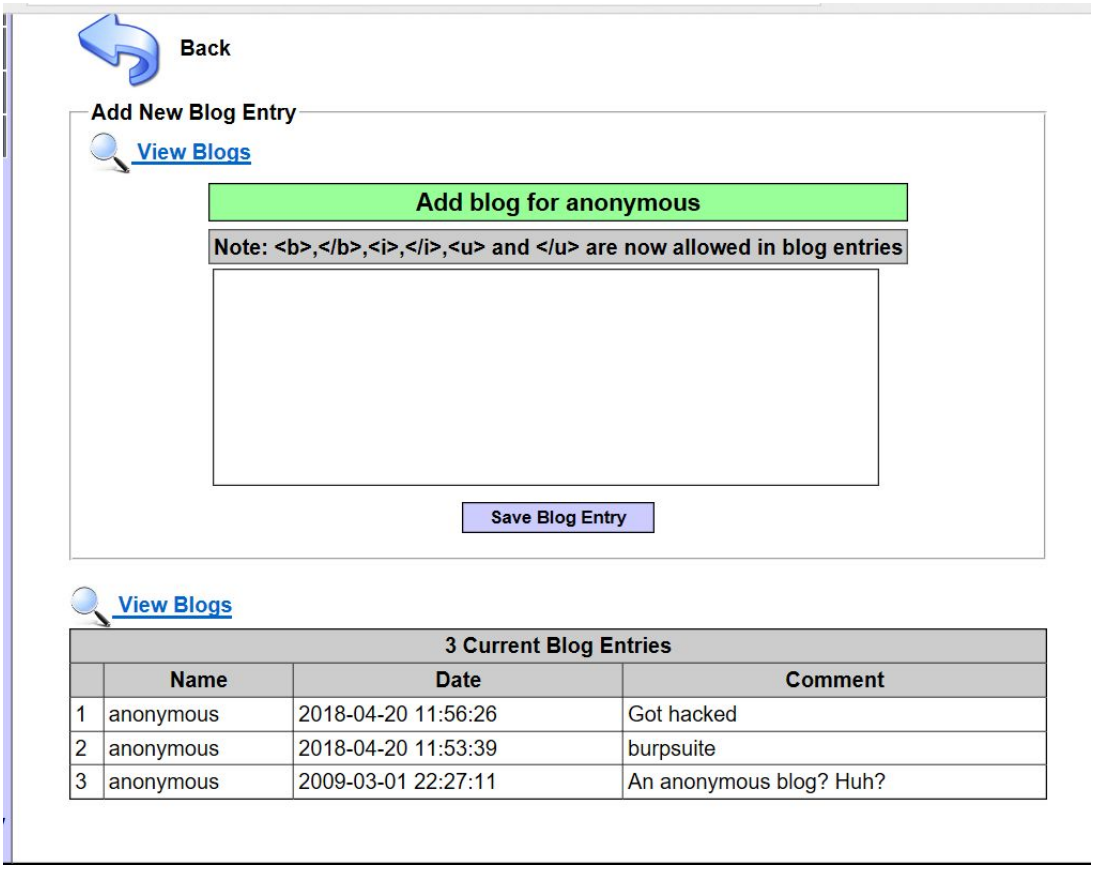


The hidden input tags contain the name of body parameters we obtained from burp suite.

```
<html>
    <center>
    <body><h1>Welcome</h1>
        <br><br>
        <img src="http://www.gstatic.com/webp/gallery/1.jpg"></body><br>


        <!-- Form Post -->
        <form name="csrf" action="http://192.168.0.74/mutillidae/index.php?page=add-to-your-blog.php"
        method="POST">
        <input type="hidden" name='csrf-token' value=''>
        <input type="hidden" name='blog_entry' value='Got hacked'>
        <input type="hidden" name='add-to-your-blog-php-submit-button' value='Save Blog Entry'>
        <input type="submit" value="Submit">
    </form>
   <!-- <script>document.csrf.submit();</script>-->
        <!-- End Form Post -->
</html>
```

Upon click submit button the message will be posted in the add-to-your-blog.php



Solution:

This exploit results in making the user submit sensitive information to unauthorized users. It can be prevented by following one of the measures listed below:

- **Synchronizer Tokens** – It is unique per sessions and has large random values. Generated by a cryptographically secure random number generator.

- **Double Cookie Defense** – Send a random value in both a cookie and as a request parameter. Have the server verify if the cookie and the request value is the same.

- **Encrypted token pattern** – After authentication, have the server generate a unique. Token comprised of the user's ID, a timestamp values and a cryptographic nonce, using a unique key available only to the server.

- **Custom Request Header** – A non-complex defense suited for AJAX endpoints. It relies on same origin policy (SOP) restrictions that only JavaScript can be used to add a custom header and only within its origin. By default, browsers do not allow JavaScript to perform Cross-origin requests.

2. SQL Injection

Give username input as ' or 1=1 -- ' and password can be blank.

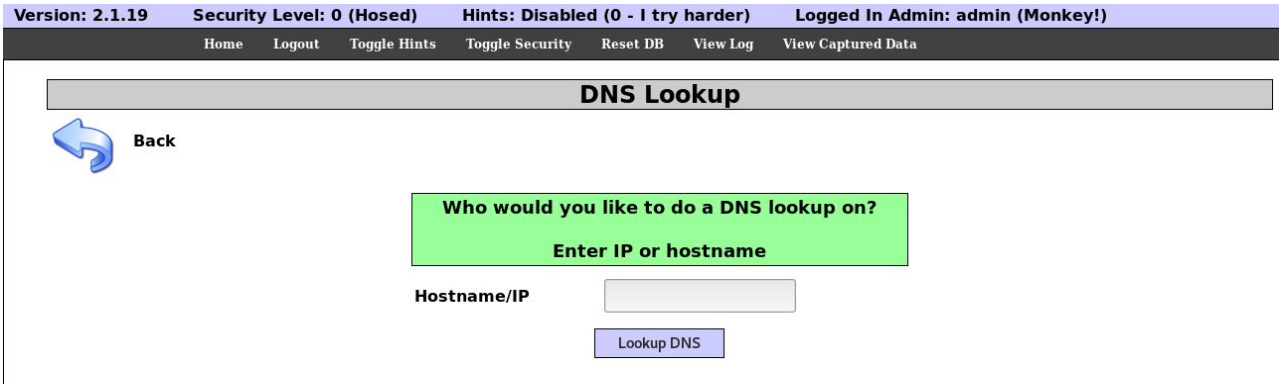
The given input for name lists all the users from the table users and their passwords.

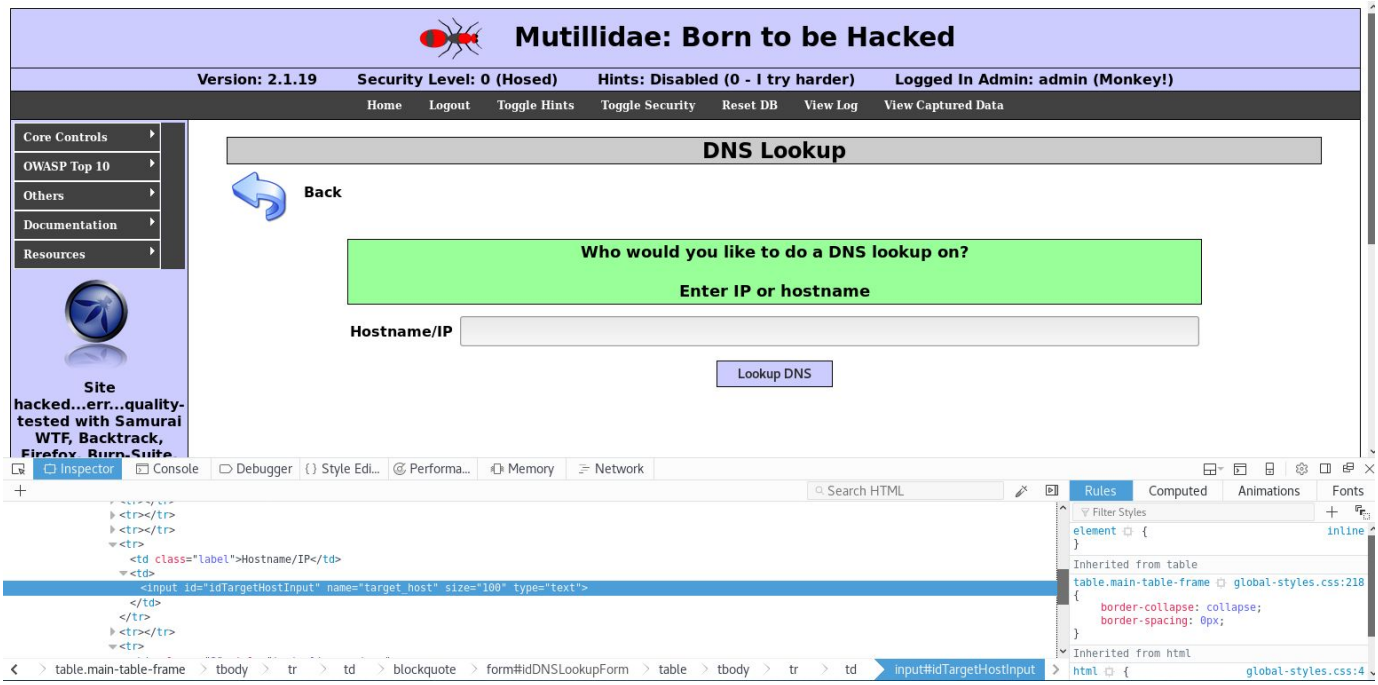
Solution:

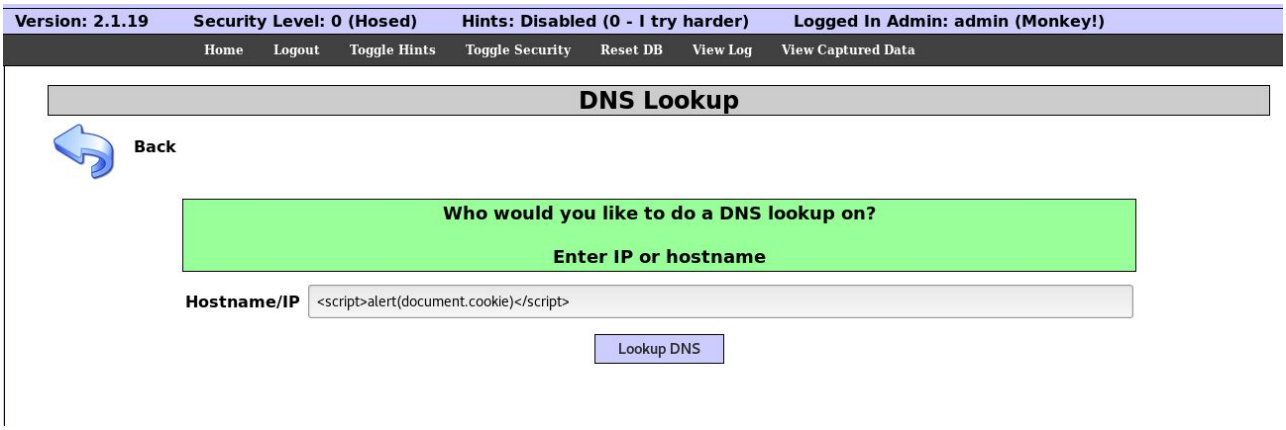
Keep the web application in sync by maintaining validation checks of all the inputs that are sent to a database.

3. Cross Site Scripting – XSS

To exploit this kind of flaw, go to DNS lookup page and using developer tools, increase the length of input textbook to add the script, as shown below:







With the sample script we added as shown above, the below alert is thrown to display the cookie information:

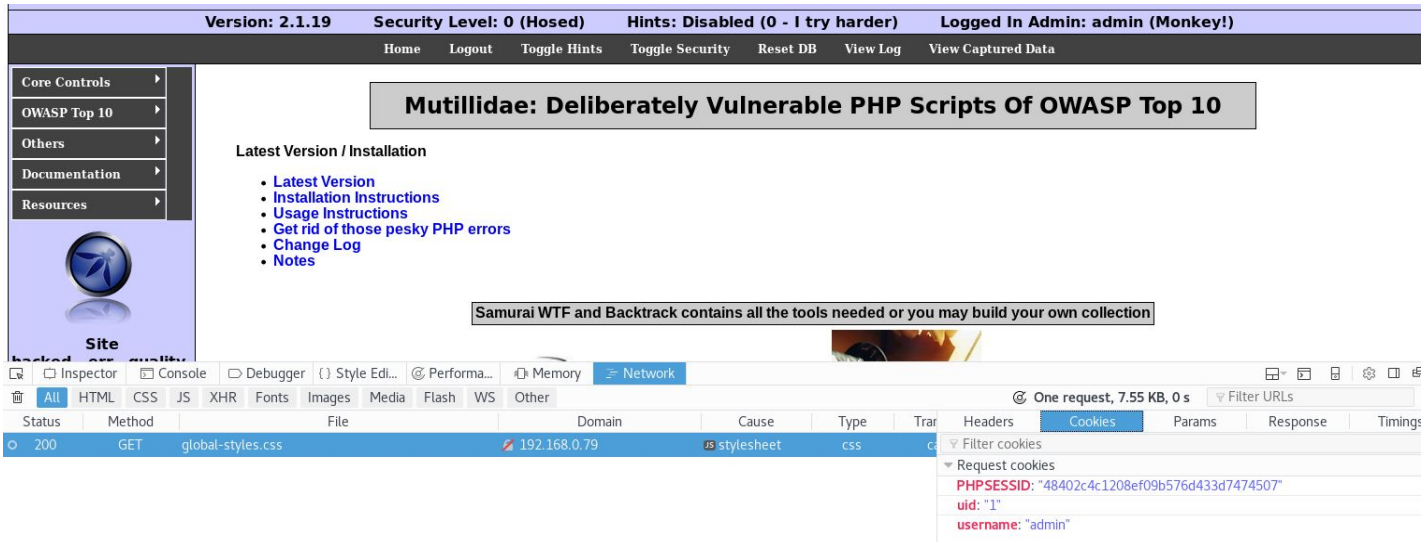
Solution:

Keep the web application in sync by maintaining validation checks of all the inputs that are sent to a database.

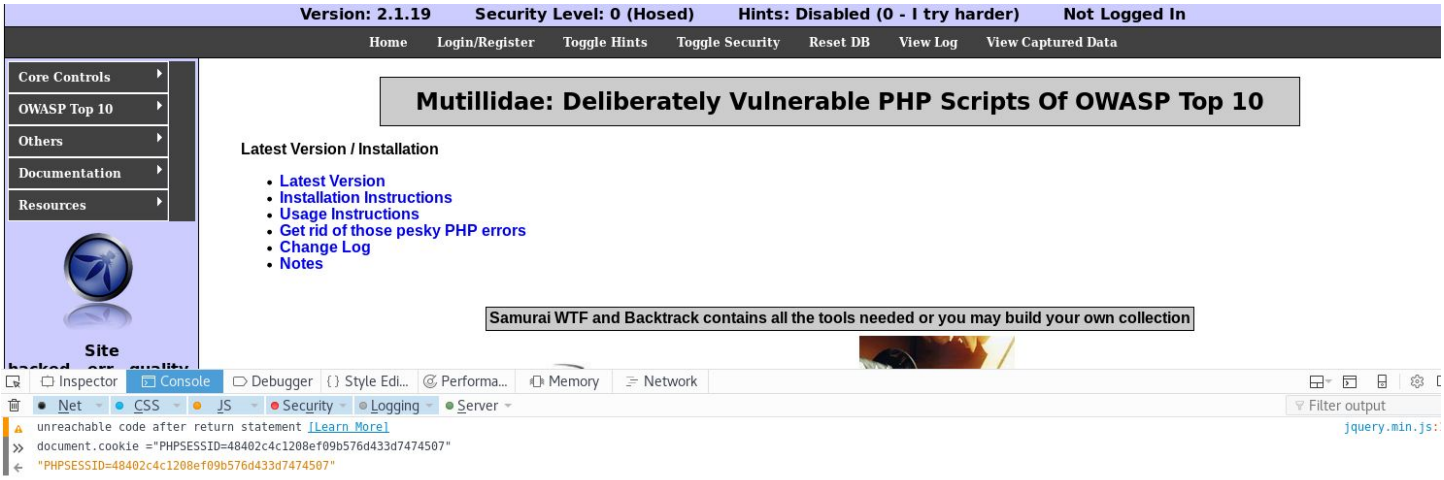
4. Broken authentication and session management

Login as admin

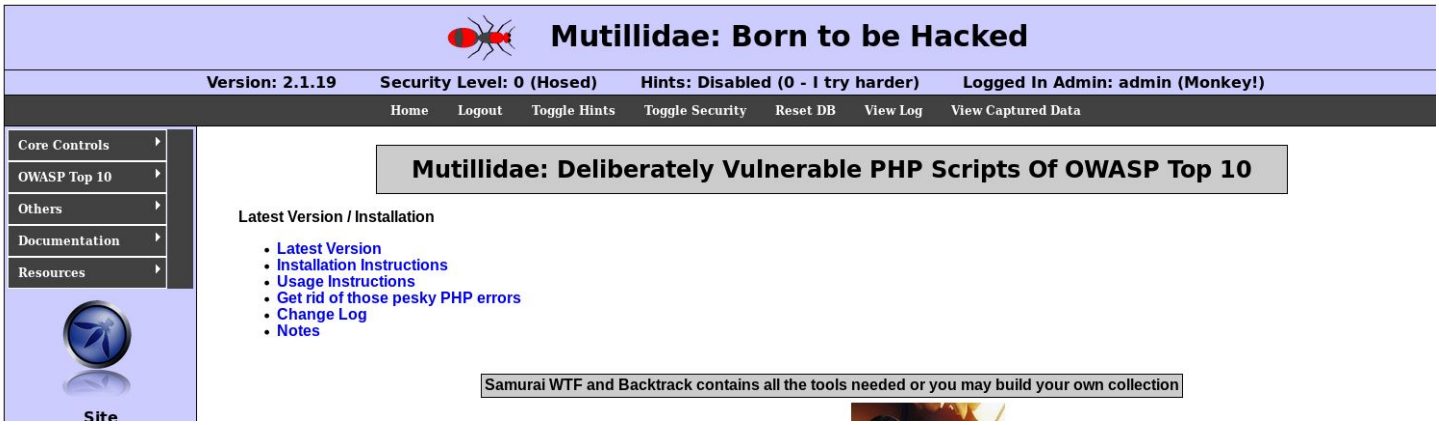
get the session id by opening developer tools and select network tab. Select cookies under network tabs.



Open a new window



refresh the page and we can see the user logged as admin



With this kind of exploits, attacker can re-use the user's session id to hack into the user's session and take control of their account.

Solution:

Attacker uses leaks and flaws in the authentication or session management functions to impersonate the users.

- **Password Strength** – Have restrictions based on a minimum size and complexity.
- **Password Use** – Have restrictions based on number of login attempts per unit time and log number of failed attempts.
- **Password Change Controls** – A single password change mechanism should be used wherever users can change a password, regardless of the situation. Users should always be required to provide both their old and new password when changing their password.
- **Password storage** – All passwords have to be stored in hashed or encrypted forms.
- **Session ID protection** – Select the session via SSL.
- **Browser Caching** – Authentication and Browser data should never be submitted as a part of GET. POST method should be used.

5. Insecure direct object reference

Go to developer tools and then change the value of one of the select tag as “/etc/passwd” as shown.

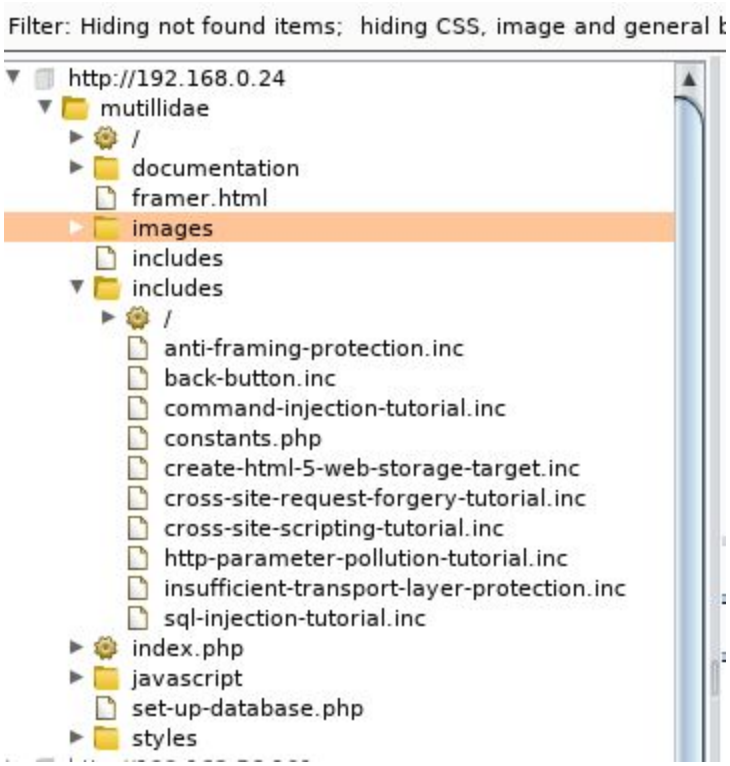
On click of View File button, list of passwords is displayed as shown below:

Solution:

To fix this issue, refrain from providing the server details on the webpage directly. Instead maintain the path details in a separate file at the server and use JavaScript and access control to the access the server path.

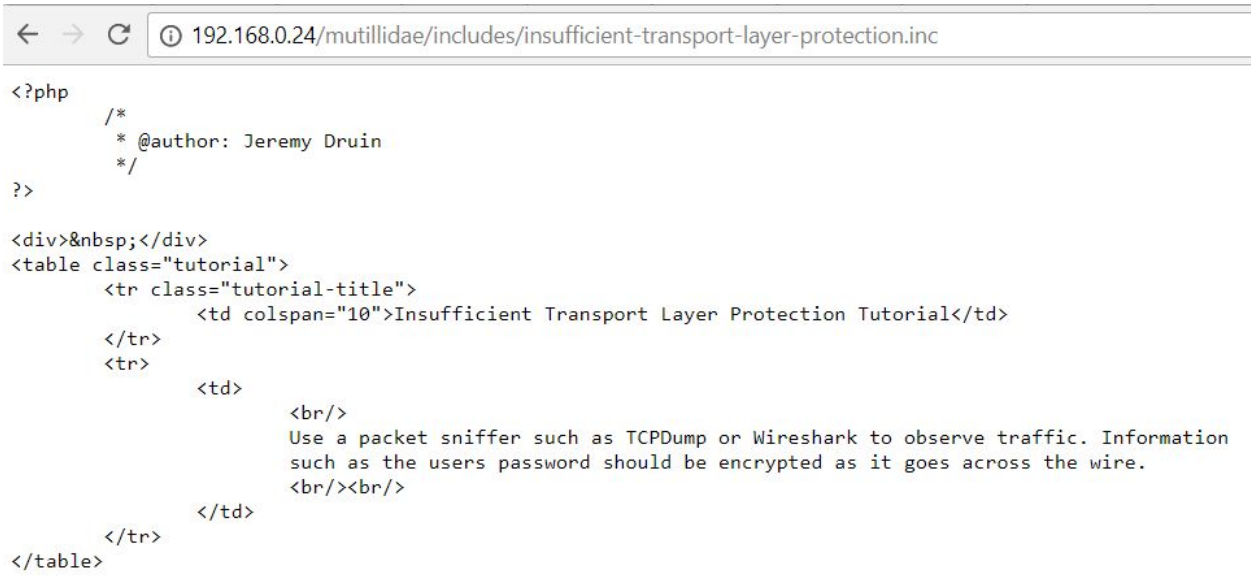
6. Security Misconfiguration

Open burp suite and then open mutillidae website.Then we can see the directory listings of that website in burp suite at 'Target' tab as shown.



We can get the url of the files and will be able to access the files as shown in below.

Accessing the file

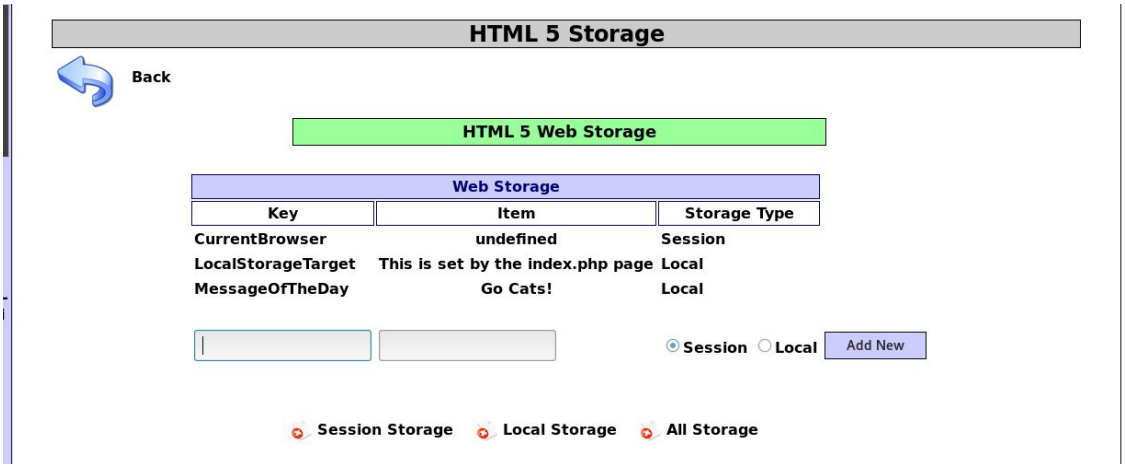


Solution:

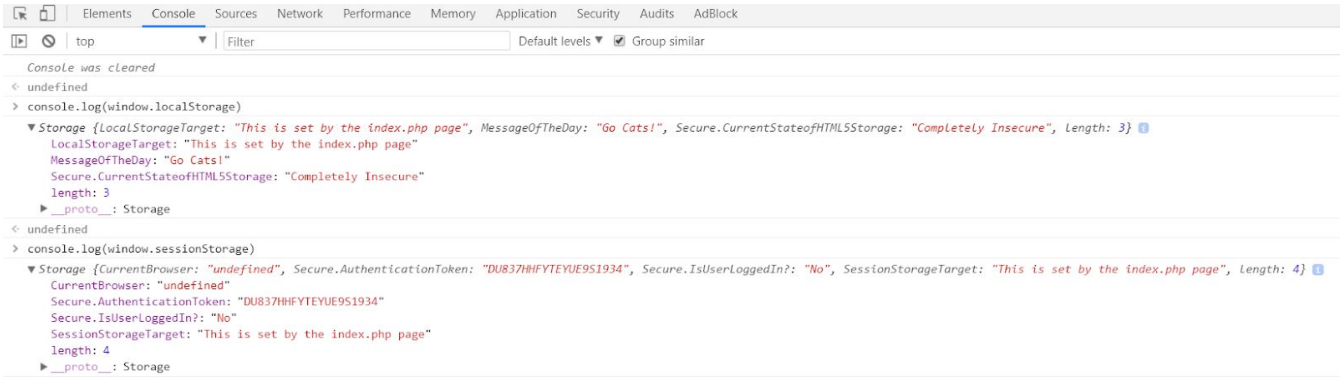
Implement proper access control and visibility of the application folder structure to prevent attackers from accessing the application folders/ data. Ensure to implement user authorization rules in the application config files.

7. Insecure Cryptographic Storage

In order to exploit this vulnerability, go to html5-storage.php



As shown below, this page is so poorly designed that from the console window of a developer tool, attacker can get the details of current session and locally stored details of the application.

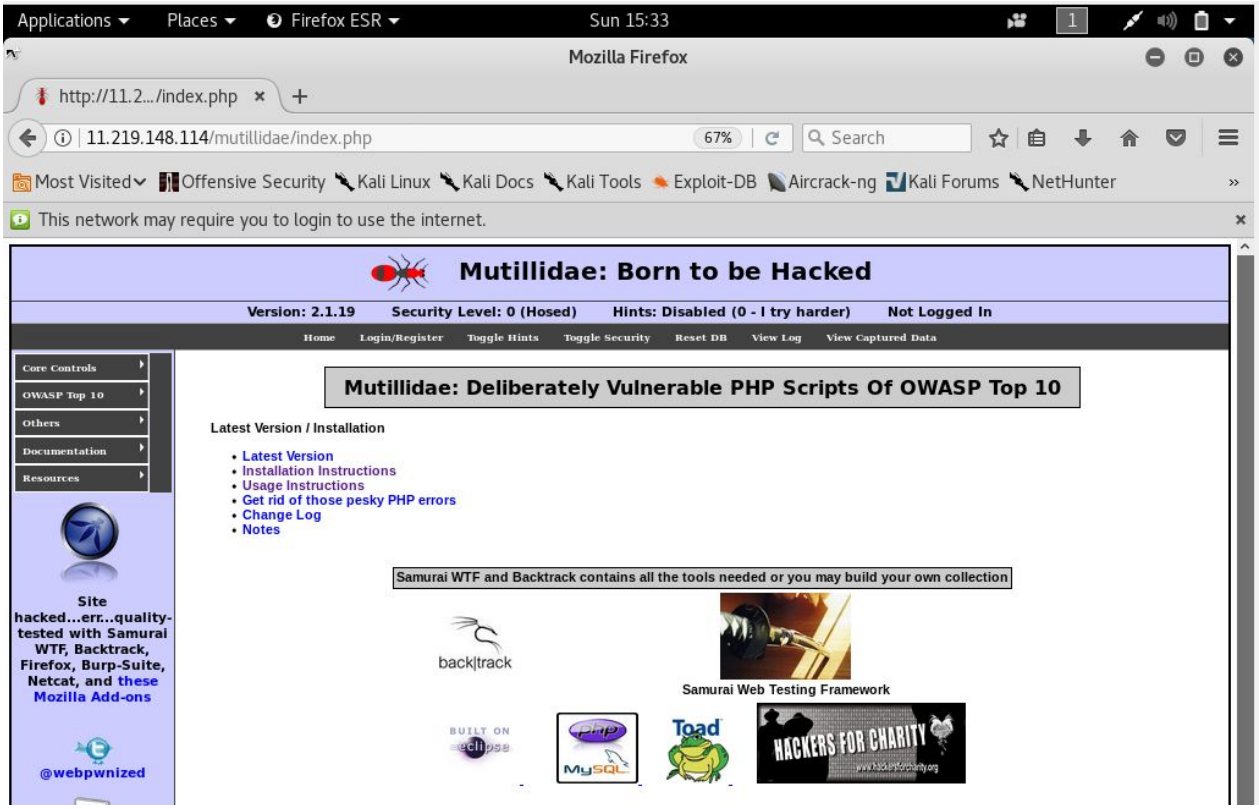


Solution:

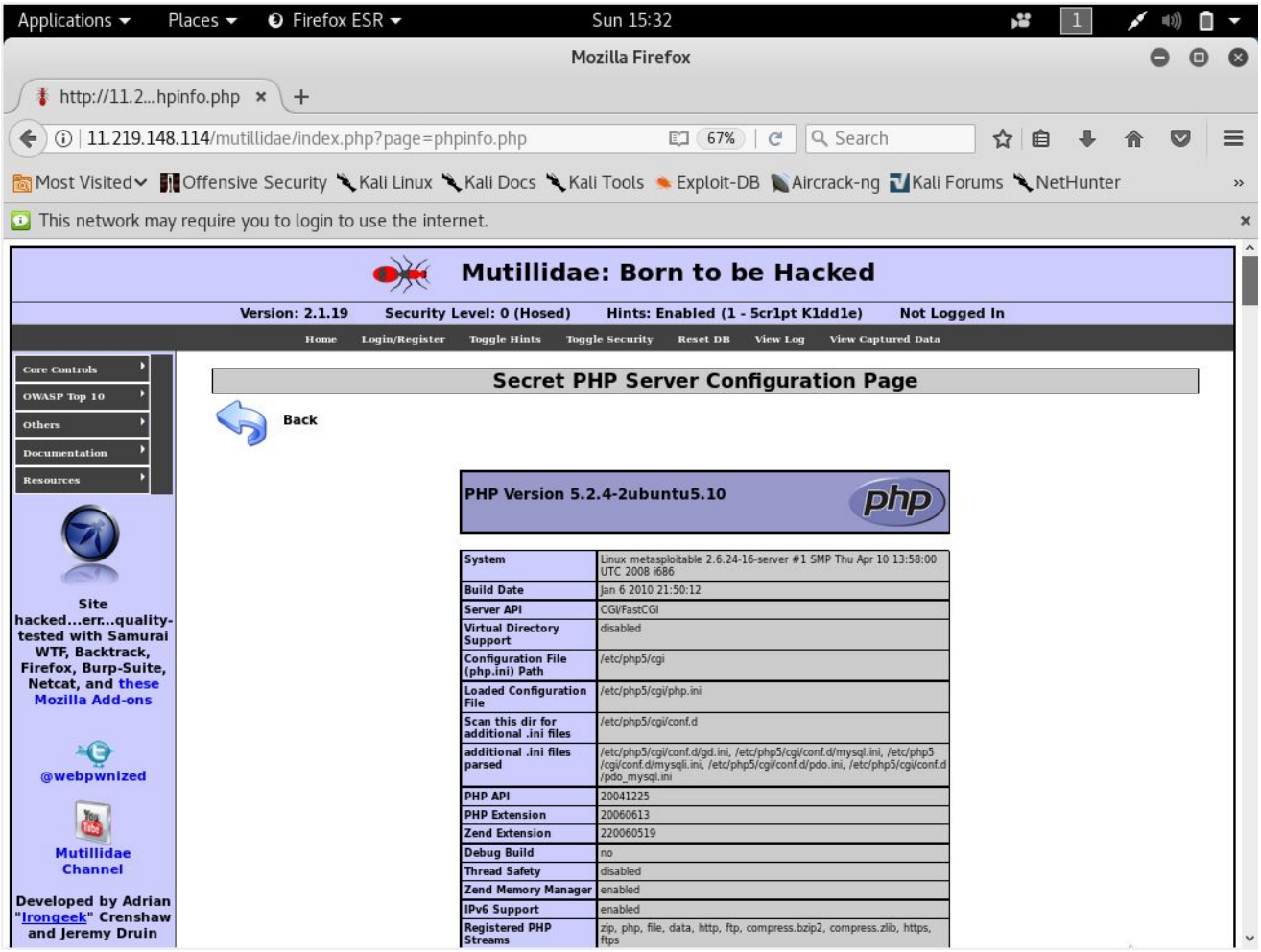
- Make sure offsite backups are encrypted, but the keys are managed and backed up separately.
- Ensure appropriate strong standard algorithms and strong keys are used, and key management is in place.
- Ensure passwords are hashed with a strong standard algorithm and an appropriate salt is used.
- Ensure all keys and passwords are protected from unauthorized access.

8. Failure to Restrict URL Access

This vulnerability can be seen if the attacker knows the application structure beforehand. Below is the index page:



Now edit the url with a query string parameter ‘page’ and the webpage redirects to an unauthorized page as shown below, which has all the server configuration details:

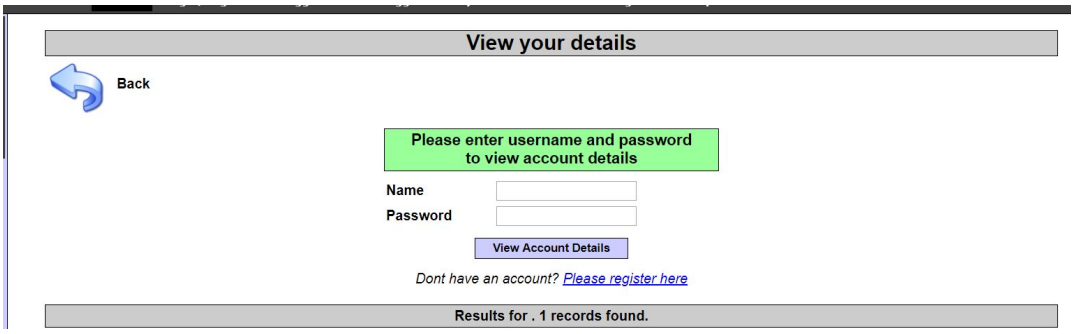


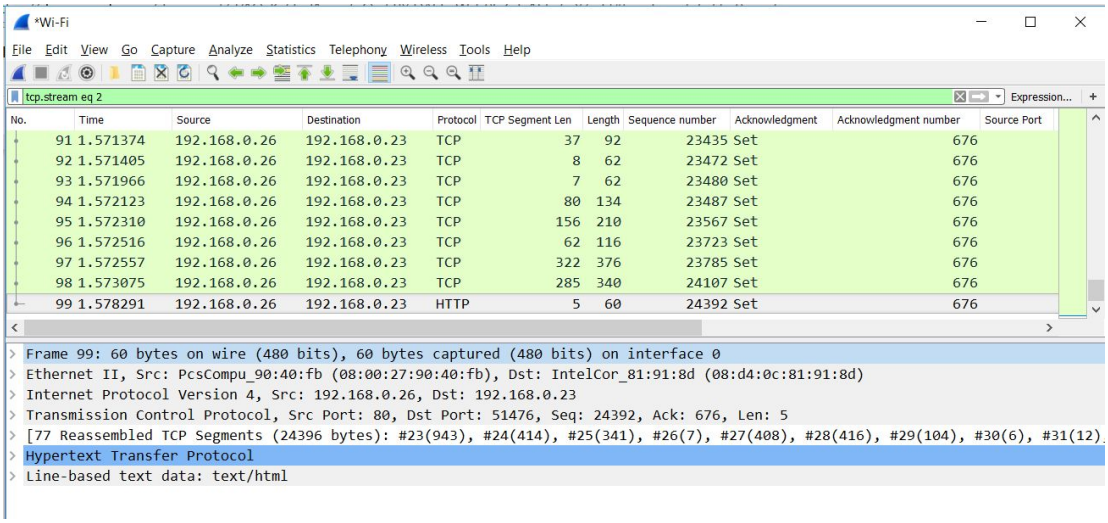
Solution:

Implement proper access control and visibility of the application folder structure to prevent attackers from accessing the application folders/ data. Ensure to implement user authorization rules in the application config files. Make sure that all url redirections pass through authentication checks in the application.

9. Insufficient Transport Layer Protection

Open user-info.php page and also start wireshark.





Enter the credentials and click on ‘view account details’. Then stop the Wireshark. Then open the TCP stream of the Wireshark for that website. We will the get the HTML page with username and password.

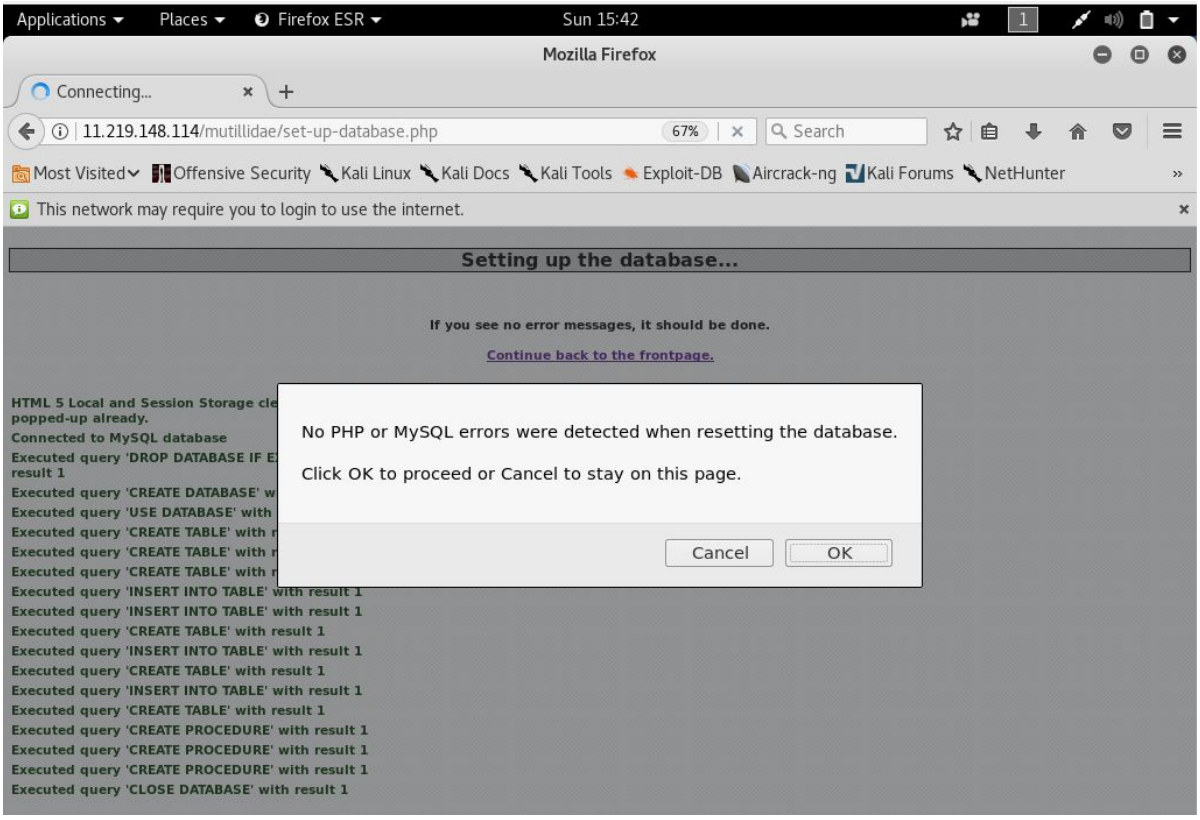
```
3a
<p class="report-header">Results for . 1 records found.<p>
19
<b>Username=</b>admin<br>
1d
<b>Password=</b>adminpass<br>
1f
<b>Signature=</b>Monkey!<br><p>
3
<p>
```

Solution:

- Implement HTTP Strict Transport Security in the browsers to enforce secure connections. Also implement Certificate and Public Key pinning in browsers where applicable.
- Redirect all HTTP requests to HTTPS
- Configure SSL properly on the server.
    - Disable weak SSL/TLS protocols
    - Disable weak ‘export’ algorithms
    - Use SSL certificate with a minimum key size of 2048 bit
- Make sure session key size is 128 bits
- Disable algorithms offering no encryption

10. Unvalidated Redirects & Forwards

The following alert pop-up explains that anyone can reset the database, regardless of user being an admin or an attacker, which is an example of the Unvalidated Redirects & Forwards vulnerability.

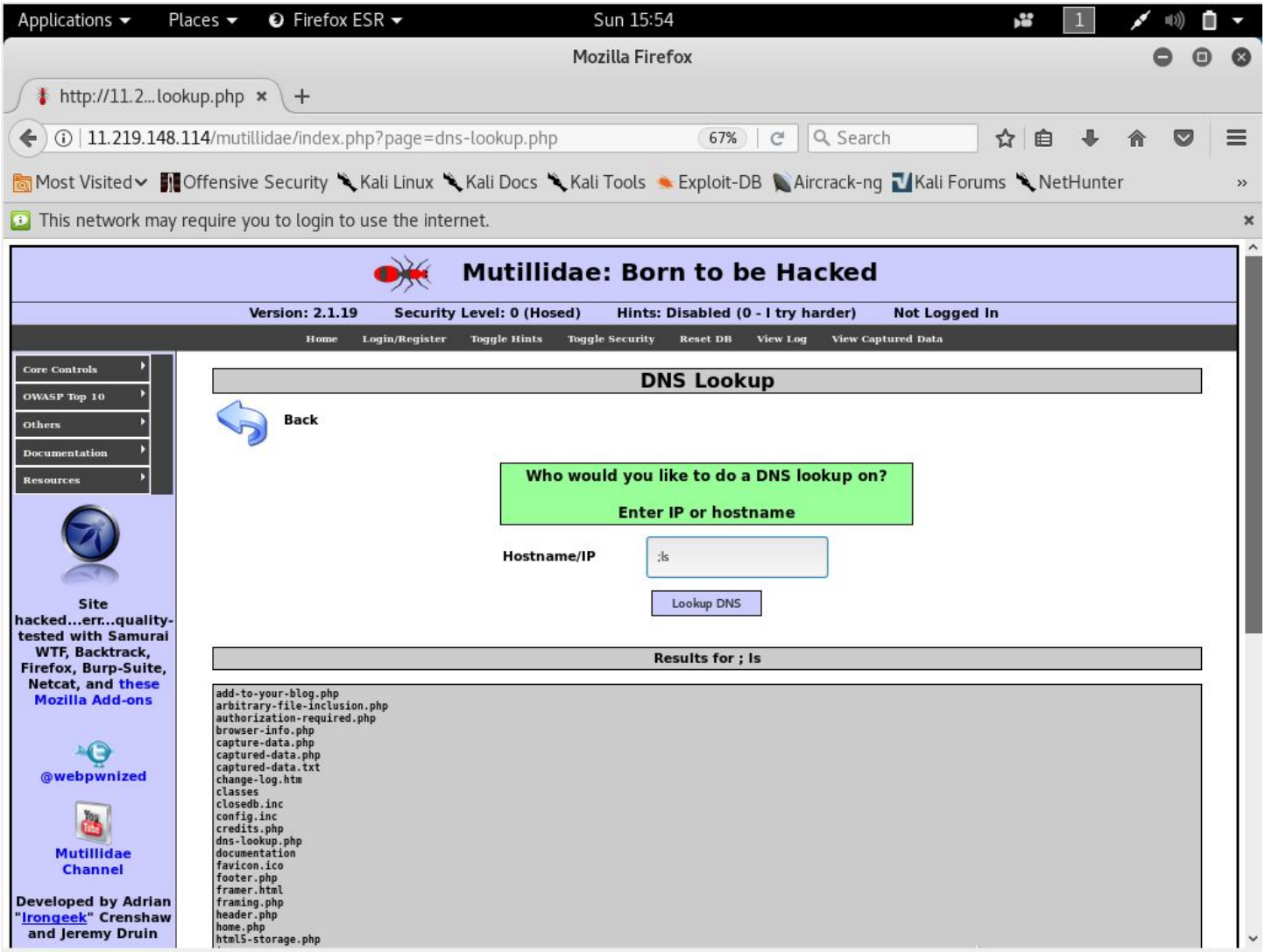


Solution:

- Avoid using redirects and forwards.
- Do not allow url as user input, if it can't be avoided, make sure to implement the authorization rules for each user role that has access to the application.
- Implement appropriate input validation rules.
- Sanitize input by creating a list of trusted URLs.

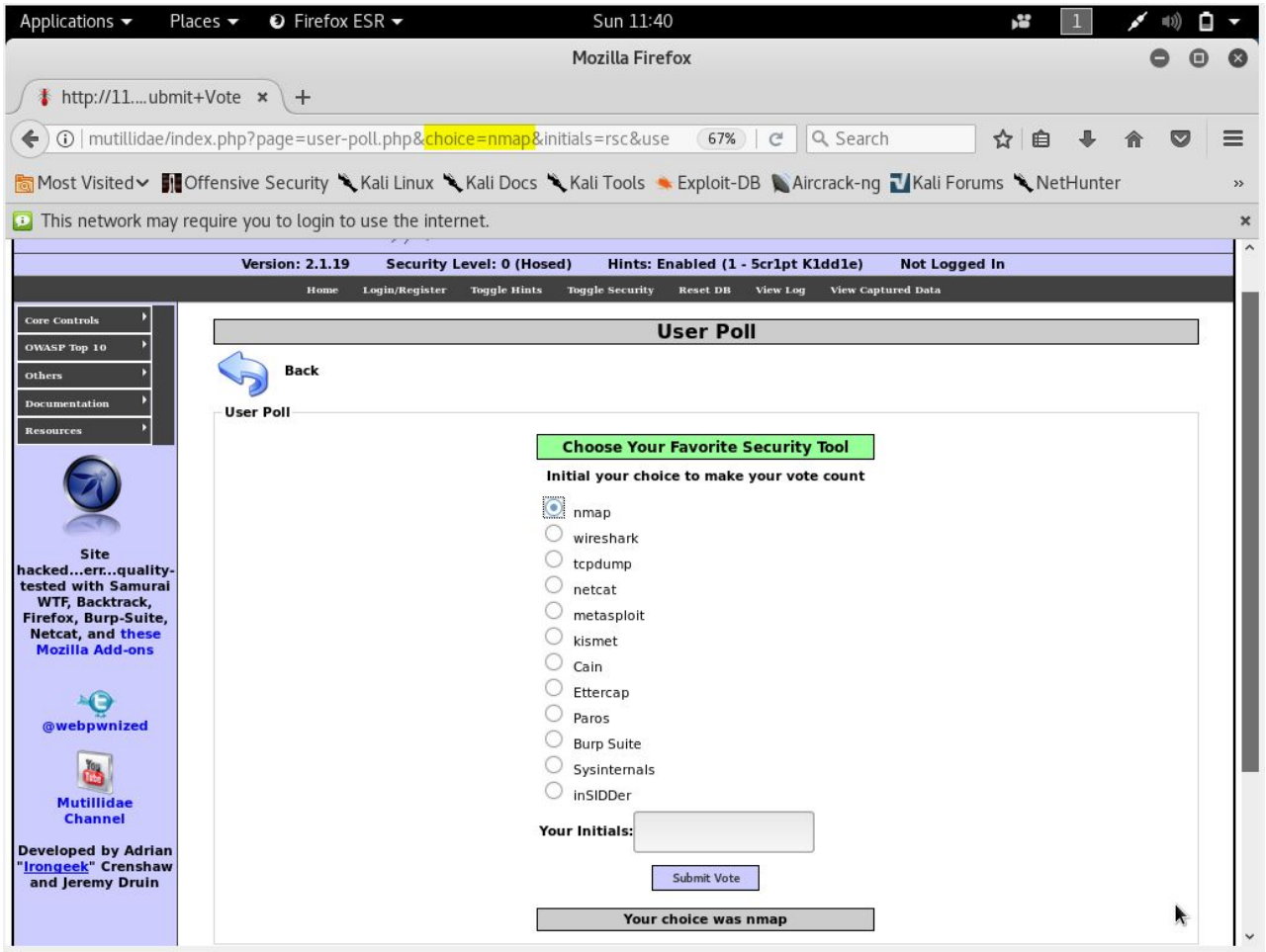
11. Command Injection

To exploit this vulnerability, in the Hostname textbox, give input as ";ls". This particular command closes the earlier command and lists all the files that are present in the same directory as that of the dns-lookup.php page, as shown below:

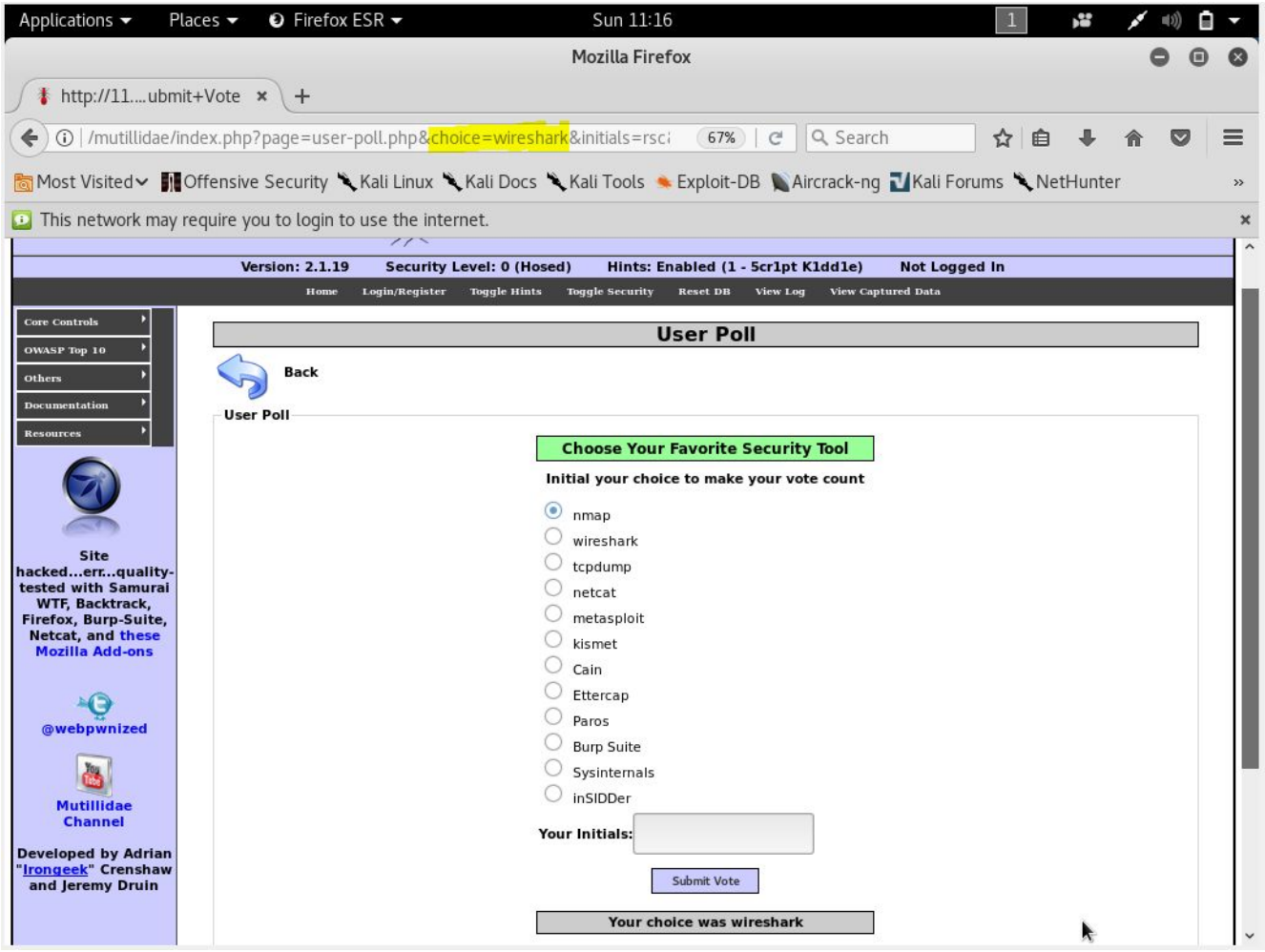


12.Security Misconfiguration – GET for POST

In the user poll page, select any of the choices from the page and notice that the selected choice is passed through the 'choice' query string of the url when submit button was hit, as shown below:



Now change the query string parameter from the user selected ‘nmap’ to wireshark to exploit the flaw, and on click of submit vote, it will be displayed that user selected wireshark.

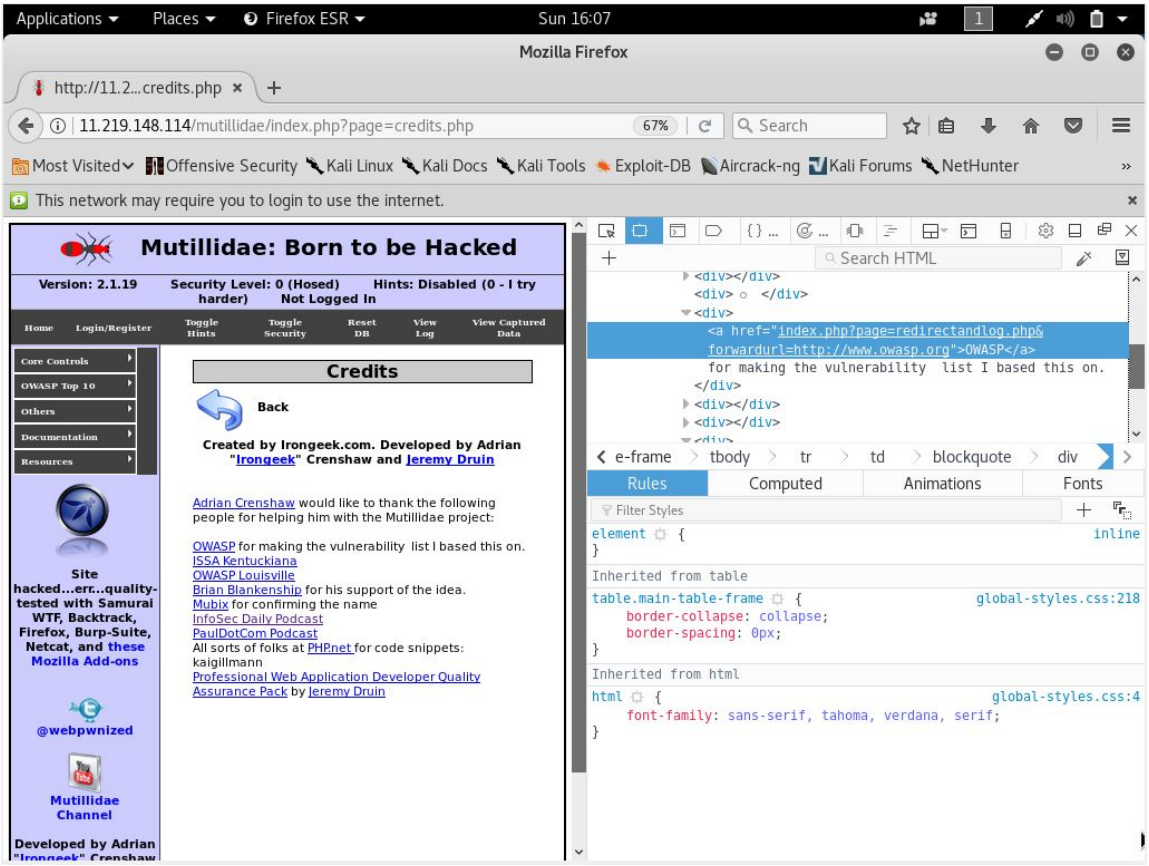


Solution:

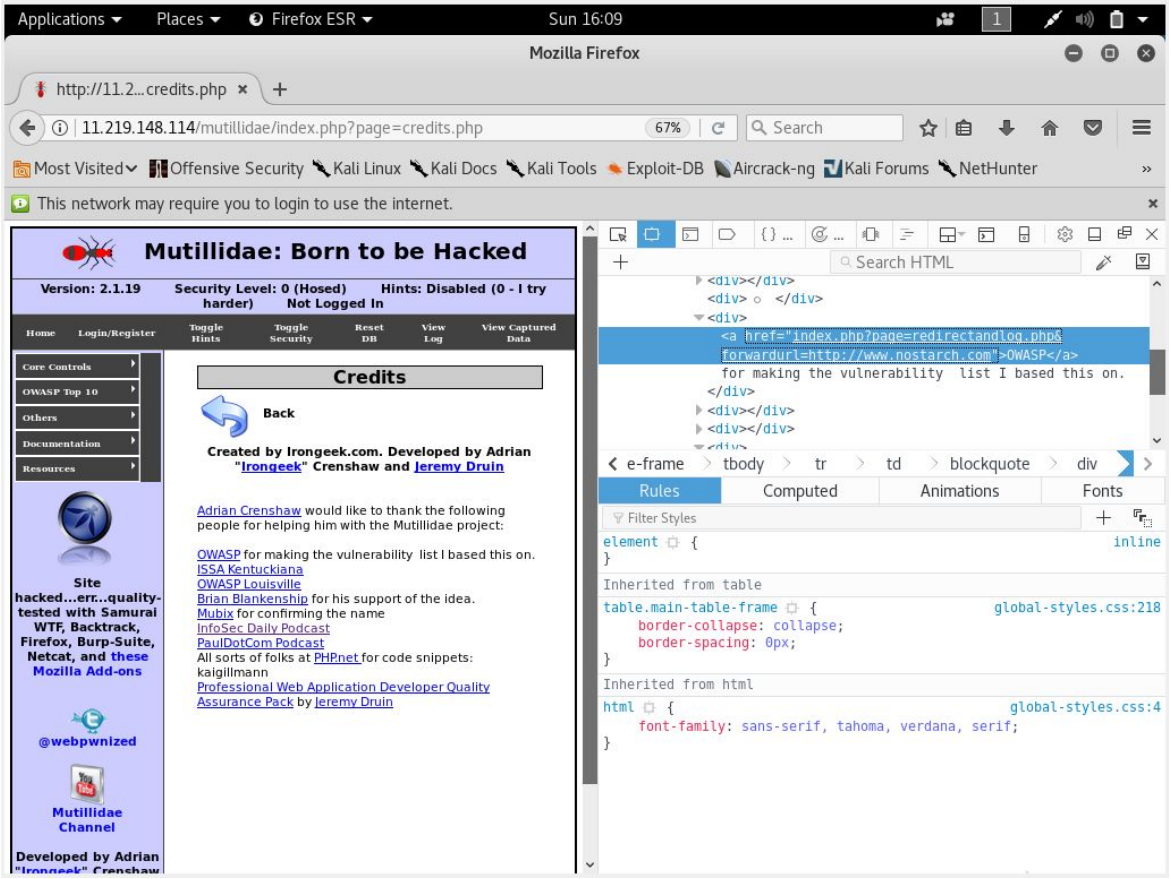
This flaw is due to the use of GET requests instead of POST requests that carry the parameters in the body of the requests instead of query strings like in GET requests.

13.Unvalidated Redirects & Forwards

To exploit this vulnerability, go to the credits.php and change the url redirection property - “forwardurl” of one of the links mentioned in this webpage.



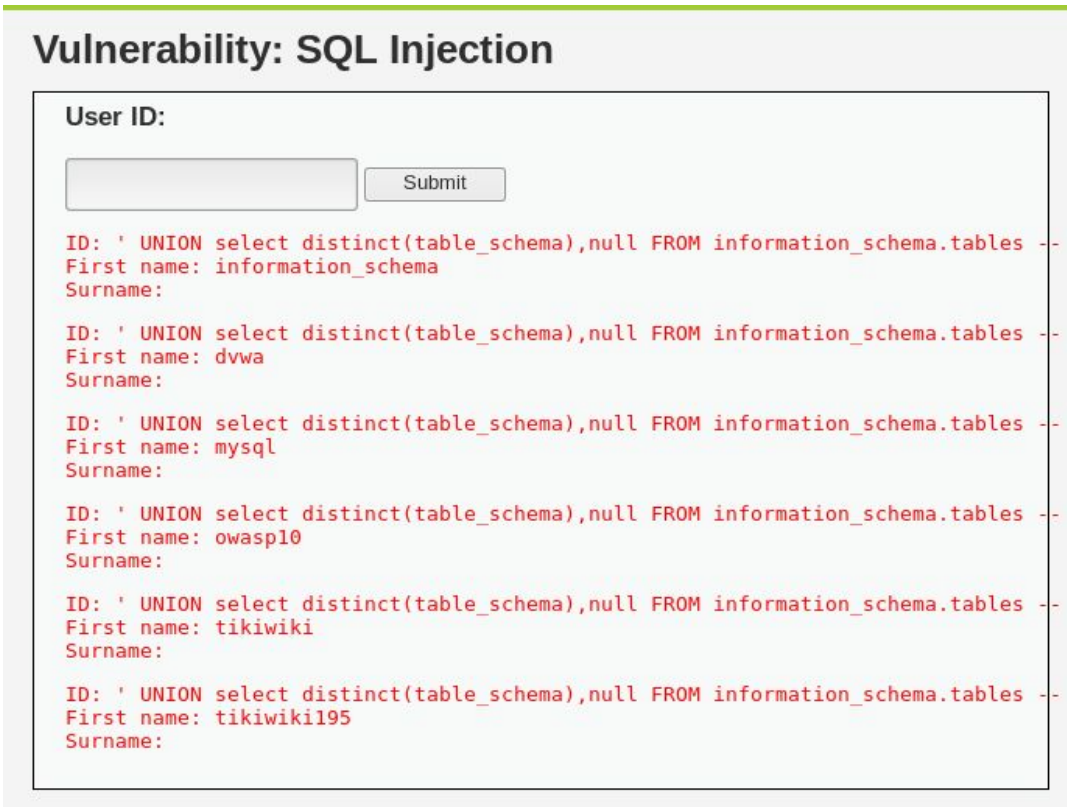
As shown below, OWASP link url is now changed to “nostarch.com”. Now, when the user clicks on this link, they will be redirected to nostarch.com website:



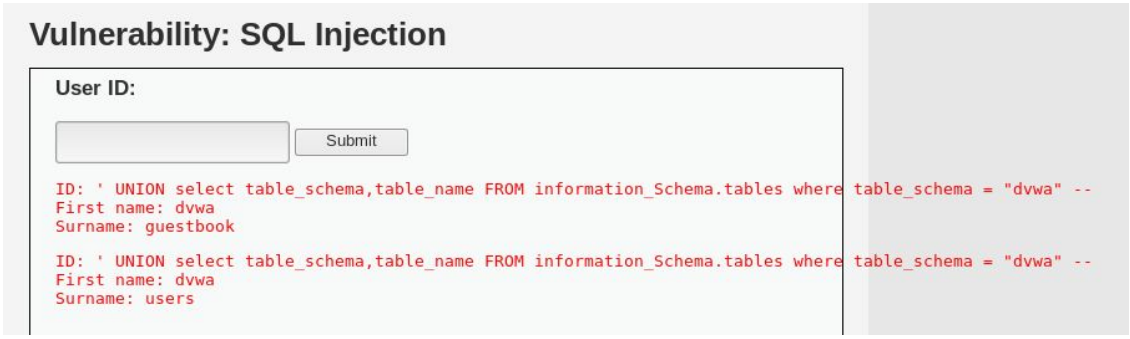
14.SQL Injection (Union exploit)

To exploit this vulnerability,go to DVWA site and select sql injection.Then pass the string as shown

' ‘*UNION select distinct(table_schema),null FROM information_schema.tables* -- ‘ and click submit.Then we get the list of database name (mentioned in First name).



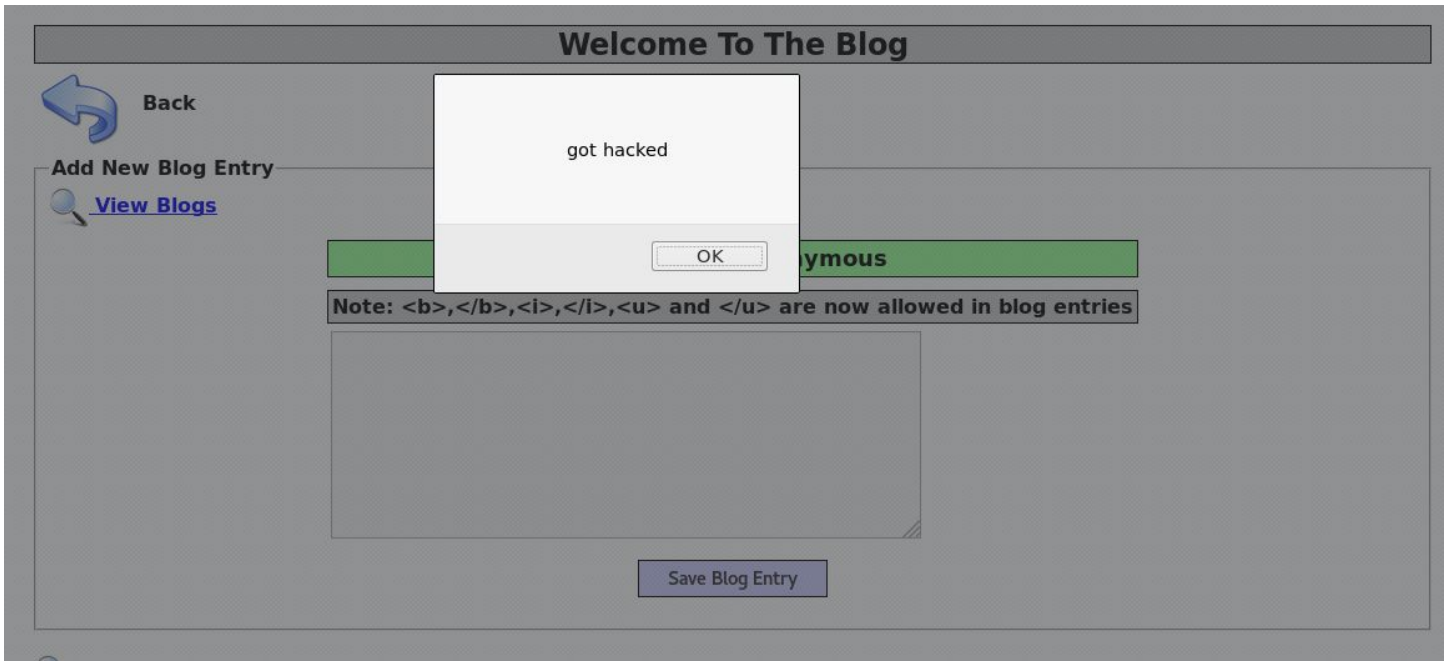
Put input as ‘ ' *UNION select table_schema,table_name FROM information_Schema.tables where table_schema = "dvwa"* -- ‘to determine the table names as shown below.First name contains the database name and Surname contains the table name.Similarly we can obtain the column names of the table also.



15.Persistent Cross Site Scripting

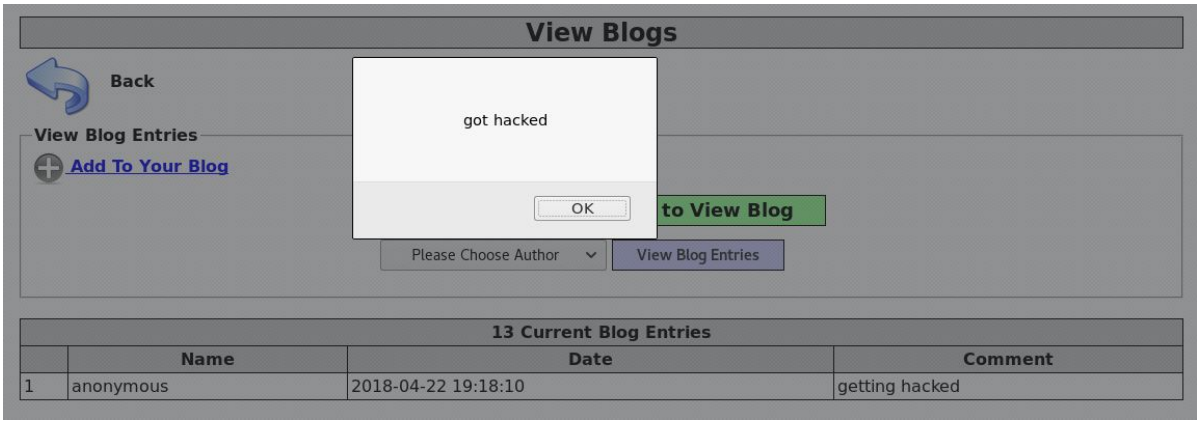
To exploit this vulnerability,go to add-to-your-blog.php page and then add the message as shown in screenshot.

Save the blog entry and then we will get an alert message since we have included a alert message as script and get excuted.



If any user goes to view-someones-blog.php and click on blog entries

The script we included in message will be executed at that page as shown below.This is know as Persistent Cross Site Scripting.



References

http://cwe.mitre.org/top25/

https://www.sans.org/top25-software-errors

https://www.owasp.org/index.php/Cross-Site_Request_Forgery_(CSRF)_Prevention_Cheat_Sheet

https://www.owasp.org/index.php/Broken_Authentication_and_Session_Management

https://www.owasp.org/index.php/OWASP_Cheat_Sheet_Series