

CEH v12 Lesson 6 : Web Application Exploitation Concepts

Learning Outcomes

In this module, you will complete the following exercises:

- Exercise 1 — Web Application Concepts
- Exercise 2 — Web Application Threats
- Exercise 3 — Web Application Hacking Methodology
- Exercise 4 — Footprint and Analyze web Infrastructure

After completing this module, you will be able to:

- Sniff Passwords
- Use Medusa to Crack Passwords
- Perform Broken Authentication Attacks
- Conduct OS Command Injection Attack
- Perform Server-side Includes Injection Attack (SSI)
- Perform Cross-site Scripting Attack
- Perform Cross-site Scripting (XSS) — Reflected (HREF) Attack
- Enumerate web Applications Using Wafwoof
- Perform Website Enumeration using Nmap
- Detect the IP addresses Using the Host Command
- Perform Information Gathering Using Legion

After completing this module, you will have further knowledge of:

- Web Applications and Their Advantages
- Web Application Architecture

- Web Services
- Vulnerability Stack in a web Application
- OWASP Top 10
- Web Application Hacking Methodology Concepts

Lab Duration

It will take approximately **1 hour and 30 minutes** to complete this lab.

Exercise 1 — Web Application Concepts

Most organizations offer their services in the form of a web application and can encompass many different services, such as a client-facing application or a collaboration platform that the users can use to communicate on a project's status. In most cases, access to these web applications is available through subscription.

In this exercise, you will learn about web application concepts.

Learning Outcomes

After completing this exercise, you will have further knowledge of:

- Web Applications and Their Advantages
- Web Application Architecture
- Web Services

Web Applications and Their Advantages

Most users may think of a web application as a website, but an application and website are different. Historically, a website usually consists of static web pages that need to be updated by a web developer when content changes are necessary. A web application is also on the web server, but consists of dynamic web pages that are created by server-side code such as PHP, Ruby, or other languages. Content is driven by user requests.

This interaction has multiple steps. First, the server-side code processes the user request. It then creates a SQL statement that is sent to the back-end database requesting the information the user wants. The database responds by sending only the requested data back to the web application. The web application processes the response and generates HTML code in a user-friendly format. This is then sent to the user's browser

for display and consumption. The interaction between the users and the web application allows each user to customize the content they consume, search, modify their profiles, place orders, and communicate with other users. Let's look at some of the key advantages of web applications.

- They are accessible through the Internet. The users can access them anytime from anywhere if they have an active subscription.
- Web applications can be accessed from different devices. For example, a user can access them from a desktop, laptop, or mobile.
- Most web applications can be customized based on user needs, and there is no dependency on developers. A user can customize the look and feel based on their specific requirements.
- Most web applications are equipped with reporting functions, allowing users to view statistics. For example, a user may want to view a report on the money spent on purchasing certain items through an online banking application.

Web Application Architecture

A web application consists of three layers, which are:

- **Presentation Layer:** also known as a client layer (or front-end), which represents information to a user. This layer includes systems used to fetch required information from the application. When a user requests something while using the application, the browser sends a request to the web server, which will respond to the request.
- **Business Logic Layer:** defines the flow of data within an application. Data is stored in a database and must be retrieved and saved. These actions are driven by user actions and are performed based on the business logic that the developers build into the application.
- **Database Layer:** contains the database and database server that hold the user's data. This data is saved or retrieved from the database based on the user's actions performed within the application.

Web Services

A web service allows communication between two applications, even if they are developed in different languages. For example, your website or application may want to

obtain data from a data provider such as a weather station. Since the data exists on another organization's server, you mostly likely don't have direct access to that data. That organization can make that data available to users by making a web service available. This access may be free or require a subscription. Either way, you have access using some rather simple coding techniques.

This is because a web service uses one of two key messaging protocols used with services, Simple Object Access Protocol (SOAP) and Representational State Transfer (REST). SOAP is an older protocol. It is based on XML and is highly structured. REST is now heavily used and works with various HTTP 1.1 verbs, such as GET, POST, PUT, and DELETE.

A web service is made up of three components, which are:

- **Service Provider:** the platform that hosts web services.
- **Service Requester:** the client that needs to use the web service.
- **Service Registry:** the location that the service provider uses to load the service descriptions, including the web service interface and its implementation details. It includes the details of network locations, bindings, and data types.

Exercise 2 — Web Application Threats

web application threats are common. They can occur for various reasons, and if not handled properly, they lead to various unfavorable outcomes, such as loss of data or application downtime. These vulnerabilities can exist because of various reasons, such as:

- The web application environment, such as the server, is vulnerable to a web application exploitation.
- The web application communicates with a vulnerable application, which leads to exploitation.
- There are inherent vulnerabilities within the web application that can be exploited.
- There are misconfigurations or programming flaws that lead to web application exploitation.

In this exercise, you will learn about various web application vulnerabilities.

Learning Outcomes

After completing this exercise, you will be able to:

- Sniffing the Passwords
- Use Medusa to Crack Passwords
- Perform Broken Authentication Attacks
- Conduct OS Command Injection Attack
- Perform Server-side Includes Injection Attack (SSI)
- Perform Cross-site Scripting Attack
- Perform Cross-site Scripting (XSS) — Reflected (HREF) Attack

After completing this exercise, you will have further knowledge of:

- Vulnerability Stack in a web Application
- OWASP Top 10

Vulnerability Stack in a Web Application

The web application infrastructure consists of seven different layers. Each layer can contain vulnerabilities that allow an attacker to exploit an application:

Layer 7 – Web Application
Layer 6 – Third-party Components
Layer 5 – Web Server
Layer 4 – Database
Layer 3 – Operating System
Layer 2 – Network
Layer 1 – Security

Figure 2.1: Diagram showing the 7 layers of infrastructure; Web Applications, Third-Party Components, Web Servers, Databases, Operating Systems, Networks, and Security.

Let's look at each layer in detail:

Layer 7 – Web Application Vulnerabilities at this layer are due to flaws in business logic. There could also be coding flaws that can lead to attacks, such as SQL Injection or cross-site scripting.
Layer 6 – Third-party Components Vulnerabilities exist due to integration of the third-party components. A web application may not have a technical flaw or vulnerability, but can still be exploited due to vulnerable components which includes things such as an add-on or web service provided by a third party.
Layer 5 –

Web Server Web servers are prone to footprinting and reconnaissance, allowing an attacker to gain a lot of information. For example, an attacker finds the name and version of a web server and can search for vulnerabilities in the vulnerabilities database. **Layer 4 — Database** A database, if not patched, can have vulnerabilities that tools such as SQLMAP can exploit. **Layer 3 — Operating System** An operating system must be patched when updates are available. An attacker can use various methods to attack an OS (such as social engineering), to deliver malware which will exploit an operating system or its vulnerabilities. **Layer 2 — Network** Attackers can attack edge routers and firewalls with DoS attacks. If not configured properly, switches can be flooded with requests to overflow their CAM tables. Attackers also attempt to sniff network traffic.

Layer 1 — Security A network can have an intrusion detection system (IDS) or intrusion prevention system (IPS) to detect and block malicious traffic. However, attackers often use IDS/IPS and firewall evasion methods to generate no alert.

OWASP Top 10

In many scenarios, organizations use off-the-shelf applications. In other scenarios, the organizations use a mix of off-the-shelf and custom applications. The patches and updates for the off-the-shelf applications are created and released by the vendor that created the application. It is now the organization's responsibility to update their existing off-the-shelf applications to patch these applications. In the case of custom applications, this is not the scenario. The custom applications are developed to meet a specific business need. An organization may have an in-house development team to create the application or outsource it to another vendor. Releasing updates is a common issue with custom applications. The vendors do not provide updates in most cases. Therefore, these inherent vulnerabilities will continue to be present until the application is used.

There are several known web application vulnerabilities. Open web Application Security Project, more commonly known as OWASP, releases the top 10 web application vulnerabilities, which are released after every few years. This data is collected from various organizations through extensive research, and then the top 10 web applications are selected. Remember — there are hundreds of web application vulnerabilities, and therefore, when doing a penetration test, you can focus on the key ones but do not ignore to test for the other vulnerabilities. The top 10 web application vulnerabilities of 2021 released by OWASP are:

- **A01:2022: Broken Access Control**

- **Ao2:202:** Cryptographic Failures
- **Ao3:2021:** Injection
- **Ao4:2021:** Insecure Design
- **Ao5:2021:** Security Misconfiguration
- **Ao6:2021:** Vulnerable and Outdated Components
- **Ao7:2021:** Identification and Authentication Failures
- **Ao8:2021:** Software and Data Integrity Failures
- **Ao9:2021:** Security Logging and Monitoring Failures
- **A10:2021:** Server-Side Request Forgery

Task 1 — Sniffing Passwords

Even though sniffing the password does not fall into “offline password cracking,” it is a great method to capture the password transmitted in unencrypted form.

In this task, you will learn about sniffing passwords. To do this, perform the following steps:

Step 1

Ensure you have powered on all the devices listed in the introduction and connect to **PLABKALI01**.

Log in using the following credentials:

Username:

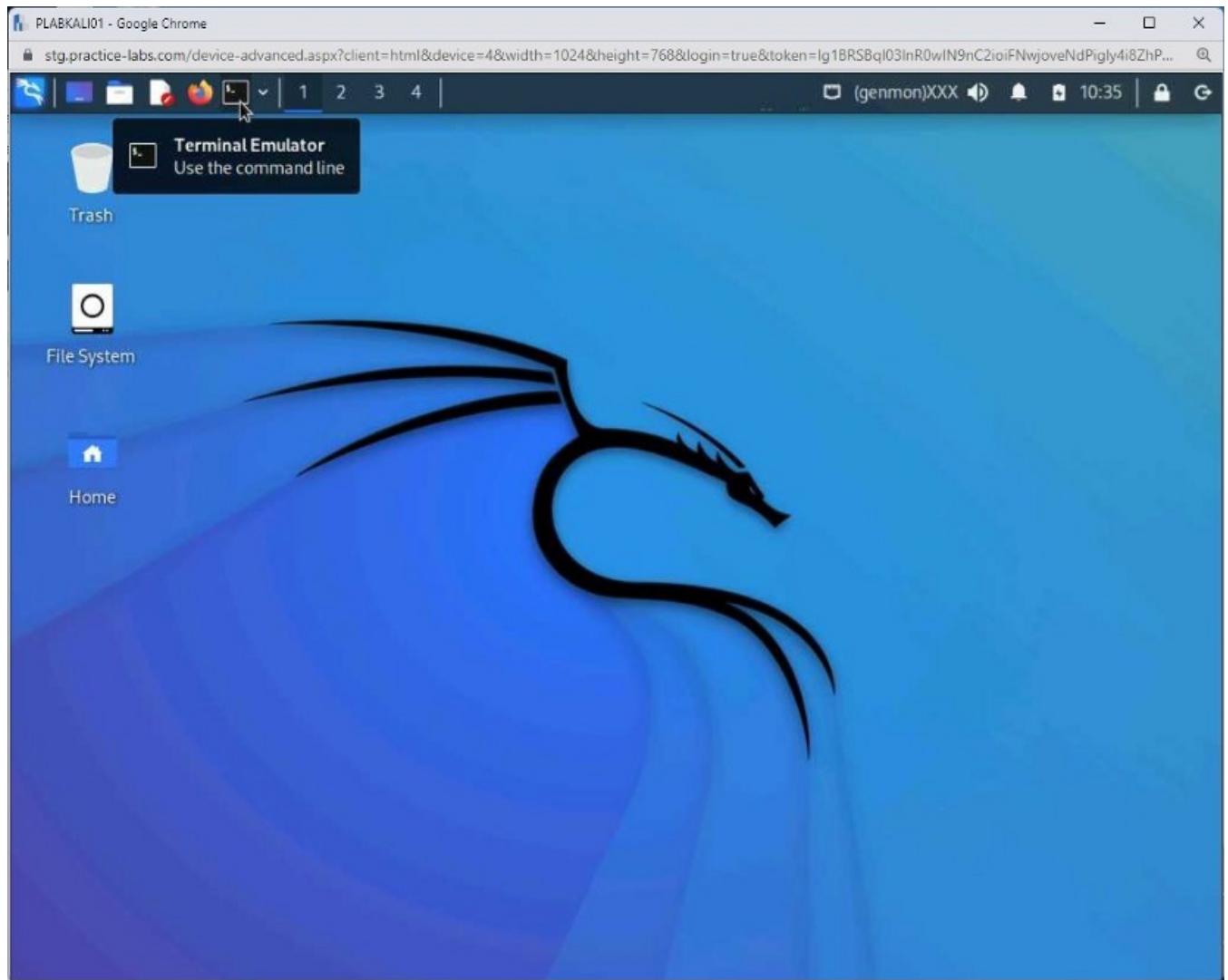
root

Password:

Password

The desktop of **PLABKALI01** is displayed.

Open a new terminal window by clicking the **Terminal Emulator** icon on the taskbar.



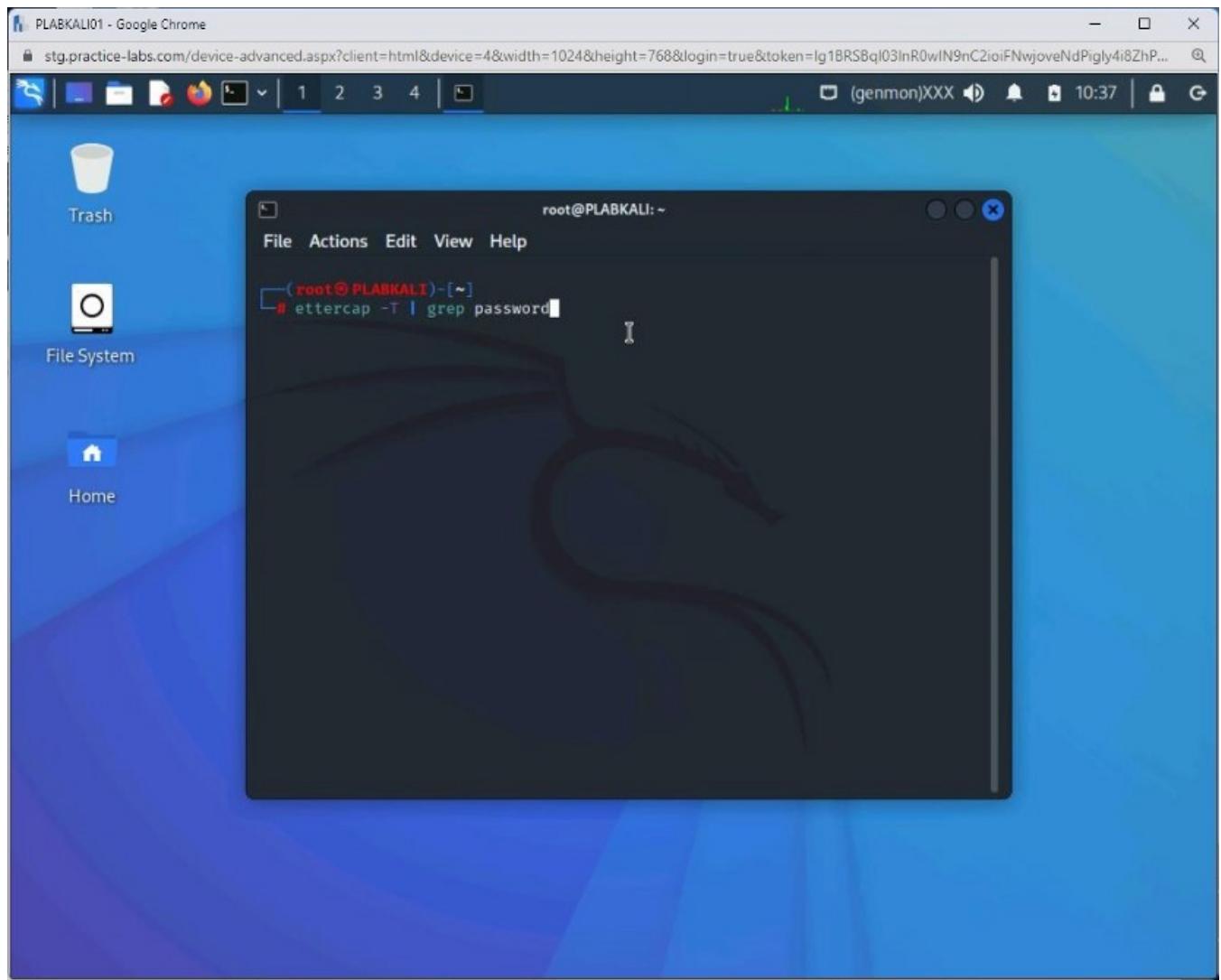
Step 2

The terminal window is displayed.

The **Ettercap** tool is used to capture a password from an unencrypted session. Type the following command:

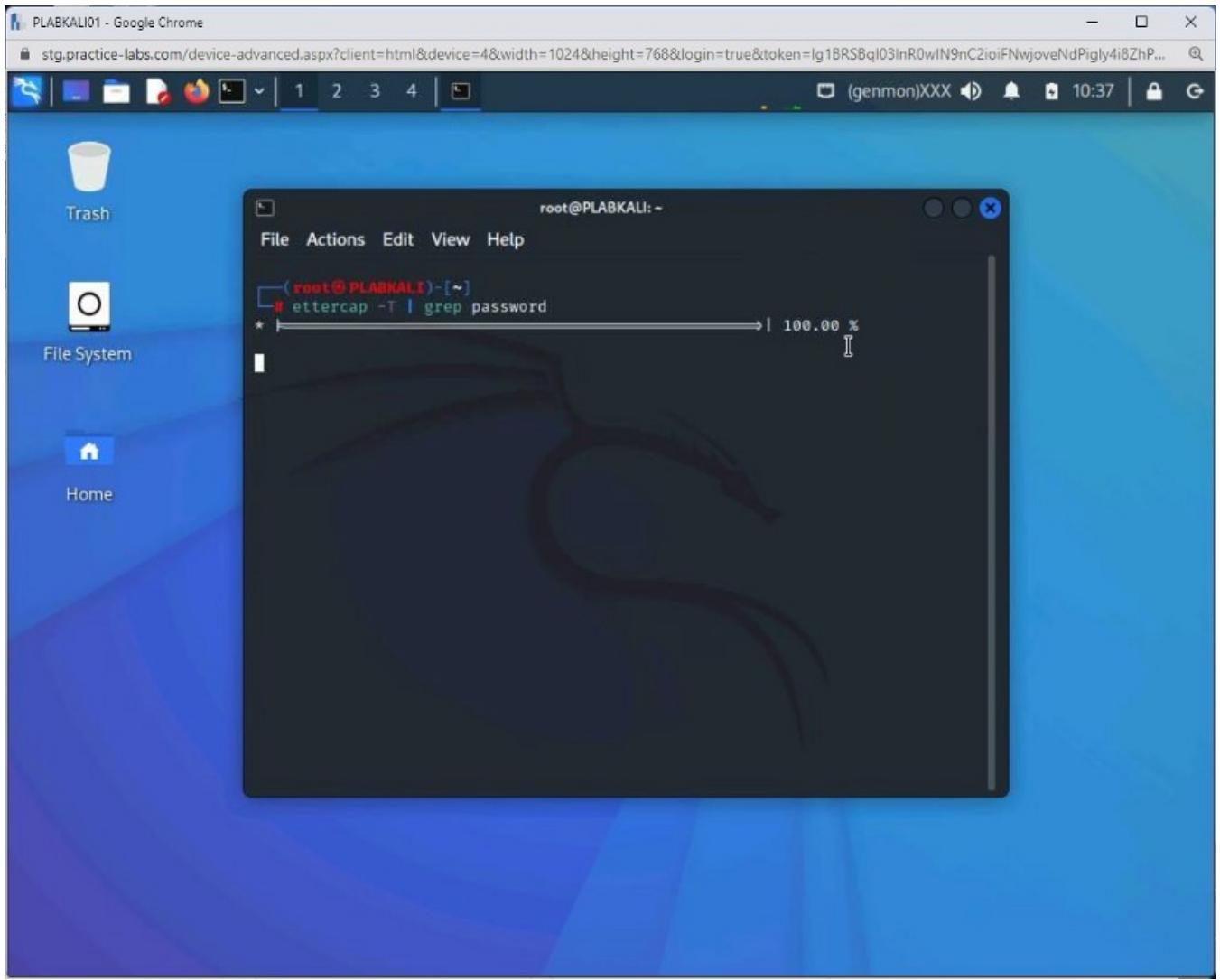
```
ettercap -T | grep password
```

Press **Enter**.



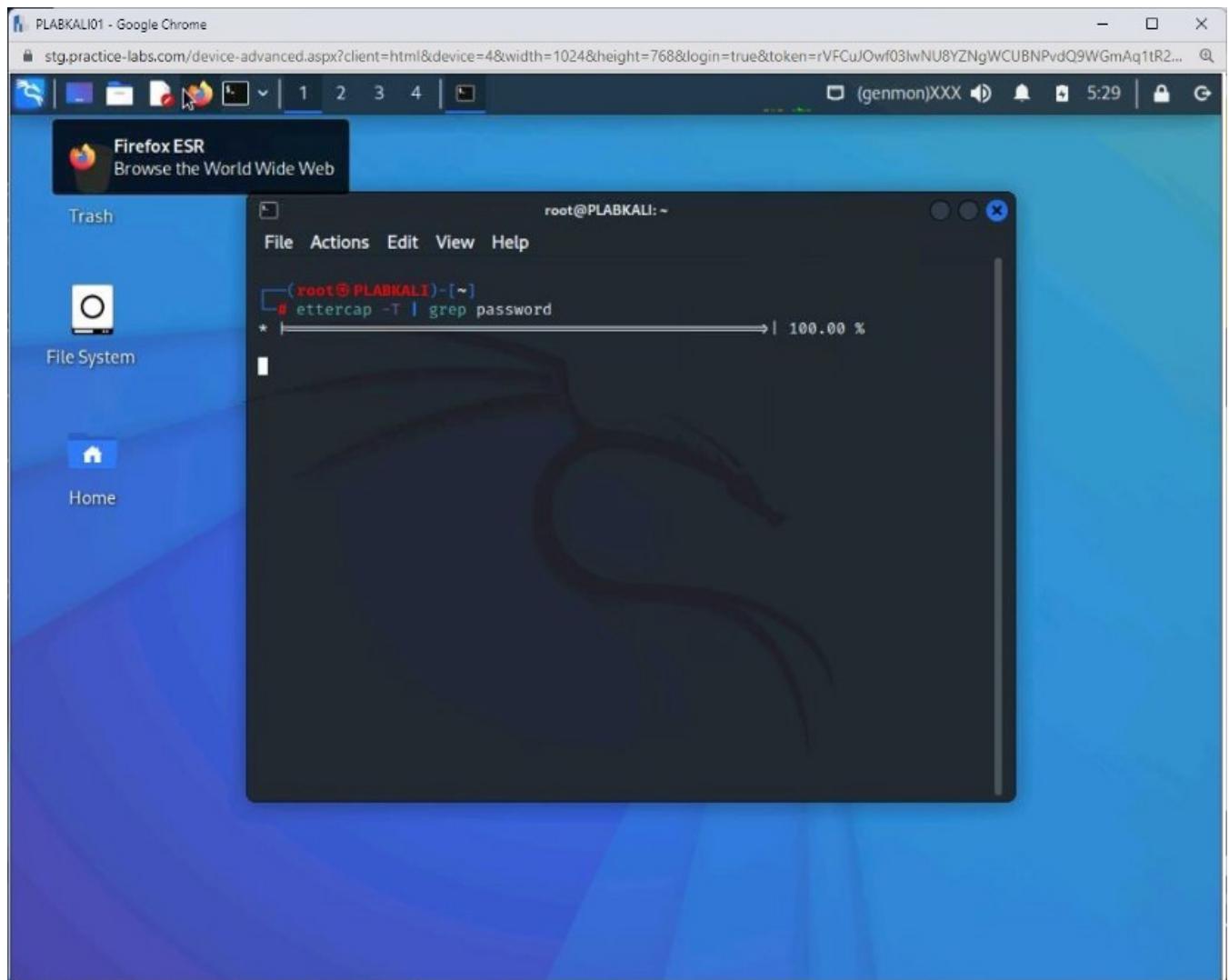
Step 3

The sniffing process starts.



Step 4

Click the **Firefox ESR** icon on the taskbar.

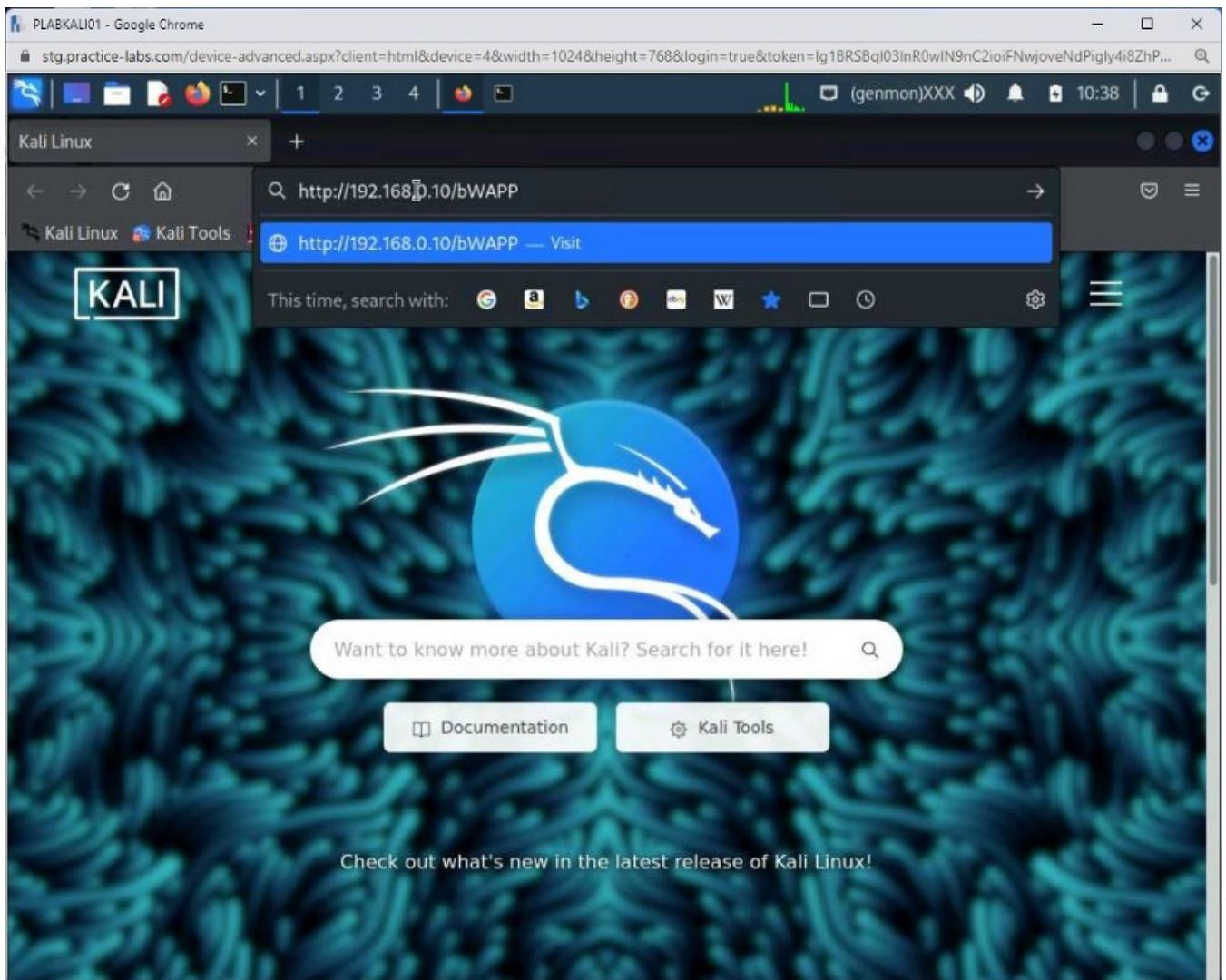


Step 5

The Firefox window is displayed. In the address bar, type the following URL:

<http://192.168.0.10/bWAPP>

Press **Enter**.



Step 6

The login page is displayed.

In the **Username** text box, type the following:

bee

In the **Password** text box, type the following:

bug

Click Login.

PLABKALI01 - Google Chrome

stg.practice-labs.com/device-advanced.aspx?client=html&device=4&width=1024&height=768&login=true&token=rVFCuJOwnf03lwNU8YZNgWCUBNPvdQ9WGmAq1tR2... (genmon)XXX

bWAPP - Login X

1 2 3 4 5:30

← → C ⌂ ⌂ 192.168.0.10/bWAPP/login.php ☆

Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hacking DB OffSec

an extremely buggy web app !

Login New User Info Talks & Training Blog

/ Login /

Enter your credentials (bee/bug).

Login:

Password:

Set the security level:









bWAPP is licensed under  © 2014 MME BVBA / Follow [@MME_IT](#) on Twitter and ask for our cheat sheet, containing all solutions! / Nee

Step 7

Click the terminal window to bring it to the foreground.

Notice that the **username** and **password** are now captured.

PLABKALI01 - Google Chrome
stg.practice-labs.com/device-advanced.aspx?client=html&device=4&width=1024&height=768&login=true&token=rVFCuJowf03lwNU8YZNgWCUBNPvdQ9WGmAq1tR2...
bWAPP - Portal

Kali Linux Kali Tools

bWAPP - Portal

File Actions Edit View Help

```
(root@PLABKALI) [~] # ettercap -T | grep password
* [100.00 %] 100.00 %

<p><label for="password">Password:</label><br />
<input type="password" id="password" name="password" size="20" autocomplete="off"></p>
login=b3e&password=bug6security_level=0&form=submit
CONTENT: login=b3e&password=bug6security_level=0&form=submit
<td><a href="#">Change Password</a></td>
```

Bugs Change f Create User Set Security Level Reset Credits

Blog Logout

/ Portal /

bWAPP, or a buggy Web application, is a free and open source deliberately insecure web application. It helps security enthusiasts, developers and students to discover and to prevent web application bugs. bWAPP covers all major known web vulnerabilities, including all risks from the OWASP Top 10 project. It is for security testing and educational purposes only.

Which bug do you want to hack today? :)

bWAPP v2.2

/ A1 - Injection /

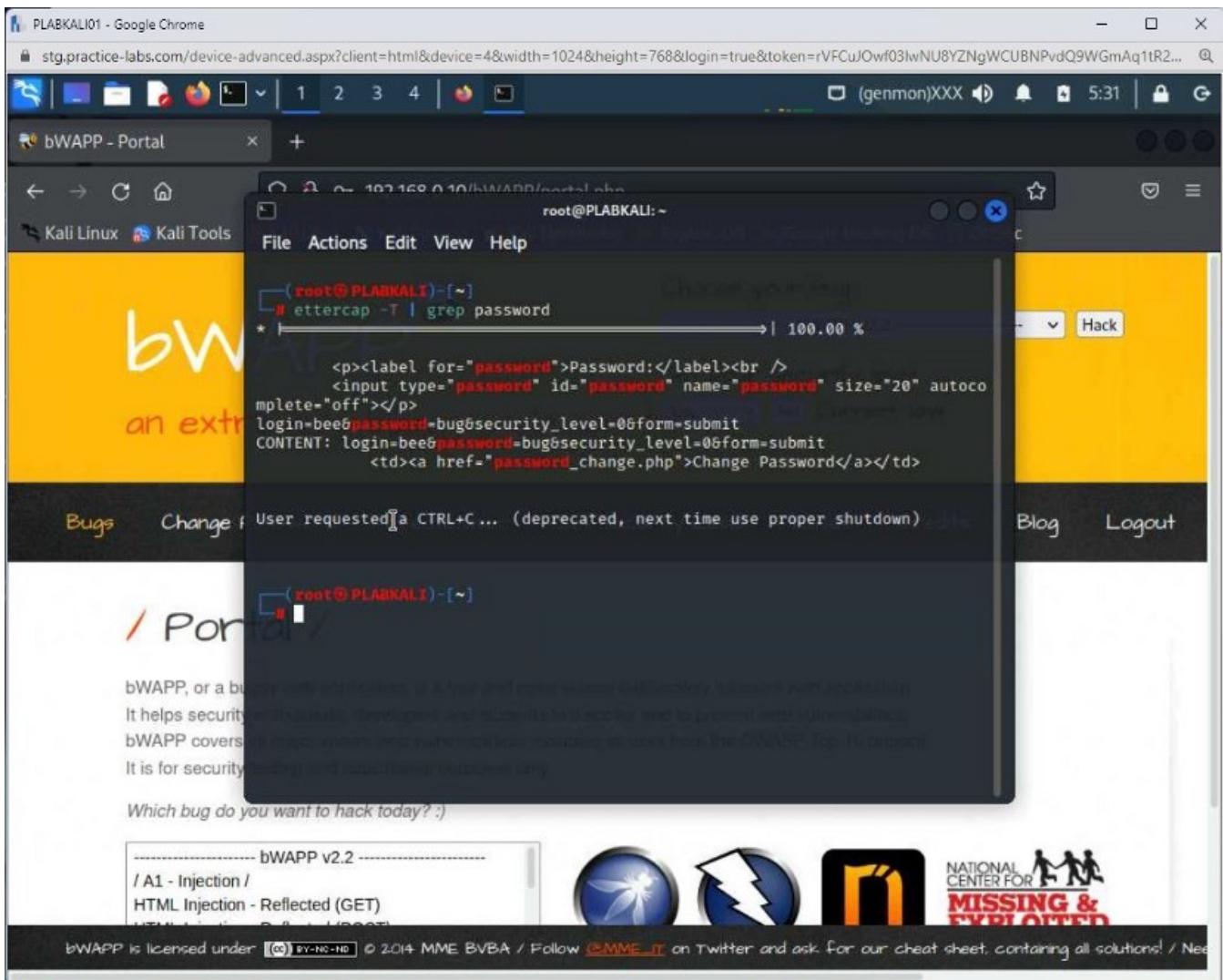
HTML Injection - Reflected (GET)

NATIONAL CENTER FOR MISSING & EXPLOITED

bWAPP is licensed under CC BY-NC-ND © 2014 MME BVBA / Follow @MME_IT on Twitter and ask for our cheat sheet, containing all solutions! / Need

Step 8

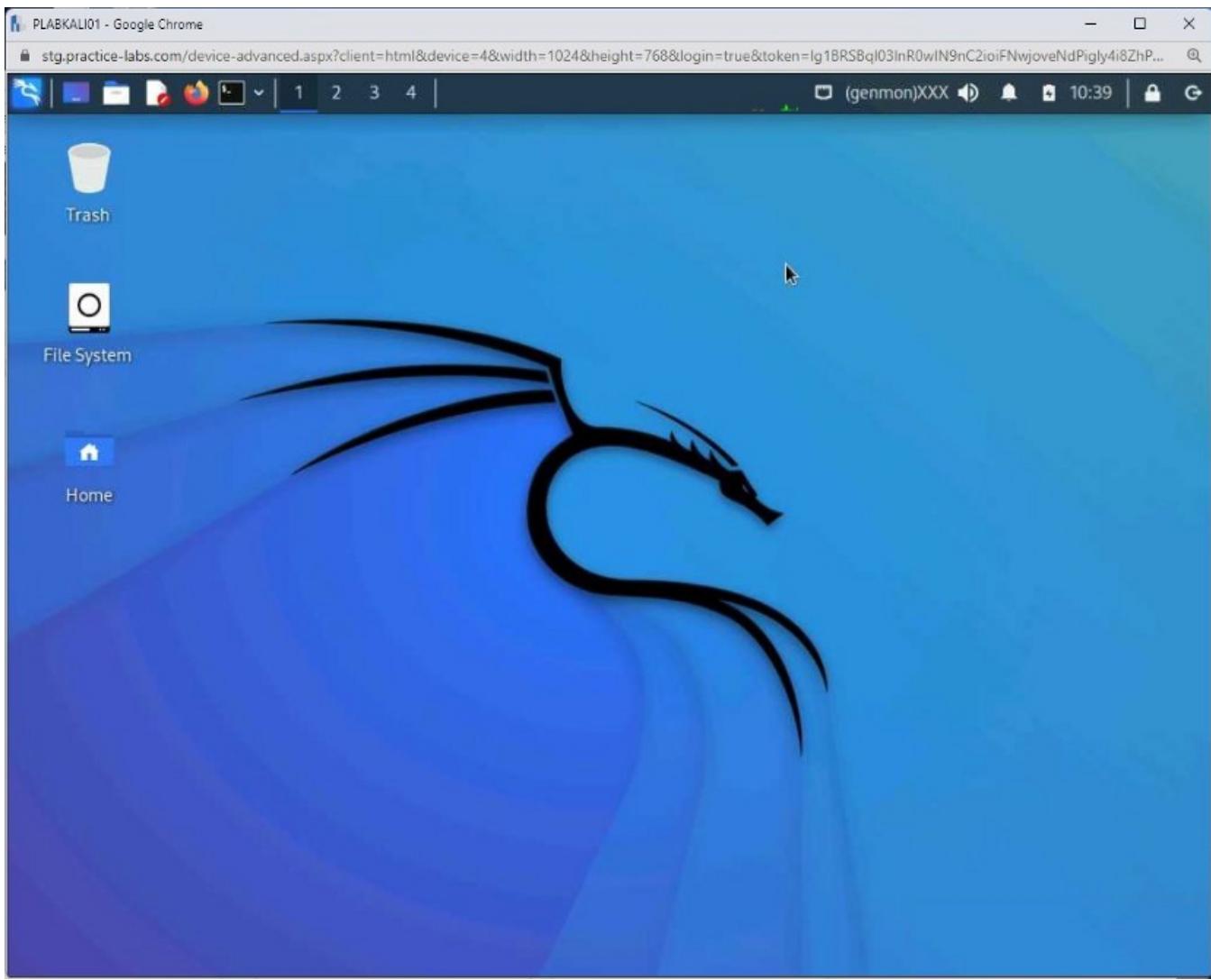
Press the **Ctrl + C** keys to break the sniffing process.



Step 9

Close all open windows.

You are now back on the **PLABKALI01** desktop.



Close the Firefox ESR window. Keep the terminal window open.

Task 2 — Use Medusa to Crack Passwords

Medusa is a login cracking application. It works well with web applications and can use different protocols, as well as not just cracking passwords but also usernames. You can supply two different wordlist files, usernames and passwords as inputs and it can crack both simultaneously.

In this task, you will use Medusa to crack the username and password on the bWAPP web application. To do this, perform the following steps:

Step 1

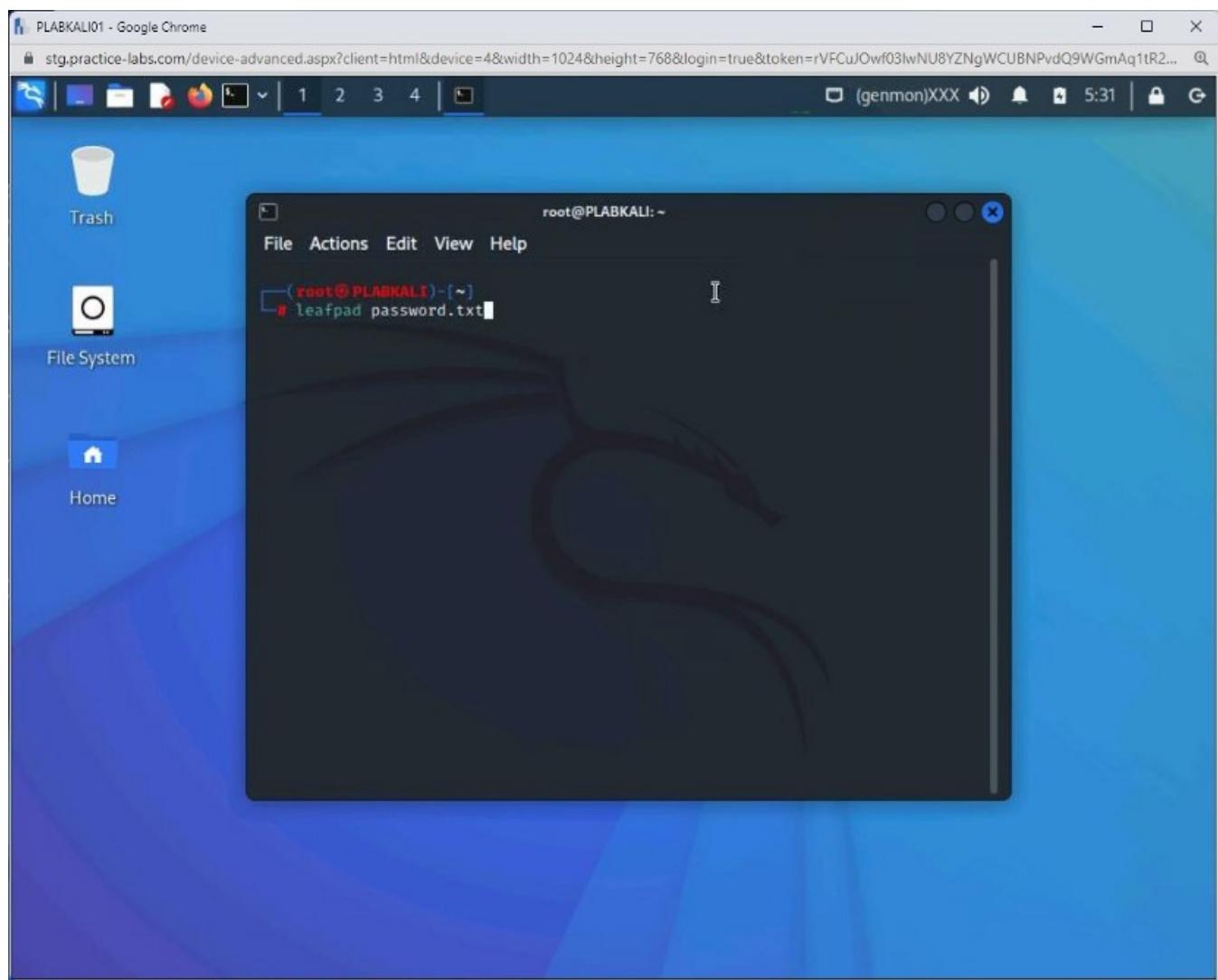
Connect to **PLABKALI01**. Open a new terminal window.

You have an option to use a pre-defined wordlist. There are several wordlists available that have grown into Gigabytes of size. You can also download the wordlists from the Internet. An alternative is to create a small wordlist manually, which you will do now.

This file will contain the keywords that will be used for guessing the password. In the command prompt window, type the following command:

```
leafpad password.txt
```

Press **Enter**.



Step 2

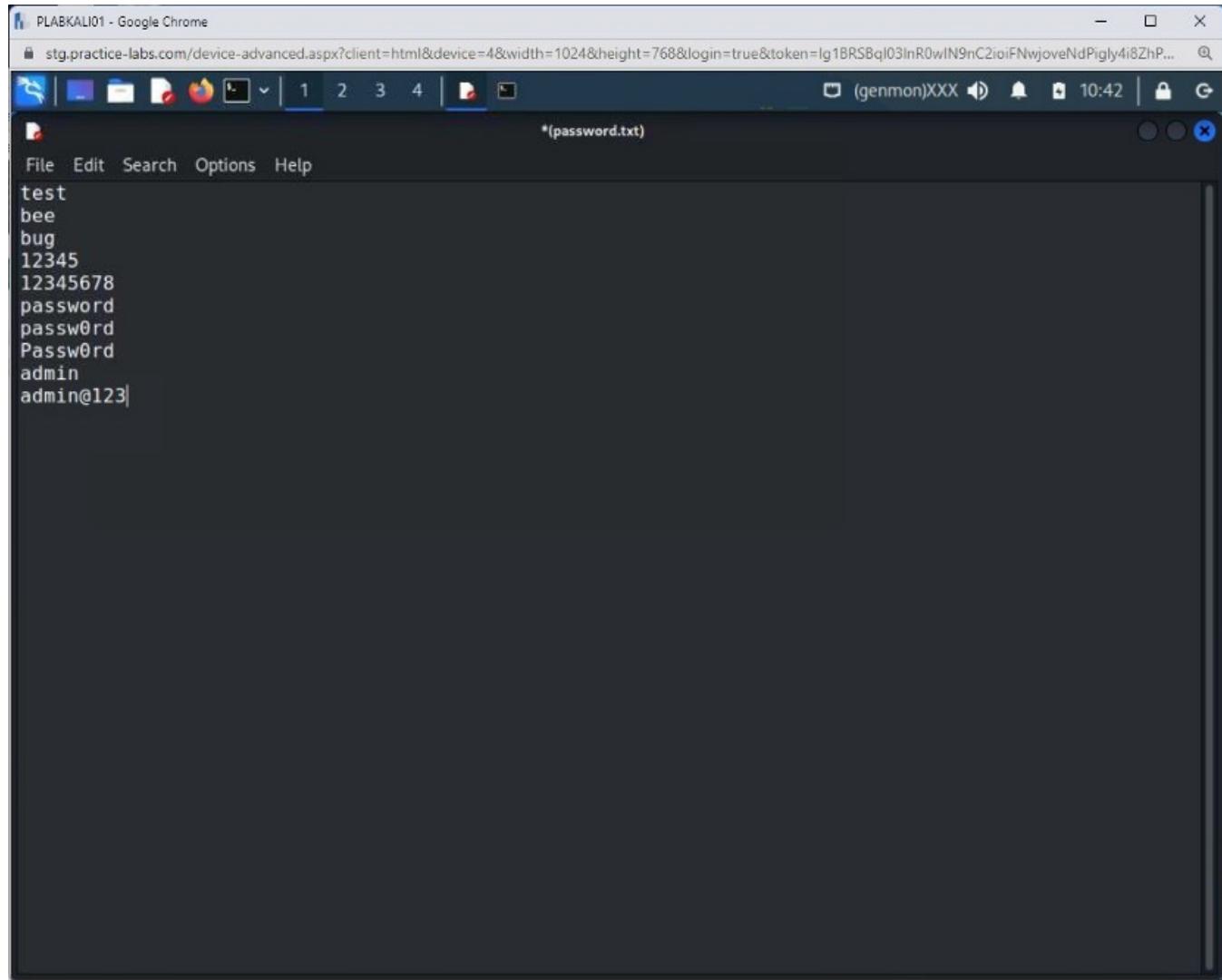
Leafpad opens with a file named (**password.txt**).

Type the following words:

```
test  
bee  
bug  
12345
```

```
12345678  
password  
passw0rd  
Passw0rd  
admin  
admin@123
```

Press **Enter** after each word except the last one.



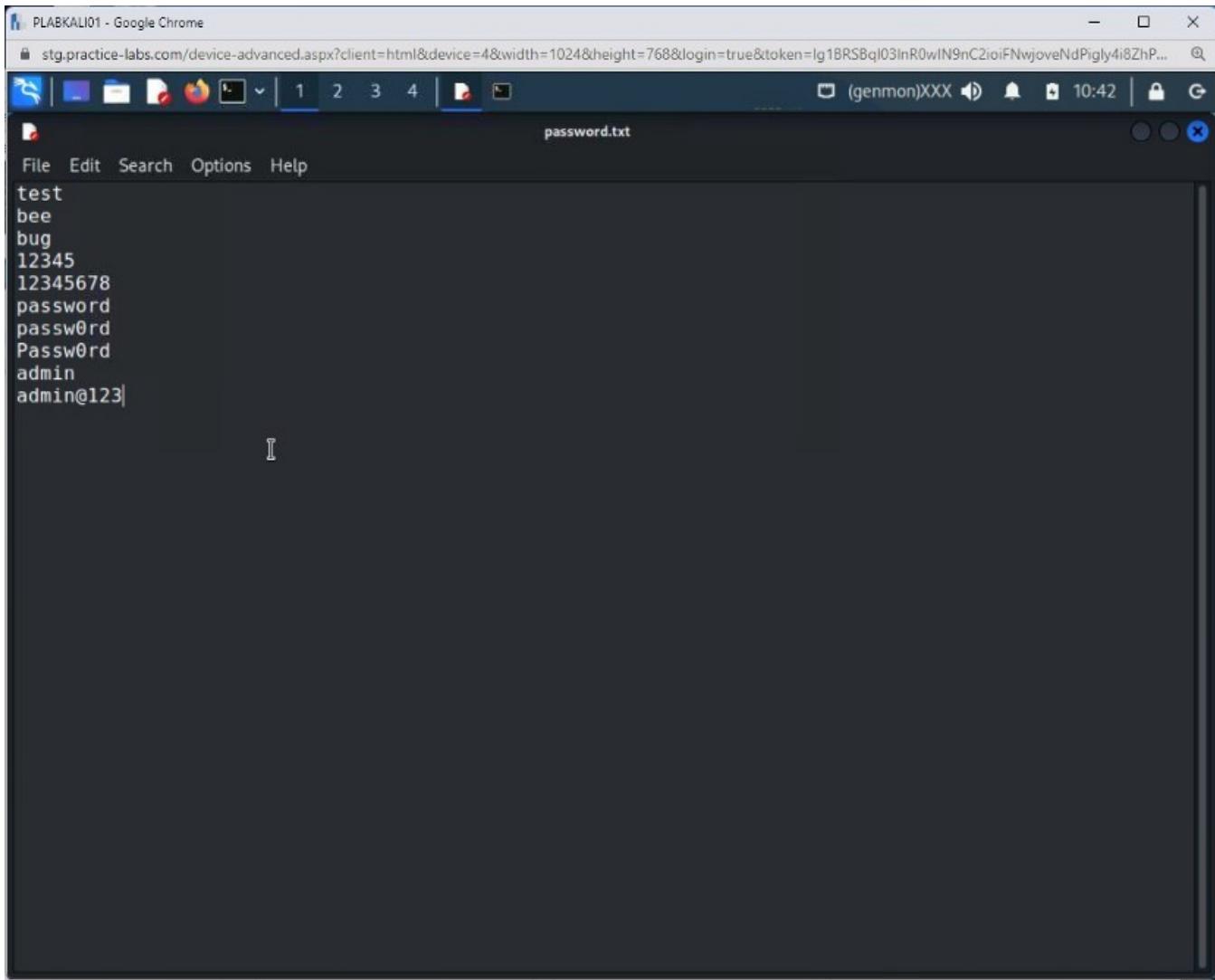
The screenshot shows a terminal window titled "PLABKALI01 - Google Chrome". The window contains a file named "password.txt" with the following content:

```
test  
bee  
bug  
12345  
12345678  
password  
passw0rd  
Passw0rd  
admin  
admin@123|
```

Step 3

Press **Ctrl + S** to save the file.

Close the **password.txt** file.



The screenshot shows a terminal window titled "password.txt". The window has a dark background and a light-colored text area. At the top, there is a menu bar with "File", "Edit", "Search", "Options", and "Help". Below the menu, there is a list of password guesses:

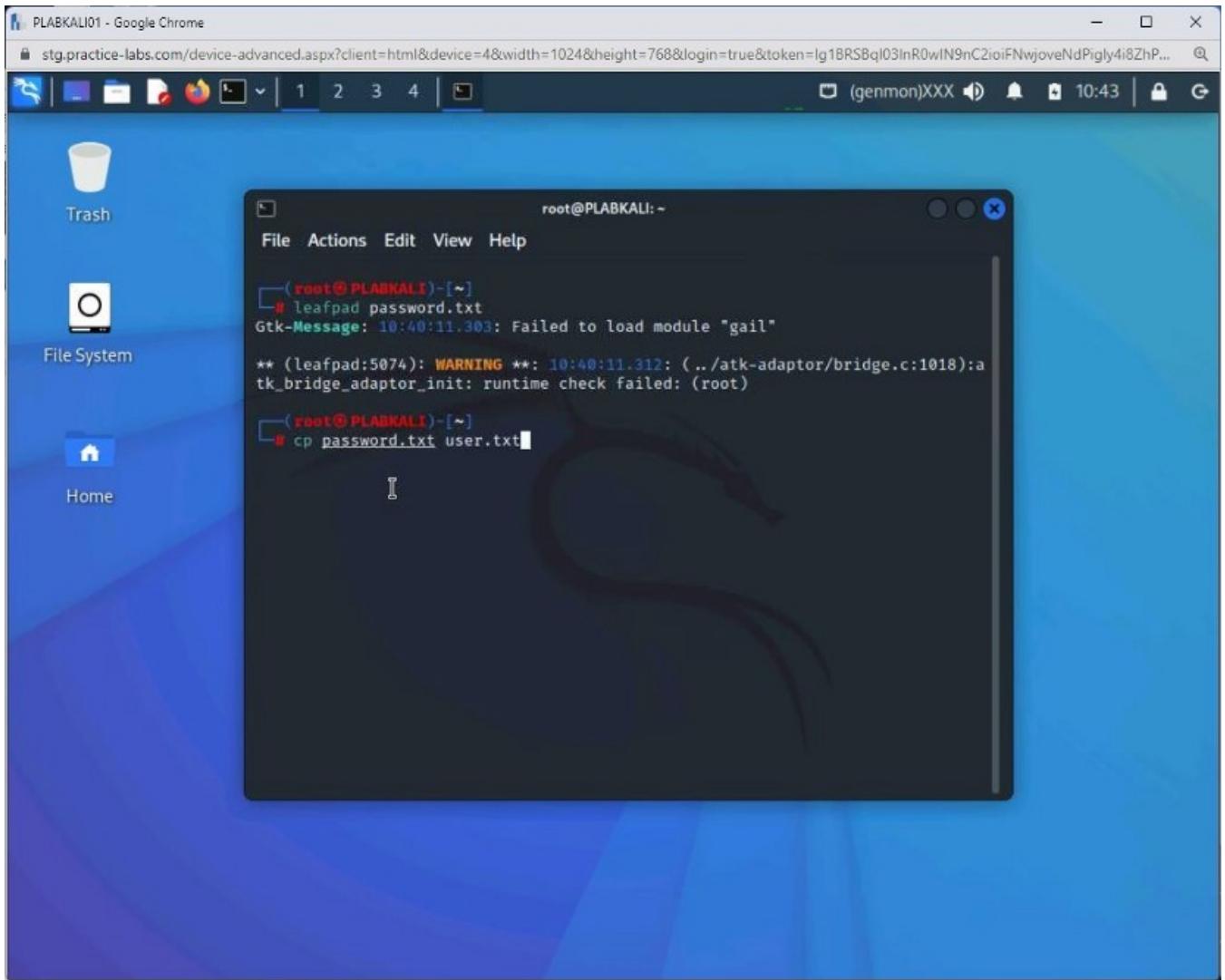
```
test
bee
bug
12345
12345678
password
passw0rd
Passw0rd
admin
admin@123|
```

Step 4

Let's make a copy of this file as the usernames file. Type the following command:

```
cp password.txt user.txt
```

Press **Enter**.

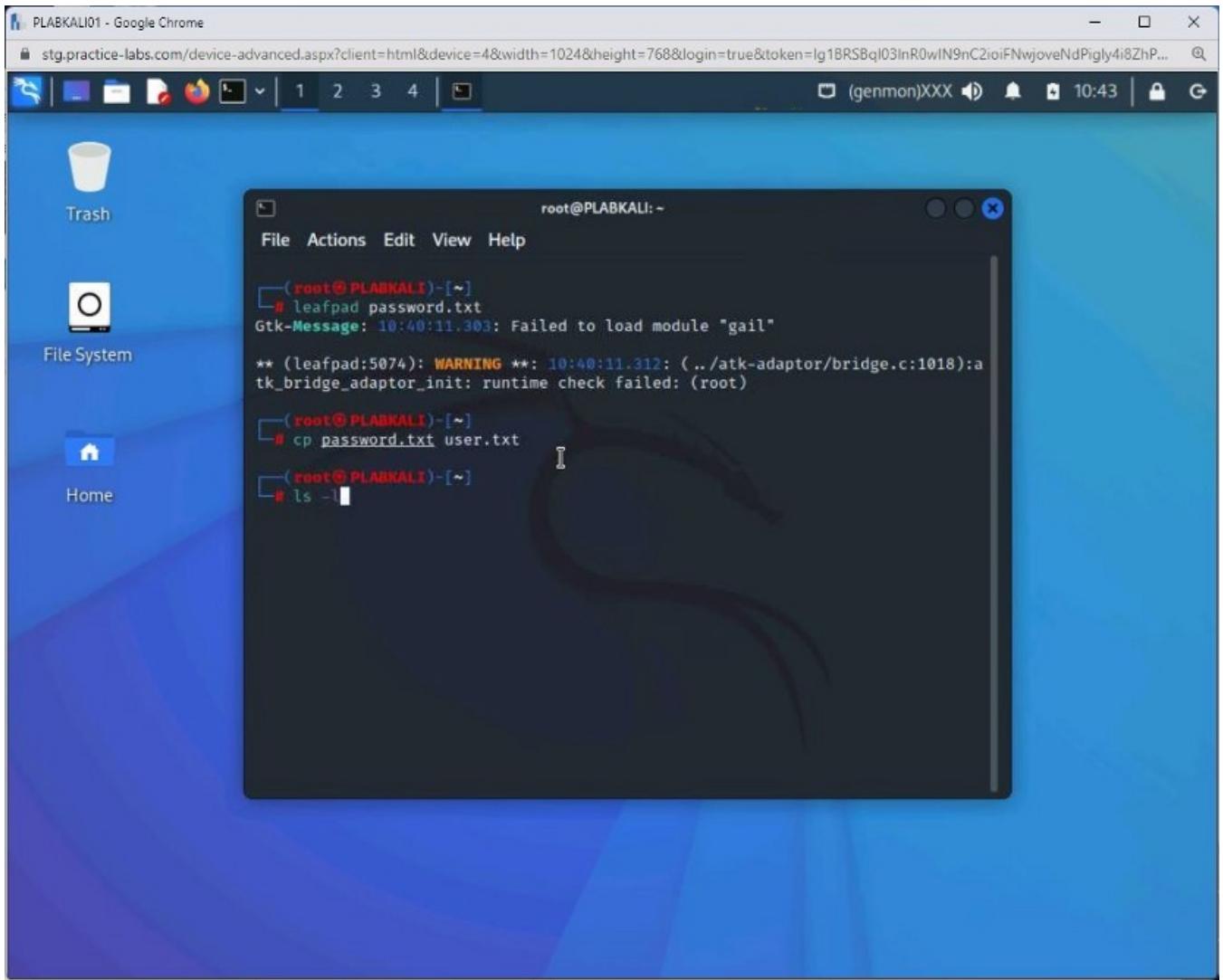


Step 5

The **cp** command does not return any output. To verify you have two files, **plab.txt** and **user.txt**, type the following command:

```
ls -l
```

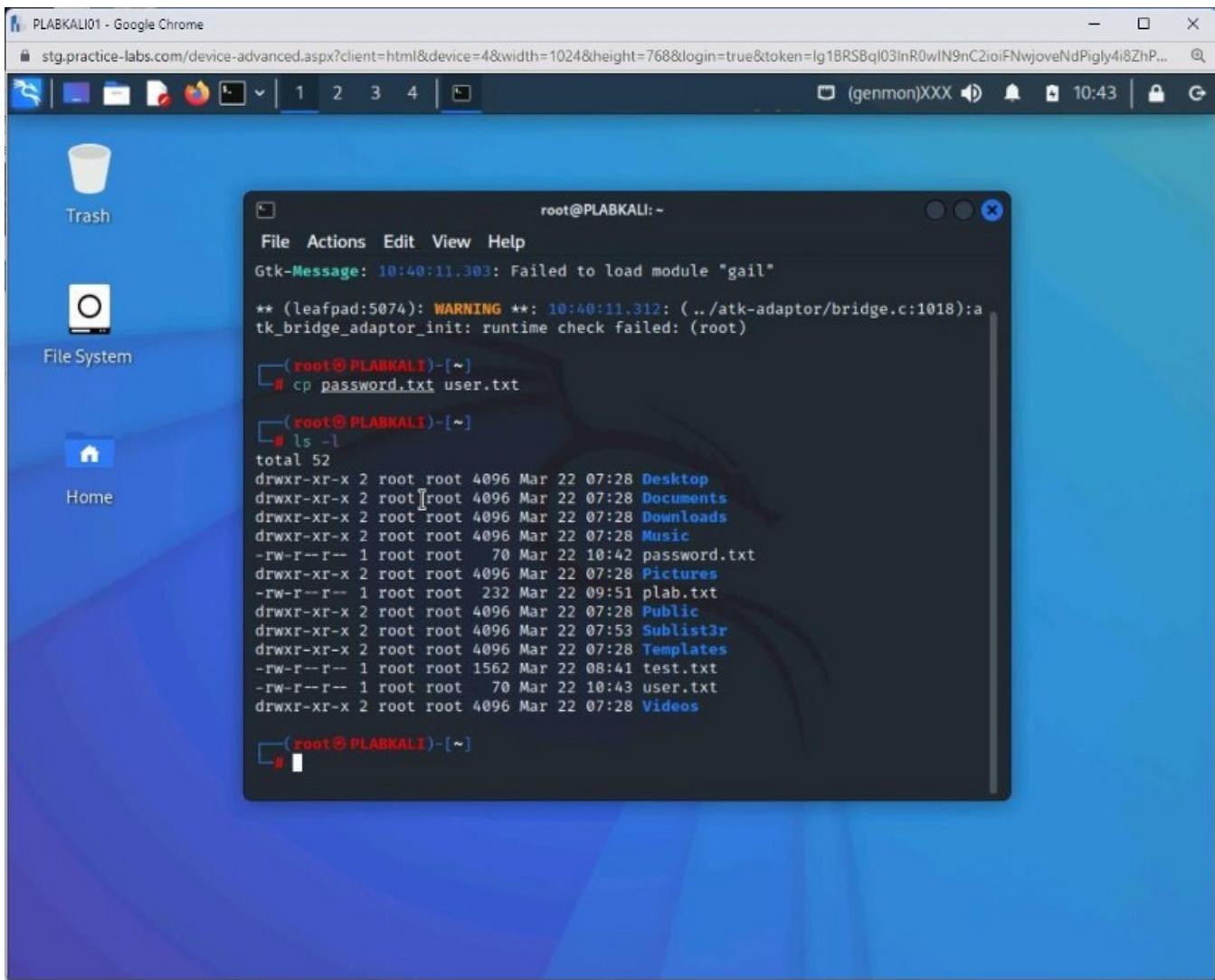
Press **Enter**.



Step 6

The file listing contains both files.

Note: Depending on which modules have been completed prior to this one, the plab.txt may or may not already exist.



Step 7

Clear the screen by entering the following command:

```
clear
```

You will now use the **plab.txt** as the password wordlist and attempt to guess the password for the username **bee**. You will attempt to break the password of the FTP account running on the webserver, **192.168.0.10**. Type the following command:

Note: The command below uses the following parameters:

-h: IP address of the target system

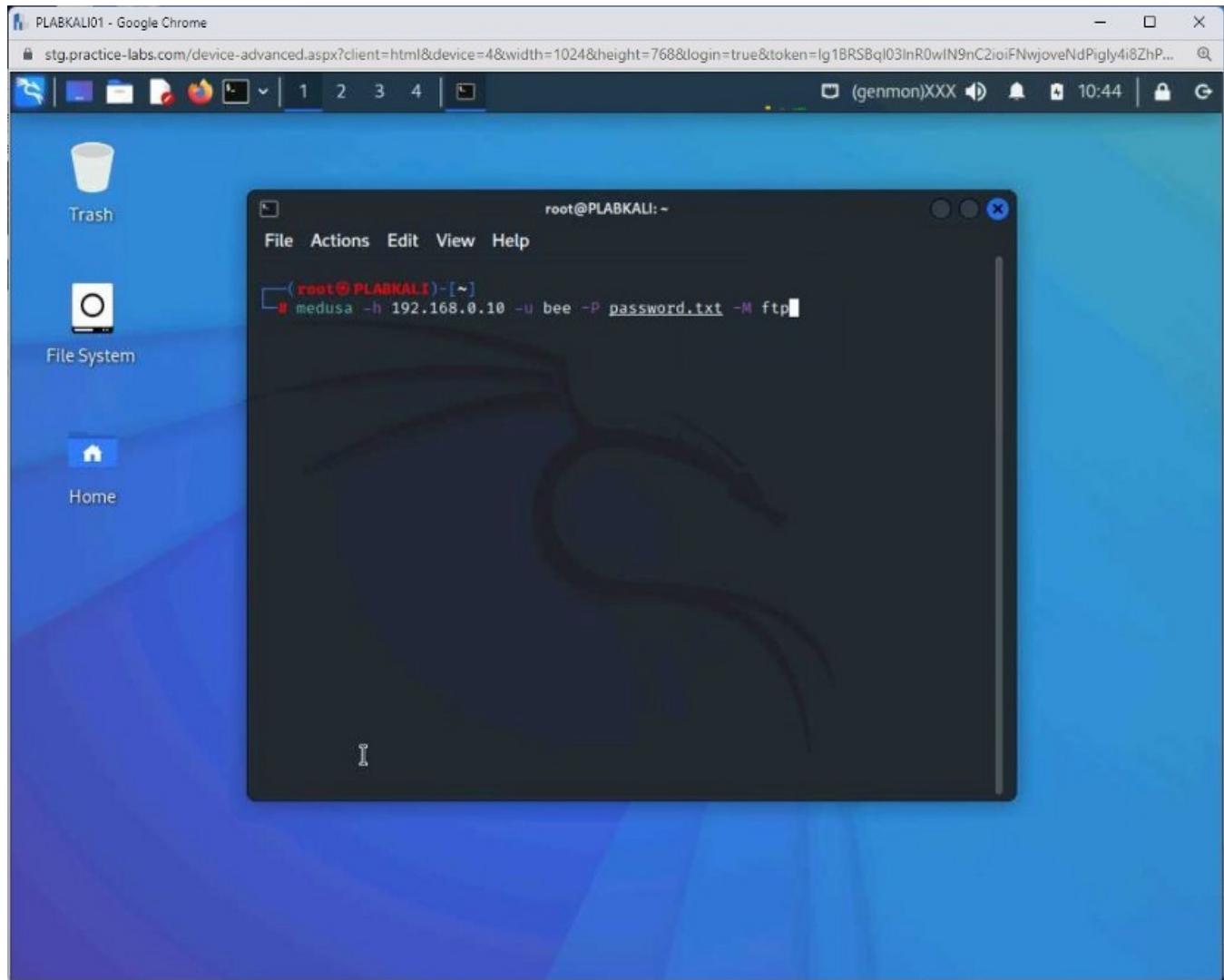
-u: Username. If **-U** is used, you need to specify the username wordlist.

-P: Password wordlist. If **-p** is used, then you can specify a single password.

-M: Module used for cracking the password

```
medusa -h 192.168.0.10 -u bee -P password.txt -M ftp
```

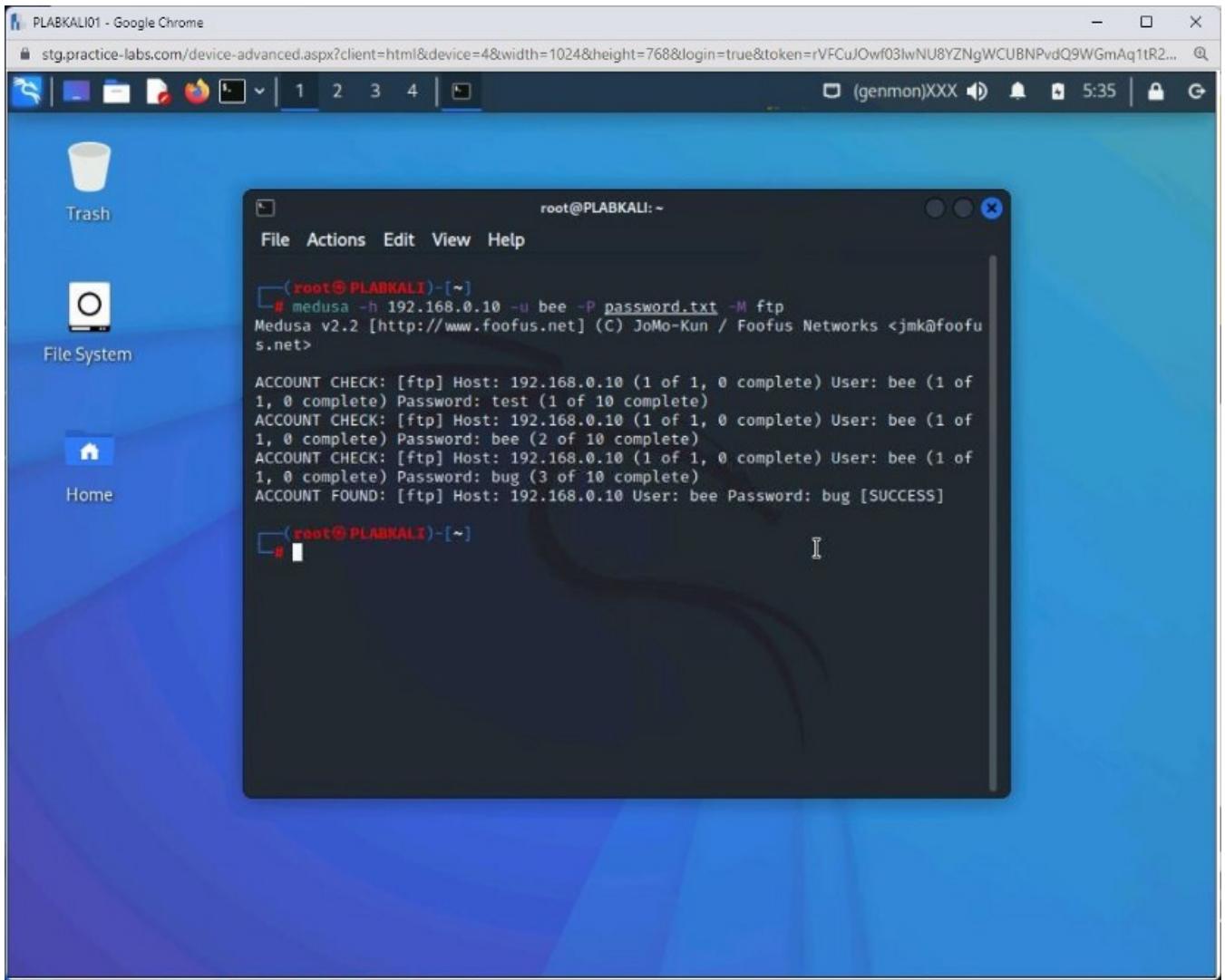
Press **Enter**.



Step 8

The password cracking process starts. It uses each word from the **password.txt** against the username **bee**.

Finally, the correct password is found. The last statement lists the password as **bug**. Notice that after the correct password is found, Medusa stops the process.

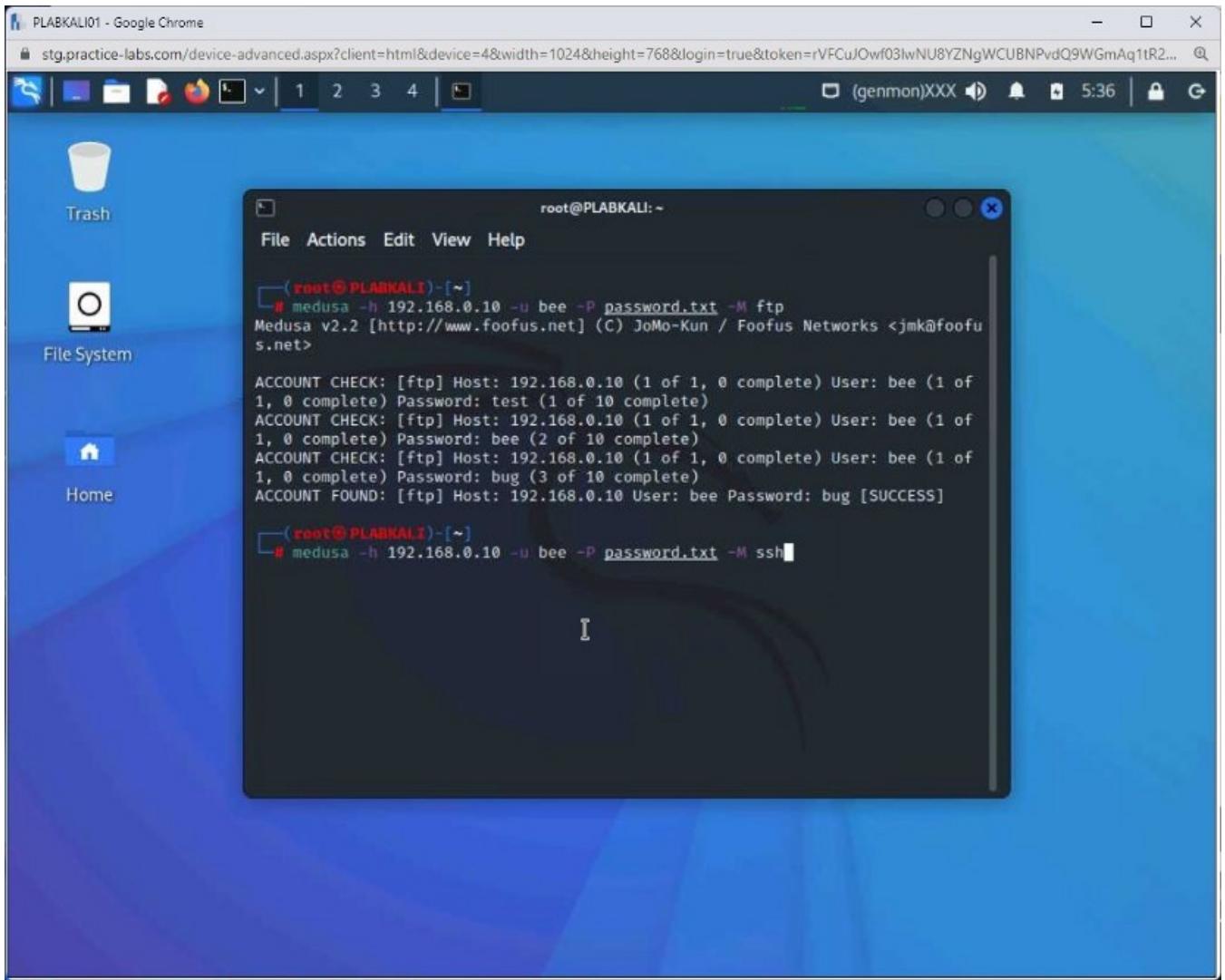


Step 9

Let's attempt to crack the password for the **SSH** module. Type the following command:

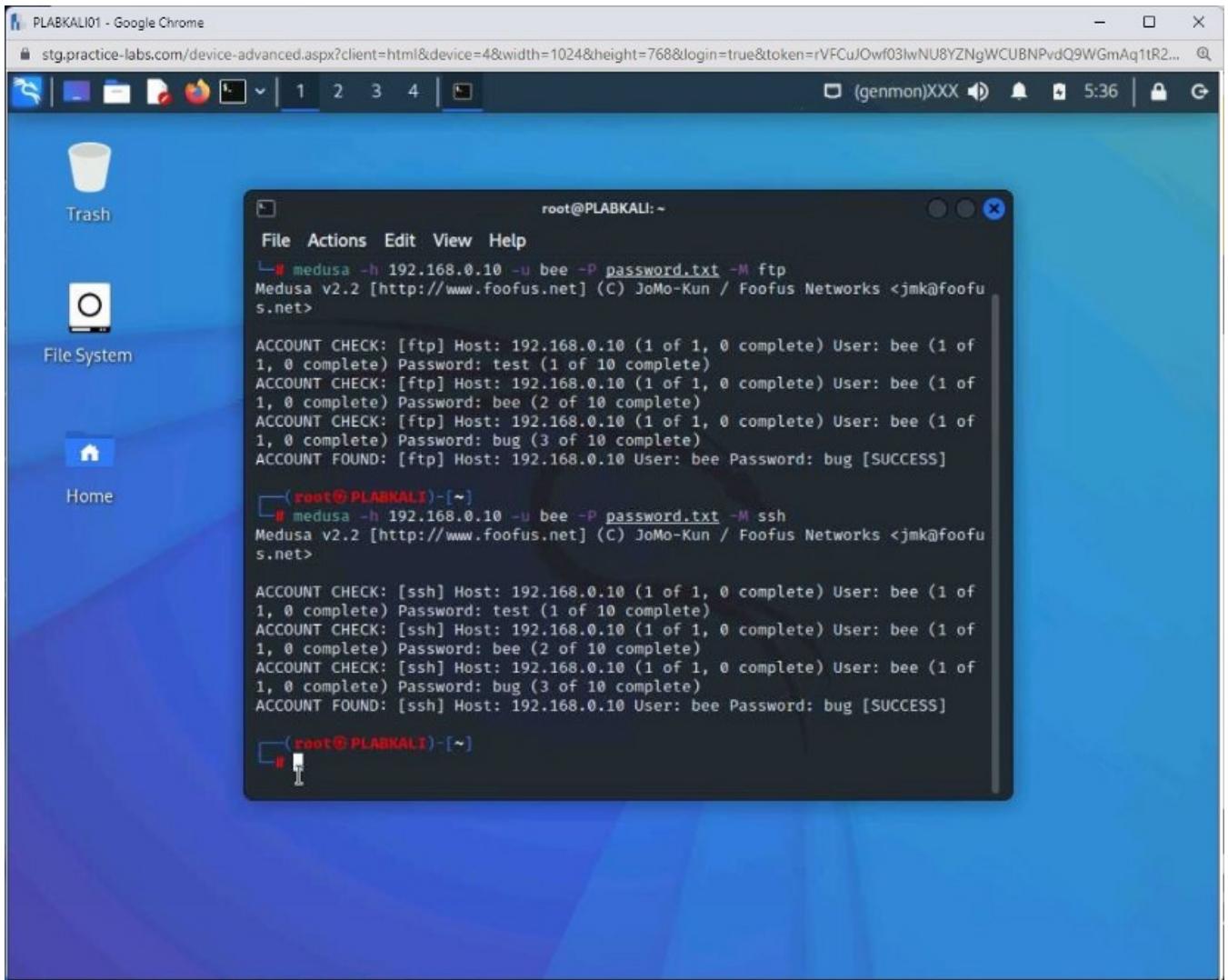
```
medusa -h 192.168.0.10 -u bee -P password.txt -M ssh
```

Press **Enter**.



Step 10

Like the FTP module, the password for SSH is also cracked for the username **bee**.



Step 11

Clear the screen by entering the following command:

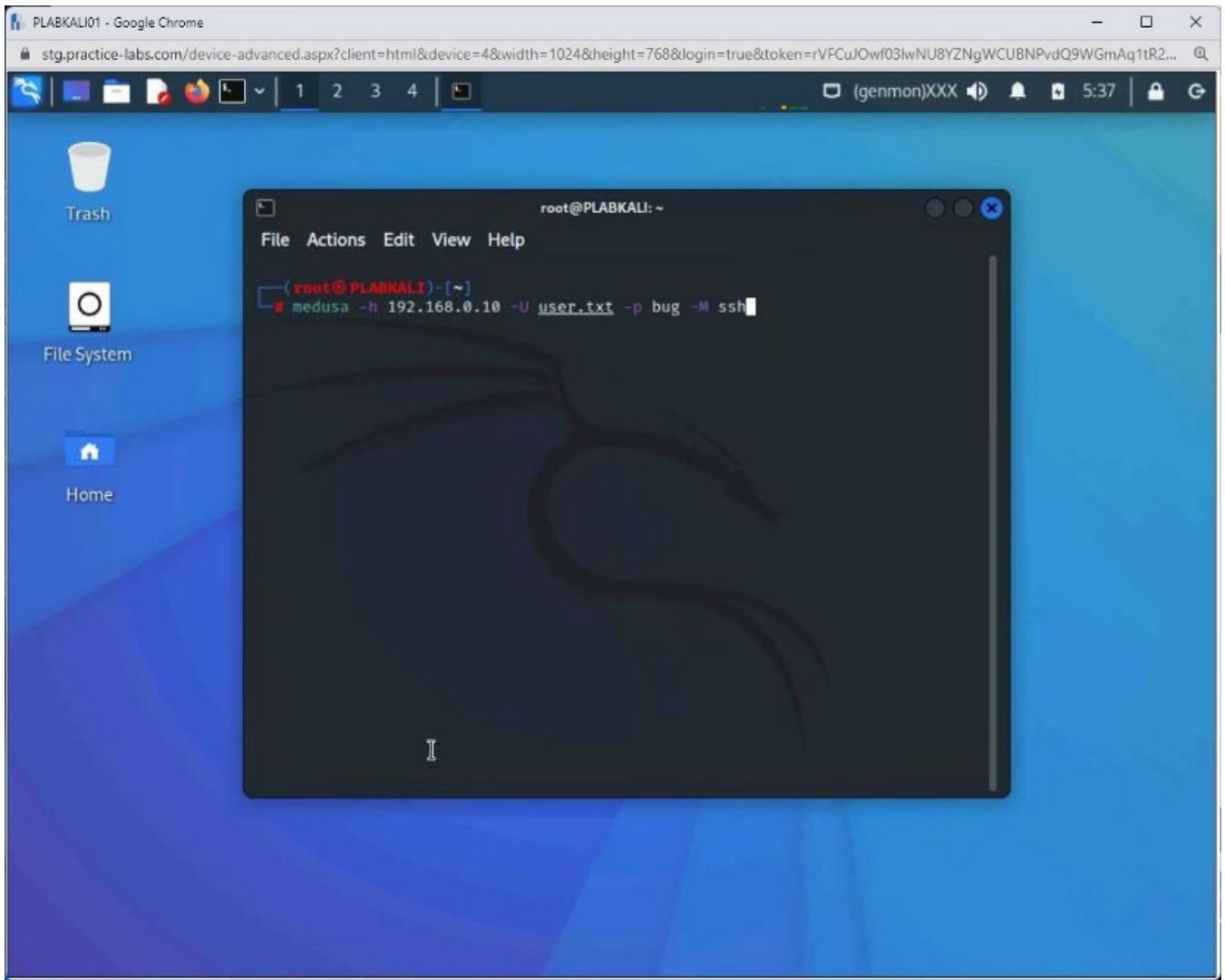
```
clear
```

Let's find the username by providing the password as an input.

Type the following command:

```
medusa -h 192.168.0.10 -U user.txt -p bug -M ssh
```

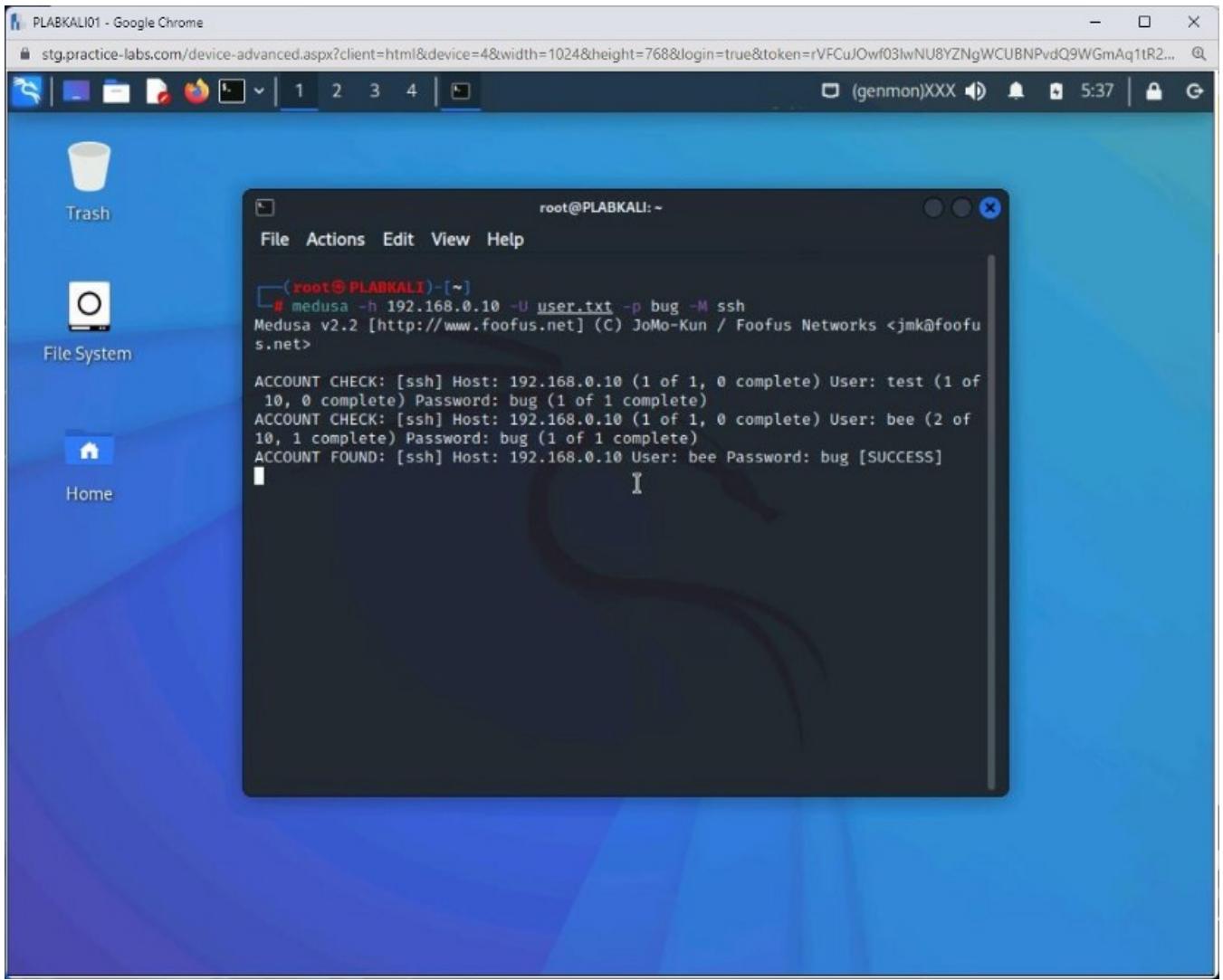
Press **Enter**.



Step 12

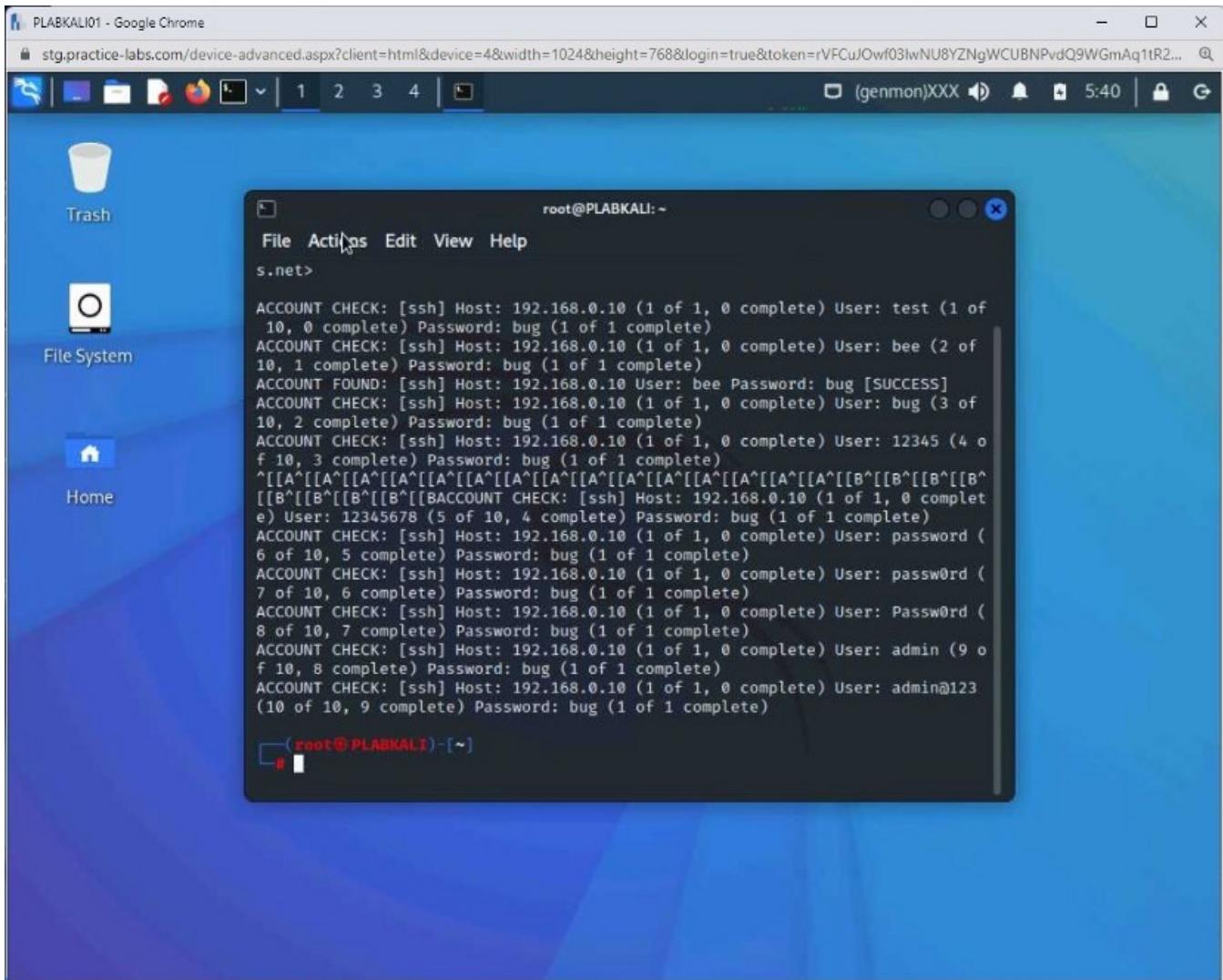
The username cracking process starts. Notice the statement with **[SUCCESS]**, meaning it has found one username.

However, unlike password cracking, the username cracking process does not stop and attempts to find more usernames from the given wordlist. It will run through all the usernames given in the wordlist.



Step 13

The username cracking process stops after running through the usernames in the wordlist.



Step 14

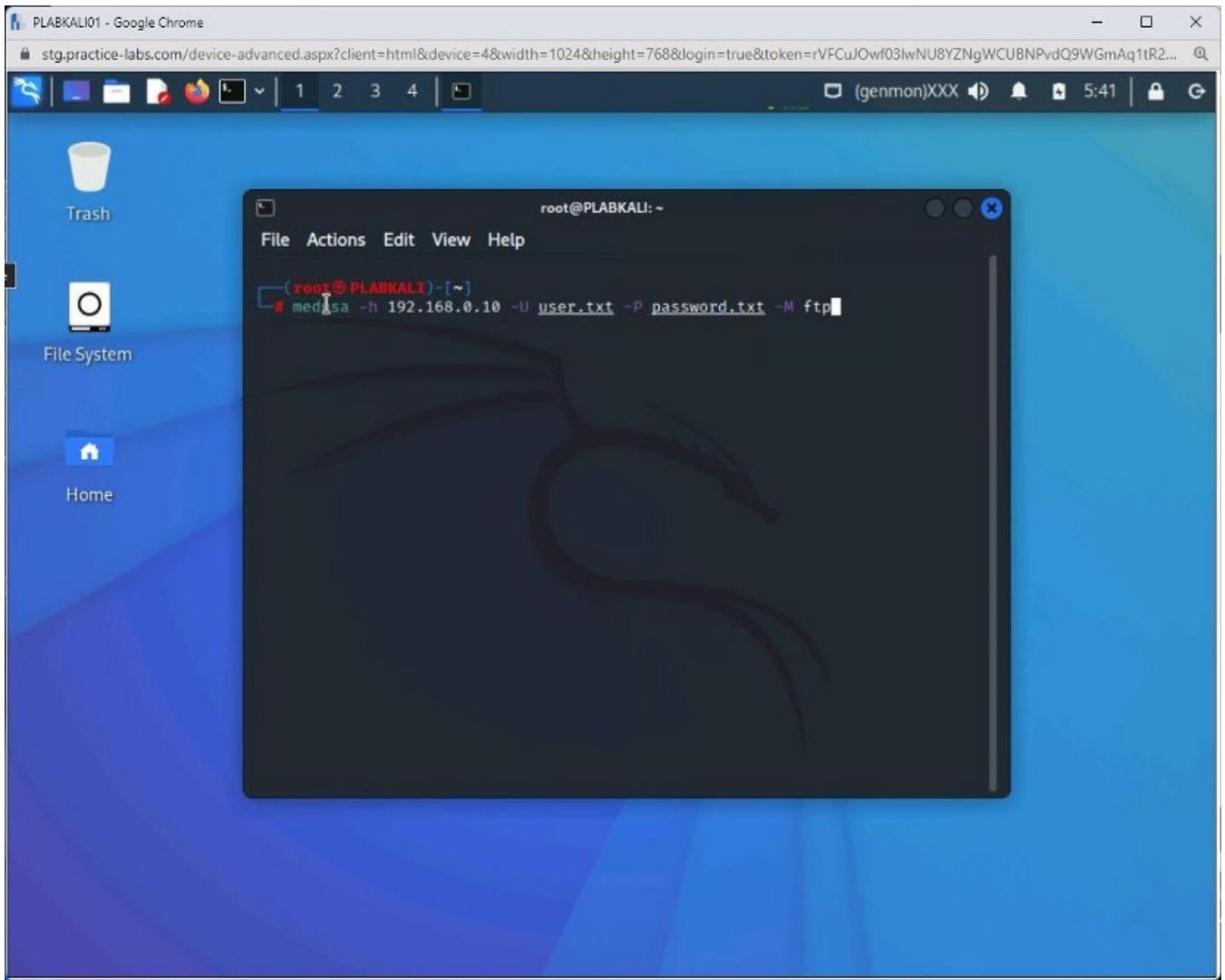
Clear the screen by entering the following command:

```
clear
```

Let's now attempt to use the username and password wordlists to find the username and the password. To do this, type the following command:

```
medusa -h 192.168.0.10 -U user.txt -P password.txt -M ftp
```

Press **Enter**.



Step 15

The username and password cracking process start. Each username will be run against each password in the **password.txt**.

Note: The username and password cracking process will take a while to complete.

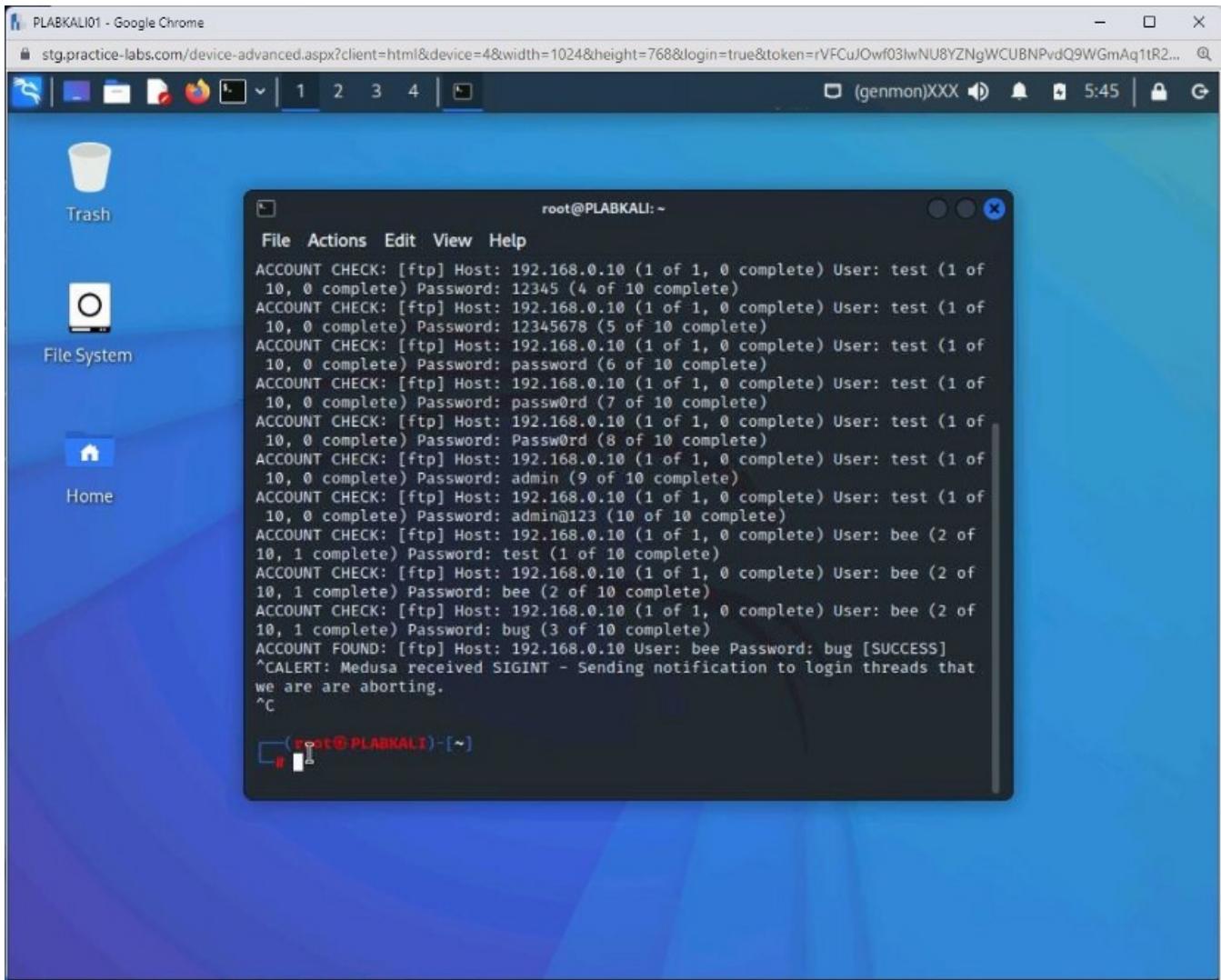
Finally, it can match the username with the correct password.

```
root@PLABKALI:~  
File Actions Edit View Help  
10, 0 complete) Password: test (1 of 10 complete)  
ACCOUNT CHECK: [ftp] Host: 192.168.0.10 (1 of 1, 0 complete) User: test (1 of  
10, 0 complete) Password: bee (2 of 10 complete)  
ACCOUNT CHECK: [ftp] Host: 192.168.0.10 (1 of 1, 0 complete) User: test (1 of  
10, 0 complete) Password: bug (3 of 10 complete)  
ACCOUNT CHECK: [ftp] Host: 192.168.0.10 (1 of 1, 0 complete) User: test (1 of  
10, 0 complete) Password: 12345 (4 of 10 complete)  
ACCOUNT CHECK: [ftp] Host: 192.168.0.10 (1 of 1, 0 complete) User: test (1 of  
10, 0 complete) Password: 12345678 (5 of 10 complete)  
ACCOUNT CHECK: [ftp] Host: 192.168.0.10 (1 of 1, 0 complete) User: test (1 of  
10, 0 complete) Password: password (6 of 10 complete)  
ACCOUNT CHECK: [ftp] Host: 192.168.0.10 (1 of 1, 0 complete) User: test (1 of  
10, 0 complete) Password: passw0rd (7 of 10 complete)  
ACCOUNT CHECK: [ftp] Host: 192.168.0.10 (1 of 1, 0 complete) User: test (1 of  
10, 0 complete) Password: Passw0rd (8 of 10 complete)  
ACCOUNT CHECK: [ftp] Host: 192.168.0.10 (1 of 1, 0 complete) User: test (1 of  
10, 0 complete) Password: admin (9 of 10 complete)  
ACCOUNT CHECK: [ftp] Host: 192.168.0.10 (1 of 1, 0 complete) User: test (1 of  
10, 0 complete) Password: admin@123 (10 of 10 complete)  
ACCOUNT CHECK: [ftp] Host: 192.168.0.10 (1 of 1, 0 complete) User: bee (2 of  
10, 1 complete) Password: test (1 of 10 complete)  
ACCOUNT CHECK: [ftp] Host: 192.168.0.10 (1 of 1, 0 complete) User: bee (2 of  
10, 1 complete) Password: bee (2 of 10 complete)  
ACCOUNT CHECK: [ftp] Host: 192.168.0.10 (1 of 1, 0 complete) User: bee (2 of  
10, 1 complete) Password: bug (3 of 10 complete)  
ACCOUNT FOUND: [ftp] Host: 192.168.0.10 User: bee Password: bug [SUCCESS]
```

Step 16

You have found a correct combination of username and password (**bee** and **bug**), and therefore, it is safe to abort the process.

Press **Ctrl + C** to break the process.



Keep the terminal window open.

Task 3 — Perform Broken Authentication Attacks

Authentication is a process that confirms a user's identity using a username and password. The server or the web application validates the user's identity. In a web application scenario, the authentication process is as follows:

1. On a login form, a user enters their login credentials, typically via username and password.
2. After a user submits the credentials, they are then sent to the application where they are verified, and then a session is created and stored in a database.
3. A cookie is sent to the user's system.
4. When a user makes a subsequent request, the session ID is verified with the one stored in the database. The browser at the client end stores and sends the token as a

cookie to the server. If both the session ID values match, the user's request is processed. However, if the values do not match, the request is not processed further.

5. After performing certain tasks, the session is destroyed when a user logs out from the application, the client, and the server.

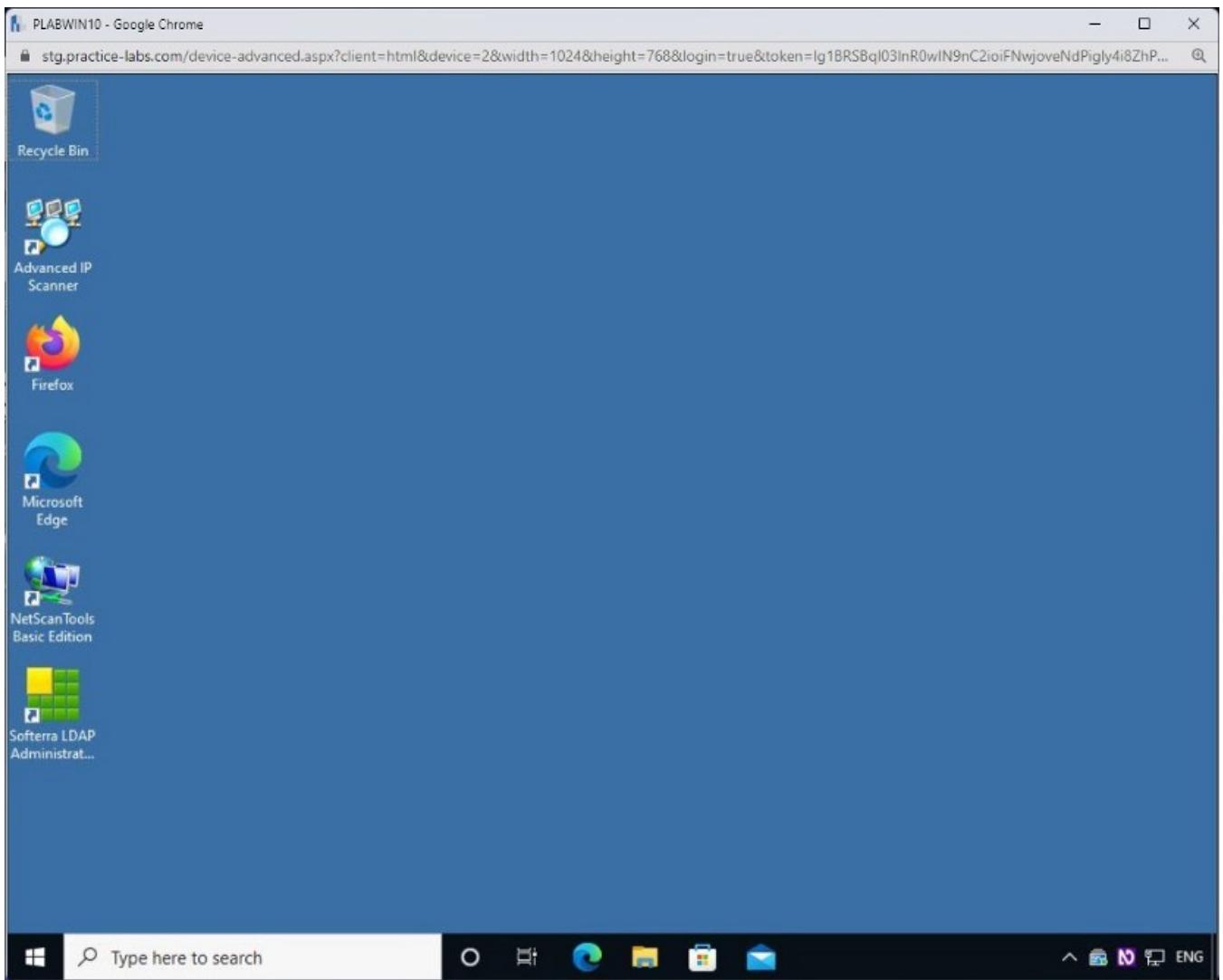
These methods are also known as Access Control attacks, in that they allow privilege escalation. Using the same method, an attacker can perform the following attacks:

- Parameter-based Access Control
- Referer-based Access Control
- Location-based Access Control

In this task, you will learn to perform broken authentication attacks. To do this, perform the following steps:

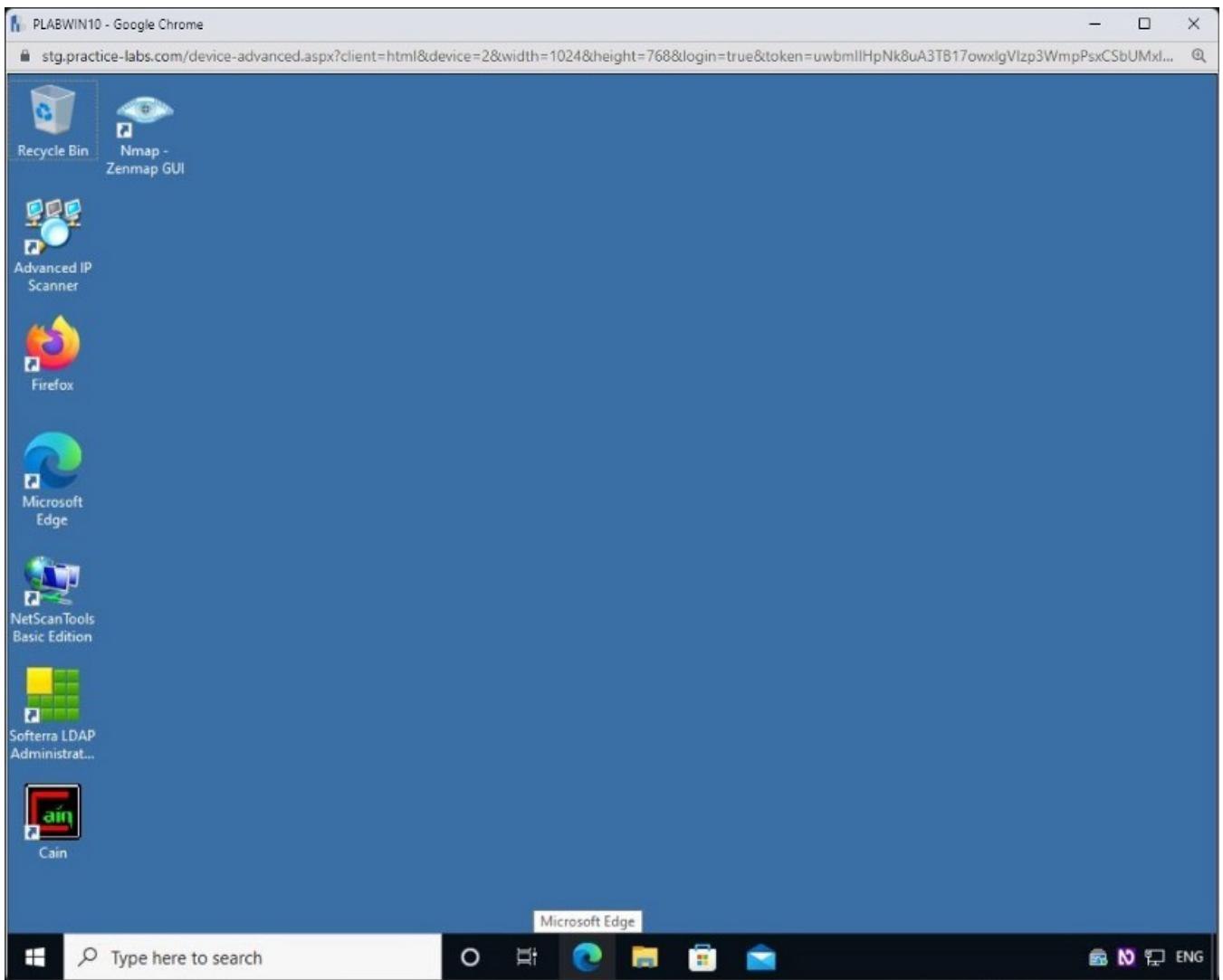
Step 1

Connect to **PLABWIN10**. The desktop is displayed.



Step 2

Open **Microsoft Edge** by clicking on the icon on the taskbar.



Step 3

The **Microsoft Edge** window is displayed. In the address bar, type the following URL:

<http://192.168.0.10/bWAPP>

Press **Enter**.

Note

We have updated this website to start offering more services. As part of this, the location of the files has changed slightly from most of the documentation.

For example, Tools and Resources > Installation_Files > Cisco is now simply Installation_Files > Cisco

Name	Created	Size
Data Files	09/04/2020	10
FTP	09/04/2020	1
Hotfix	09/04/2020	5
Installation_Files	09/04/2020	76
Tools	09/04/2020	59

Step 4

The login page of the **bWAPP** web application is displayed. In the **Login** text box, type the following username:

bee

In the **Password** text box, type the following password:

bug

Click Login.

PLABWIN10 - Google Chrome

stg.practice-labs.com/device-advanced.aspx?client=html&device=2&width=1024&height=768&login=true&token=rVFCuJOwnf03lwNU8YZNgWCUBNPvdQ9WGmAq1tR2...

bWAPP - Login

Not secure | 192.168.0.10/bWAPP/login.php

bWAPP an extremely buggy web app !

Login New User Info Talks & Training Blog

/ Login /

Enter your credentials (bee/bug).

Login:

Password:

Set the security level:

NATIONAL CENTER FOR
MISSING &
EXPLOITED
CHILDREN

Scan your website for XSS and SQL Injection vulnerabilities

bWAPP is licensed under © 2014 MME BVBA / Follow [@MME_IT](#) on Twitter and ask for our cheat sheet, containing all solutions! / Ne

Type here to search ENG

Step 5

A notification bar is displayed about remembering the password. Click **Not for this site.**

PLABWIN10 - Google Chrome

stg.practice-labs.com/device-advanced.aspx?client=html&device=2&width=1024&height=768&login=true&token=rVFCuJOwnf03lwNU8YZNgWCUBNPvdQ9WGmAq1tR2... ...

bWAPP - Portal X

Not secure | 192.168.0.10/bWAPP/portal.php ...

Choose your bug:
bWAPP v2.2 Hack

Set your security level:
low Set Current low

Bugs Change Password Create User Set Security Level Reset Credits Blog Logout

/ Portal /

bWAPP, or a buggy web application, is a free and open source deliberately insecure web application. It helps security enthusiasts, developers and students to discover and to prevent web vulnerabilities. bWAPP covers all major known web vulnerabilities, including all risks from the OWASP Top 10 project! It is for security-testing and educational purposes only.

Which bug do you want to hack today? :)

bWAPP v2.2

- / A1 - Injection /
- HTML Injection - Reflected (GET)
- HTML Injection - Reflected (POST)
- HTML Injection - Reflected (Current URL)
- HTML Injection - Stored (Blog)

bWAPP is licensed under © 2014 MME BVBA / Follow [@MME_IT](#) on Twitter and ask for our cheat sheet, containing all solutions! / Ne

Type here to search Windows Start button File Explorer Edge File Explorer Mail Cloud Network ENG

Step 6

From the **Choose your bug:** drop-down, select **Broken Authentication – Insecure Login Forms**, and click **Hack**.

PLABWIN10 - Google Chrome

stg.practice-labs.com/device-advanced.aspx?client=html&device=2&width=1024&height=768&login=true&token=rVFCuJOwnf03lwNU8YZNgWCUBNPvdQ9WGmAq1tR2...

bWAPP - Portal

Not secure | 192.168.0.10/bWAPP/portal.php

Choose your bug:
Broken Authentication - Insecure Login Forms Hack

Set your security level:
low Set Current low

bWAPP an extremely buggy web app!

Bugs Change Password Create User Set Security Level Reset Credits Blog Logout

/ Portal /

bWAPP, or a buggy web application, is a free and open source deliberately insecure web application. It helps security enthusiasts, developers and students to discover and to prevent web vulnerabilities. bWAPP covers all major known web vulnerabilities, including all risks from the OWASP Top 10 project! It is for security-testing and educational purposes only.

Which bug do you want to hack today? :)

bWAPP v2.2

/A1 - Injection /

- HTML Injection - Reflected (GET)
- HTML Injection - Reflected (POST)
- HTML Injection - Reflected (Current URL)
- HTML Injection - Stored (Blog)

NATIONAL CENTER FOR MISSING & EXPLOITED CHILDREN

bWAPP is licensed under © 2014 MME BVBA / Follow [@MME_IT](#) on Twitter and ask for our cheat sheet, containing all solutions! / Ne

Type here to search ENG

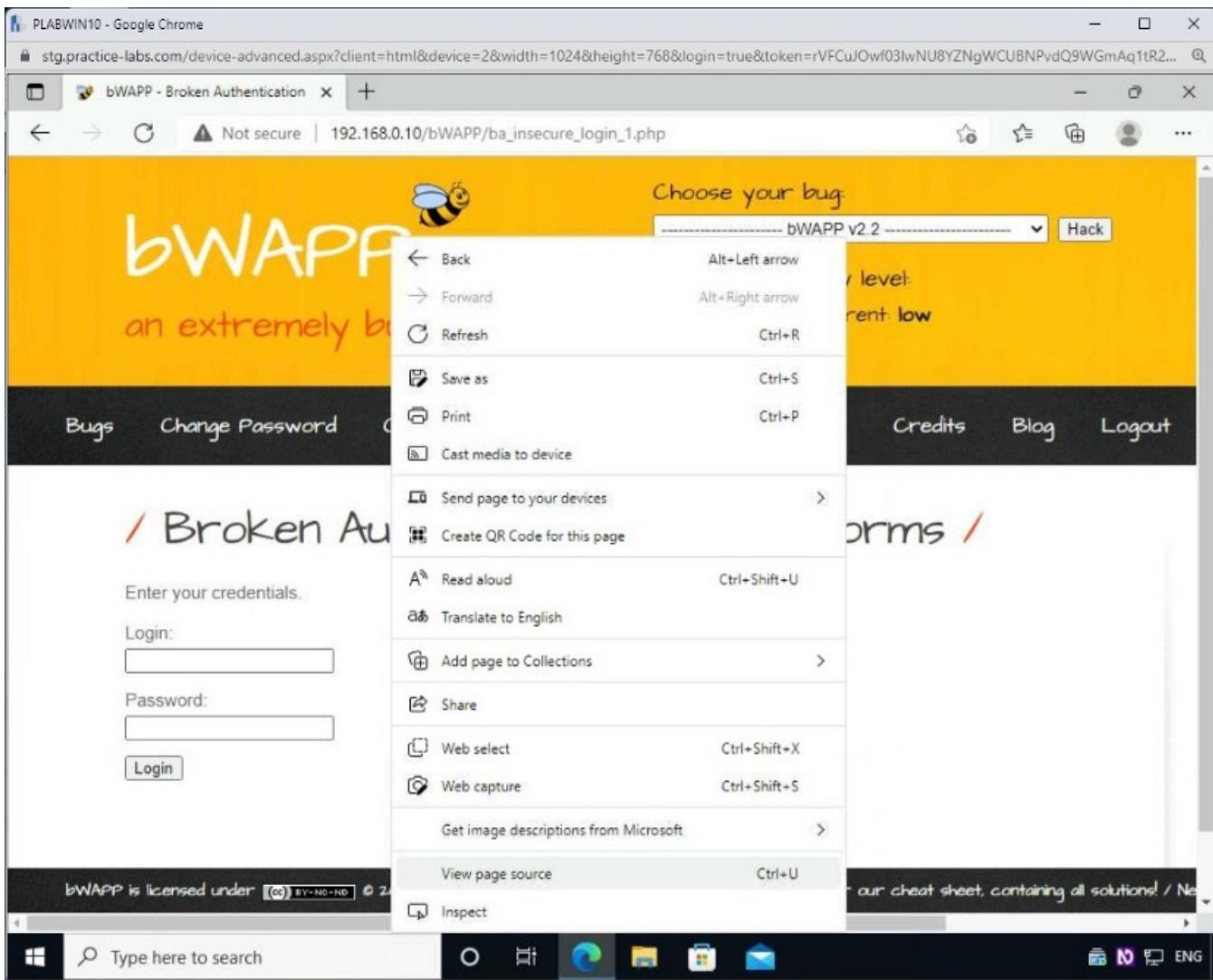
Step 7

The **login** page is displayed.

The screenshot shows a web browser window for Google Chrome on a Windows 10 desktop. The title bar reads "PLABWIN10 - Google Chrome". The address bar shows the URL "stg.practice-labs.com/device-advanced.aspx?client=html&device=2&width=1024&height=768&login=true&token=rVFCuJOwnf03lwNU8YZNgWCUBNPvdQ9WGmAq1tR2...". The main content area displays the bWAPP homepage with a yellow header featuring the bWAPP logo and a bee icon. The header also includes a dropdown menu set to "bWAPP v2.2" and a "Hack" button. Below the header, there's a section to "Set your security level" with a dropdown set to "low" and a "Set" button. A navigation bar at the bottom has links for "Bugs", "Change Password", "Create User", "Set Security Level", "Reset", "Credits", "Blog", and "Logout". The main content area has a heading "/ Broken Auth. - Insecure Login Forms /" and a sub-section titled "Enter your credentials." It contains fields for "Login:" and "Password:", both represented by empty input boxes. A "Login" button is located below the password field. At the bottom of the page, a footer bar includes a license notice about bWAPP being licensed under CC BY-NC-ND, a copyright notice for 2014 MME BVBA, and a Twitter handle @MME_IT. The Windows taskbar at the bottom shows the Start button, a search bar with "Type here to search", and icons for File Explorer, Edge, Task View, and Mail. The system tray shows battery status, network connectivity, and language settings (ENG).

Step 8

Several times, the developers hard-code the login information in the login form. It is quite easy to extract login information. To do this, right-click anywhere on the login form and select **View source**.



Step 9

A **Debugger** window opens in the bottom section of the webpage and displays the source code for the login page. Review the code from lines **59** to **63**.

Notice the username and password. It displays the login name as **tonystark** and password as **I am Iron Man**.

PLABWIN10 - Google Chrome

stg.practice-labs.com/device-advanced.aspx?client=html&device=2&width=1024&height=768&login=true&token=rVFCuJOwnf03lwNU8YZNgWCUBNPvdQ9WGmAq1tR2...

bWAPP - Broken Authentication | view-source:192.168.0.10/bWAPP | view-source:192.168.0.10/bWAPP/ba_insecure_login_1.php

Not secure

```
51 <div id="main">
52     <h1>Broken Auth. - Insecure Login Forms</h1>
53     <p>Enter your credentials.</p>
54     <form action="/bWAPP/ba_insecure_login_1.php" method="POST">
55         <p><label for="login">Login:</label><font color="white">tonystark</font><br />
56             <input type="text" id="login" name="login" size="20" /></p>
57
58         <p><label for="password">Password:</label><font color="white">I am Iron Man</font><br />
59             <input type="password" id="password" name="password" size="20" /></p>
60
61         <button type="submit" name="form" value="submit">Login</button>
62     </form>
63     <br />
64 </div>
65
66 <div id="side">
67     <a href="http://twitter.com/MME_IT" target="blank_" class="button"></a>
68     <a href="http://be.linkedin.com/in/malikmesellem" target="blank_" class="button"></a>
69     <a href="http://www.facebook.com/pages/MME-IT-Audits-Security/104153019664877" target="blank_" class="button">
70     <a href="http://itsecgames.blogspot.com" target="blank_" class="button"></a>
71 </div>
72
73 <div id="disclaimer">
74     <p>bWAPP is licensed under <a rel="license" href="http://creativecommons.org/licenses/by-nc-nd/4.0/" target="_blank"><img style="ver
75     </div>
76
77 <div id="bee">
78     
79 </div>
```

Step 10

Close the **view-source** tab.

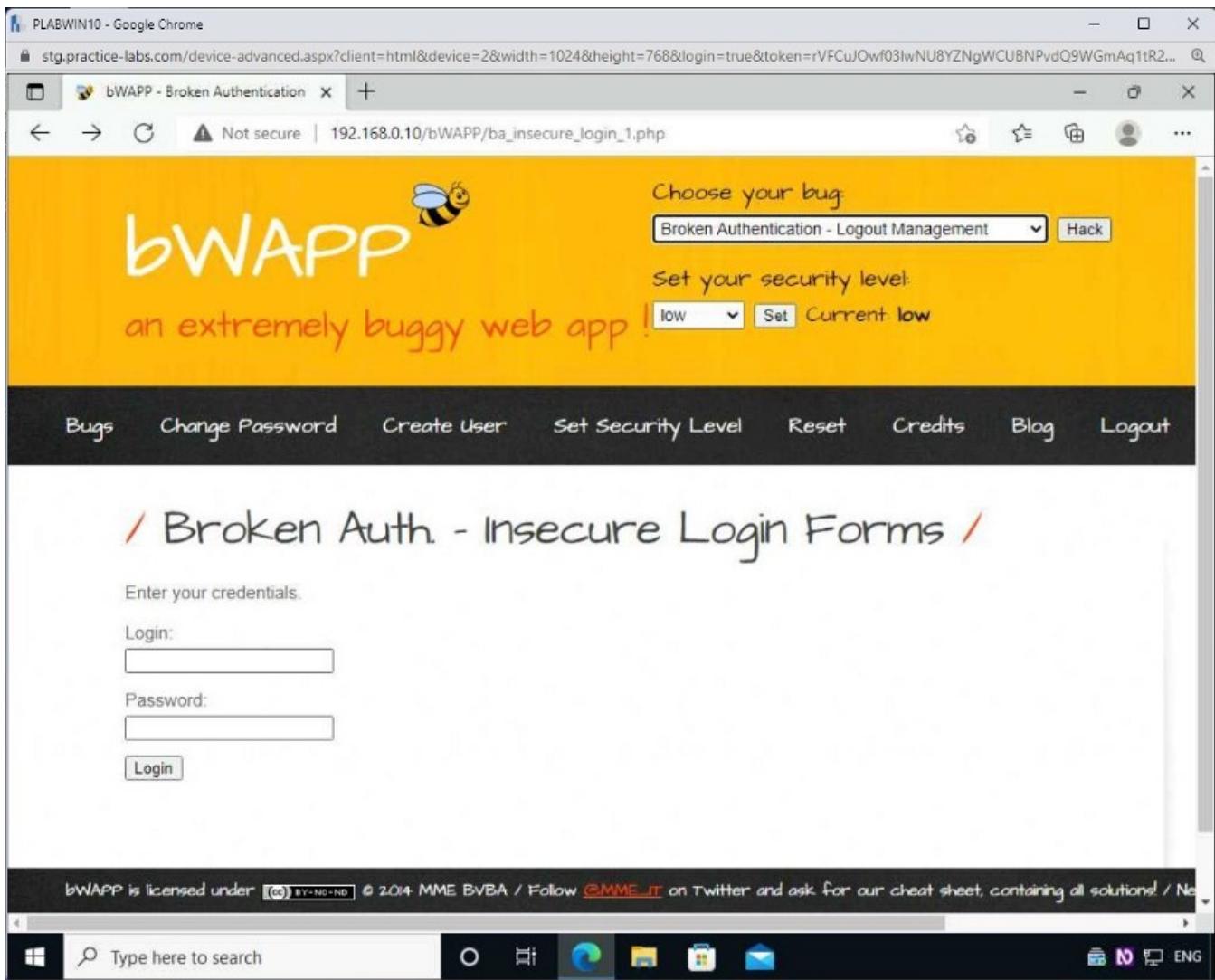
```
51 <div id="main">
52   <h1>Broken Auth. - Insecure Login Forms</h1>
53   <p>Enter your credentials.</p>
54   <form action="/bWAPP/ba_insecure_login_1.php" method="POST">
55     <p><label for="login">Login:</label><font color="white">tonystark</font><br />
56       <input type="text" id="login" name="login" size="20" /></p>
57
58     <p><label for="password">Password:</label><font color="white">I am Iron Man</font><br />
59       <input type="password" id="password" name="password" size="20" /></p>
60
61     <button type="submit" name="form" value="submit">Login</button>
62   </form>
63   <br />
64 </div>
65 <div id="side">
66   <a href="http://twitter.com/MIE_IT" target="blank_" class="button"></a>
67   <a href="http://be.linkedin.com/in/malikmesellem" target="blank_" class="button"></a>
68   <a href="http://www.facebook.com/pages/MIE-IT-Audits-Security/104153019664877" target="blank_" class="button">
69   <a href="http://itsecgames.blogspot.com" target="blank_" class="button"></a>
70 </div>
71 <div id="disclaimer">
72   <p>bWAPP is licensed under <a rel="license" href="http://creativecommons.org/licenses/by-nc-nd/4.0/" target="_blank"><img style="ver
73   </div>
74 <div id="bee">
75   
76 </div>
```

Step 11

Logout management is one of the common vulnerabilities in applications. Even if the user logs out, the user session is still live. One of the common methods is to click the Back button on the browser to get back to the same session after logging out.

Let's perform a logout management attack on this application.

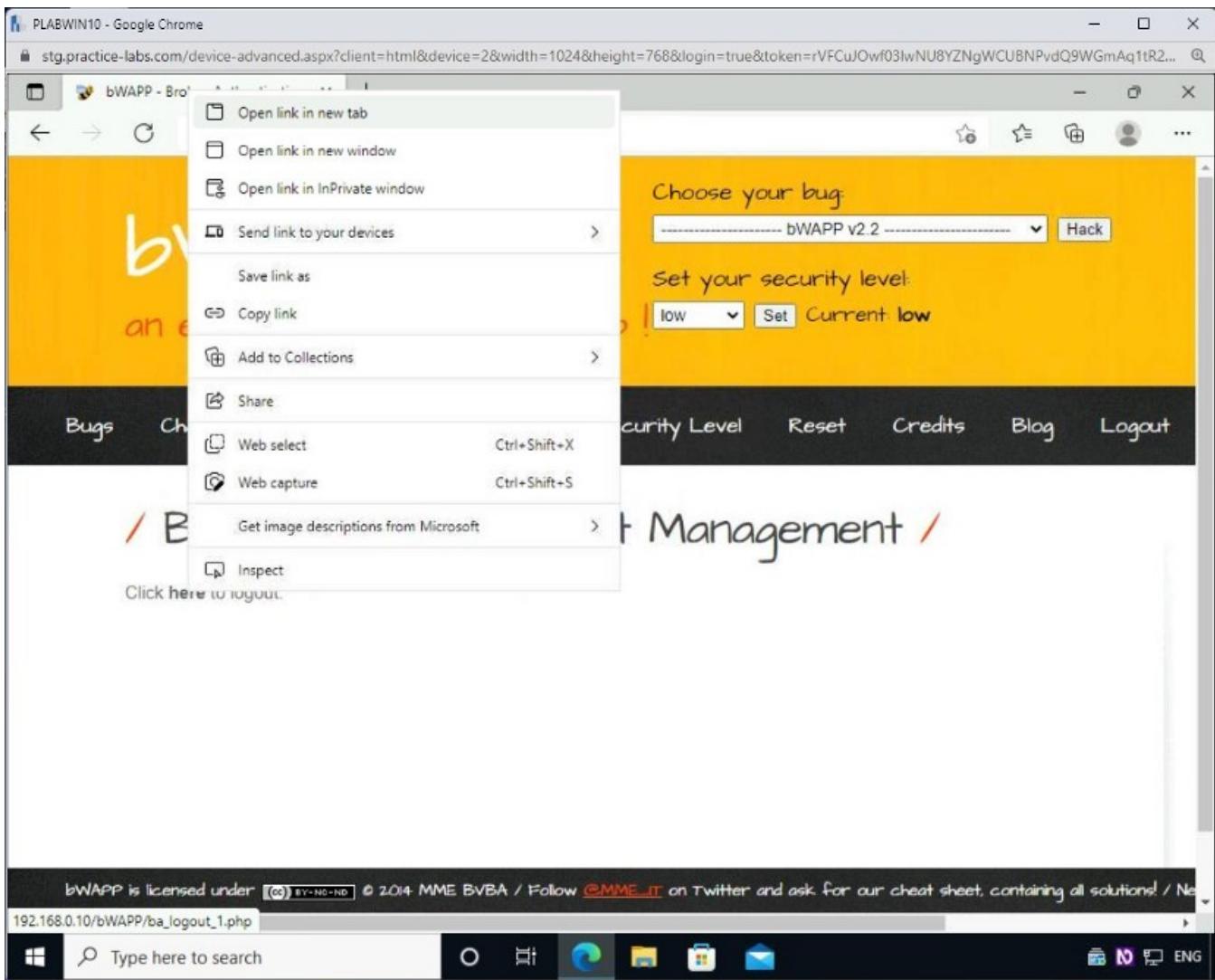
Back on the **bWAPP-Broken Authentication** tab, From the **Choose your bug** drop-down, select **Broken Authentication — Logout Management**, and click **Hack**.



Step 12

The **Broken Auth – Logout Management** webpage is loaded.

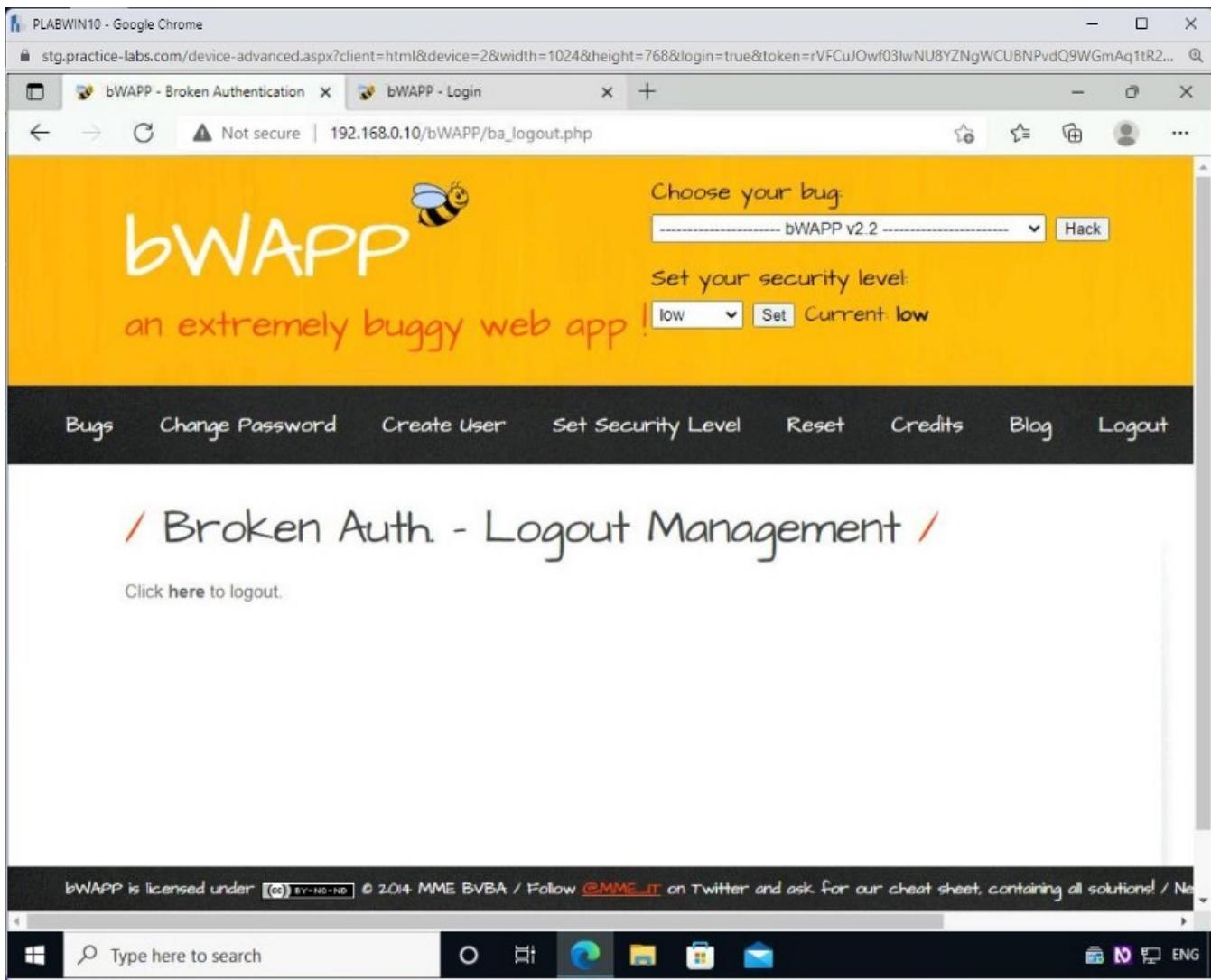
Right-click the **here** in the **Click here to logout** statement and select **Open in new tab**.



Step 13

The **bWAPP — Login** tab opens.

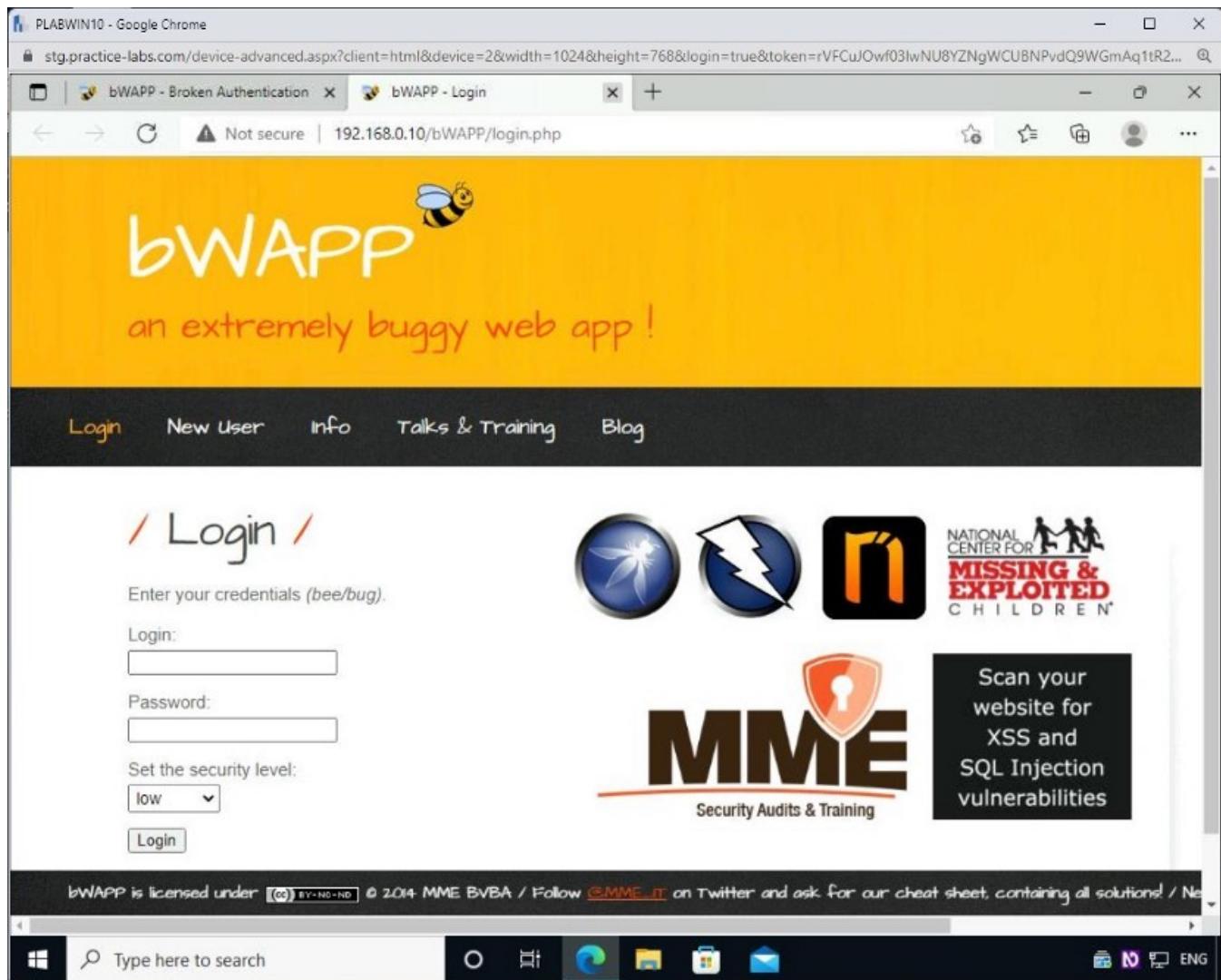
Click the **bWAPP — Login** tab.



Step 14

Notice that you are logged out successfully.

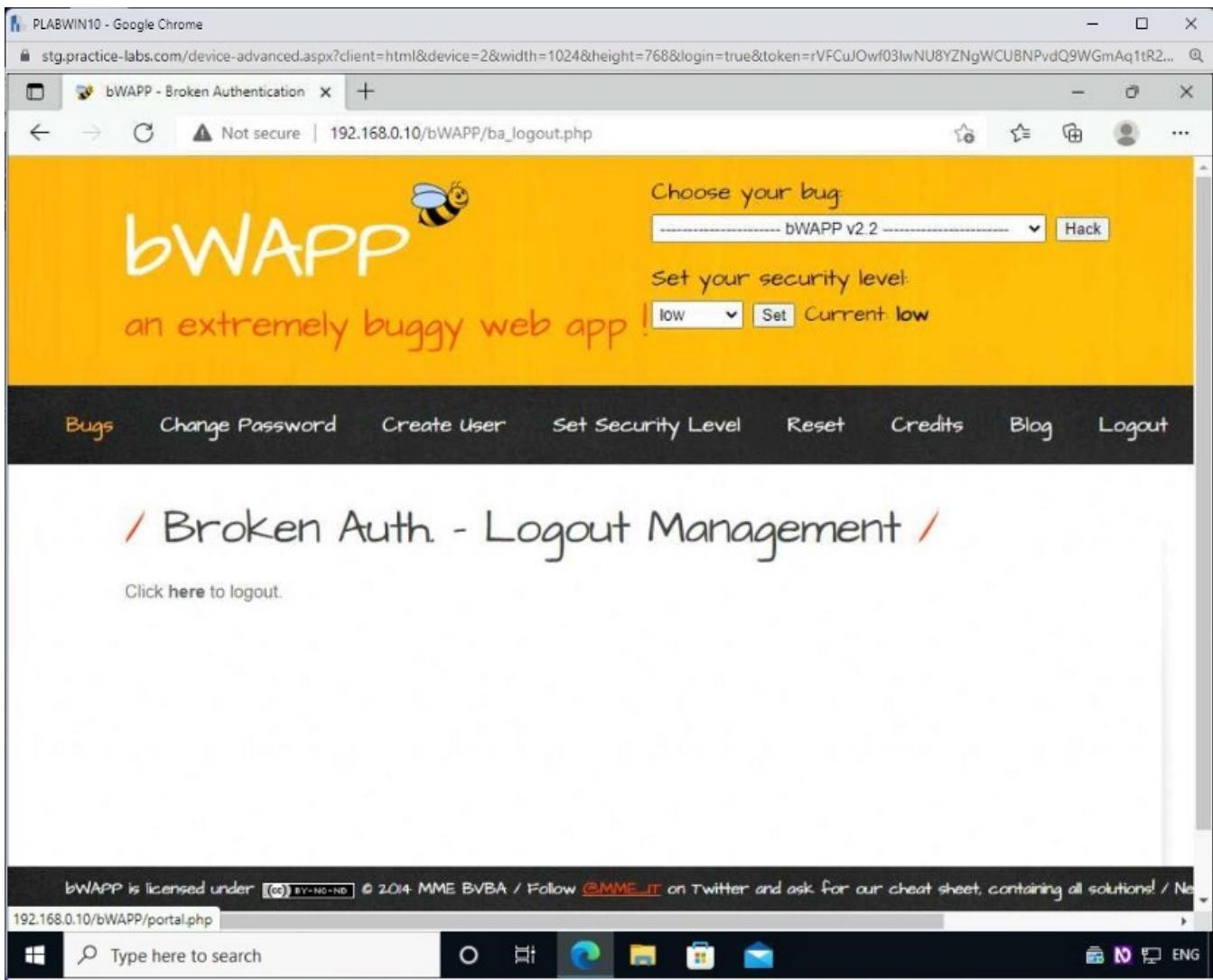
Close this tab.



Step 15

Notice that you are back on the first original tab. You are still logged in to this tab.

Click Bugs.



Step 16

The **Portal** webpage is loaded successfully.

This means that you were still able to work within the web application.



Keep the Microsoft Edge window open.

Task 4 — Conduct OS Command Injection Attack

Command injection attacks occur because input fields on an application accept arbitrary data without sufficient input validation. An attacker can use various commands, both for Linux and Windows, to discover required information. For example, an attacker may use the **net user** command to know the existing users in an operating system.

In this task, you will learn to conduct an OS command injection attack. To do this, perform the following step:

Step 1

Connect to **PLABWIN10**. The **Microsoft Edge** window should be open on the beep portal.

From the **Choose your bug** drop-down, select **OS Command Injection**, and click **Hack**.

The screenshot shows a Google Chrome window with the URL <http://192.168.0.10/bWAPP/portal.php>. The title bar says "bWAPP - Portal". The page has a yellow header with the bWAPP logo and a bee icon. It says "an extremely buggy web app!". On the right, there's a sidebar with "Choose your bug:" dropdown set to "OS Command Injection", a "Hack" button, "Set your security level:" dropdown set to "low", and a "Set Current low" button. Below the header is a black navigation bar with links: Bugs, Change Password, Create User, Set Security Level, Reset, Credits, Blog, Logout. The main content area has a red header "Portal /". Below it, a text block says: "bWAPP, or a buggy web application, is a free and open source deliberately insecure web application. It helps security enthusiasts, developers and students to discover and to prevent web vulnerabilities. bWAPP covers all major known web vulnerabilities, including all risks from the OWASP Top 10 project! It is for security-testing and educational purposes only." A sub-header "Which bug do you want to hack today? :" is followed by a dropdown menu titled "bWAPP v2.2" containing: "/A1 - Injection /", "HTML Injection - Reflected (GET)", "HTML Injection - Reflected (POST)", "HTML Injection - Reflected (Current URL)", and "HTML Injection - Stored (Blog)". To the right of the dropdown are four icons: a blue circle with a white dragonfly, a blue circle with a white lightning bolt, an orange square with a white stylized letter 'n', and the "MISSING & EXPLOITED CHILDREN" logo. At the bottom of the page is a footer with "bWAPP is licensed under CC BY-NC-ND © 2014 MME BVBA / Follow @MME_IT on Twitter and ask for our cheat sheet, containing all solutions! / Ne". The Windows taskbar at the bottom shows the search bar with "Type here to search" and various pinned icons.

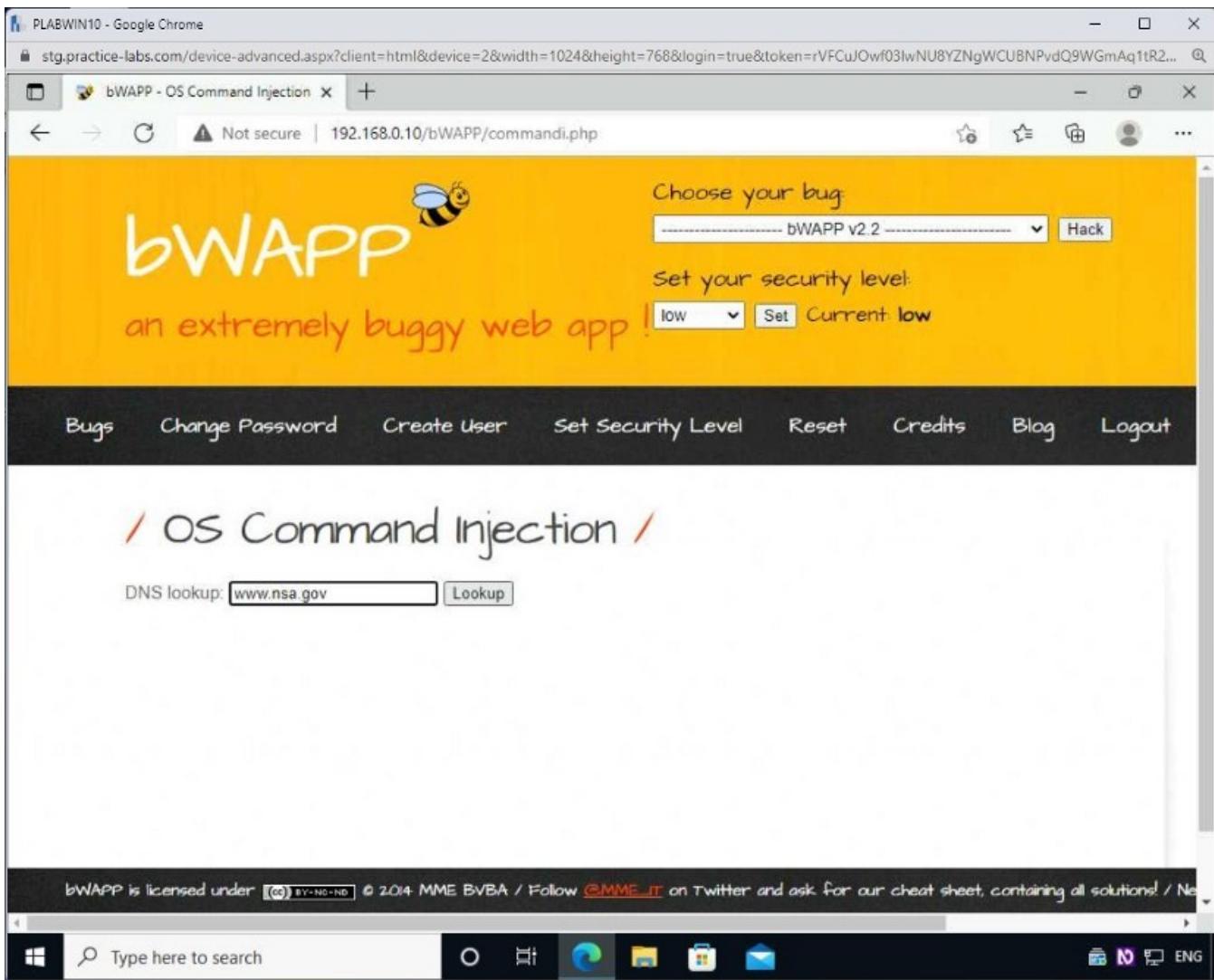
Step 2

The **OS Command Injection** webpage is displayed. Notice that in the DNS Lookup text box, the following text is entered by default:

www.nsa.gov

Click **Lookup**.

Note: this may take up to a minute to execute.



Step 3

Notice the output. The firewall blocks the DNS query from going out of the lab environment, and therefore, you get a connection timed out error.

PLABWIN10 - Google Chrome

stg.practice-labs.com/device-advanced.aspx?client=html&device=2&width=1024&height=768&login=true&token=rVFCuJOwnf03lwNU8YZNgWCUBNPvdQ9WGmAq1tR2... ...

bWAPP - OS Command Injection Not secure | 192.168.0.10/bWAPP/commcmdi.php

Choose your bug: bWAPP v2.2 Hack

Set your security level: low Set Current low

Bugs Change Password Create User Set Security Level Reset Credits Blog Logout

/ OS Command Injection /

DNS lookup: www.nsa.gov Lookup

:: connection timed out; no servers could be reached

bWAPP is licensed under (cc) BY-NC-ND © 2014 MME BVBA / Follow [@MME_IT](#) on Twitter and ask for our cheat sheet, containing all solutions! / Ne

Type here to search ○ Hi ENGLISH

Step 4

Replace the existing text in the **DNS lookup** text box and type the following:

| hostname

Click Lookup.

The screenshot shows a web browser window titled "PLABWIN10 - Google Chrome" displaying the "bWAPP - OS Command Injection" page at the URL "stg.practice-labs.com/device-advanced.aspx?client=html&device=2&width=1024&height=768&login=true&token=rVFCuJOwnf03lwNU8YZNgWCUBNPvdQ9WGmAq1tR2...". The page has a yellow header with the bWAPP logo and a bee icon. It says "Choose your bug: bWAPP v2.2" and "Hack". Below that, it says "Set your security level: low Set Current low". A navigation bar at the top includes links for "Bugs", "Change Password", "Create User", "Set Security Level", "Reset", "Credits", "Blog", and "Logout". The main content area is titled "/ OS Command Injection /". It contains a "DNS lookup" input field with "hostname" and a "Lookup" button. Below the input field, the text ";; connection timed out; no servers could be reached" is displayed. At the bottom of the page, there is a footer with the text "bWAPP is licensed under CC BY-NC-ND © 2014 MME BVBA / Follow @MME_IT on Twitter and ask for our cheat sheet, containing all solutions! / Ne" and a Windows taskbar with a search bar, file explorer, and other icons.

Step 5

The output is now displayed.

In the notification bar regarding **AutoComplete** to remember web entries, click **No**.

PLABWIN10 - Google Chrome

stg.practice-labs.com/device-advanced.aspx?client=html&device=2&width=1024&height=768&login=true&token=rVFCuJOwnf03lwNU8YZNgWCUBNPvdQ9WGmAq1tR2... ...

bWAPP - OS Command Injection Not secure | 192.168.0.10/bWAPP/commcmdi.php

Choose your bug:
bWAPP v2.2 Hack

Set your security level:
low Set Current low

Bugs Change Password Create User Set Security Level Reset Credits Blog Logout

/ OS Command Injection /

DNS lookup: Lookup

bee-box

bWAPP is licensed under [\(cc\) BY-NC-ND](#) © 2014 MME BVBA / Follow [@MME_IT](#) on Twitter and ask for our cheat sheet, containing all solutions! / Ne

Type here to search ○ Hi ENG

Step 6

Replace the existing text in the **DNS lookup** text box and type the following:

```
| net user
```

Click Lookup.

The screenshot shows a web browser window for 'PLABWIN10 - Google Chrome' displaying the 'bWAPP - OS Command Injection' page at '192.168.0.10/bWAPP/commands.php'. The page has a yellow header with the 'bWAPP' logo and a bee icon. It says 'an extremely buggy web app!'. On the right, there's a dropdown menu 'Choose your bug:' set to 'bWAPP v2.2' with a 'Hack' button. Below it is a 'Set your security level:' dropdown set to 'low' with a 'Set' button and 'Current low' label. A navigation bar at the top includes links for 'Bugs', 'Change Password', 'Create User', 'Set Security Level', 'Reset', 'Credits', 'Blog', and 'Logout'. The main content area features a red banner with '/ OS Command Injection /'. Below it is a 'DNS lookup:' input field containing 'inet user' with a 'Lookup' button. The Windows taskbar at the bottom shows the search bar with 'Type here to search' and various pinned icons.

Step 7

The output is now displayed. There are two users, **nobody** and **bee**.



Keep the Microsoft Edge window open.

Task 5 — Perform Server-side Includes Injection Attack (SSI)

SSIs directives are used in web applications to provide dynamic content to HTML pages. SSIs execute a set of defined actions before a webpage is loaded. Therefore, the server which hosts the application, analyses the SSI before the HTML page is loaded in a user's web browser.

The SSI directives are injected in input fields and sent to the webserver. When the field input is submitted, the script, which was added to the input field, is executed. In an SSI attack, an attacker injects malicious scripts in HTML pages.

To perform an SSI attack, perform the following steps:

Step 1

Ensure you have powered on all the devices listed in the introduction and connect to **PLABWIN10**. The **Microsoft Edge** window should be open.

From the **Choose your bug** drop-down, select **Server-side Includes (SSI) Injection**, and click **Hack**.

The screenshot shows a web browser window titled 'PLABWIN10 - Google Chrome' displaying the 'bWAPP - OS Command Injection' page. The URL is 'stg.practice-labs.com/device-advanced.aspx?client=html&device=2&width=1024&height=768&login=true&token=rVFCuJOWf03lwNU8YZNgWCU8NPvdQ9WGmAq1tR2...'. The page has a yellow header with the 'bWAPP' logo and the tagline 'an extremely buggy web app!'. It features a 'Choose your bug:' dropdown set to 'Server-Side Includes (SSI) Injection' with a 'Hack' button next to it. Below that is a 'Set your security level:' dropdown set to 'low' with a 'Set Current low' button. A navigation bar at the bottom includes links for 'Bugs', 'Change Password', 'Create User', 'Set Security Level', 'Reset', 'Credits', 'Blog', and 'Logout'. The main content area is titled '/ OS Command Injection /' and contains a 'DNS lookup:' input field with 'www.nsa.gov' and a 'Lookup' button. Below the input field, the text 'nobody bee' is displayed. At the bottom of the page, there's a footer note: 'bWAPP is licensed under CC BY-NC-ND © 2014 MME BVBA / Follow @MME_IT on Twitter and ask for our cheat sheet, containing all solutions! / Ne'. The browser's taskbar at the bottom shows the Windows Start button, a search bar with 'Type here to search', and various pinned icons.

Step 2

The **Server-Side Includes (SSI) Injection** webpage is loaded. In the **First name** text box, type the following:

plab

In the **Last name** text box, type the following:

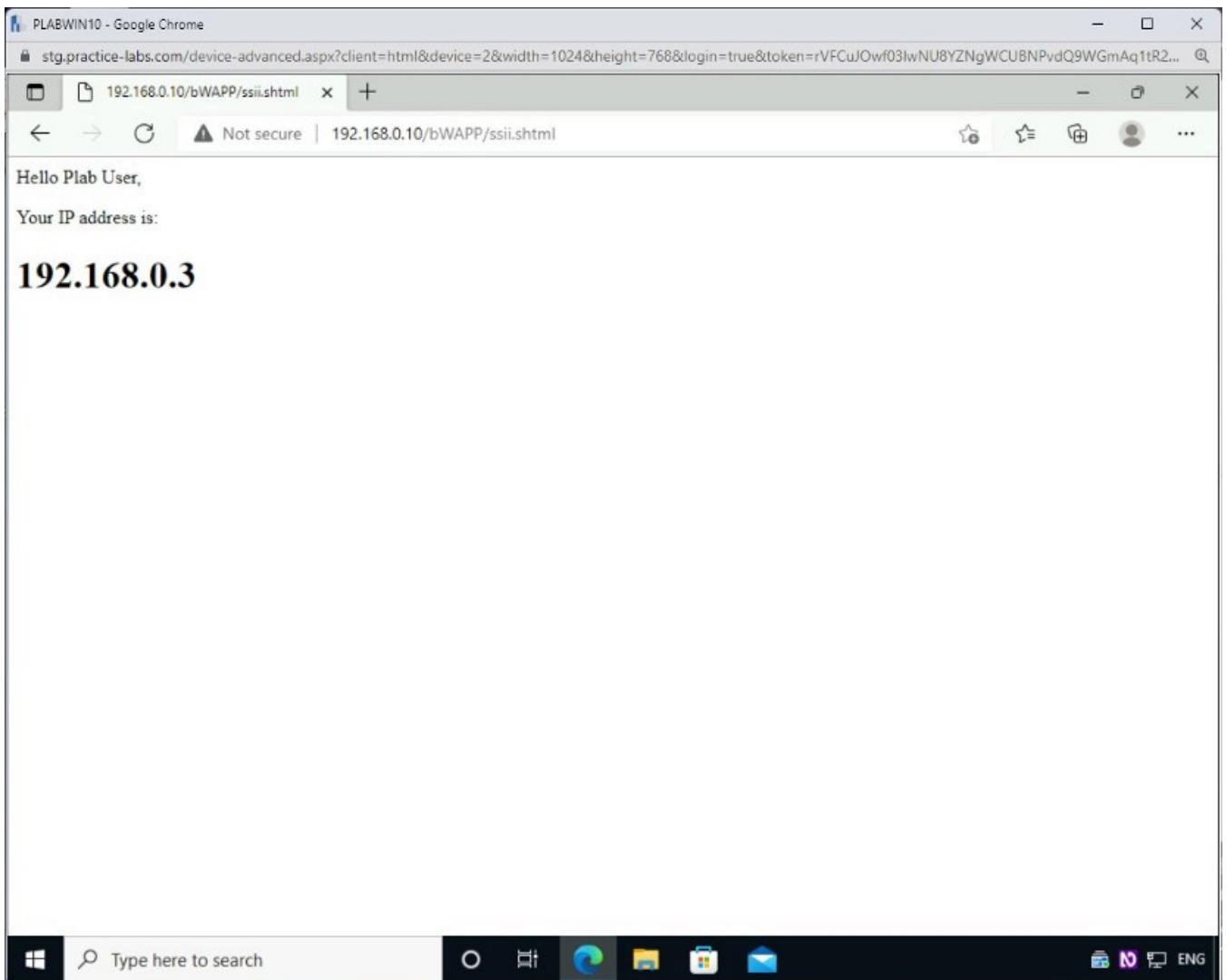
user

Click **Lookup**.

The screenshot shows a browser window titled "PLABWIN10 - Google Chrome" displaying the "bWAPP - SSI Injection" page at the URL "stg.practice-labs.com/device-advanced.aspx?client=html&device=2&width=1024&height=768&login=true&token=rVFCuJOwnf03lwNU8YZNgWCUBNPvdQ9WGmAq1tR2...". The page has a yellow header with the bWAPP logo and a bee icon. It says "Choose your bug: bWAPP v2.2" and "Hack". Below that, it says "Set your security level: low Set Current low". A navigation bar at the top includes links for "Bugs", "Change Password", "Create User", "Set Security Level", "Reset", "Credits", "Blog", and "Logout". The main content area features a red banner with the text "/ Server-Side Includes (SSI) Injection /". Below the banner, there's a form asking "What is your IP address? Lookup your IP address... (bee-box only)". The form fields are "First name: plab", "Last name: user", and a "Lookup" button. At the bottom of the page, there's a footer bar with the text "bWAPP is licensed under CC BY-NC-ND © 2014 MME BVBA / Follow @MME_IT on Twitter and ask for our cheat sheet, containing all solutions! / Ne". The bottom of the screen shows the Windows taskbar with the Start button, a search bar, and various pinned icons.

Step 3

Notice that the output is displayed with an IPv4 address.



Step 4

Click the **Back** arrow to navigate back to the previous page.

Back on the **Server-Side Includes (SSI) Injection** webpage, you need to insert the following into the **First name** text box:

```
<script>alert("Hacked")</script>
```

In the **Last name** text box, type the following:

user

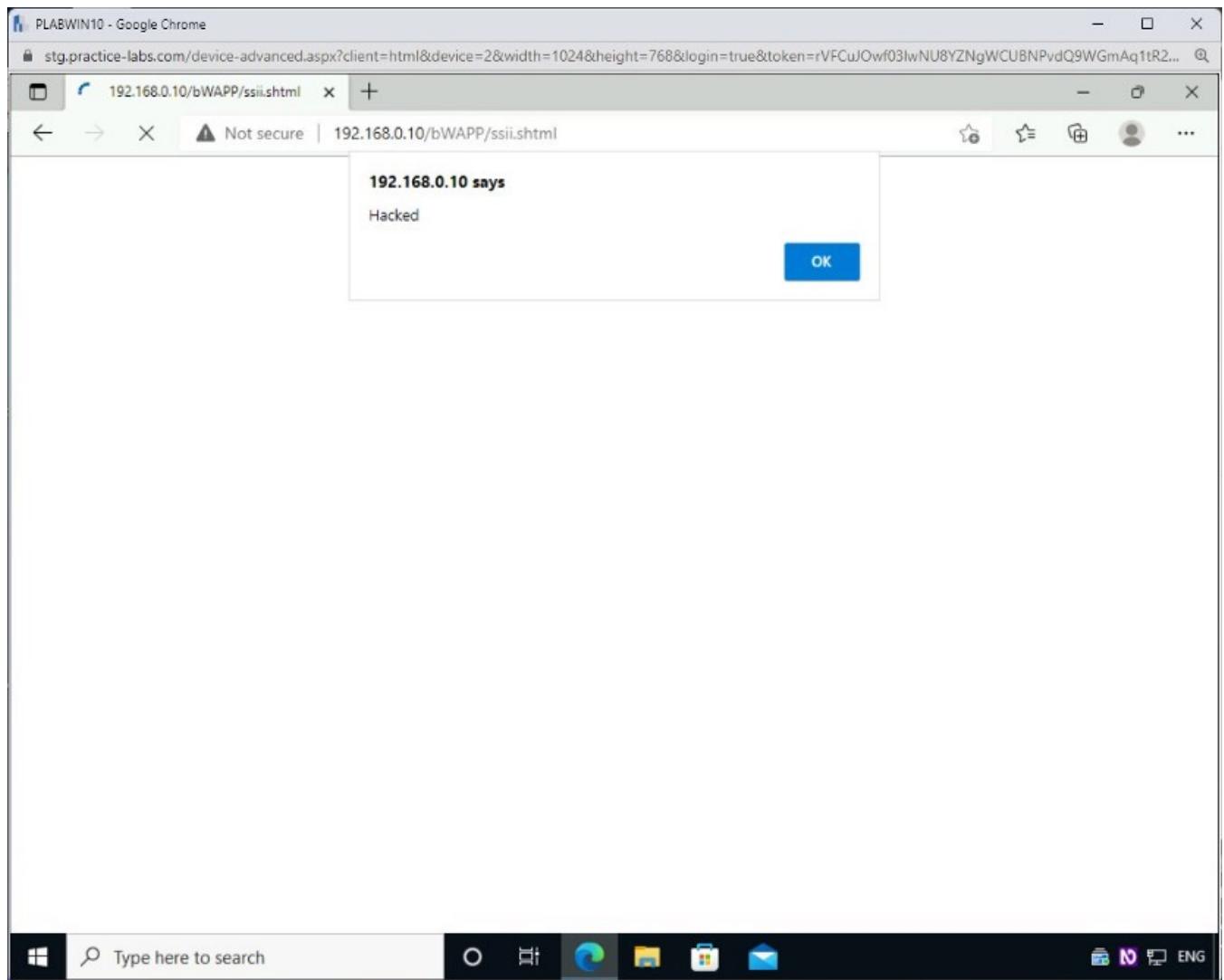
Click **Lookup**.

The screenshot shows a browser window titled "PLABWIN10 - Google Chrome" displaying the "bWAPP - SSI Injection" page at "stg.practice-labs.com/device-advanced.aspx?client=html&device=2&width=1024&height=768&login=true&token=rVFCuJOwnf03lwNU8YZNgWCUBNPvdQ9WGmAq1tR2...". The page has a yellow header with the bWAPP logo and a bee icon. It says "Choose your bug: bWAPP v2.2" and "Hack". Below that, it says "Set your security level: low Set Current low". A navigation bar at the top includes links for "Bugs", "Change Password", "Create User", "Set Security Level", "Reset", "Credits", "Blog", and "Logout". The main content area features a red banner with the text "/ Server-Side Includes (SSI) Injection /". Below the banner, there's a note: "What is your IP address? Lookup your IP address... (bee-box only)". There are two input fields: "First name:" containing "<script>alert('Hacked')</script>" and "Last name:" containing "user". A "Lookup" button is present. At the bottom of the page, a footer bar displays "bWAPP is licensed under CC BY-NC-ND © 2014 MME BVBA / Follow @MME_IT on Twitter and ask for our cheat sheet, containing all solutions! / Ne". The Windows taskbar at the bottom shows the search bar, Start button, and various pinned icons.

Step 5

Notice the output. The script embedded in the **First name** text box has been executed successfully.

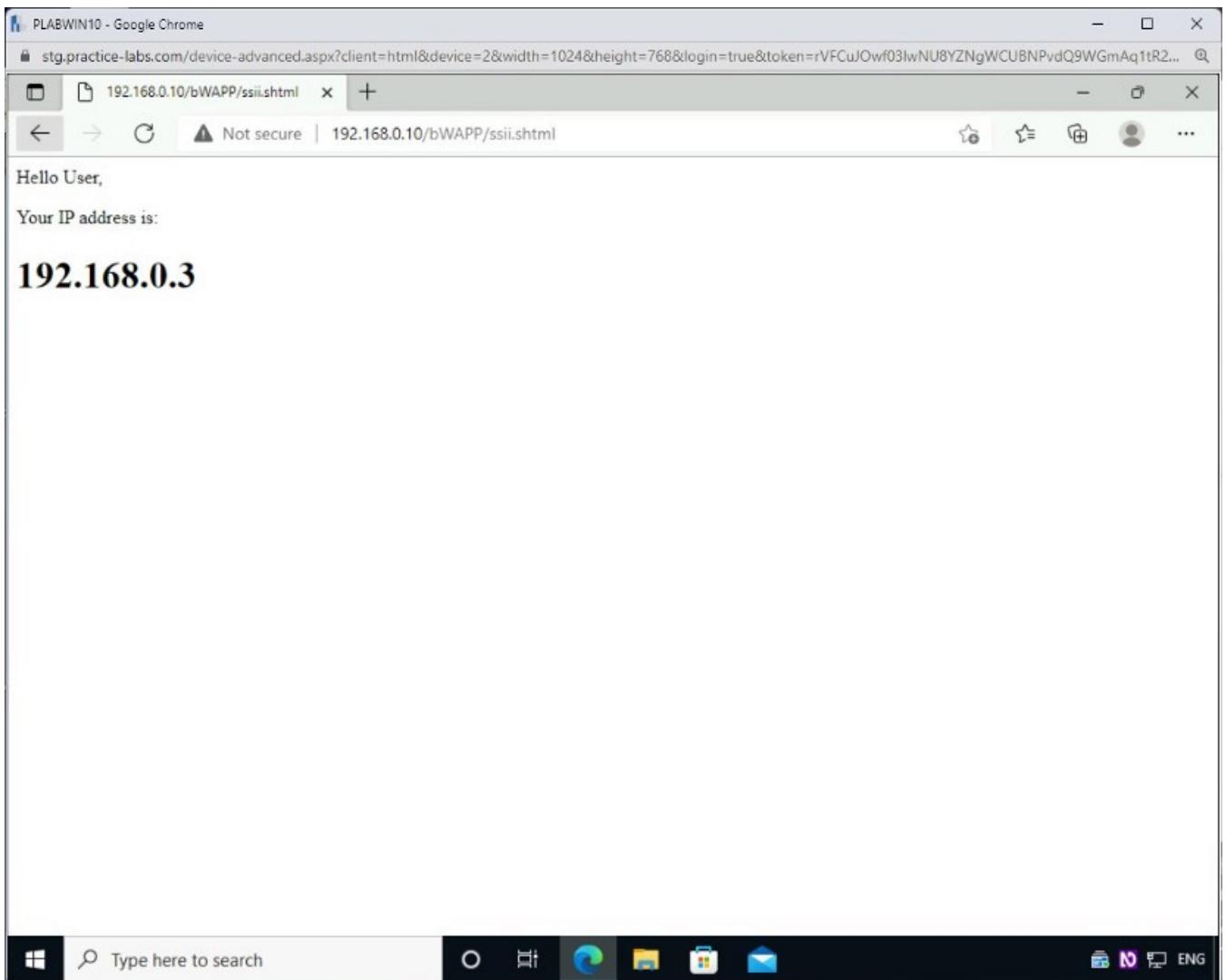
Click **OK**.



Step 6

Again, the text on the webpage is changed to the IPv4 address.

Click the **Back** button on the browser.



Step 7

Back on the **Server-Side Includes (SSI) Injection** webpage, you can also fetch the cookie from the web server. you need to insert the following into the **First name** text box:

```
<script>alert (document.cookie)</script>
```

In the **Last name** text box, type the following:

user

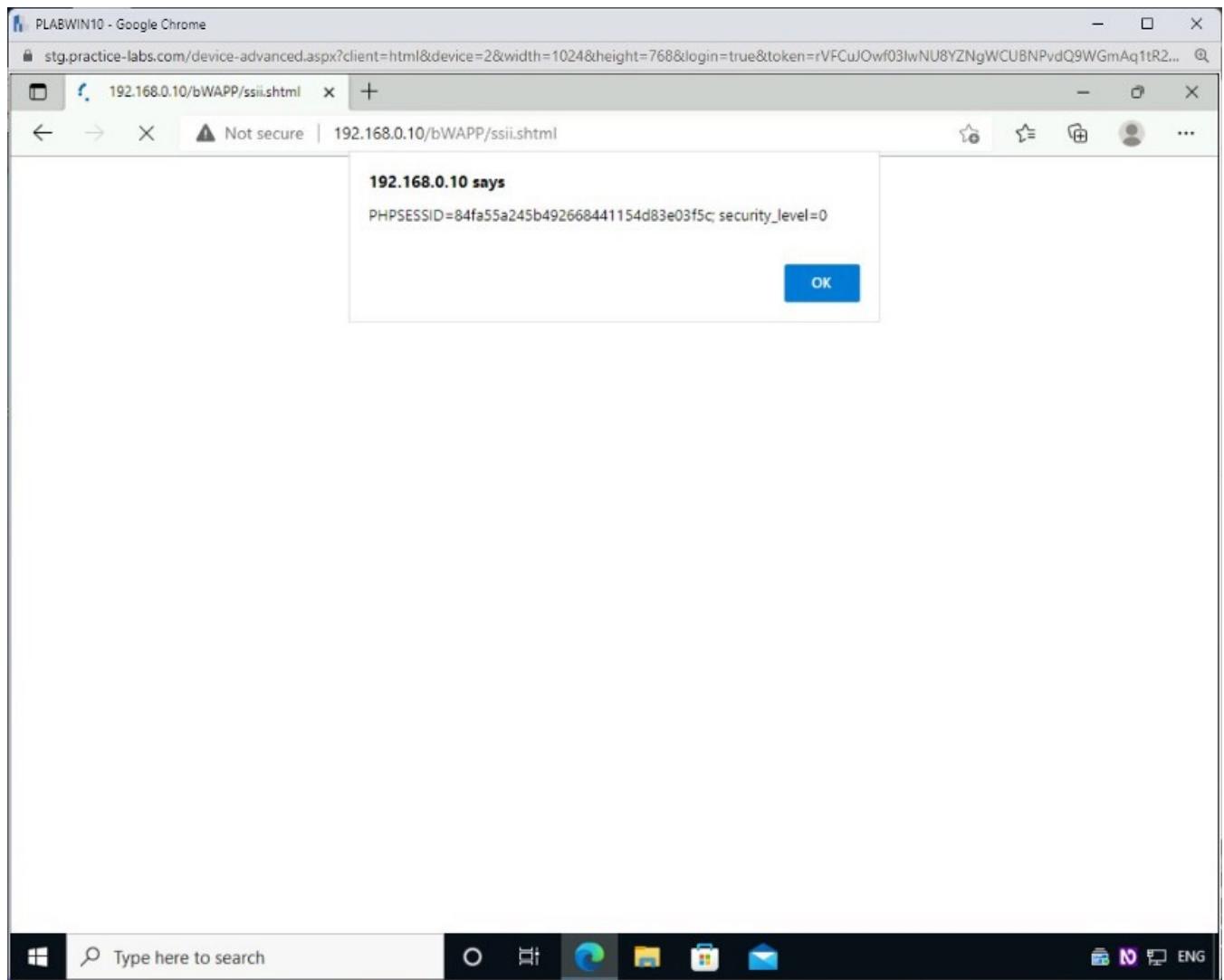
Click Lookup.

The screenshot shows a browser window titled "PLABWIN10 - Google Chrome" displaying the "bWAPP - SSI Injection" page at the URL "stg.practice-labs.com/device-advanced.aspx?client=html&device=2&width=1024&height=768&login=true&token=rVFCuJOwnf03lwNU8YZNgWCUBNPvdQ9WGmAq1tR2...". The page has a yellow header with the text "bWAPP" and "an extremely buggy web app!". It features a "Set your security level:" dropdown set to "low". Below the header is a navigation bar with links: "Bugs", "Change Password", "Create User", "Set Security Level", "Reset", "Credits", "Blog", "Logout", and "Web". A sidebar on the right contains social media icons for Twitter, LinkedIn, Facebook, and Email. The main content area has a heading "/ Server-Side Includes (SSI) Injection /". It includes a question "What is your IP address? Lookup your IP address... (bee-box only)". There are two input fields: "First name:" containing "<script>alert(document.cookie)" and "Last name:" containing "user". A "Lookup" button is present. At the bottom of the page, a footer states "bWAPP is licensed under CC BY-NC-ND © 2014 MME BVBA / Follow @MME_IT on Twitter and ask for our cheat sheet, containing all solutions! / Ne". The Windows taskbar at the bottom shows the search bar, Start button, and various pinned icons.

Step 8

Notice the output. The script embedded in the **First name** text box has been executed successfully.

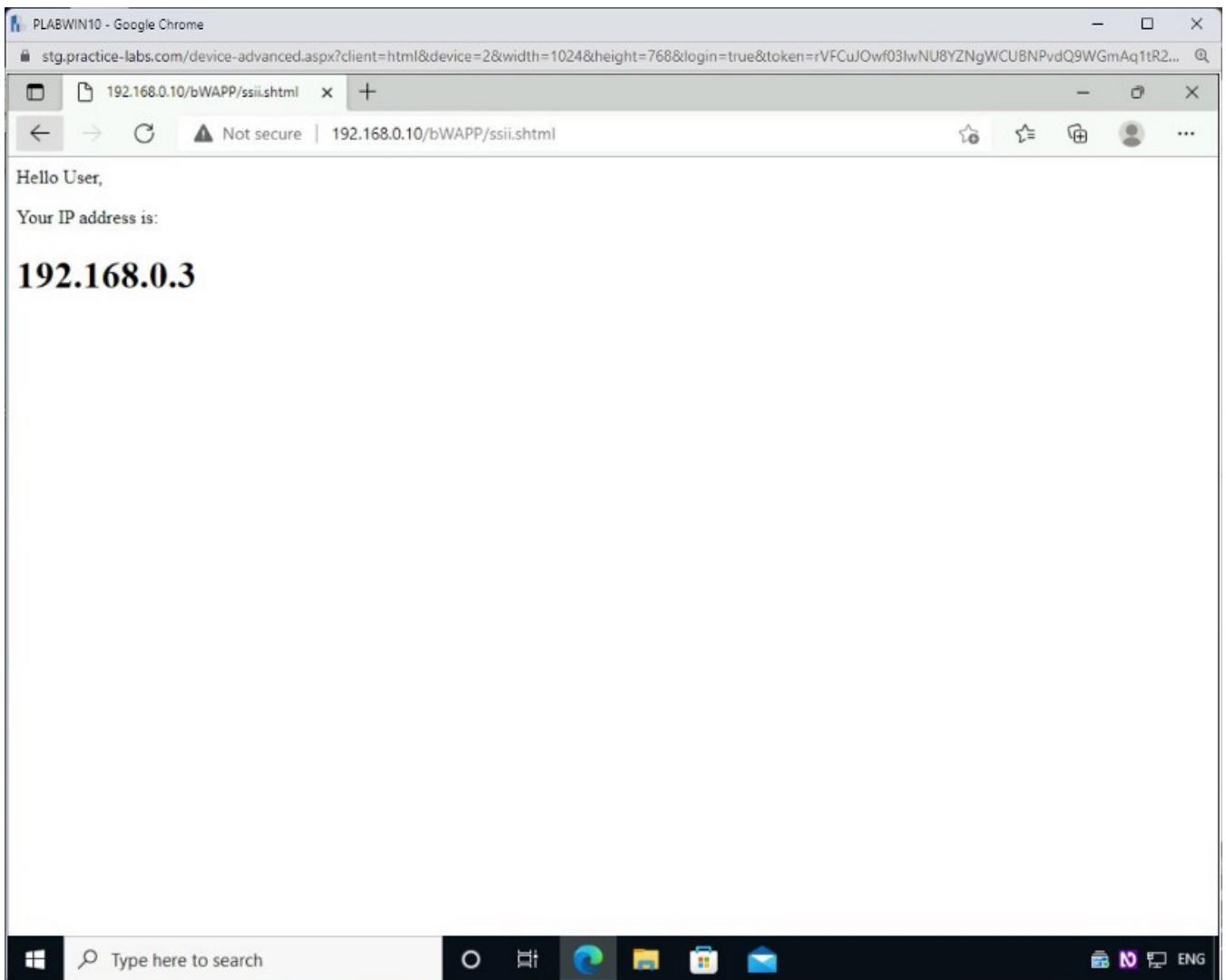
The cookie is being displayed in the dialog box. Click **OK**.



Step 9

Again, the text on the webpage is changed to the IPv4 address.

Click the **Back** button on the browser.



Step 10

Back on the **Server-Side Includes (SSI) Injection** webpage, you can show the current document name. you need to insert the following into the **First name** text box:

```
<!--#echo var="DOCUMENT_NAME" -->
```

In the **Last name** text box, type the following:

plab

Click Lookup.

PLABWIN10 - Google Chrome

stg.practice-labs.com/device-advanced.aspx?client=html&device=2&width=1024&height=768&login=true&token=rVFCuJOwnf03lwNU8YZNgWCUBNPvdQ9WGmAq1tR2...

bWAPP - SSI Injection Not secure | 192.168.0.10/bWAPP/ssii.php

bWAPP v2.2 Hack

Set your security level:
low Set Current: low

Bugs Change Password Create User Set Security Level Reset Credits Blog Logout

/ Server-Side Includes (SSI) Injection /

What is your IP address? Lookup your IP address... (bee-box only)

First name:
`<!--#echo var="DOCUMENT"`

Last name:
`plab`

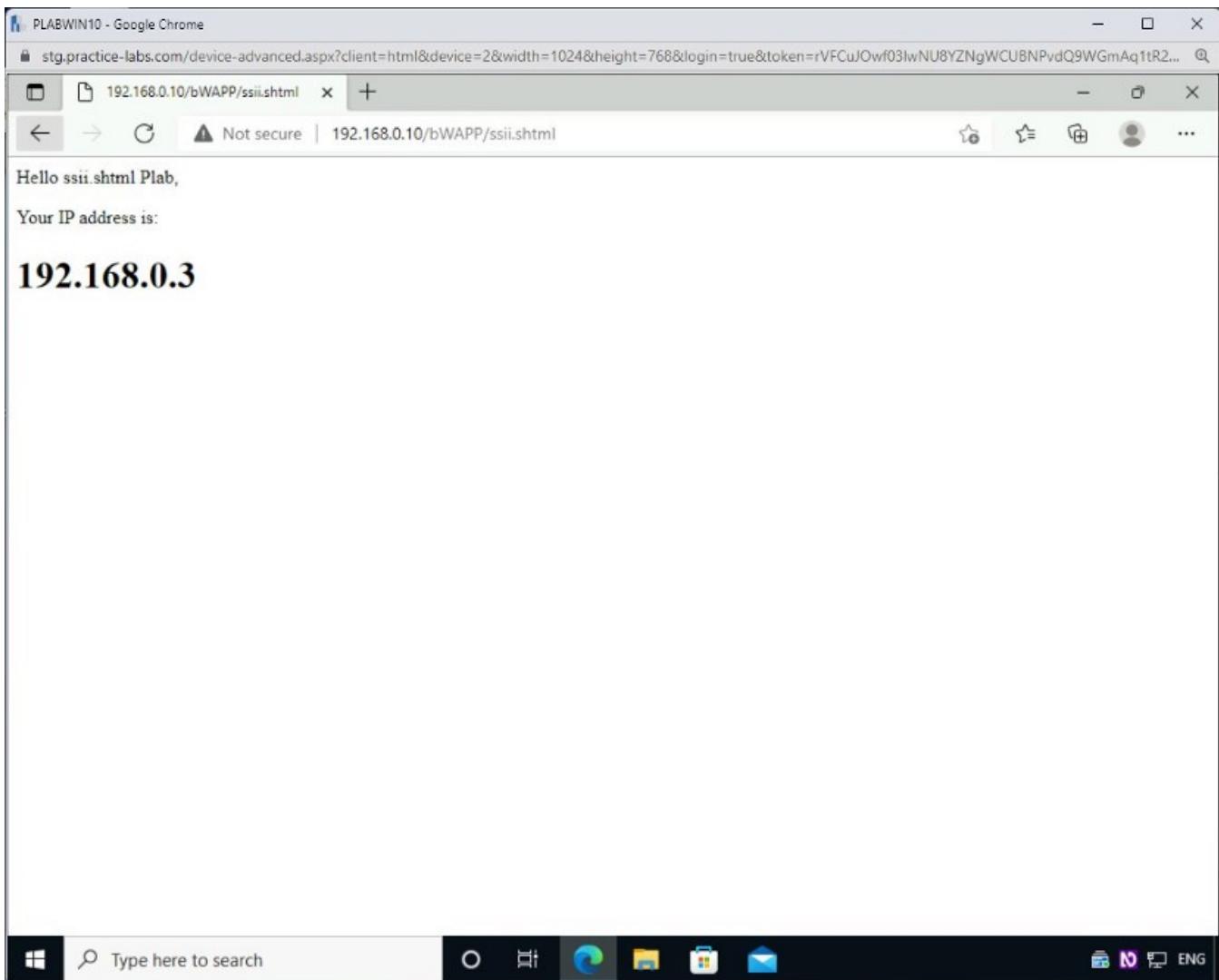
Lookup

bWAPP is licensed under CC BY-NC-ND © 2014 MME BVBA / Follow @MME_IT on Twitter and ask for our cheat sheet, containing all solutions! / Ne

Step 11

Notice the output. It displays the current document name before the last name, plab.

Click the **Back** button on the web browser window.



Step 12

Back on the **Server-Side Includes (SSI) Injection** webpage, you can show the virtual path and filename. To do this, you need to insert the following into the **First name** text box:

```
<!--#echo var="DOCUMENT_URI" -->
```

In the **Last name** text box, type the following:

plab

Click Lookup.

PLABWIN10 - Google Chrome

stg.practice-labs.com/device-advanced.aspx?client=html&device=2&width=1024&height=768&login=true&token=rVFCuJOwnf03lwNU8YZNgWCUBNPvdQ9WGmAq1tR2...

bWAPP - SSI Injection

Not secure | 192.168.0.10/bWAPP/ssii.php

bWAPP v2.2

an extremely buggy web app!

Set your security level:
low Set Current: low

Bugs Change Password Create User Set Security Level Reset Credits Blog Logout

/ Server-Side Includes (SSI) Injection /

What is your IP address? Lookup your IP address... (bee-box only)

First name:
<!--#echo var="DOCUMENT"

Last name:
plab

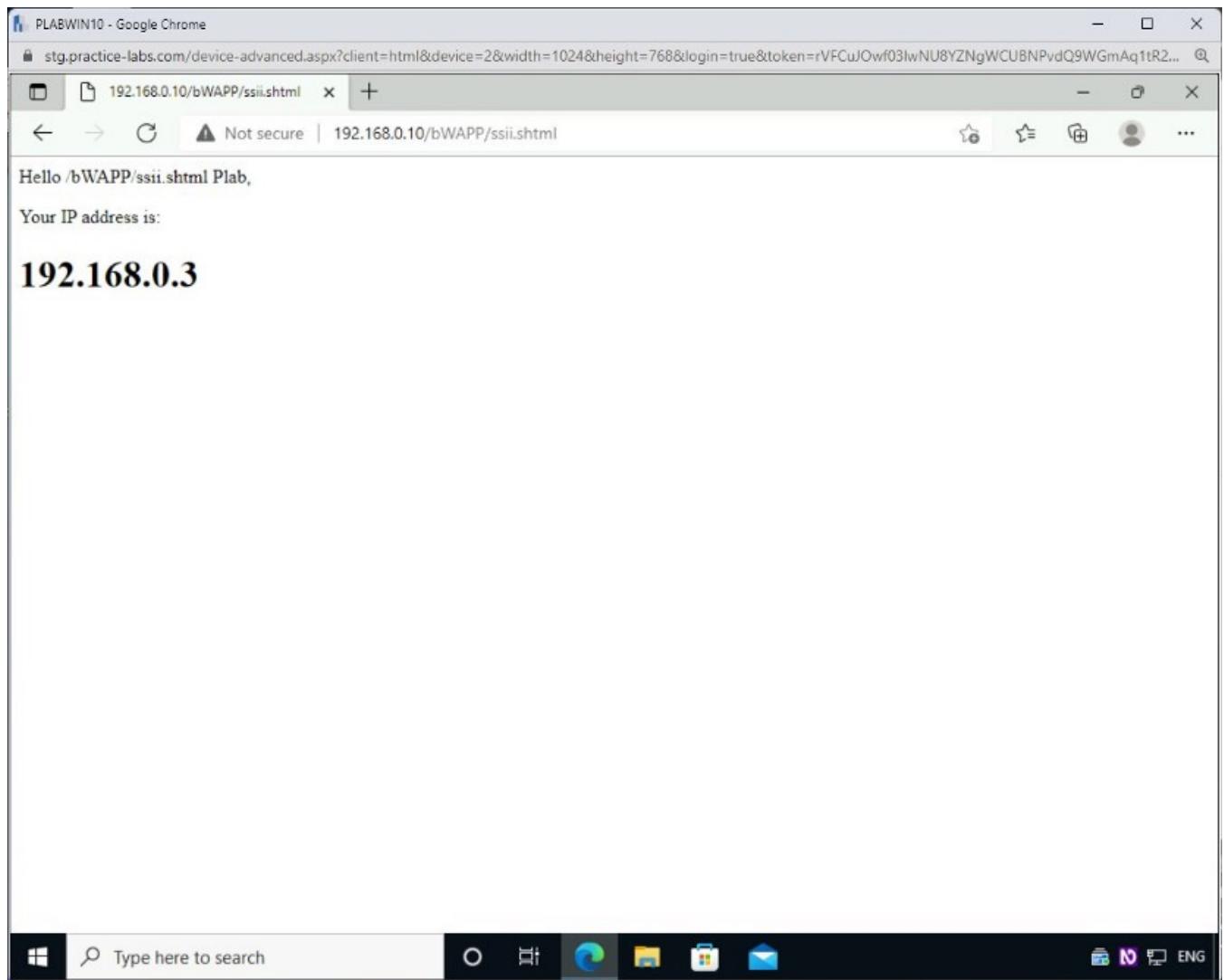
Lookup

bWAPP is licensed under CC BY-NC-ND © 2014 MME BVBA / Follow @MME_IT on Twitter and ask for our cheat sheet, containing all solutions! / Ne

Step 13

Notice that the output shows the virtual path and the file name.

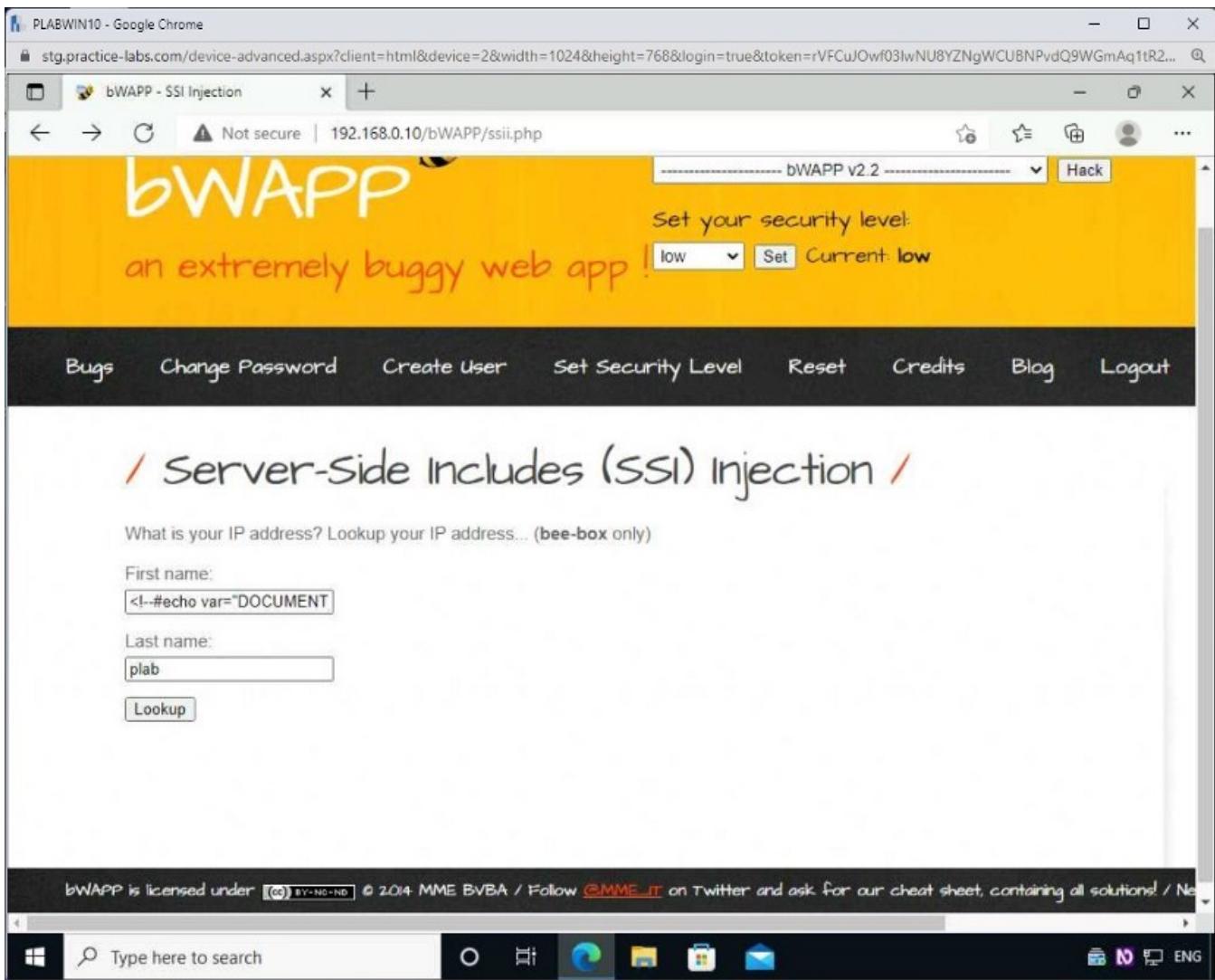
Click the **Back** button on the web browser.



Step 14

Click the **Back** button on the web browser window.

You should be back on the **Server-Side Includes (SSI) Injection** webpage.



Keep the web browser window open.

Task 6 — Perform Cross-site Scripting Attack

Cross-site scripting (XSS) is an attack where an attacker injects malicious scripts into vulnerable web applications, thereby causing serious damage. The malicious script is executed in a victim's web browser when they visit the infected page. JavaScript is the most common scripting language used for developing malicious code. There are two broad categories of cross-site scripting attacks. These are as follows:

- Reflected XSS
- Stored XSS

A reflected XSS attack is also known as a non-persistent XSS attack. An attacker sends the malicious code as a link in an email or posts it on a website in this type of attack. Once a victim is infected, the malicious script embedded gets executed and is reflected on a victim's web browser. The browser then sends cookie information of a victim's session to an attacker.

A stored XSS attack is also known as a persistent XSS attack. In this type of attack, an attacker injects malicious script directly into a vulnerable web application. The malicious script steals cookie information of victim's sessions and sends it to an attacker. Therefore, in stored XSS attacks, the risk is higher as malicious script executes on every visit to the application.

In this task, you will perform cross-site scripting attacks on the **bWAPP** application.

Step 1

Connect to **PLABWIN10**. The **Microsoft Edge** window should be open.

To begin reflected cross-site scripting attack, from the **Choose your bug** drop-down, click **Cross-site Scripting – Reflected (GET)**, and click **Hack**.

The screenshot shows a Microsoft Edge browser window with the following details:

- Title Bar:** PLABWIN10 - Google Chrome
- Address Bar:** stg.practice-labs.com/device-advanced.aspx?client=html&device=2&width=1024&height=768&login=true&token=rVFCuJQwf03lwNU8YZNgWCUBNPvdQ9WGmAq1tR2... (Note: The URL is truncated)
- Page Title:** bWAPP - SSL Injection
- Content Area:**
 - bWAPP Logo:** A cartoon bee icon above the text "bWAPP".
 - Tagline:** "an extremely buggy web app!"
 - User Interface Elements:**
 - Choose your bug:** A dropdown menu currently set to "Cross-Site Scripting - Reflected (GET)".
 - Hack:** A button next to the dropdown.
 - Set your security level:** A dropdown menu set to "low".
 - Current:** A button labeled "Current low".
 - Navigation Bar:** Bugs, Change Password, Create User, Set Security Level, Reset, Credits, Blog, Logout.
 - Section Header:** // Server-Side Includes (SSI) Injection //
 - Form Fields:** First name: <!--#echo var="DOCUMENT" -->, Last name: plab, Lookup button.
 - Footer:** bWAPP is licensed under (CC BY-NC-ND) © 2014 MME BVBA / Follow [MME.IT](#) on Twitter and ask for our cheat sheet, containing all solutions! / Ne
 - Bottom Bar:** Windows Start button, Type here to search, Task View, File Explorer, Internet Explorer, File Manager, Mail, Network, Battery, ENG.

Step 2

On the **XSS – Reflected (GET)** webpage, In the **First name** text box, type the following name:

Plab

In the **Last name** text box, type the following name:

User

Click **Go**.

The screenshot shows a Google Chrome window titled 'PLABWIN10 - Google Chrome' displaying the bWAPP application at '192.168.0.10/bWAPP/xss_get.php'. The page has a yellow header with the bWAPP logo and a bee icon. It says 'Choose your bug: bWAPP v2.2' and 'Set your security level: low'. Below the header is a navigation bar with links: Bugs, Change Password, Create User, Set Security Level, Reset, Credits, Blog, and Logout. The main content area features a title 'XSS - Reflected (GET)' with a red and green border. It contains fields for 'First name:' (containing 'plab') and 'Last name:' (containing 'user'), followed by a 'Go' button. At the bottom of the page, there is a footer note: 'bWAPP is licensed under CC BY-NC-ND © 2014 MME BVBA / Follow @MME_IT on Twitter and ask for our cheat sheet, containing all solutions! / Ne'. The browser's taskbar at the bottom shows the search bar and various pinned icons.

Step 3

Observe the output displayed in the address bar.

The input passed to the server is reflected in the application. This indicates a good entry point for reflected XSS attacks as the response is getting reflected.

PLABWIN10 - Google Chrome
stg.practice-labs.com/device-advanced.aspx?client=html&device=2&width=1024&height=768&login=true&token=rVFCuJOwnf03lwNU8YZNgWCUBNPvdQ9WGmAq1tR2...
bWAPP - XSS
Not secure | 192.168.0.10/bWAPP/xss_get.php?firstname=plab&lastname=user&form=submit
hash key - Bing Search
vertical slash
vertical line symbol
an extremely buggy web app!
low Set Current low
Bugs Change Password Create User Set Security Level Reset Credits Blog Logout
/ XSS - Reflected (GET) /
Enter your first and last name:
First name:
Last name:
Go
Welcome plab user
bWAPP is licensed under CC BY-NC-ND © 2014 MME BVBA / Follow @MME_IT on Twitter and ask for our cheat sheet, containing all solutions! / Ne
Type here to search O Hi E Mail ENG

Step 4

On the **XSS – Reflected (GET)** webpage, In the **First name** text box, inject a JavaScript code by typing the following:

```
<script>alert ("Hacked")</script>
```

In the **Last name** text box, type the following name:

User

Click **Go**.

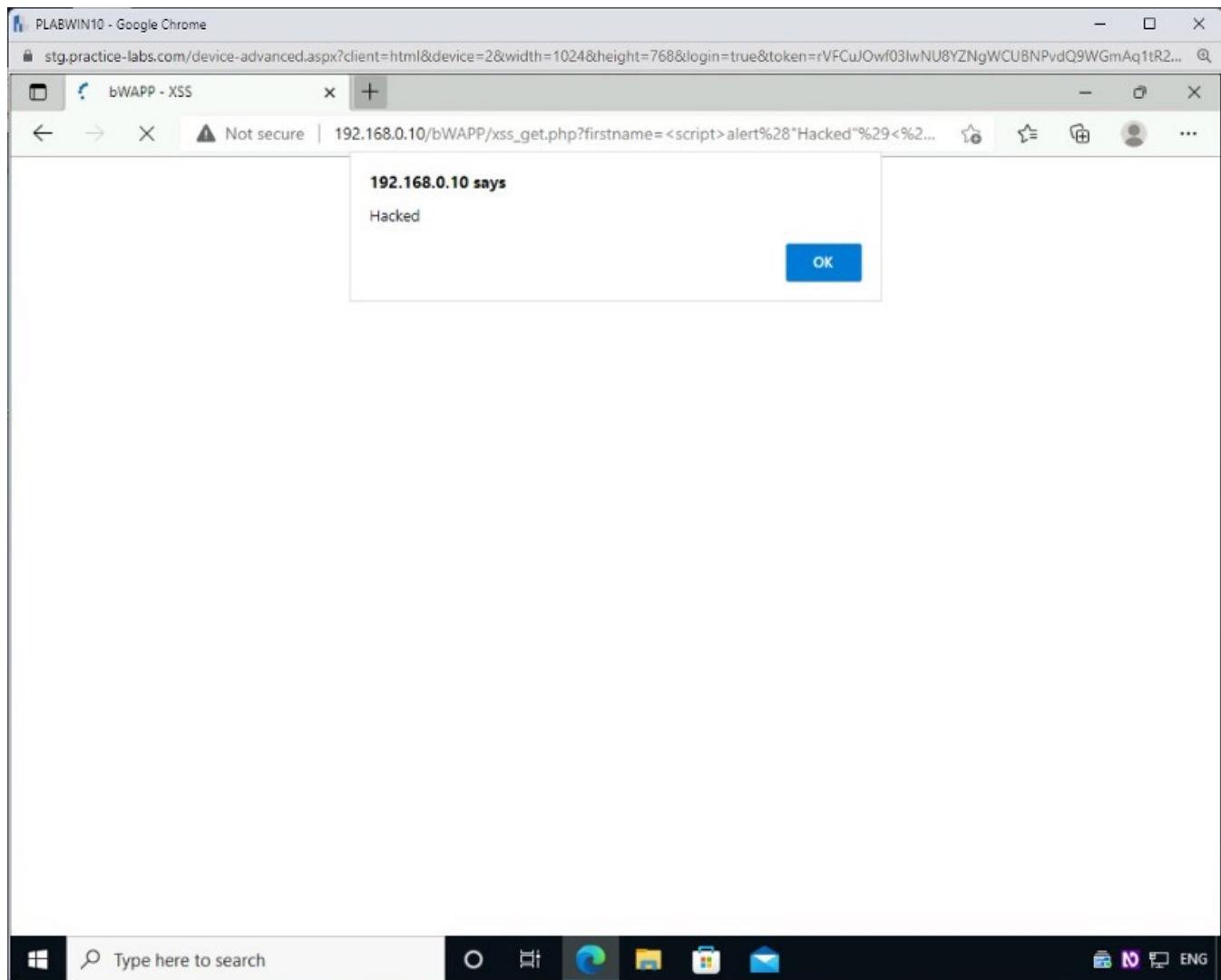
The screenshot shows a web browser window for Google Chrome on a Windows 10 desktop. The title bar says 'PLABWIN10 - Google Chrome'. The address bar shows the URL 'stg.practice-labs.com/device-advanced.aspx?client=html&device=2&width=1024&height=768&login=true&token=rVFCuJOwnf03lwNU8YZNgWCUBNPvdQ9WGmAq1tR2...'. The main content area displays the 'bWAPP' logo and the tagline 'an extremely buggy web app!'. A navigation bar at the top includes links for 'Bugs', 'Change Password', 'Create User', 'Set Security Level', 'Reset', 'Credits', 'Blog', and 'Logout'. Below the navigation bar, a section titled '/ XSS - Reflected (GET) /' contains a form with fields for 'First name' (containing '<script>alert("Hacked")</script>'), 'Last name' (containing 'User'), and a 'Go' button. The response from the server is 'Welcome alert("Hacked") user'. The status bar at the bottom of the browser window shows 'bWAPP is licensed under CC BY-NC-ND © 2014 MME BVBA / Follow @MME_IT on Twitter and ask for our cheat sheet, containing all solutions! / Ne...' and the Windows taskbar with icons for File Explorer, Mail, and Start.

Step 5

The server processes the script and displays the alert message box.

Click **OK**.

Note: The script is executed and reflected in the server response, a vulnerability. Using this vulnerability in a real-time environment, an attacker can inject malicious scripts in less secure applications to steal cookies, learn about the document location, and so on.



Step 6

You are back on the **XSS – Reflected (GET)** webpage.

From the **Choose your bug** drop-down, select **Cross-Site Scripting – Stored (Blog)**, and click **Hack**.

The screenshot shows a web browser window titled "PLABWIN10 - Google Chrome". The address bar displays "stg.practice-labs.com/device-advanced.aspx?client=html&device=2&width=1024&height=768&login=true&token=rVFCuJOwnf03lwNU8YZNgWCUBNPvdQ9WGmAq1tR2...". The main content area is the "bWAPP" application, which has a yellow header with the text "Choose your bug: Cross-Site Scripting - Stored (Blog)" and "Set your security level: low". Below the header, there are links for "Bugs", "Change Password", "Create User", "Set Security Level", "Reset", "Credits", "Blog", and "Logout". The main content area is titled "/ XSS - Reflected (GET) /". It contains fields for "First name:" and "Last name:", both represented by empty input boxes. A "Go" button is located below these fields. The text "Welcome User" is displayed. At the bottom of the page, a footer bar includes the text "bWAPP is licensed under CC BY-NC-ND © 2014 MME BVBA / Follow @MME_IT on Twitter and ask for our cheat sheet, containing all solutions! / Ne", a search bar, and a language switcher showing "ENG".

Step 7

On the **XSS — Stored (Blog)** webpage, type the following message in the text box:

Welcome to the PLAB blog!

Click Submit.

The screenshot shows a web browser window for Google Chrome on a Windows 10 desktop. The title bar reads "PLABWIN10 - Google Chrome". The address bar shows the URL "stg.practice-labs.com/device-advanced.aspx?client=html&device=2&width=1024&height=768&login=true&token=rVFCuJOwnf03lwNU8YZNgWCUBNPvdQ9WGmAq1tR2...". The main content area displays the bWAPP web application. At the top, there's a yellow header with the bWAPP logo and a bee icon. It says "Choose your bug: bWAPP v2.2" and "Hack". Below that, it says "Set your security level: low Set Current low". A navigation bar at the bottom includes links for "Bugs", "Change Password", "Create User", "Set Security Level", "Reset", "Credits", "Blog", and "Logout". The main content area has a title "XSS - Stored (Blog)". Below it is a text input field containing "Welcome to the PLAB blog!". Underneath the input field are buttons for "Submit", "Add: ", "Show all: ", and "Delete: ". A table header row follows, with columns labeled "#", "Owner", "Date", and "Entry". The footer of the page includes a license notice: "bWAPP is licensed under [\(cc\) BY-NC-ND](#) © 2014 MME BVBA / Follow [@MME_IT](#) on Twitter and ask for our cheat sheet, containing all solutions! / Ne". The Windows taskbar at the bottom shows the search bar, pinned icons for File Explorer, Edge, and Mail, and the language setting "ENG".

Step 8

Observe the output displayed on the screen.

The entered values are stored in the application's database and are displayed permanently on the application's **XSS – Stored (Blog)** page. This could be a potential entry point for stored XSS attacks.

The screenshot shows a web browser window for 'PLABWIN10 - Google Chrome' displaying the 'bWAPP - XSS' page at '192.168.0.10/bWAPP/xss_stored_1.php'. The page title is 'bWAPP' and the subtitle is 'an extremely buggy web app!'. A navigation bar includes links for 'Bugs', 'Change Password', 'Create User', 'Set Security Level', 'Reset', 'Credits', 'Blog', and 'Logout'. A dropdown menu 'bWAPP v2.2' is open, showing 'Hack' as the selected option. A 'Set your security level:' dropdown is set to 'low'. Below the navigation is a large title 'XSS - Stored (Blog)' with a hand-drawn style. A text input field is present. Below it is a message: 'Submit' button, 'Add: ', 'Show all: ', 'Delete: ', and 'Your entry was added to our blog!'. A table lists one entry: #1, Owner bee, Date 2022-03-23 13:50:56, Entry 'Welcome to the PLAB blog!'. The status bar at the bottom indicates 'bWAPP is licensed under CC BY-NC-ND © 2014 MME BVBA / Follow @MME_IT on Twitter and ask for our cheat sheet, containing all solutions! / Ne...'.

Step 9

On the **XSS — Stored (Blog)** webpage, type the following message in the text box:

```
<script>alert ("Hacked")</script>
```

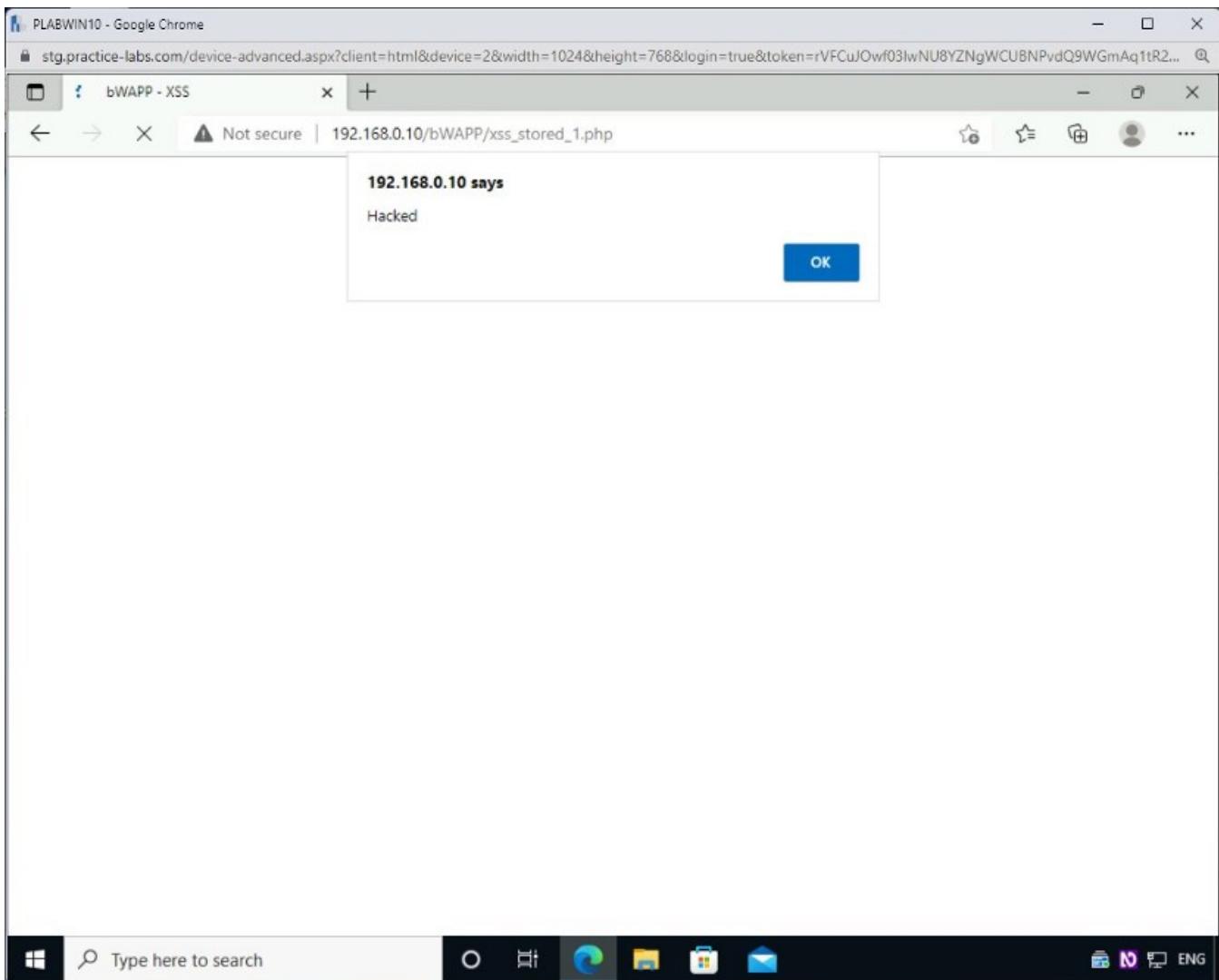
Click **Submit**.

The screenshot shows a web browser window for Google Chrome on a Windows 10 desktop. The title bar reads "PLABWIN10 - Google Chrome". The address bar shows the URL "stg.practice-labs.com/device-advanced.aspx?client=html&device=2&width=1024&height=768&login=true&token=rVFCuJOwnf03lwNU8YZNgWCUBNPvdQ9WGmAq1tR2...". The main content area displays the bWAPP homepage with a yellow header featuring the text "an extremely buggy web app!". Below the header is a navigation bar with links for "Bugs", "Change Password", "Create User", "Set Security Level", "Reset", "Credits", "Blog", and "Logout". A dropdown menu titled "Choose your bug:" is set to "bWAPP v2.2" with a "Hack" button. A "Set your security level:" dropdown is set to "low". The main content area has a heading "/ XSS - Stored (Blog) /". A text input field contains the script "<script>alert('Hacked')</script>". Below the input field are buttons for "Submit", "Add: ", "Show all: ", and "Delete: ". A success message "Your entry was added to our blog!" is displayed. A table lists the stored entry: #1, Owner bee, Date 2022-03-23 13:50:56, Entry "Welcome to the PLAB blog!". At the bottom of the page, a footer notes "bWAPP is licensed under CC BY-NC-ND © 2014 MME BVBA / Follow @MME_IT on Twitter and ask for our cheat sheet, containing all solutions! / Ne". The Windows taskbar at the bottom includes icons for Start, Search, Task View, File Explorer, Edge, File Explorer, Mail, and a network icon.

Step 10

The server processes the script and displays the alert message box.

Click **OK**.



Step 11

You are back on the **XSS – Stored (Blog)** webpage. Notice that there is a second entry in the list, but it is empty. The entered JavaScript gets reflected in the stored XSS section of the web page. Unlike reflected XSS, stored XSS is permanent since the entered values are stored in the application's database.

Therefore, attackers could use this vulnerability to deface any website by displaying an image using stored XSS, stealing users' cookies, and so on.

The screenshot shows a web browser window for the bWAPP application. The URL is `stg.practice-labs.com/device-advanced.aspx?client=html&device=2&width=1024&height=768&login=true&token=rVFCuJOwnf03lwNU8YZNgWCUBNPvdQ9WGmAq1tR2...`. The page title is "bWAPP - XSS". A warning message "Not secure | 192.168.0.10/bWAPP/xss_stored_1.php" is displayed. The main content area has a yellow header with the text "Set your security level: low Set Current: low". Below the header is a navigation bar with links: Bugs, Change Password, Create User, Set Security Level, Reset, Credits, Blog, Logout. The main content is titled "/ XSS - Stored (Blog) /". There is a large empty text input field. Below it are buttons for "Submit", "Add: ", "Show all: ", and "Delete: ". A success message "Your entry was added to our blog!" is shown. A table lists two entries:

#	Owner	Date	Entry
1	bee	2022-03-23 13:50:56	Welcome to the PLAB blog!
2	bee	2022-03-23 13:52:12	

At the bottom, a footer bar includes the text "bWAPP is licensed under CC BY-NC-ND © 2014 MME BVBA / Follow @MME_IT on Twitter and ask for our cheat sheet, containing all solutions! / Ne", a search bar, and a language switcher showing "ENG".

Step 12

Once again, select **Cross-Site Scripting – Stored (Blog)** from the **Choose your bug** drop-down and click **Hack**.

The screenshot shows a web browser window for Google Chrome on a Windows 10 desktop. The title bar says 'PLABWIN10 - Google Chrome' and the address bar shows 'stg.practice-labs.com/device-advanced.aspx?client=html&device=2&width=1024&height=768&login=true&token=rVFCuJOwnf03lwNU8YZNgWCUBNPvdQ9WGmAq1tR2...'. The page itself is the 'bWAPP - XSS' section of the application. At the top, it says 'Choose your bug: Cross-Site Scripting - Stored (Blog)' and 'Set your security level: low'. Below that, there's a form with a large input field and a 'Submit' button. The message 'Your entry was added to our blog!' is displayed below the form. A table shows the stored entry: #1, Owner 'bee', Date '2022-03-23 13:50:56', and Entry 'Welcome to the PLAB blog!'. At the bottom of the page, a footer notes 'bWAPP is licensed under CC BY-NC-ND © 2014 MME BVBA / Follow @MME_IT on Twitter and ask for our cheat sheet, containing all solutions! / Ne...'.

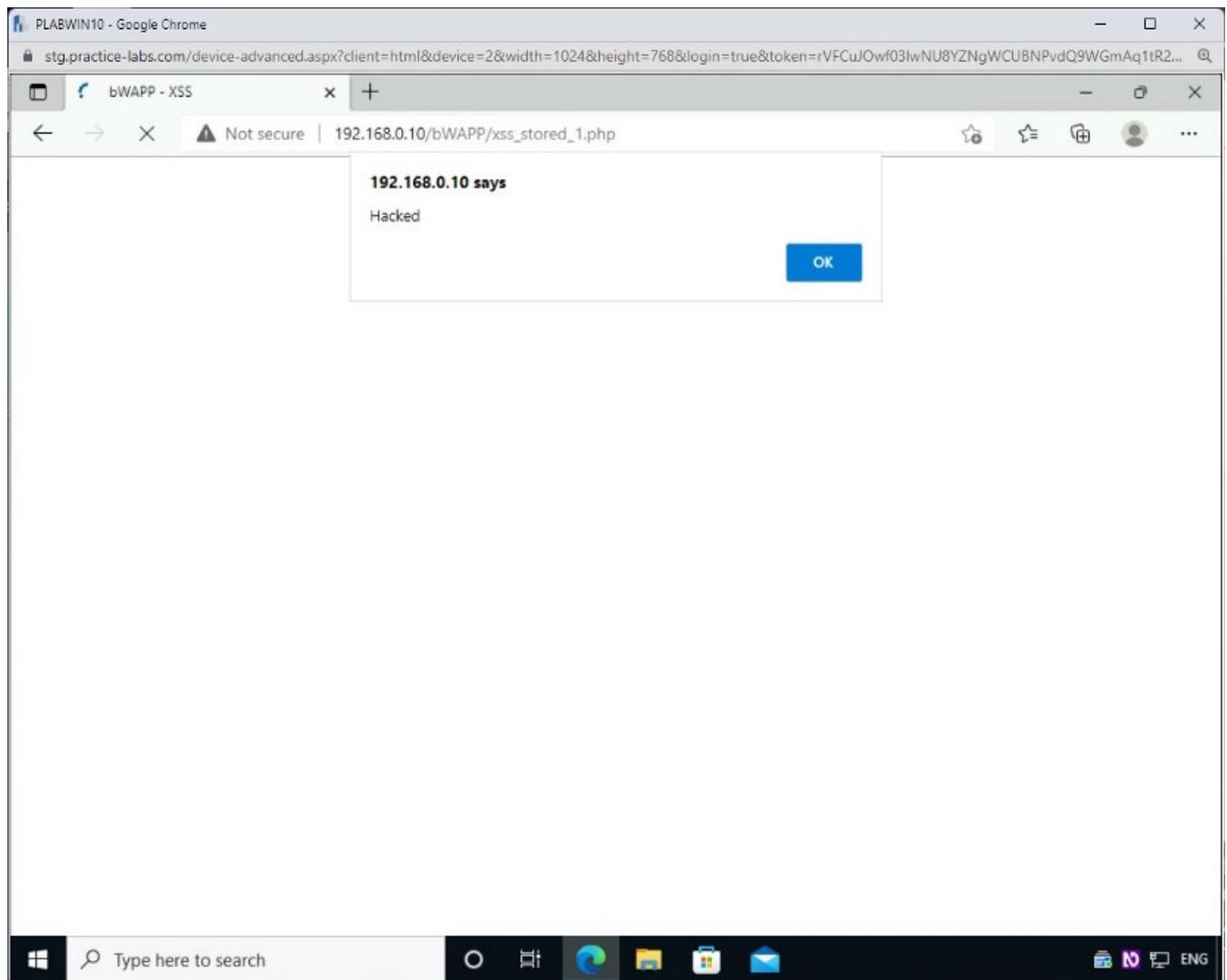
Step 13

The stored XSS script you used on this page is automatically displayed.

Click **OK** in the alert message box.

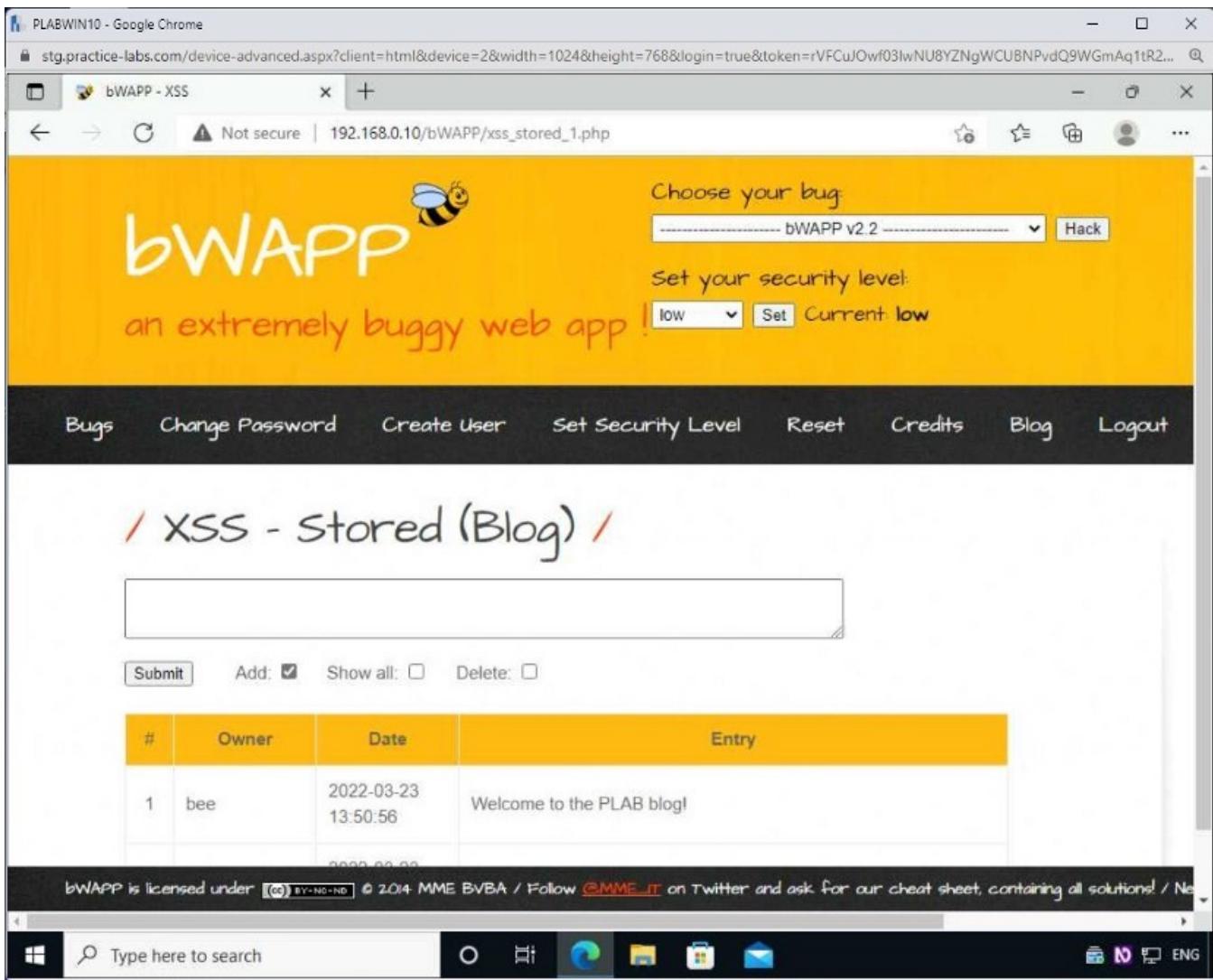
This is because the injected JavaScript is stored in the database, and it gets executed every time you navigate to the stored XSS section.

However, in reflected XSS, this does not happen since the values are not stored in the application's database.



Step 14

The **XSS – Stored (Blog)** webpage is now displayed.



Keep the Microsoft Edge window open.

Task 7 — Perform Cross-site Scripting (XSS) — Reflected (HREF) Attack

In a Reflected (HREF) attack, an attacker can insert a malicious script in the URL itself. This problem occurs when a web page requests a string of information, such as a username, and displays it in the URL. If the input is not sanitized, an attacker can leverage this vulnerability to run a malicious script.

In this task, you will perform an XSS — Reflected (HREF) attack. To do this, perform the following steps:

Step 1

Ensure you have powered on all the devices listed in the introduction and connect to **PLABWIN10**.

The **Microsoft Edge** window should be open.

To begin reflected cross-site scripting attack, from the **Choose your bug** drop-down, click **Cross-site Scripting — Reflected (HREF)** and click **Hack**.

The screenshot shows a web browser window for 'PLABWIN10 - Google Chrome' displaying the 'bWAPP - XSS' page at '192.168.0.10/bWAPP/xss_stored_1.php'. The page has a yellow header with the 'bWAPP' logo and a bee icon. It features a 'Choose your bug:' dropdown set to 'Cross-Site Scripting - Reflected (HREF)' with a 'Hack' button. Below it is a 'Set your security level:' dropdown set to 'low' with a 'Set Current low' button. A navigation bar at the top includes links for 'Bugs', 'Change Password', 'Create User', 'Set Security Level', 'Reset', 'Credits', 'Blog', and 'Logout'. The main content area is titled '/ XSS - Stored (Blog) /' and contains a large text input field. Below it are buttons for 'Submit', 'Add: ', 'Show all: ', and 'Delete: '. A table lists one entry:

#	Owner	Date	Entry
1	bee	2022-03-23 13:50:56	Welcome to the PLAB blog!

A footer note states: 'bWAPP is licensed under [\(CC\) BY-NC-ND](#) © 2014 MME BVBA / Follow [@MME_IT](#) on Twitter and ask for our cheat sheet, containing all solutions! / Ne...'. The Windows taskbar at the bottom shows the search bar, Start button, and various pinned icons.

Step 2

The **XSS — Reflected (HREF)** webpage is displayed. In the text box, type the following:

PLAB

Click **Continue**.

The screenshot shows a web browser window for Google Chrome on a Windows 10 desktop. The title bar reads "PLABWIN10 - Google Chrome". The address bar shows the URL "stg.practice-labs.com/device-advanced.aspx?client=html&device=2&width=1024&height=768&login=true&token=rVFCuJOwnf03lwNU8YZNgWCUBNPvdQ9WGmAq1tR2...". The main content area displays the bWAPP homepage with a yellow header. The header includes the bWAPP logo (a bee), the text "Choose your bug: bWAPP v2.2 Hack", and "Set your security level: low Set Current low". Below the header is a navigation bar with links: Bugs, Change Password, Create User, Set Security Level, Reset, Credits, Blog, and Logout. The main content area has a red banner with the text "/ XSS - Reflected (HREF) /". Below the banner, a message says "In order to vote for your favorite movie, your name must be entered:". There is an input field containing "PLAB" and a "Continue" button. At the bottom of the page, there is a footer with the text "bWAPP is licensed under CC BY-NC-ND © 2014 MME BVBA / Follow @MME_IT on Twitter and ask for our cheat sheet, containing all solutions! / Ne" and a Windows taskbar with search, start, and pinned app icons.

Step 3

Notice that the entered name is now embedded in the URL and displayed on the webpage.

The screenshot shows a Google Chrome window with the title "PLABWIN10 - Google Chrome". The address bar displays the URL "stg.practice-labs.com/device-advanced.aspx?client=html&device=2&width=1024&height=768&login=true&token=rVFCuJOwnf03lwNU8YZNgWCUBNPvdQ9WGmAq1tR2...". The page content is from the "bWAPP - XSS" lab. At the top, there's a navigation bar with links: "Bugs", "Change Password", "Create User", "Set Security Level", "Reset", "Credits", "Blog", and "Logout". Below this is a large heading "XSS - Reflected (HREF)". The main area contains two paragraphs: "Hello PLAB, please vote for your favorite movie." and "Remember, Tony Stark wants to win every time...". Below these is a table with the following data:

Title	Release	Character	Genre	Vote
G.I. Joe: Retaliation	2013	Cobra Commander	action	Vote
Iron Man	2008	Tony Stark	action	Vote
Man of Steel	2013	Clark Kent	action	Vote
Terminator Salvation	2009	John Connor	sci-fi	Vote
The Amazing Spider-Man	2012	Peter Parker	action	Vote
The Cabin in the Woods	2011	Some zombies	horror	Vote

At the bottom of the page, there's a footer note: "bWAPP is licensed under CC BY-NC-ND © 2014 MME BVBA / Follow @MME_IT on Twitter and ask for our cheat sheet, containing all solutions! / Ne". The Windows taskbar at the bottom shows the search bar, Start button, and various pinned icons.

Step 4

In the URL, replace the name **PLAB** with the following script:

```
><script>alert(1)</script><
```

Press **Enter**.

The screenshot shows a Google Chrome window titled "PLABWIN10 - Google Chrome". The address bar contains the URL "stg.practice-labs.com/device-advanced.aspx?client=html&device=2&width=1024&height=768&login=true&token=rVFCuJOwnf03lwNU8YZNgWCUBNPvdQ9WGmAq1tR2...". The main content area displays a page titled "/ XSS - Reflected (HREF) /". The page includes a message: "Hello PLAB, please vote for your favorite movie." and "Remember, Tony Stark wants to win every time...". Below this is a table with movie information:

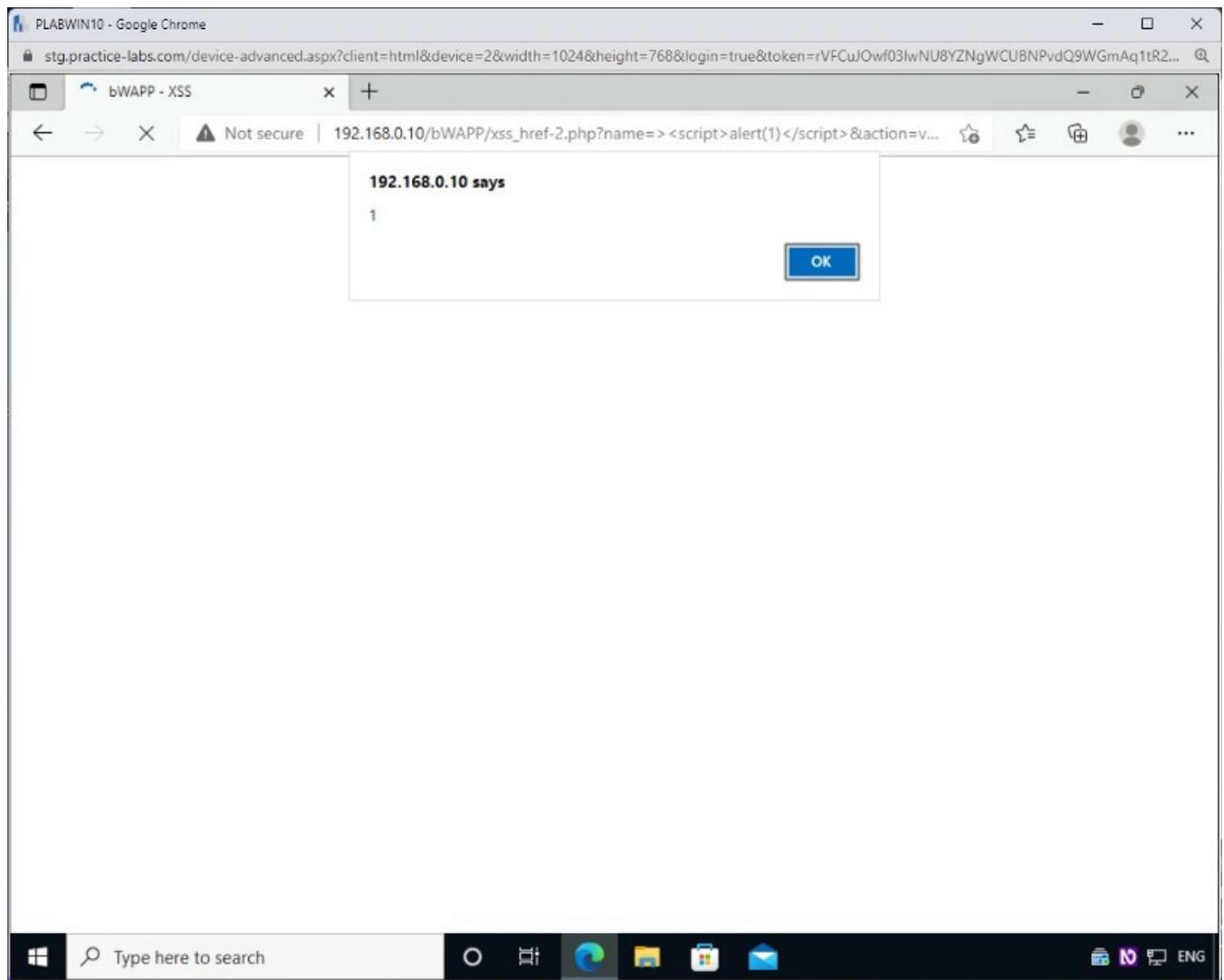
Title	Release	Character	Genre	Vote
G.I. Joe: Retaliation	2013	Cobra Commander	action	Vote
Iron Man	2008	Tony Stark	action	Vote
Man of Steel	2013	Clark Kent	action	Vote
Terminator Salvation	2009	John Connor	sci-fi	Vote
The Amazing Spider-Man	2012	Peter Parker	action	Vote
The Cabin in the Woods	2011	Some zombies	horror	Vote

At the bottom of the page, there is a footer note: "bWAPP is licensed under [\(cc\) BY-NC-ND](#) © 2014 MME BVBA / Follow [@MME_IT](#) on Twitter and ask for our cheat sheet, containing all solutions! / Ne". The Windows taskbar at the bottom of the screen shows the search bar, Start button, and various pinned icons.

Step 5

You are navigated to another page. An alert is displayed with the value **1**.

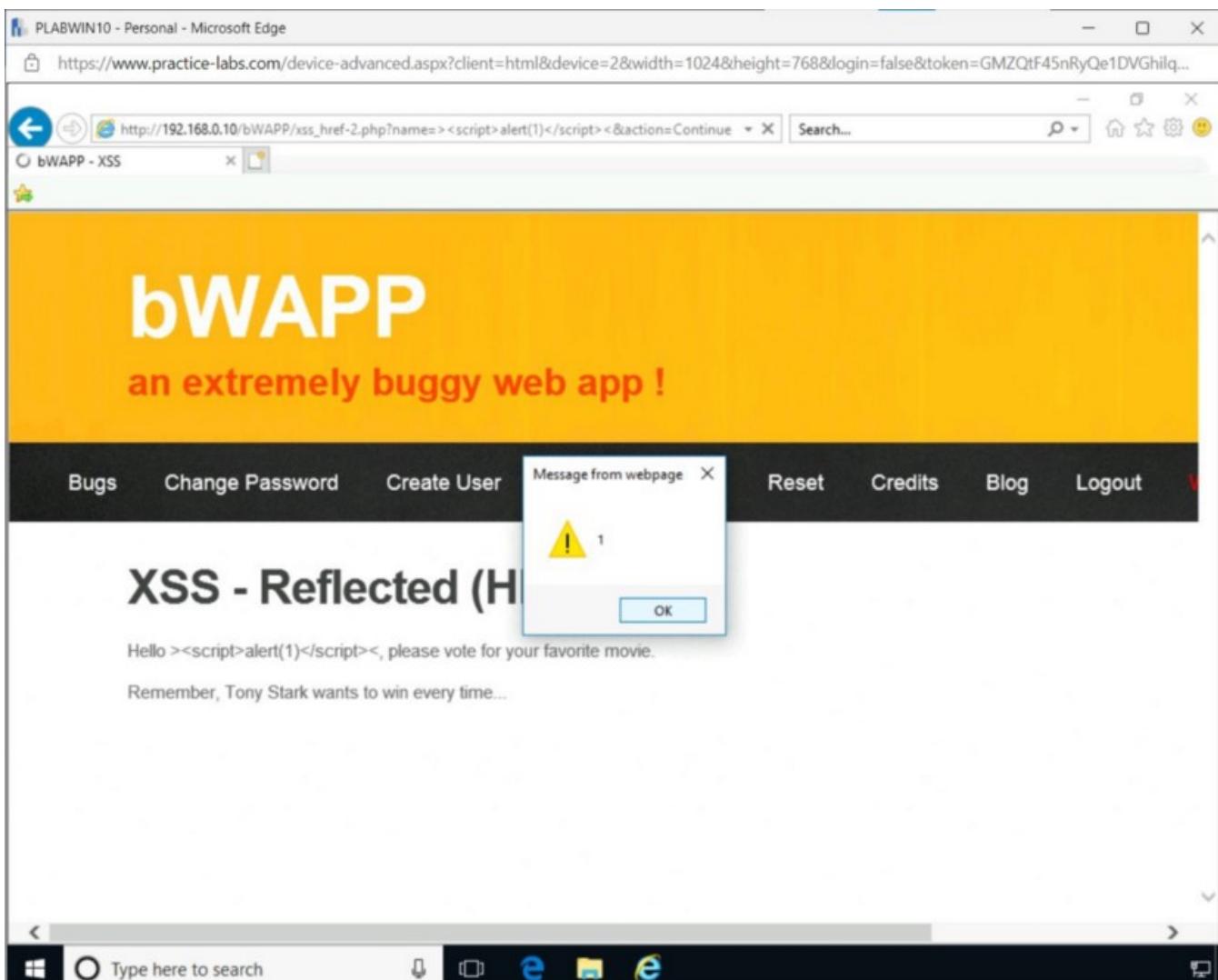
Click **OK**.



Step 6

The alert box continues to display even after you click **OK**.

Alert: If you wanted to return to the page before, you will have to be quick to be able to navigate to another page via the back button.



Exercise 3 — Web Application Hacking Methodology

Every attack follows a specific methodology, nothing but a series of defined steps. Similarly, in web application hacking, hackers follow a methodology covering web applications from all aspects.

In this exercise, you will learn about the web application hacking methodology.

Learning Outcomes

After completing this exercise, you will have further knowledge of:

- Web Application Hacking Methodology Concepts

The Concept of Web Application Hacking Methodology

Attackers use a series of steps to attack web applications. These series of steps are known as web application hacking methodology. The key steps in this methodology are:

- **Footprinting web infrastructure:** is performed using various methods, such as Whois lookup, DNS interrogation, hidden content discovery, detecting firewalls and load balancers, and port scanning.
- **Analyzing target web applications:** is performed using various methods, such as identifying the server-side and endpoint technologies. It also includes identifying server-side functionalities, files and directories, web application vulnerabilities, and identifying the attack surface.
- **Bypassing client-side controls:** is performed by attacking hidden form fields and web browser extensions, identifying the source code flaws by reviewing them and evading XSS filters.
- **Attacking authentication methods:** is performed by bypassing authentication, performing username enumeration, cookie exploitation, password and session attacks.
- **Attacking authorization schemes:** is compromising an attack and then escalating privileges. It is also performed by manipulating HTTP requests.
- **Attacking access controls:** includes an attack on applications using different user accounts, attacking a low-privileged user account, then escalating privileges, and attacking the static resources mentioned in the URL
- **Attacking session management controls:** are attacks such as man-in-the-middle, session replay, and session hijacking attacks.
- **Conducting injection attacks:** are attacks such as OS command, SMTP, SQL, LDAP, XPath, and file injection.
- **Conducting application logic flaw attacks:** includes logic flows, programming errors, and misconfiguration exploitation.
- **Conducting shared environment attacks:** includes attacks on access methods and launching attacks by running scripts from one application to another one.
- **Conducting database connectivity attacks:** includes various attacks, such as connection pool DoS, connection string injection, and Connection String Parameter Pollution (CSPP) attacks.
- **Conducting web application client attacks:** includes various attacks, such as cross-site scripting, HTTP header injection, request forgery, redirection, frame

injection, and session fixation.

- **Conduct web service attacks:** includes various attacks, such as SOAP and XML injection, WSDL probing, application logic, and database attacks.

Exercise 4 — Footprint and Analyze Web Infrastructure

The web application attack methodology defines an attacker who starts the attack by footprinting and analyzing the web infrastructure and applications. It is required that an attacker has information regarding the web infrastructure to devise a correct plan for proceeding further.

In this exercise, you will learn to footprint and analyze the web infrastructure.

Learning Outcomes

After completing this exercise, you will be able to:

- Web Applications Enumeration Using Wafwoof
- Perform website Enumeration using Nmap
- Detect the IP addresses Using the Host Command
- Information Gathering Using Legion

Task 1 — Enumerate web Applications Using Wafw00f

A web application enumeration can be prevented using a web Application Firewall (WAF). The wafwoof tool helps determine whether the web application is behind a WAF.

A hacker may want to analyze the web application before launching an attack. wafwoof is a useful tool to determine whether the application is behind a firewall. Accordingly, an attacker may decide the next course of action.

In this task, you will learn to use wafwoof. To do this, perform the following steps:

Step 1

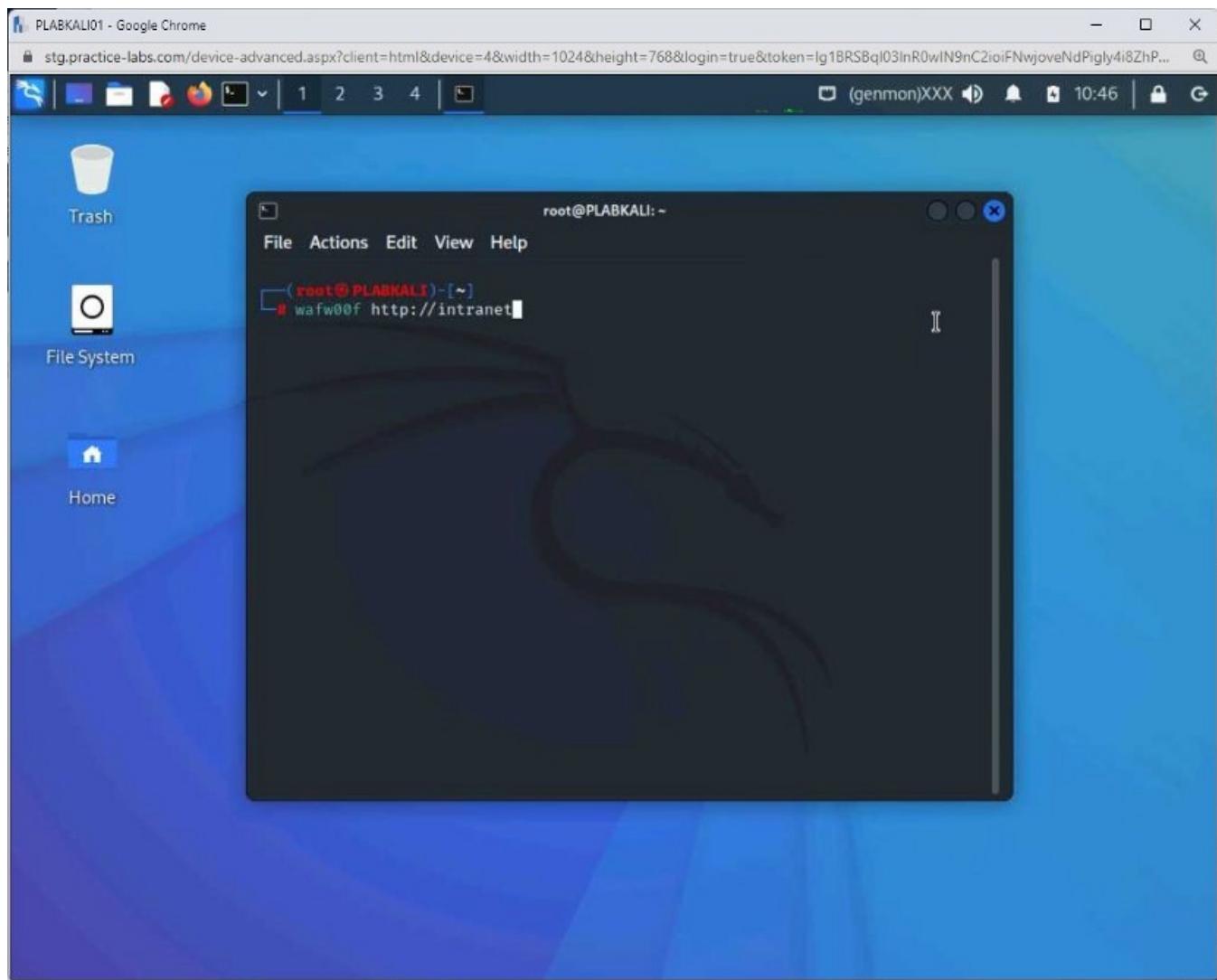
Ensure you have powered on all the devices listed in the introduction and connect to **PLABKALI01**.

You will attempt to determine whether a web application is behind the **web Application Firewall (WAF)**.

You will use a tool named **wafwoof** for this purpose. Type the following command:

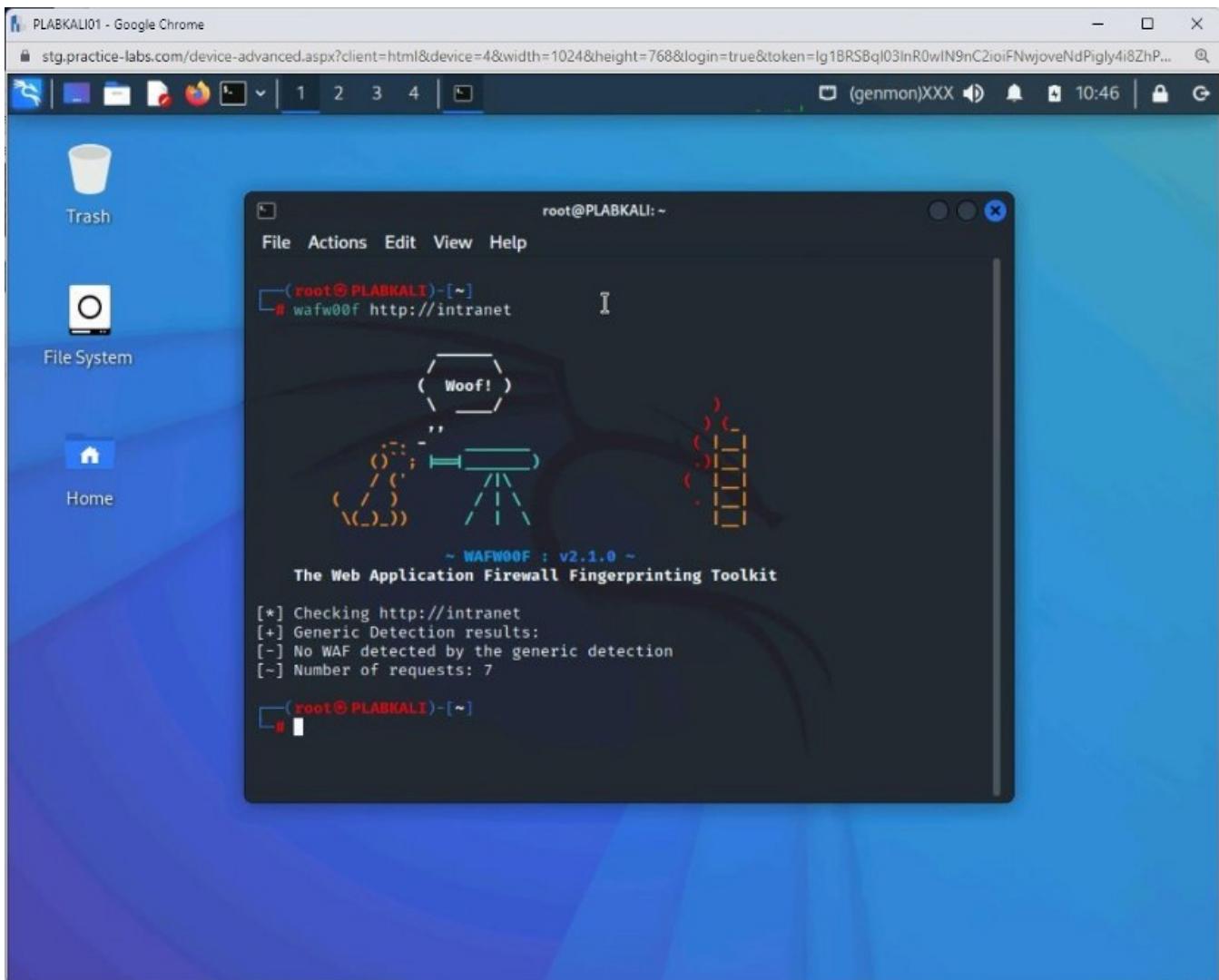
```
wafw00f http://intranet
```

Press **Enter**.



Step 2

Notice that **wafwoof** did not detect a **WAF**.



Keep the terminal window open.

Task 2 — Perform Website Enumeration using Nmap

There are different methods to enumerate a website. For example, you can use a manual method using a web browser. You can try:

<http://www.plab.com/admin>

After the URL, you can add a directory name, such as admin. You are likely to get one of the following responses:

- **200** – OK
- **401** – Unauthorized
- **402** – Payment Required

- **403** – Forbidden
- **404** – Not Found

If the admin does not return a **404** error but something else, such as **403**, it indicates clearly that this directory exists.

You can also enumerate a website using Nmap, which provides several scripts to enumerate different types of websites, such as WordPress or Drupal.

In this task, you will perform website enumeration using Nmap. To do this, perform the following steps:

Step 1

Connect to **PLABKALIo1**.

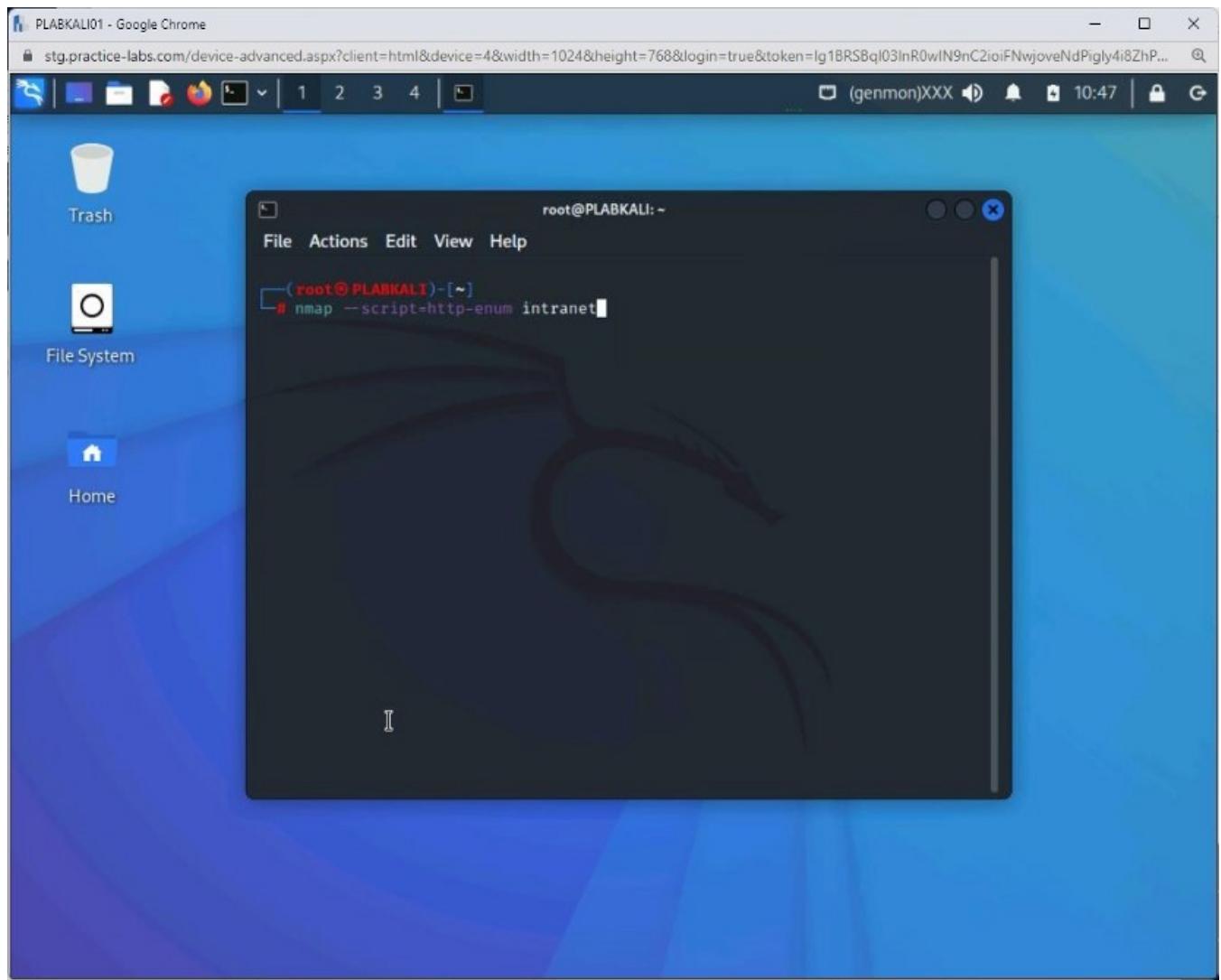
Clear the screen by entering the following command:

```
clear
```

To perform a website enumeration, type the following command:

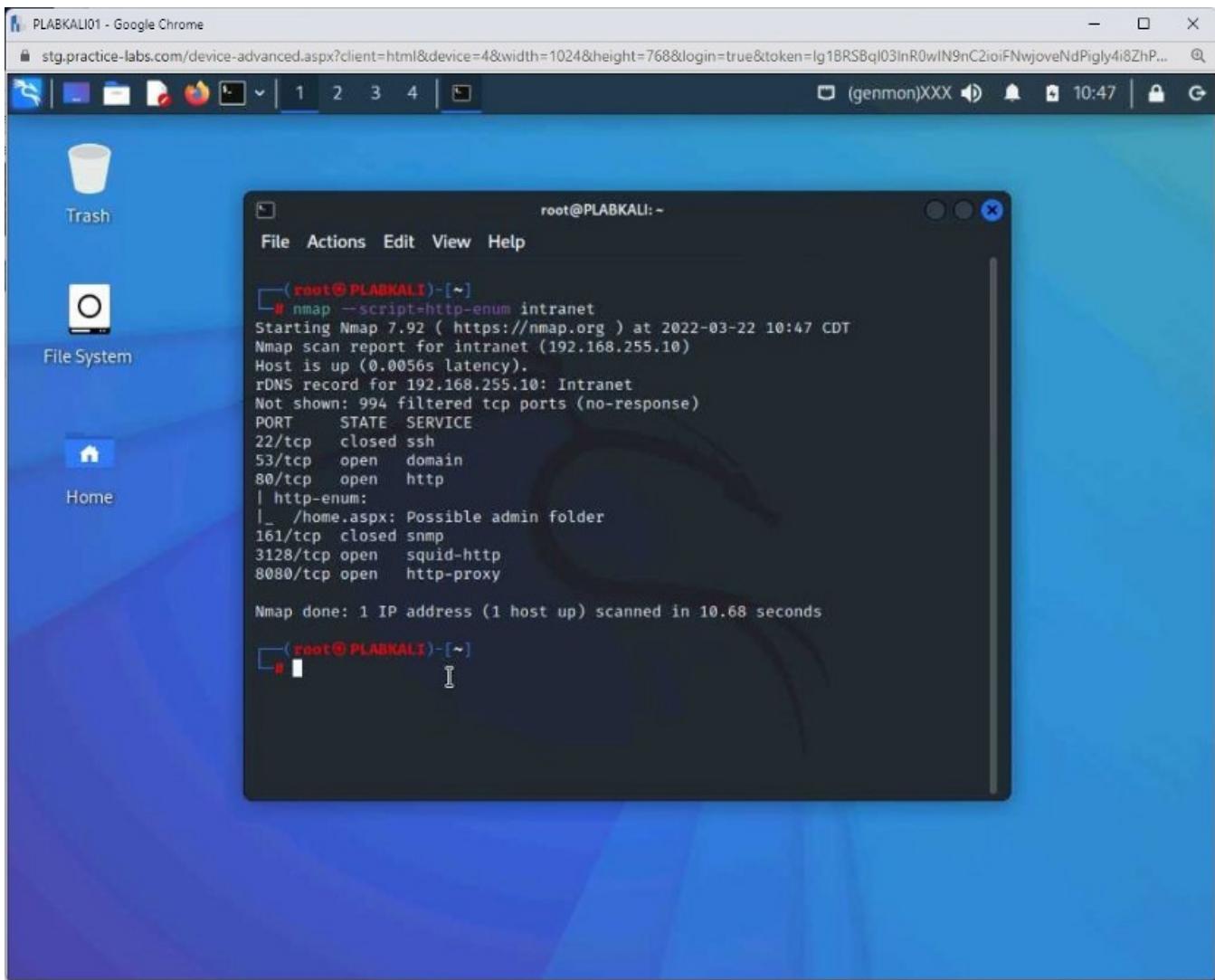
```
nmap --script=http-enum intranet
```

Press **Enter**.



Step 2

Notice the output. It has been able to list the open ports and a possible admin folder.



Keep the terminal window open.

Task 3 — Detect the IP addresses Using the Host Command

Using the host command, an attacker can detect the IP addresses of the web application. If more than one IP address is assigned to the web application, it can be detected. It is also useful to determine if the web application uses a load balancer, taking many web applications. If an attacker is planning to conduct a DoS attack, this command can determine a load balancer.

To use the host command, perform the following steps:

Step 1

Connect to **PLABKALI01**.

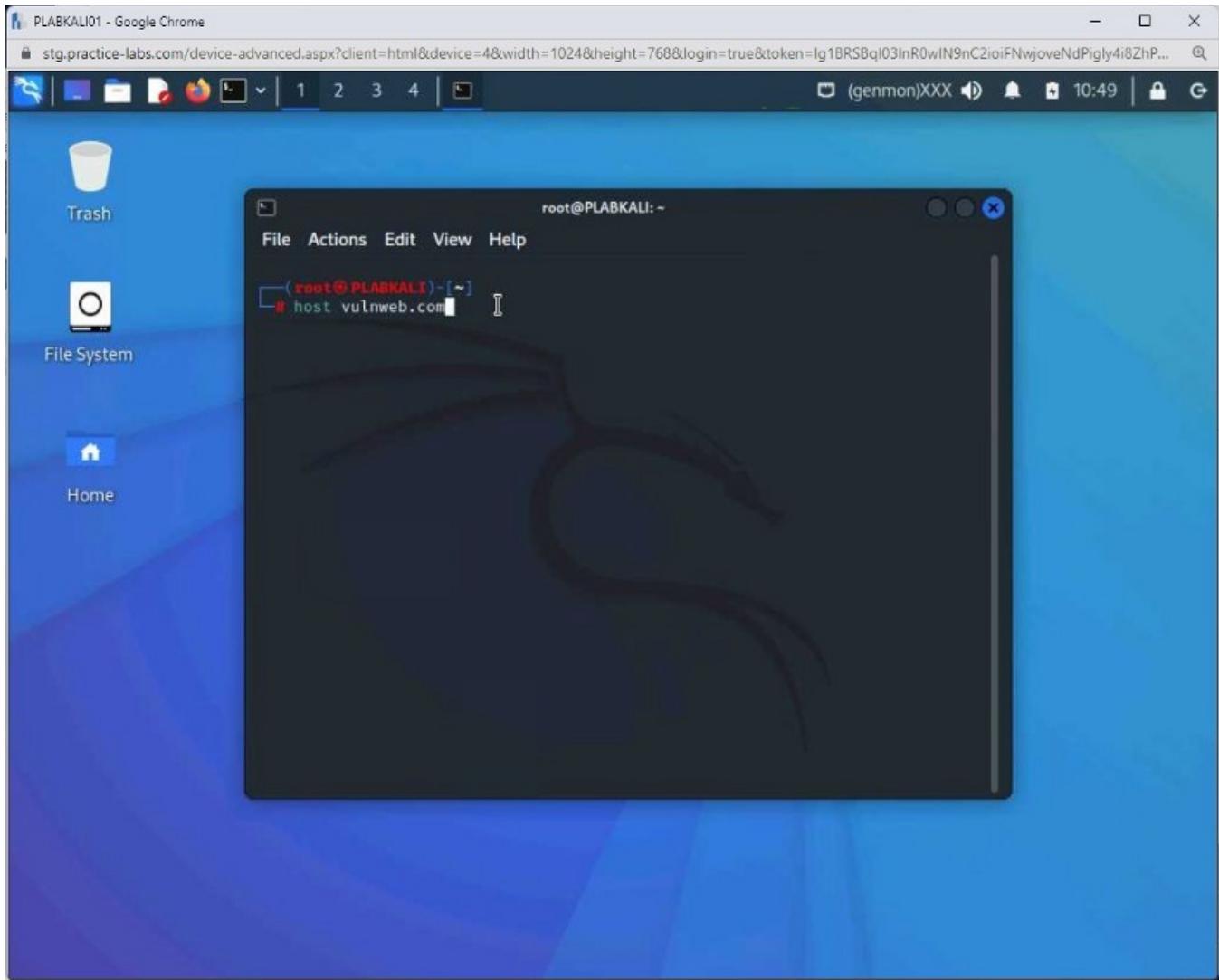
Clear the screen by entering the following command:

```
clear
```

To determine the IP address(es), type the following command:

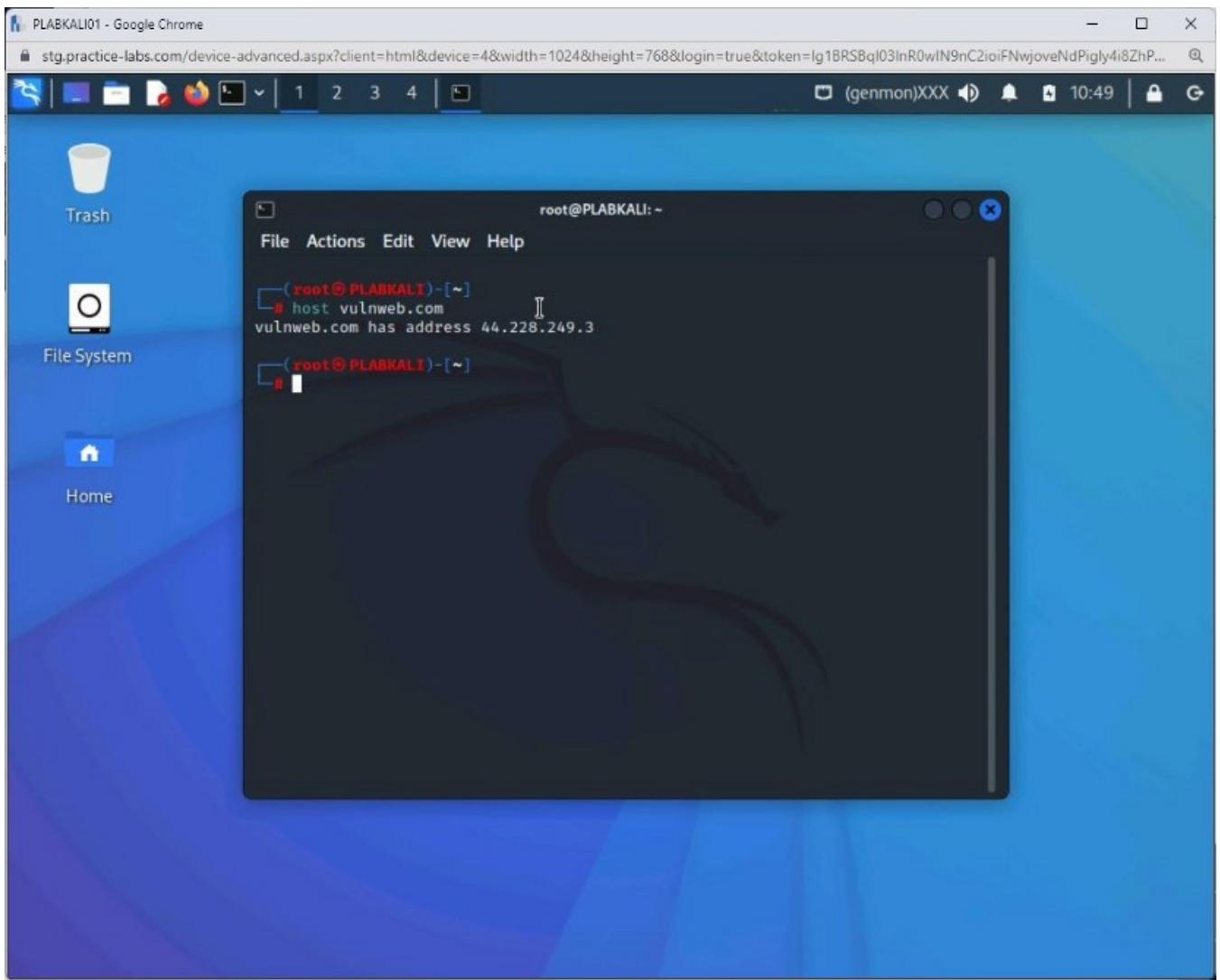
```
host vulnweb.com
```

Press **Enter**.



Step 2

The output shows only one IP address, which means the web application is not using a load balancer.



Keep the terminal window open.

Task 4 — Information Gathering Using Legion

Legion is a Python-based tool with several capabilities, such as vulnerability scanning and information gathering. It uses several tools, such as Nmap, Telnet, and Nikto.

To perform information gathering using Legion, perform the following steps:

Step 1

Connect to **PLABKALI01**.

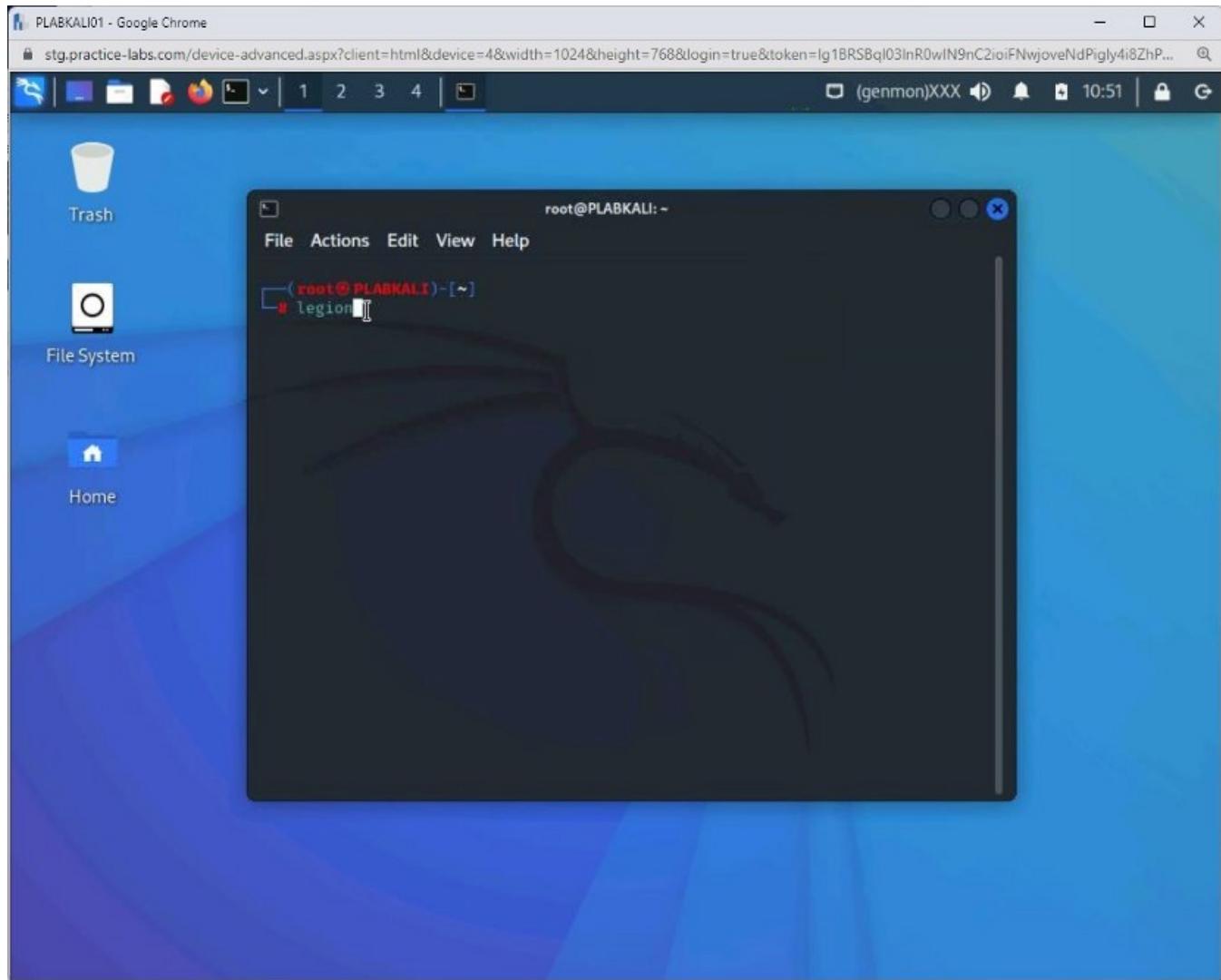
Clear the screen by entering the following command:

```
clear
```

To start Legion, type the following command:

legion

Press **Enter**.

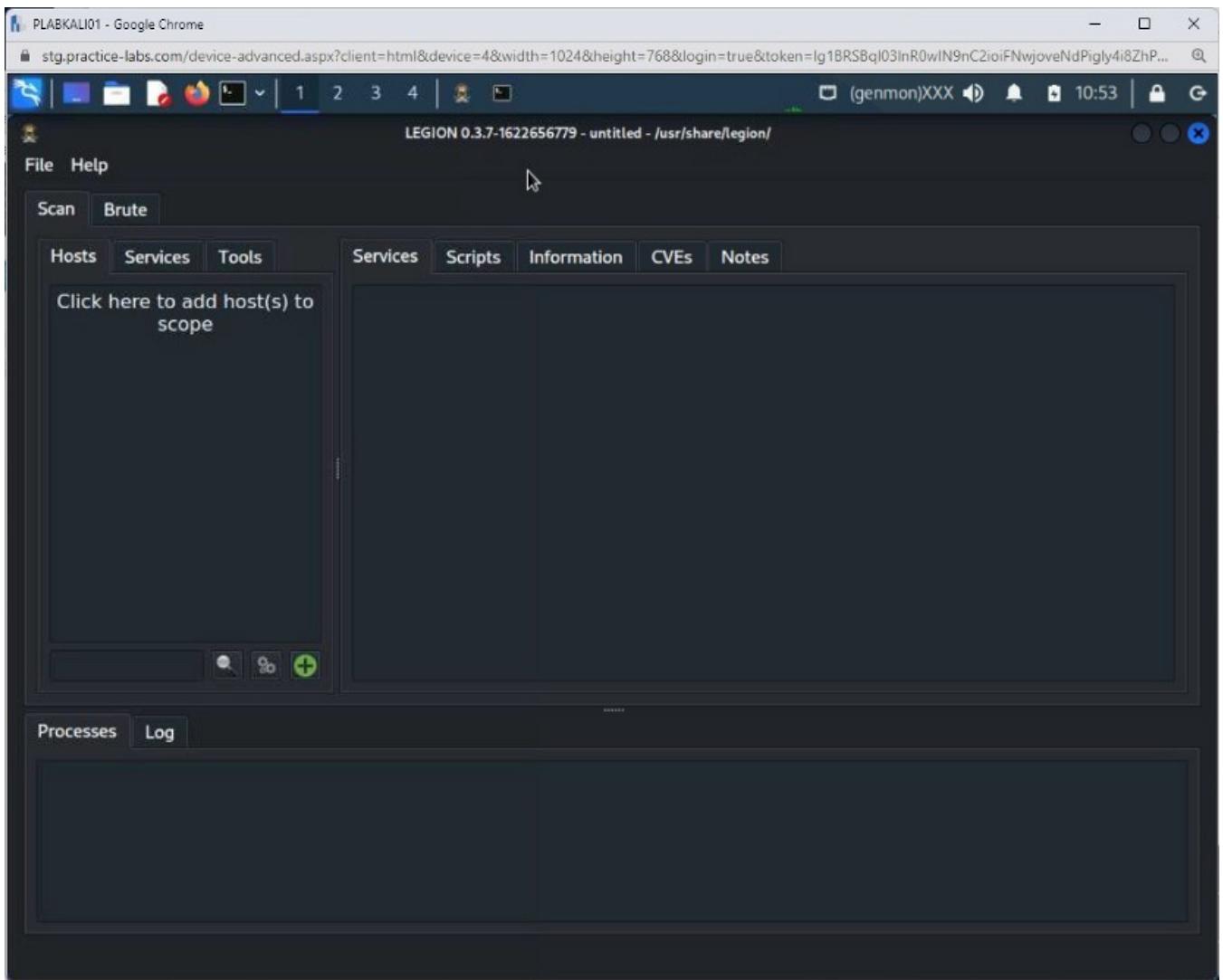


Step 2

The graphical interface of **Legion** is displayed. On the **Hosts** tab in the left pane, click in the **Click here to add host(s) to scope** textbox.

Notice that the right pane currently has four tabs: **Services**, **Scripts**, **Information**, and **Notes**.

Note: You may have to manually resize the screen inside of the lab environment.

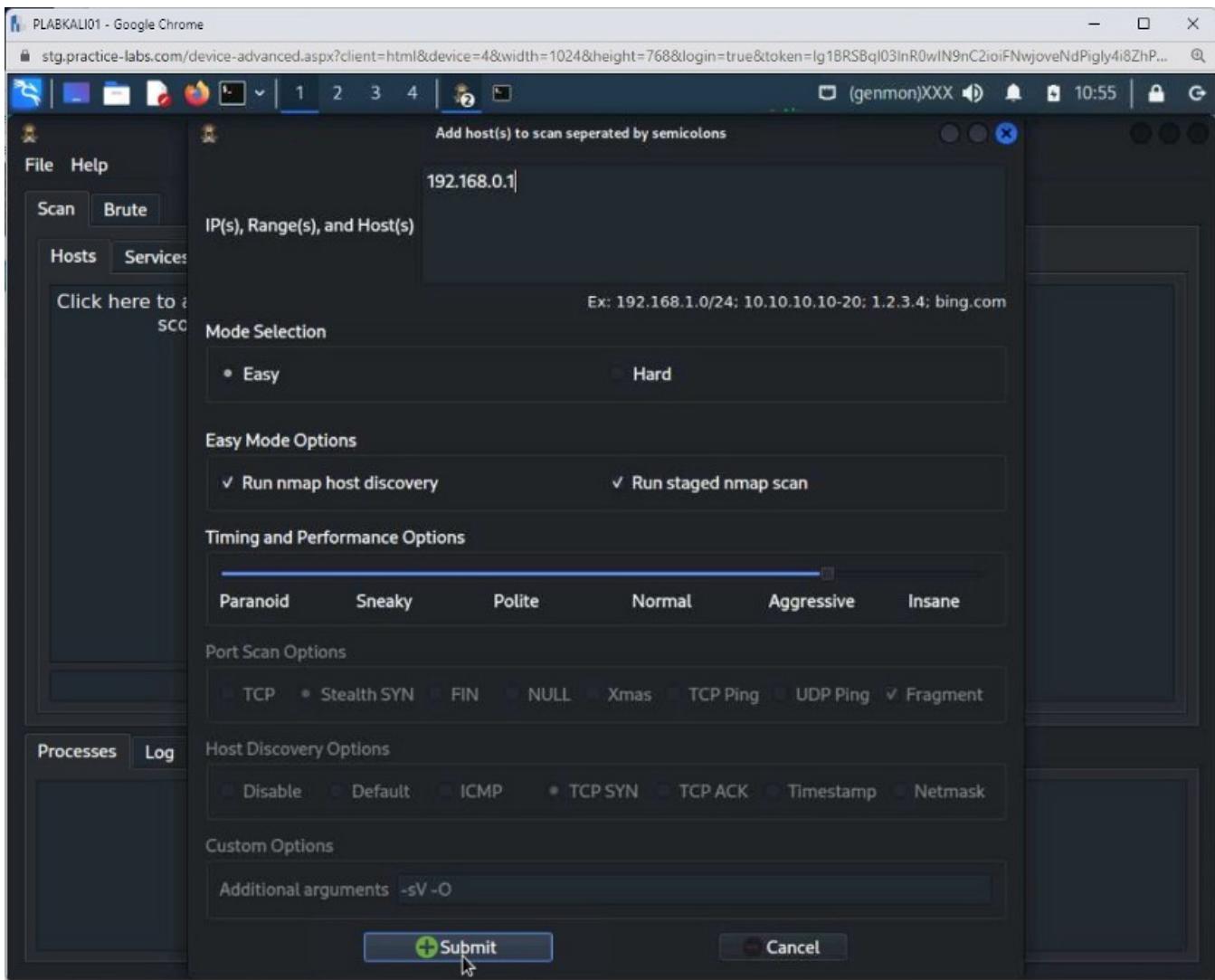


Step 3

The **Add host(s) to scope** dialog box is displayed. In the **IP Range** text box, type the following IP address:

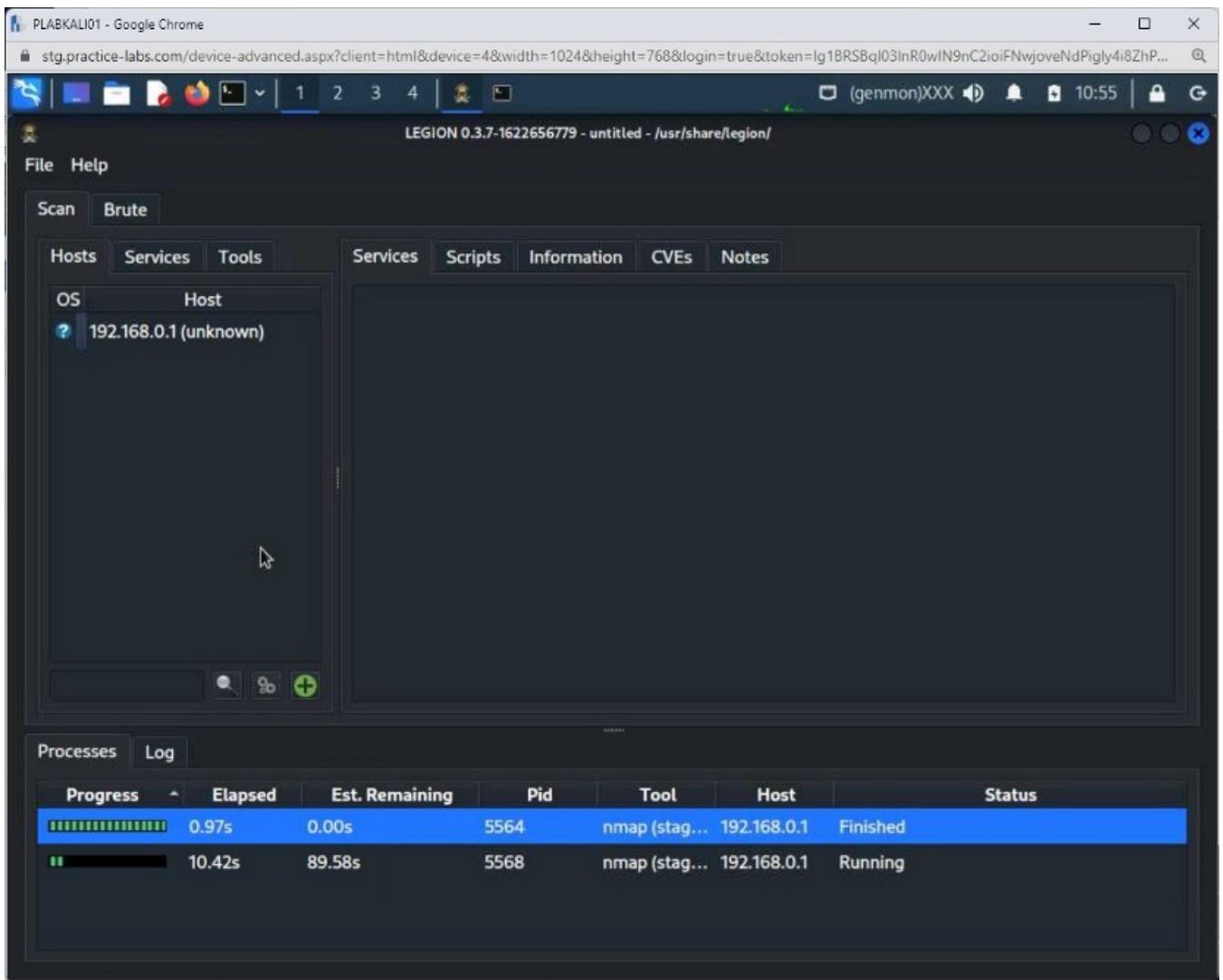
192.168.0.1

Keep the default selection of the remaining options and click **Submit**.



Step 4

The bottom pane shows the scan progress, which takes a while to finish.



Step 5

Notice that the new tabs are now being added to the right pane. Open ports and running services list are now populated in the Services tab.

It now only shows the open ports, but also the version numbers. This information can be a goldmine for an attacker.

The screenshot shows the LEGION 0.3.7 interface running in a browser window. The title bar reads "PLABKALI01 - Google Chrome" and the address bar shows "stg.practice-labs.com/device-advanced.aspx?client=html&device=4&width=1024&height=768&login=true&token=lg1BRSBql03lnR0wIN9nC2ioiFNwjoveNdPigly4i8ZhP...". The main window has tabs for "Scan" and "Brute", with "Scan" selected. In the "Scan" tab, there are three sub-tabs: "Hosts", "Services", and "Tools", with "Services" selected. The left pane shows a table with columns "OS" and "Host", where "192.168.0.1 (unknown)" is listed. The right pane shows a table of services with columns "Port", "Protocol", "State", "Name", and "Version". Services listed include domain (Simple DNS Plus), kerberos-sec (Microsoft Windows Kerberos), msrpc (Microsoft Windows RPC), netbios-ns (Microsoft Windows netbios-ns), netbios-ssn (Microsoft Windows netbios-ssn), snmp (SNMPv1 server (public)), ldap (Microsoft Windows Active Directory LD...), microsoft-ds (microsoft-ds), kpasswd5 (kpasswd5), ncacn_http (Microsoft Windows RPC over HTTP 1.0), and tcpwrapped (tcpwrapped). The bottom pane shows a table for "Processes" and "Log" with columns "Progress", "Elapsed", "Est. Remaining", "Pid", "Tool", "Host", and "Status". There are four entries: one for smbenum (status: Finished, elapsed: 1.93s), one for smbenum (status: Finished, elapsed: 2.52s), one for snmp-defa... (status: Running, elapsed: 21.30s), and one for nmap (status: Running, elapsed: 11.32s).

Step 6

Click on the **Information** tab in the right pane. Notice that it displays information in three sections: **Host Status**, **Addresses**, and **Operating System**.

Note: You can also click on each tab after the notes section to see the results of each scan.

PLABKALI01 - Google Chrome

stg.practice-labs.com/device-advanced.aspx?client=html&device=4&width=1024&height=768&login=true&token=lg1BRSBql03lnR0wIN9nC2ioiFNwjoveNdPigly4i8ZhP... (genmon)XXX

LEGION 0.3.7-1622656779 - untitled - /usr/share/legion/ 10:58

File Help

Scan Brute

Hosts Services Tools Services Scripts Information CVEs Notes smbenum (445/tcp) snmp-default (161)

OS	Host
?	192.168.0.1 (unknown)

Host Status	Addresses	Location
State: up	IPv4: 192.168.0.1 IPv6: unknown	Country Code: unknown
Open Ports: 14	MAC: 00:15:5D:60:64:47	City: unknown
Closed Ports: 0	Vendor: Microsoft	
Filtered Ports: 65521	ASN: unknown	Latitude: unknown
	ISP: unknown	Longitude: unknown

Operating System	
Name: unknown	
Accuracy: NaN	

Processes Log

Progress	Elapsed	Est. Remaining	Pid	Tool	Host	Status
[Progress Bar]	1.93s	0.00s	5573	smbenum (...)	192.168.0.1	Finished
[Progress Bar]	2.52s	0.00s	5608	smbenum (...)	192.168.0.1	Finished
[Progress Bar]	33.30s	66.70s	5578	snmp-defa...	192.168.0.1	Running
[Progress Bar]	23.32s	76.68s	5607	nmap (stag...)	192.168.0.1	Running