CySA+ Lab Series

Lab 02: Web Application Scanning

Document Version: 2022-10-10

Material in this Lab Aligns to the Following	
CompTIA CySA+ (CS0-002) Exam Objectives	1.2 - Given a scenario, utilize threat intelligence to support organizational security 1.3 - Given a scenario, perform vulnerability managemenactivitieses 1.6 - Explain the threats and vulnerabilities associated with operating in the cloud 1.7 - Given a scenario, implement controls to mitigate attacks and software vulnerabilities 2.1 - Given a scenario, apply security solutions for infrastructure management 4.1 - Explain the importance of the incident response process 4.3 - Given an incident, analyze potential indicators of compromise 4.4 - Given a scenario, utilize basic digital forensics techniques
All-In-One CompTIA CySA+ Second Edition ISBN-13: 978-1260464306 Chapters	2: Threat Intelligence in Support of Organizational Security 3: Vulnerability Management Activities 6: Threats and Vulnerabilities Associated with Operating in the Cloud 7: Mitigating Controls for Attacks and Software Vulnerabilities 8: Security Solutions for Infrastructure Management 15: The Importance of the Incident Response Process 17: Analyze Potential Indicators of Compromise 18: Utilize Basic Digital Forensics Techniques

Contents

Introduction 3
Objective 3
Lab Topology 4
Lab Settings 5
1 Allowing Web Server Access Through the Firewall 6
1.1 Opening HTTP to External Traffic 6
2 Scanning a Website for Vulnerabilities 12
2.1 Test the Website for Misconfigurations Using Nikto 12
2.2 Test the Website for Vulnerabilities using OWASP ZAP 16
2.3 Exploiting the Vulnerable Website After OWASP ZAP Discovery 22

Introduction

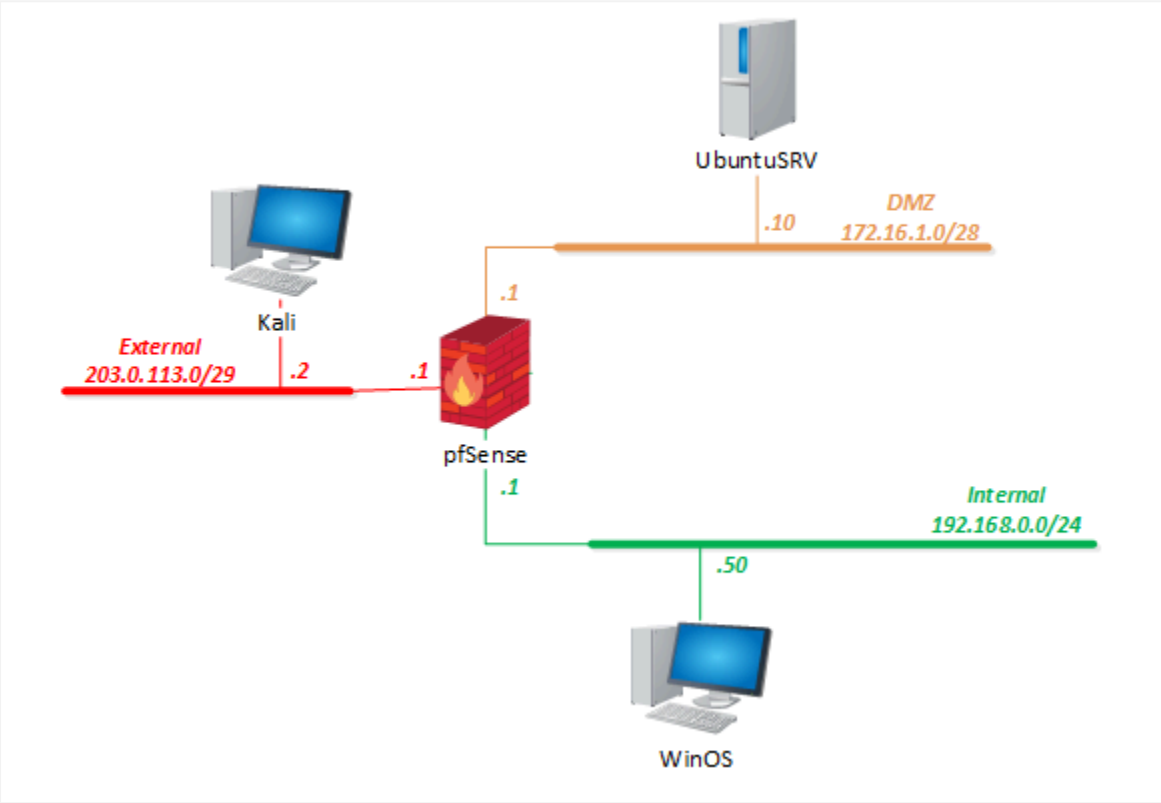
Web application penetration testing is a complete field within the penetration testing discipline. All of the action takes place at the application level. Many of the same types of tactics that are used for a general penetration test also apply to web application testing. In this lab, you will be using the *Kali* machine to attack the *Ubuntu* machine.

Objective

In this lab, you will be conducting web application scans using various tools. You will be performing the following tasks:

- Scan websites for vulnerabilities with *Nikto Web Server Scanner*
- Scan websites for vulnerabilities with *OWASP ZAP*

Lab Topology



Lab Settings

The information in the table below will be needed in order to complete the lab. The task sections below provide details on the use of this information.

Virtual Machine	IP Address	Account	Password
WinOS (Server 2019)	192.168.0.50	Administrator	NDGlabpass123!
MintOS (Linux Mint)	192.168.0.60	sysadmin	NDGlabpass123!
OSSIM (Alien Vault)	172.16.1.2	root	NDGlabpass123!
UbuntuSRV (Ubuntu Server)	172.16.1.10	sysadmin	NDGlabpass123!
Kali	203.0.113.2	sysadmin	NDGlabpass123!
pfSense	203.0.113.1 172.16.1.1 192.168.0.1	admin	NDGlabpass123!

1 Allowing Web Server Access Through the Firewall

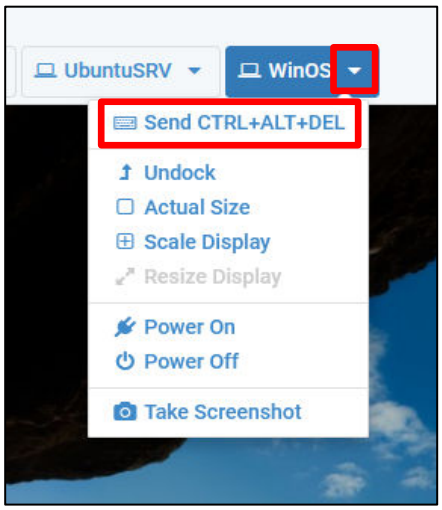
One of the main functions of a DMZ is to allow specific external traffic to use an organization's resources, such as web servers, email servers, VoIP servers, etc., without being able to access any internal LAN resources.

In order to allow external traffic to flow into the DMZ, the firewall must allow the traffic through. This entails opening specific ports on the firewall to be directed to specific hosts and ports.

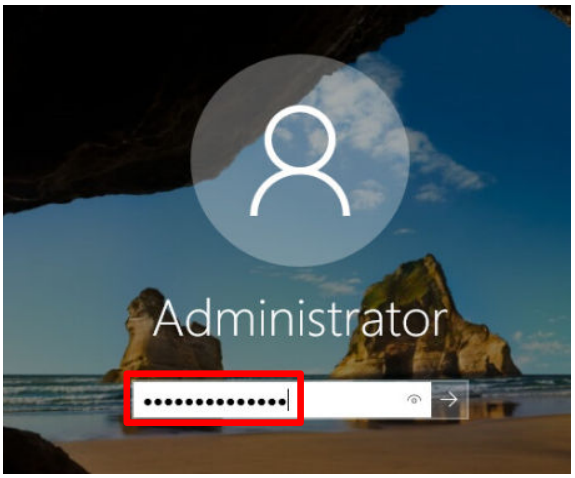
> The combination of a host IP address with a port number is called a ***socket***.

1.1 Opening HTTP to External Traffic

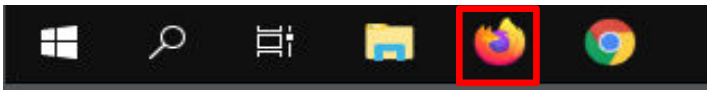
1. Change the focus to the **WinOS** computer.
2. Bring up the login window by sending a Ctrl + Alt + Delete. To do this, click the **WinOS** dropdown menu and click **Send CTRL+ALT+DEL**.



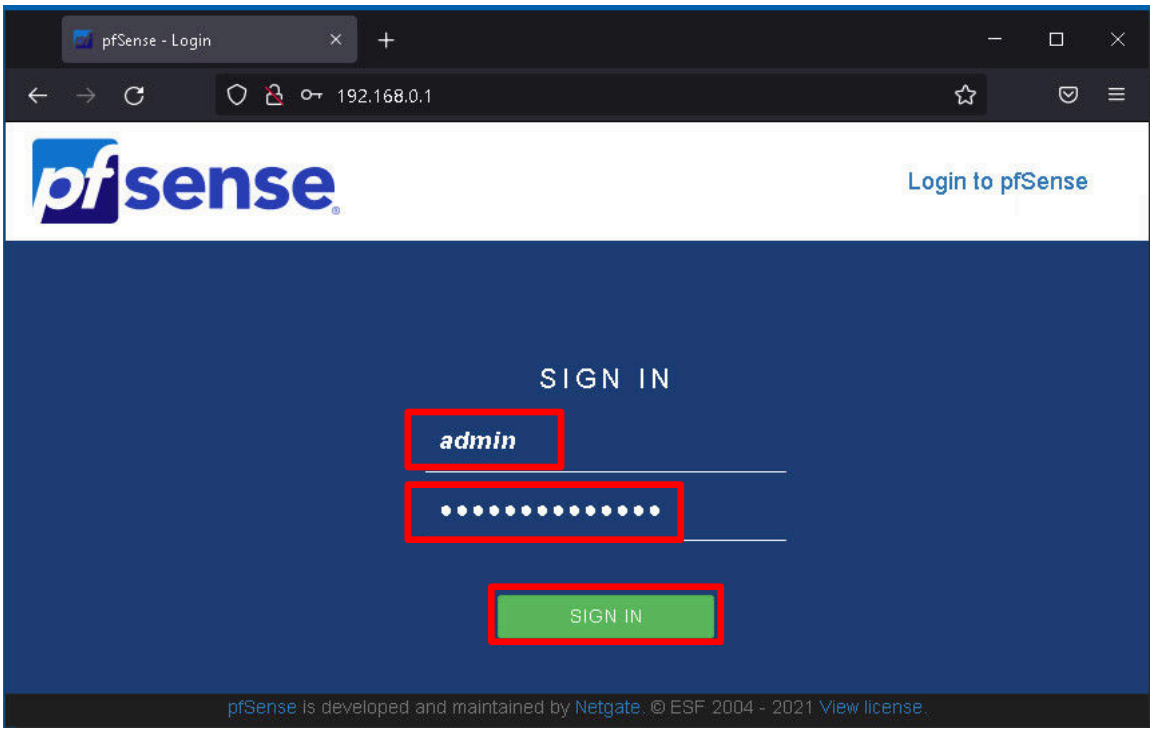
3. Log in as *Administrator* using the password: `NDGlabpass123!`



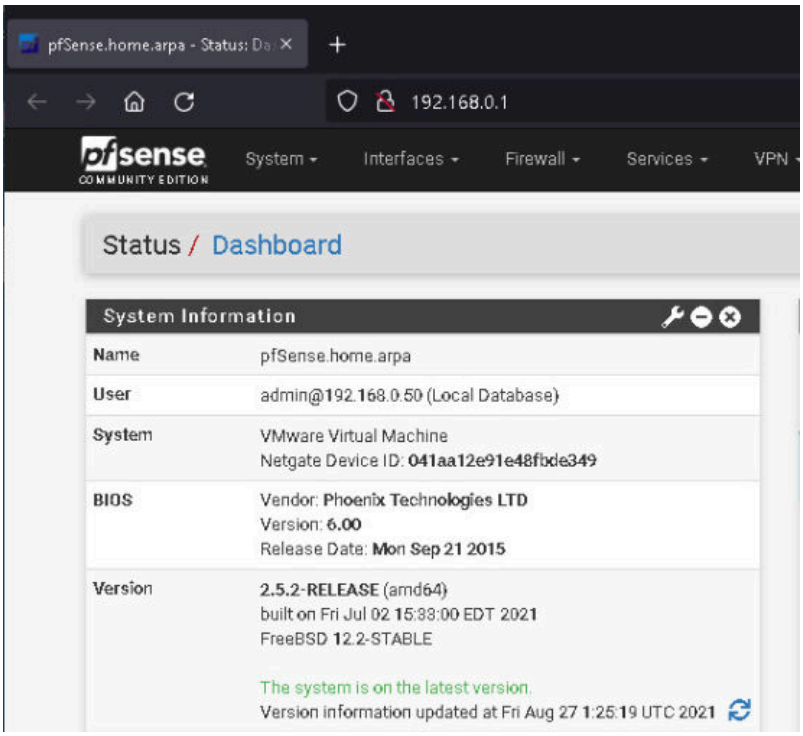
4. Click on the **Firefox** browser icon in the taskbar to open a web browser.



5. In the address bar of the browser, type `192.168.0.1`, the IP address of the *pfSense* server.
6. Log in as `admin` using the password `NDGlabpass123!` and click the **SIGN IN** button.

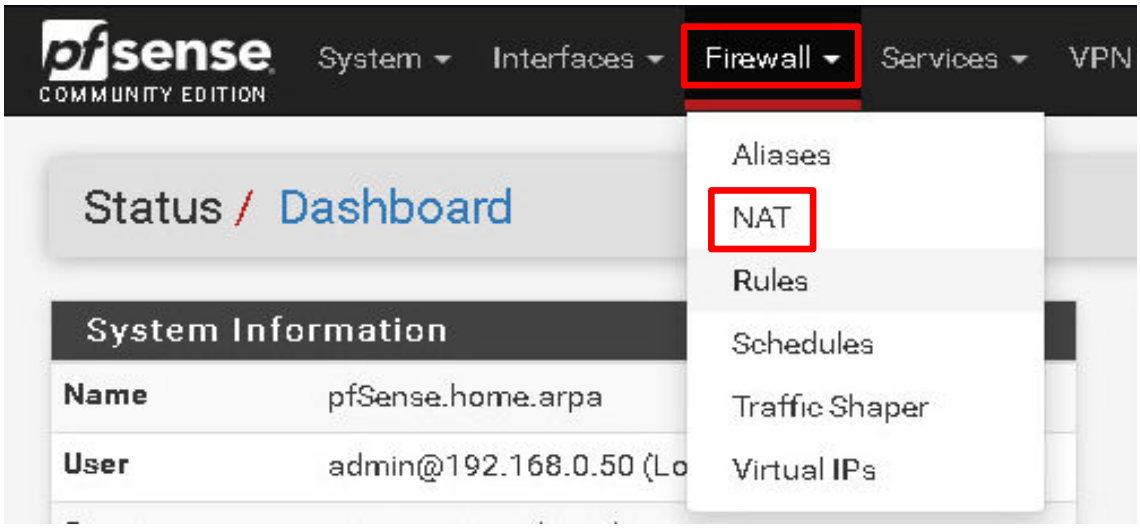


7. You will be presented with the Dashboard for the *pfSense* firewall.

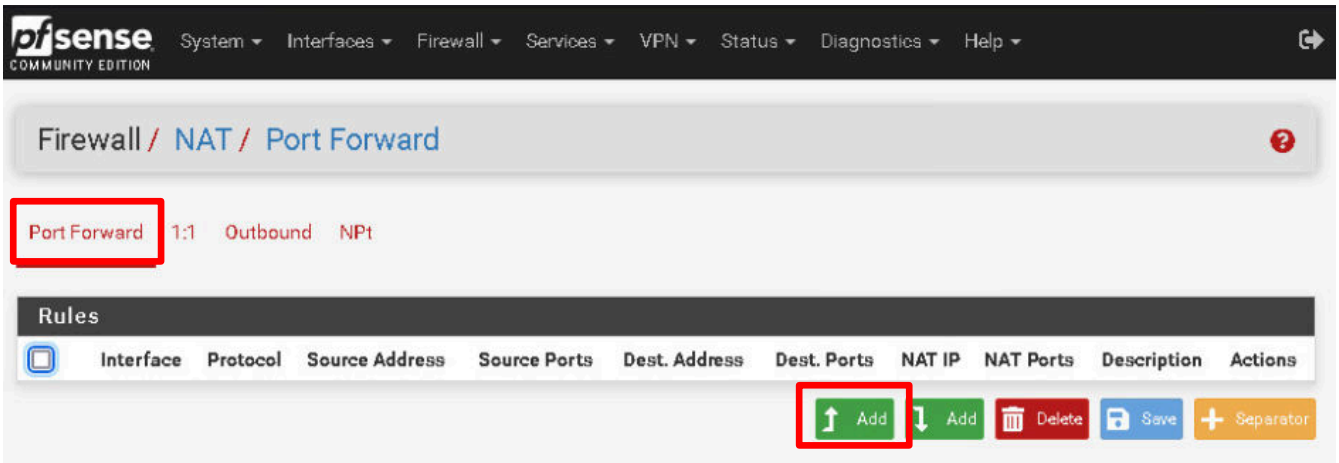


We need to add a NAT entry to forward traffic on Port 80 from the External network forwarded to 172.16.1.10 (the **UbuntuSRV** computer) on DMZ.

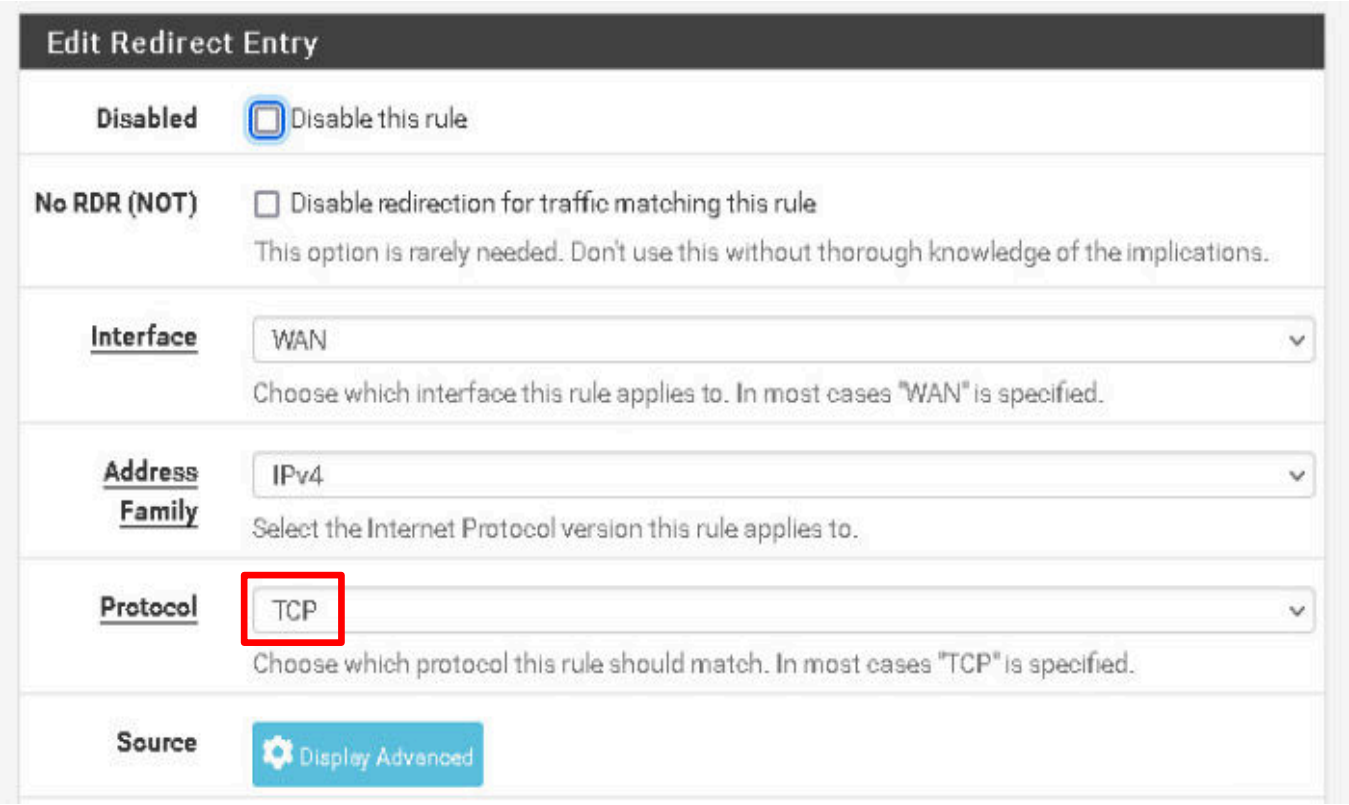
8. Click on the **Firewall** menu item and then click on **NAT.**



9. Make sure the **Port Forward** item is selected and click the **Add to Top** button.

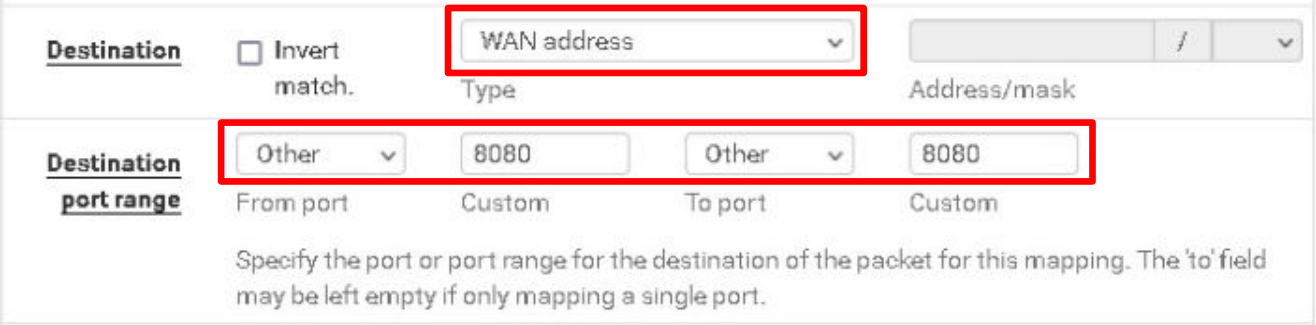


10. In the *Edit Redirect Entry***,** use the list arrow to change the Protocol to **TCP** if it's not already set.

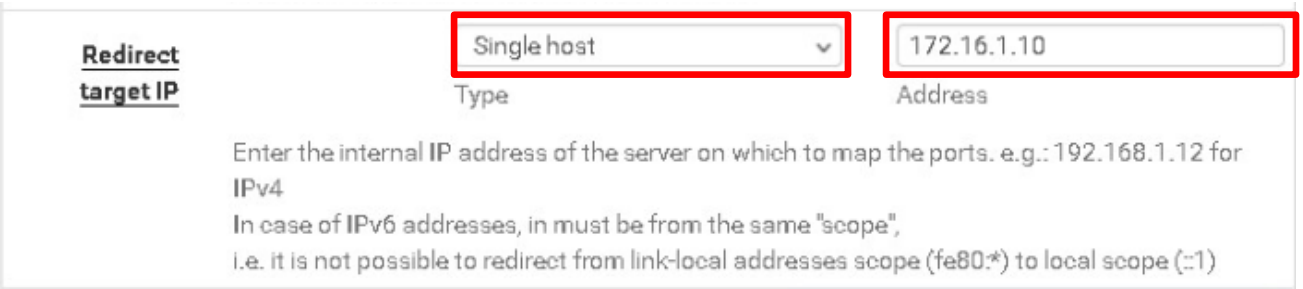


11. Scroll down to the *Destination* section. Then use the list arrow to select **WAN Address** if not already selected.

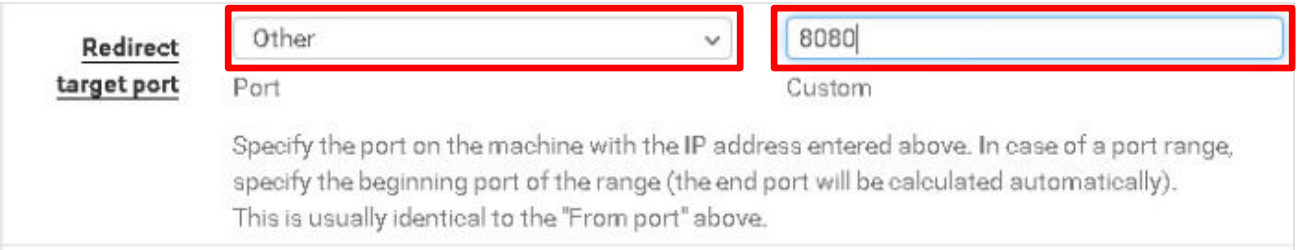
    In the *Destination Port Range*, click the list arrow in the *From Port* and select **Other**, if not already selected, then type `8080` for the port in the *Custom* box. Repeat for the *To Port* and its *Custom* port.



12. In the *Redirect Target IP* section, click on the list arrow and change the option to **Single Host**, if not already selected, then change the *Address* to `172.16.1.10`.

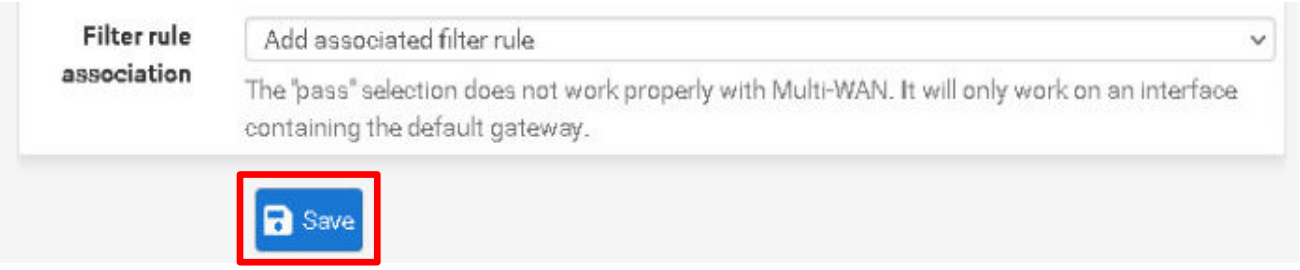


13. In the *Redirect Target Port* section, click on the list arrow and change the option to **Other** if it is not already selected, and change the *Custom* port to `8080`.

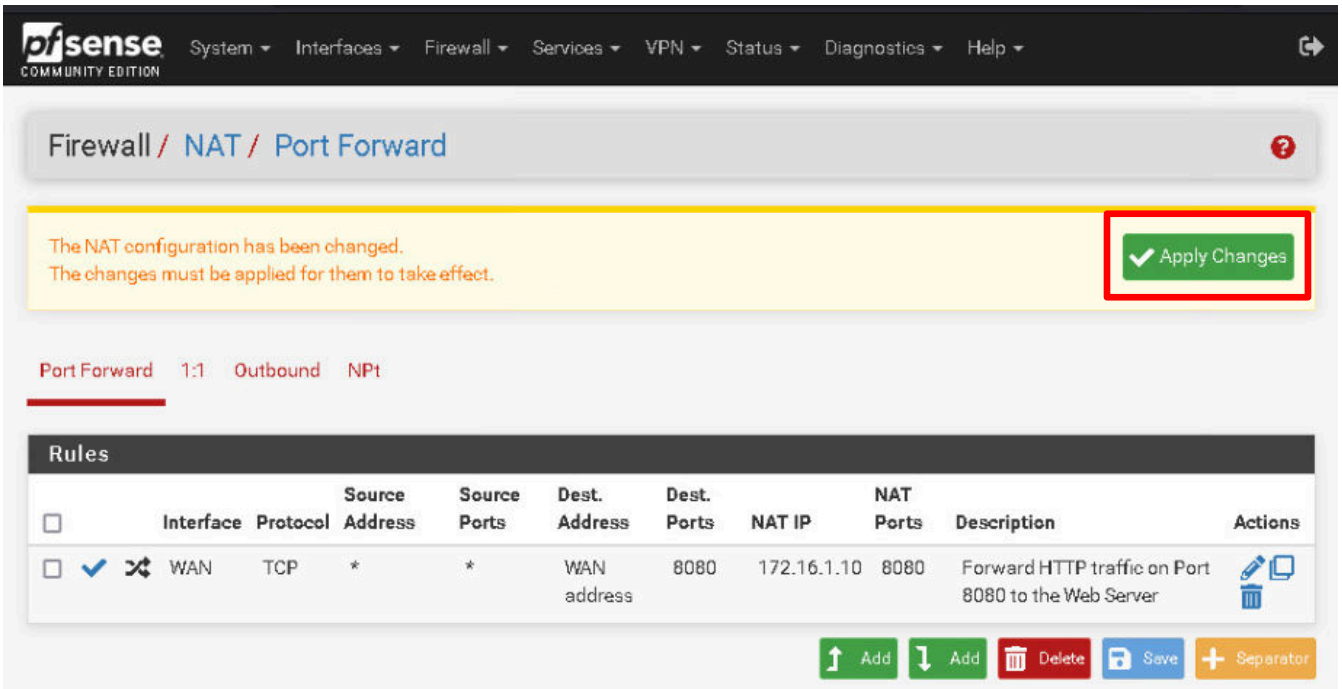


14. In the *Description* section, change the description to `Forward HTTP Traffic on Port 8080 to the Web Server.`

15. Scroll to the bottom of the web page and click the **Save** button at the bottom of the window.



16. Click the **Apply Changes** button.



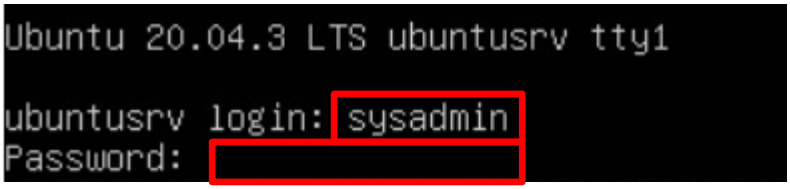
17. Close the *Firefox* browser.

2 Scanning a Website for Vulnerabilities

2.1 Test the Website for Misconfigurations Using Nikto

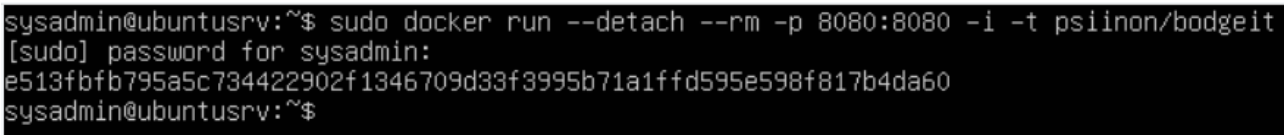
Nikto is used to test for website misconfigurations that could allow an attacker to compromise the web server. Once you have finished, an HTML file reporting any problems or vulnerabilities that were discovered will be generated. You can find more information on *Nikto* at https://cirt.net/Nikto2.

1. Set the focus to the **UbuntuSRV** computer.
2. Log in as `sysadmin` using the password: `NDGlabpass123!`

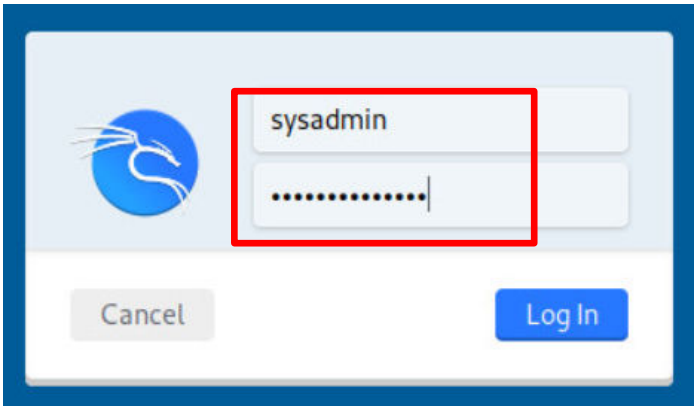


3. Begin by starting the *Bodgeit* website inside a *Docker* container. A Docker container is a form of virtualization that utilizes the OS in order to allow software to run inside of an isolated, virtual instance in any Linux environment. In order to start the *Bodgeit* docker container, type the following command using the password `NDGlabpass123!` when prompted:

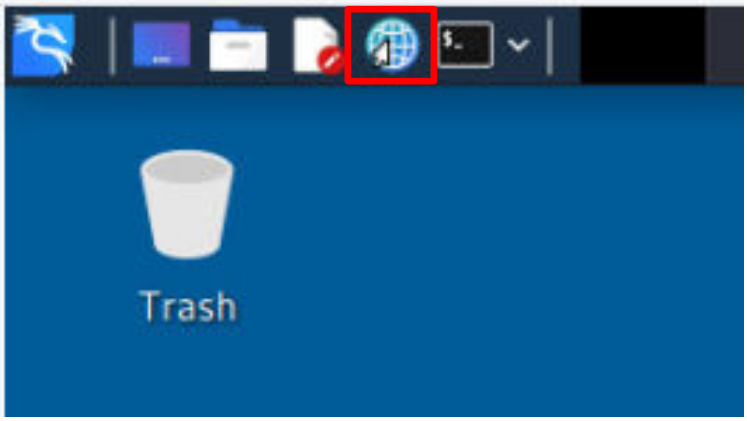
```
sudo docker run --detach --rm -p 8080:8080 -i -t psiinon/bodgeit
```



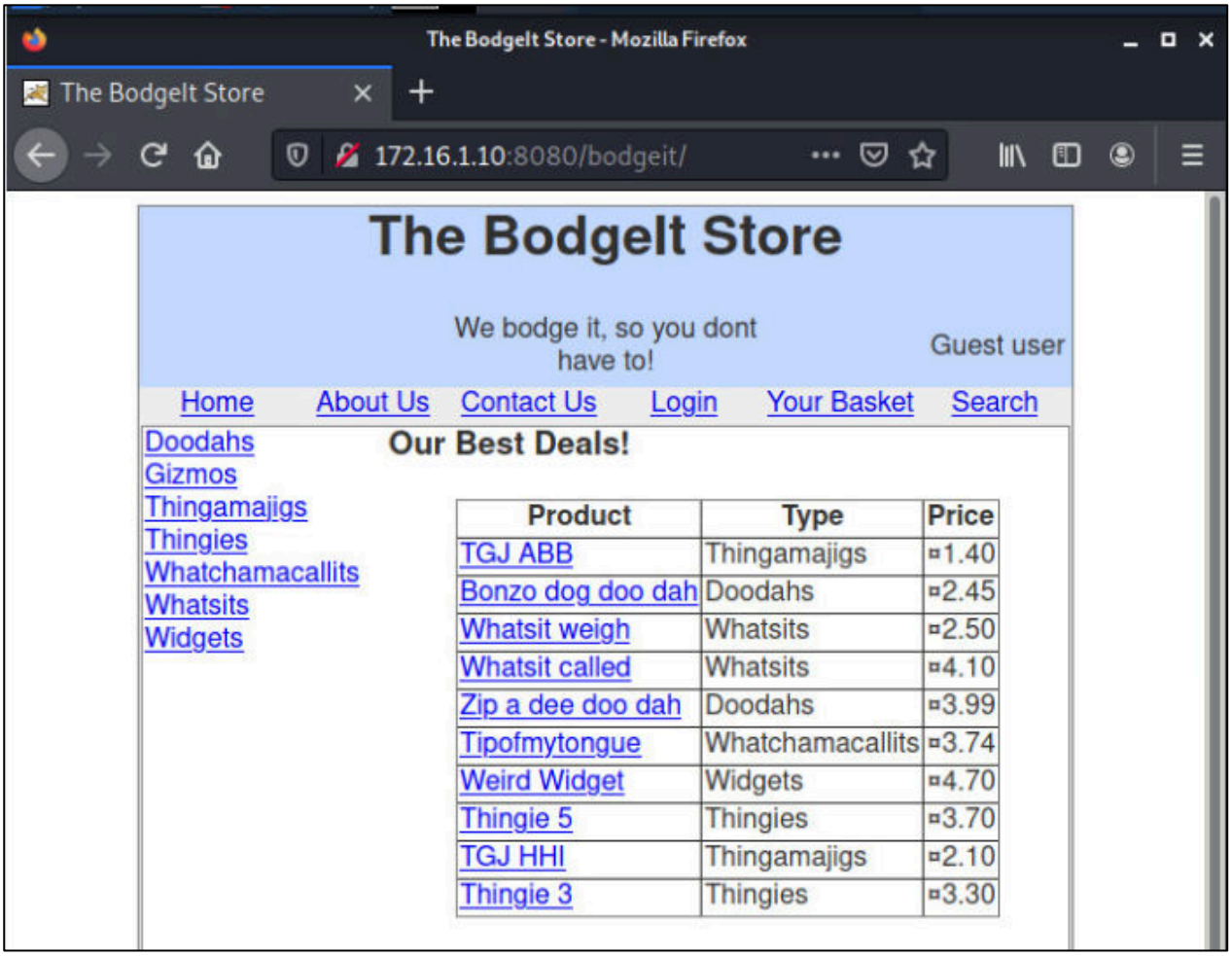
4. Change focus to the **Kali** computer.
5. Log in as `sysadmin` using the password: `NDGlabpass123!`



6. Open the **Web Browser** application on the taskbar.

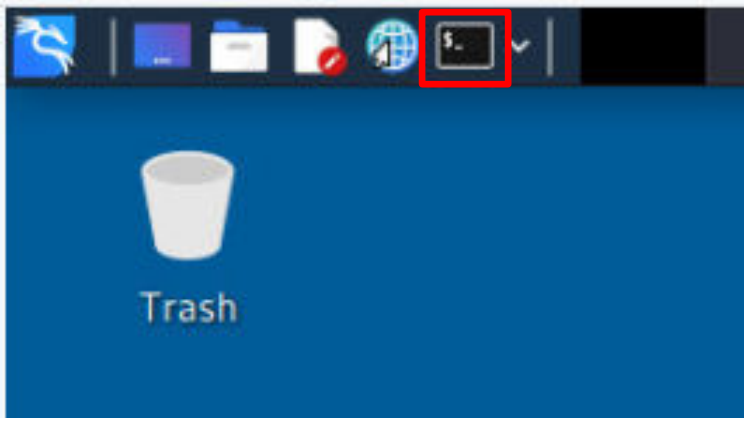


7. In order to ensure that the application is correctly running on the target, in the URL bar, navigate to `http://172.16.1.10:8080/bodgeit/`. Confirm that the website has successfully loaded.



8. Close the web browser.

9. Click on the **Terminal** icon in the taskbar at the top of the screen.



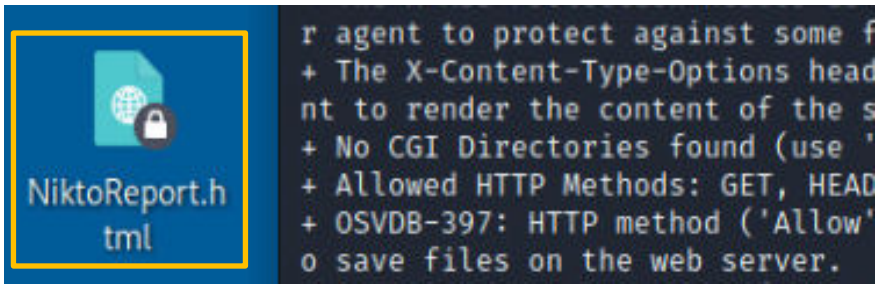
10. Execute the following command to use *Nikto* to test the *Bodgeit* site for misconfigurations:

```
sudo nikto -host 172.16.1.10 -port 8080 -root psiinon/bodgeit -Format htm
-output Desktop/NiktoReport.html
```

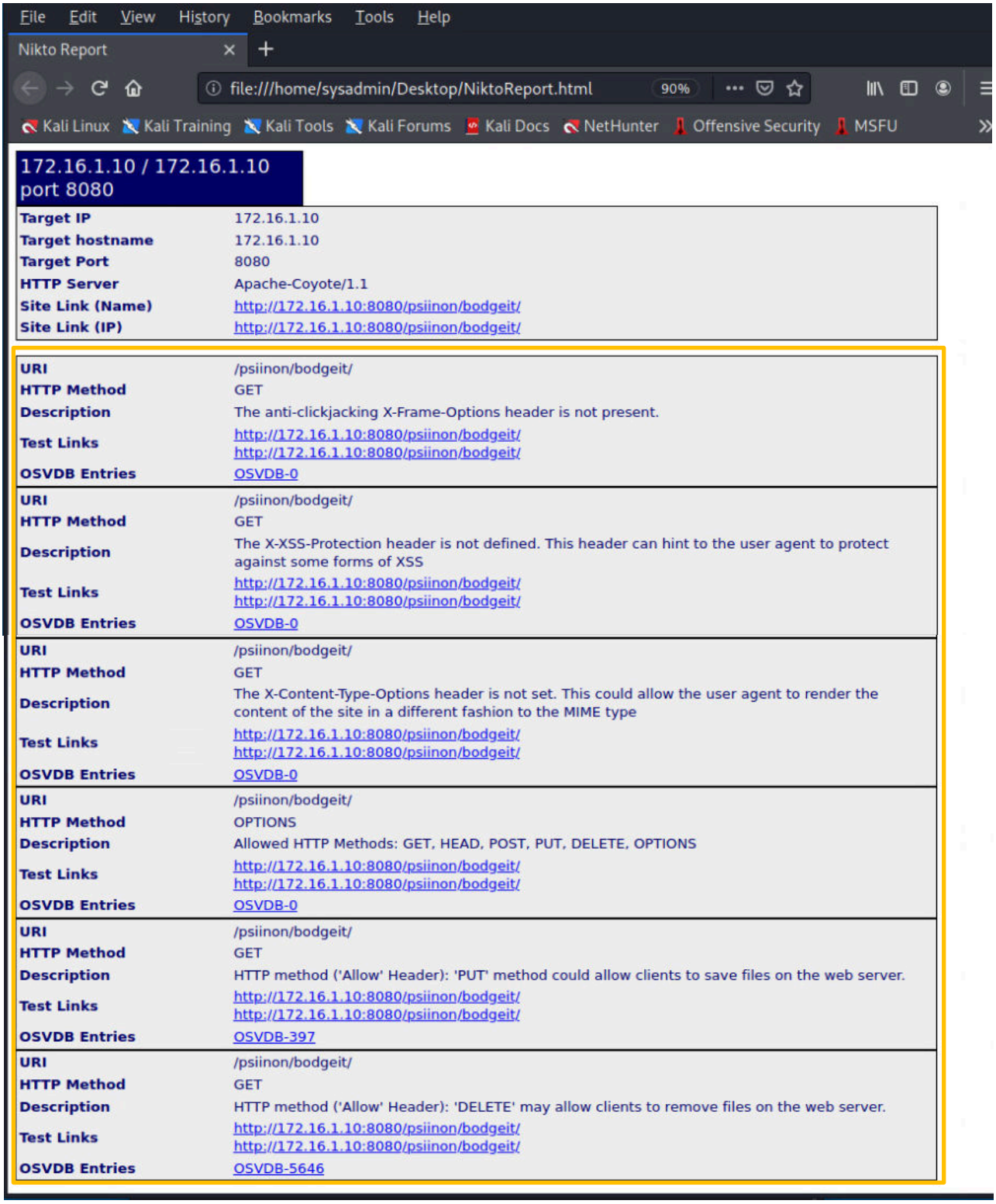
If asked for the *sysadmin* password, type: `NDGlabpass123!`

```
┌──(sysadmin㉿kali)-[~]
└─$ sudo nikto -host 172.16.1.10 -port 8080 -root psiinon/bodgeit -Format htm -output Desktop/NiktoReport.html
[sudo] password for sysadmin:
- Nikto v2.1.6
---------------------------------------------------------------------------
+ Target IP:          172.16.1.10
+ Target Hostname:    172.16.1.10
+ Target Port:        8080
+ Target Path:        /psiinon/bodgeit
+ Start Time:         2022-05-03 13:08:30 (GMT-4)
---------------------------------------------------------------------------
+ Server: Apache-Coyote/1.1
+ The anti-clickjacking X-Frame-Options header is not present.
+ The X-XSS-Protection header is not defined. This header can hint to the user agent to protect against some fo
rms of XSS
+ The X-Content-Type-Options header is not set. This could allow the user agent to render the content of the si
te in a different fashion to the MIME type
+ No CGI Directories found (use '-C all' to force check all possible dirs)
+ Allowed HTTP Methods: GET, HEAD, POST, PUT, DELETE, OPTIONS
+ OSVDB-397: HTTP method ('Allow' Header): 'PUT' method could allow clients to save files on the web server.
+ OSVDB-5646: HTTP method ('Allow' Header): 'DELETE' may allow clients to remove files on the web server.
+ 7916 requests: 0 error(s) and 6 item(s) reported on remote host
+ End Time:           2022-05-03 13:08:42 (GMT-4) (12 seconds)
---------------------------------------------------------------------------
+ 1 host(s) tested
```

11. On the desktop, you should see a file named **NiktoReport.html**. This file contains the report on the website's vulnerabilities. Double-click on the file, and it will open in a web browser window.



You will see the issues found by *Nikto*. A *Google* search will allow you to elaborate on the vulnerabilities and how to fix them.



From here, the report's HTML file could be printed or saved for later documentation

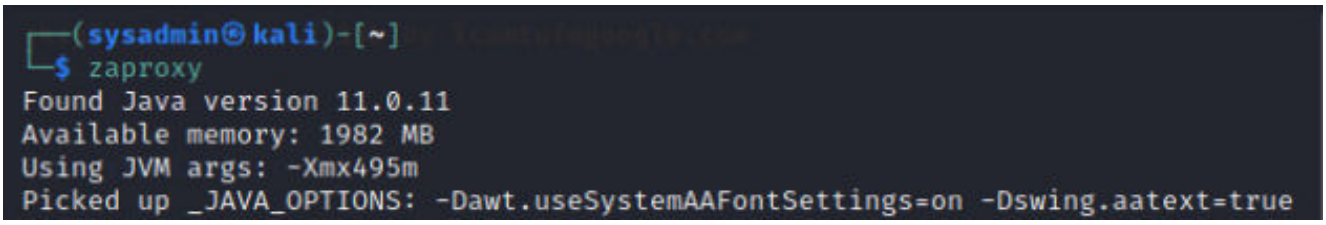
18. Close the web browser.
19. Leave the *Kali* computer open for the next section of the lab.

2.2 Test the Website for Vulnerabilities using OWASP ZAP

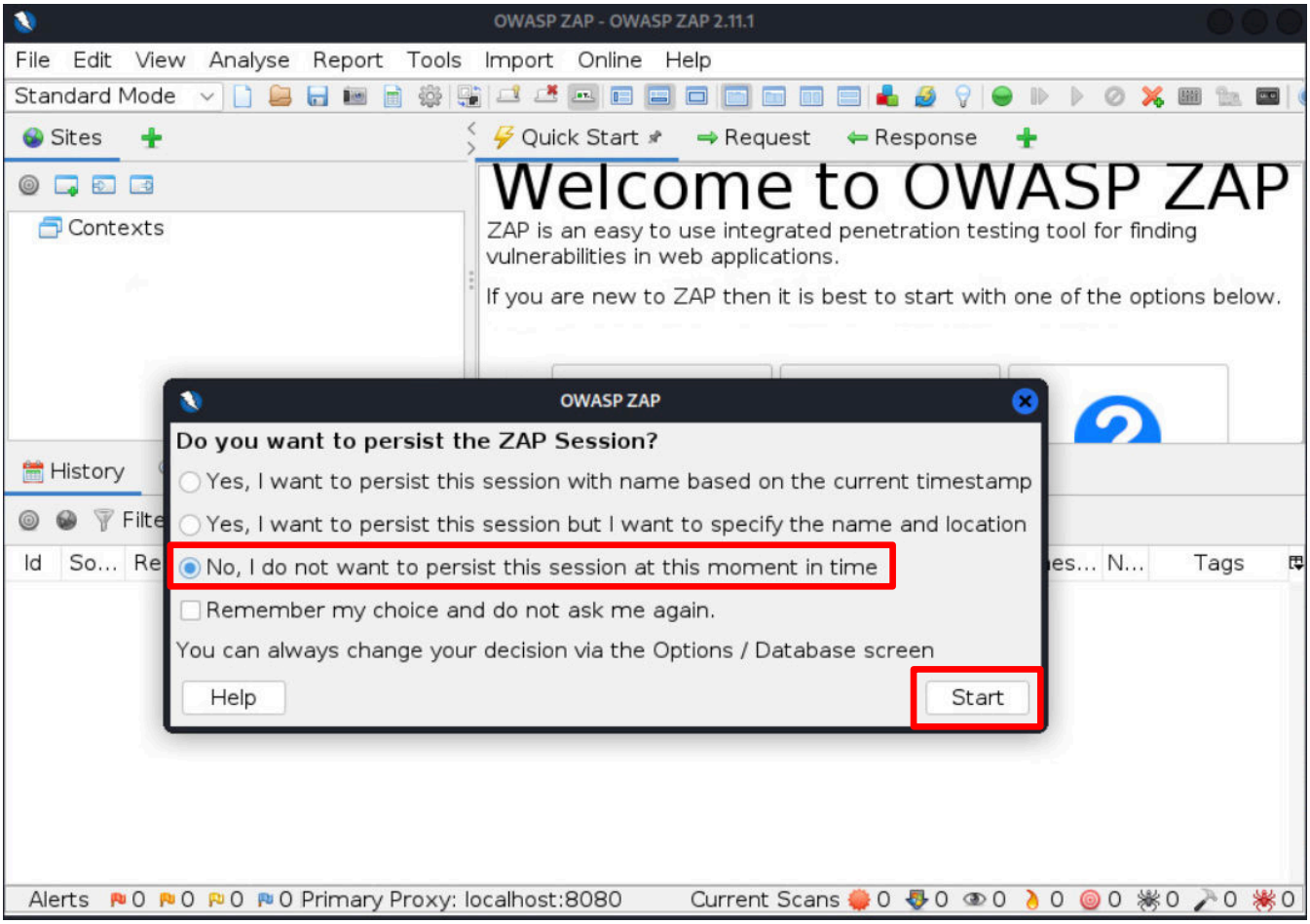
OWASP ZAP (*Zed Attack Proxy*) is one of the most popular free web security tools. Not only can it help to find security vulnerabilities in your web applications automatically, but experienced penetration testers can also use the program for manual security testing, as well.

1. We will use *OWASP ZAP* to scan the *Bodgeit* site. In order to launch *ZAP*, in the terminal window, type the following command:

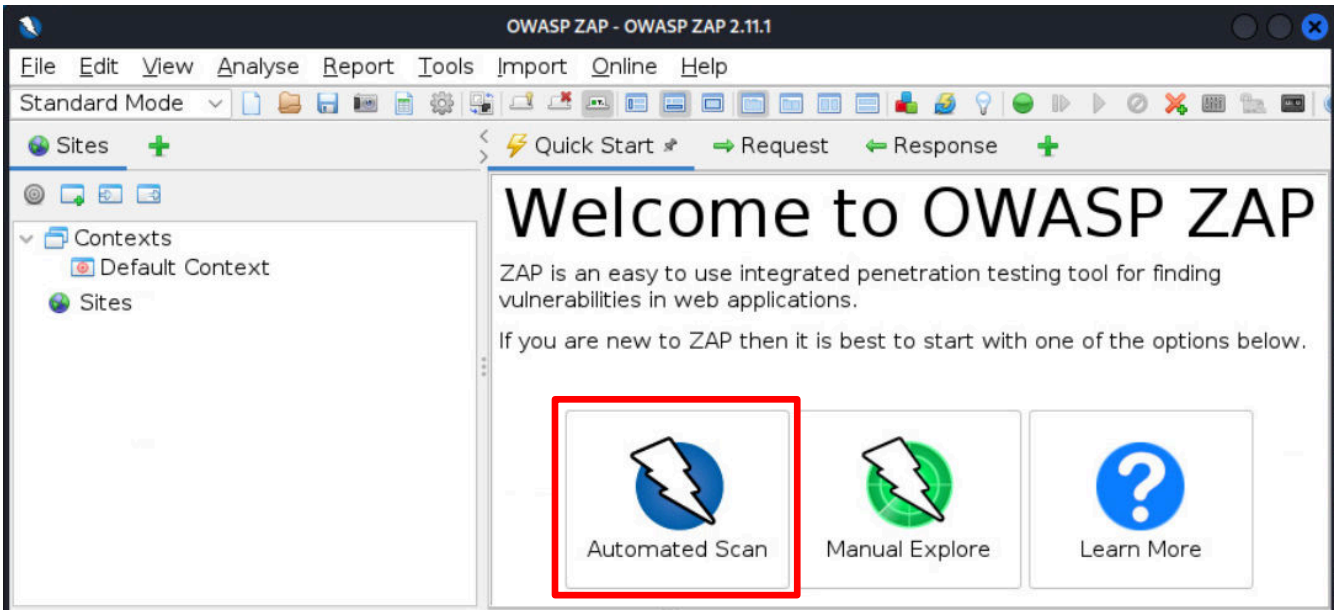
```
zaproxy
```



2. The *OWASP ZAP* page will open, and there will be a popup window asking *Do you want to persist the ZAP Session*? Click **No, I do not want to persist this session at this moment in time** and then click **Start.**

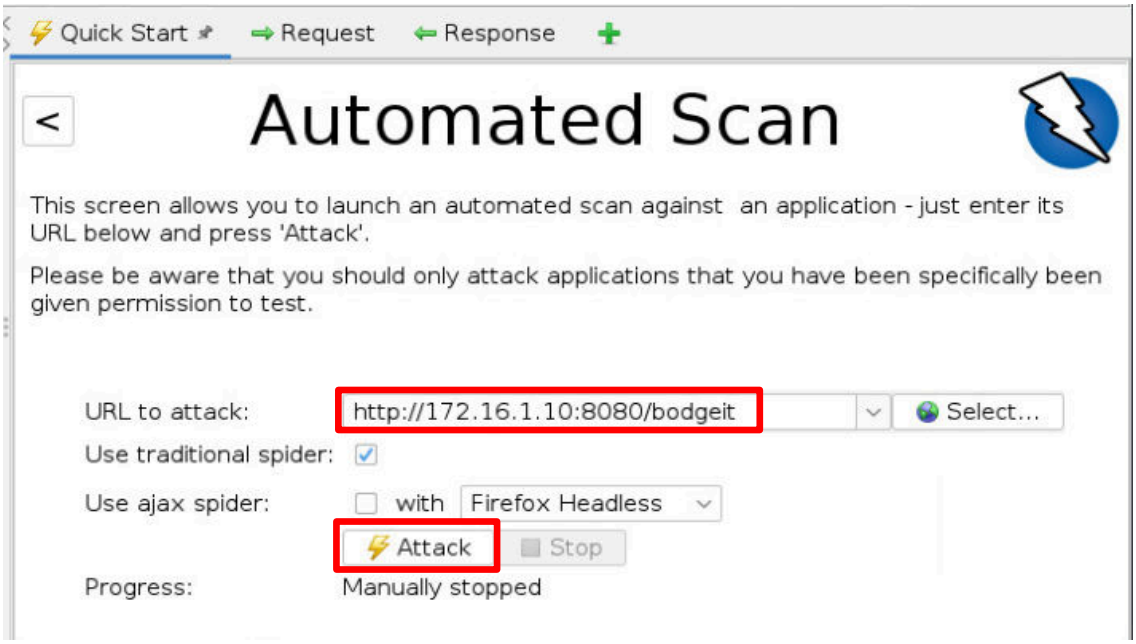


3. On the *Quick Start* tab on the main page, click on the **Automated Scan** button.



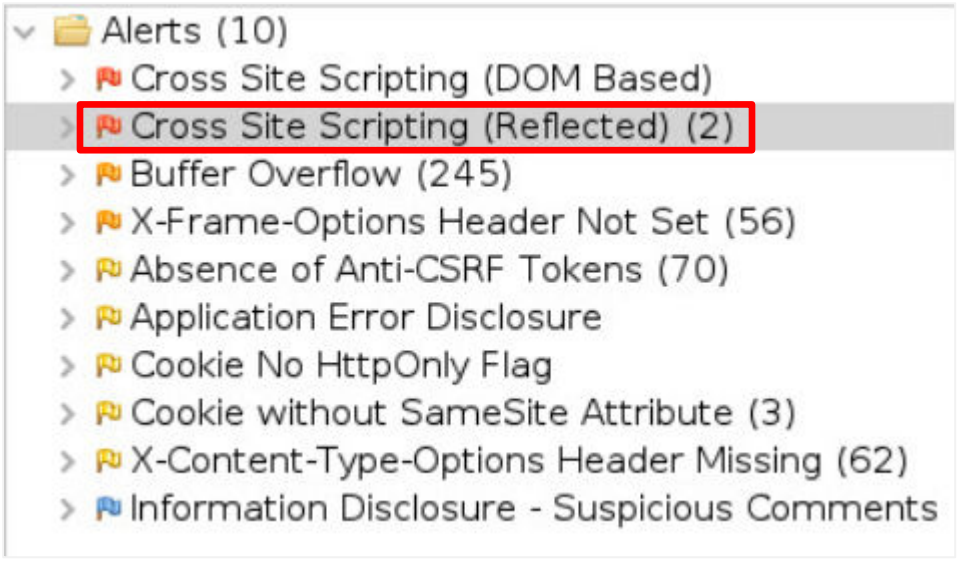
You might need to adjust the size of the panes by clicking and dragging the borders of each pane.

4. Once the GUI loads, type the target URL of `http://172.16.1.10:8080/bodgeit` and click **Attack**.



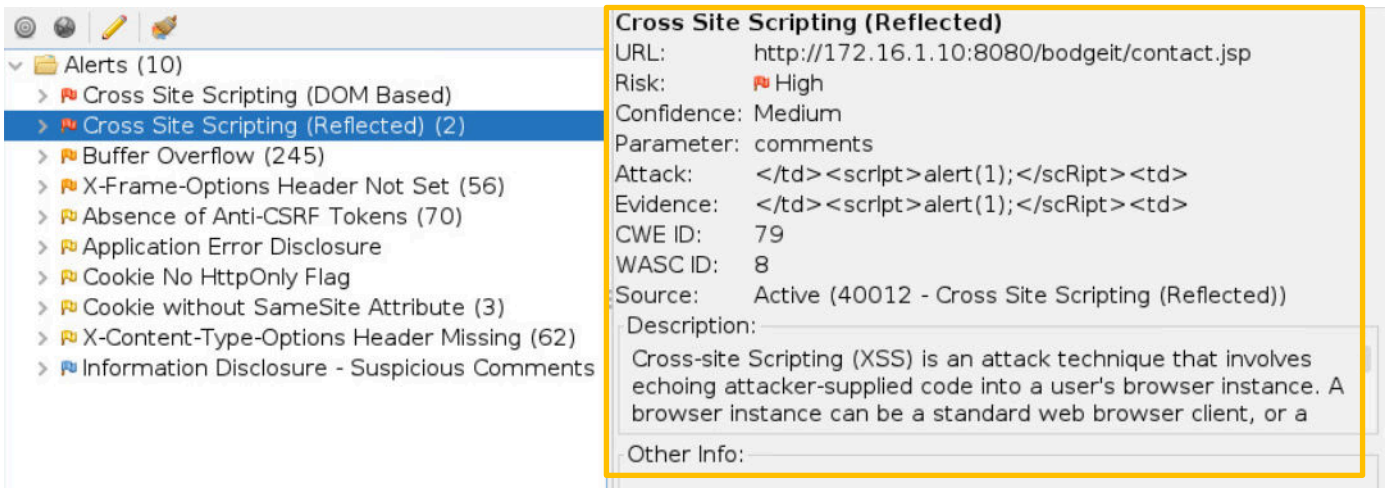
Please note that this scan will take approximately 7-8 minutes, as it is running around 80,000 tests.

5. Once the scan is finished, review the scan results by navigating to the **Alerts** tab in the lower-left corner. Click on **Cross Site Scripting (Reflected)** in the list.

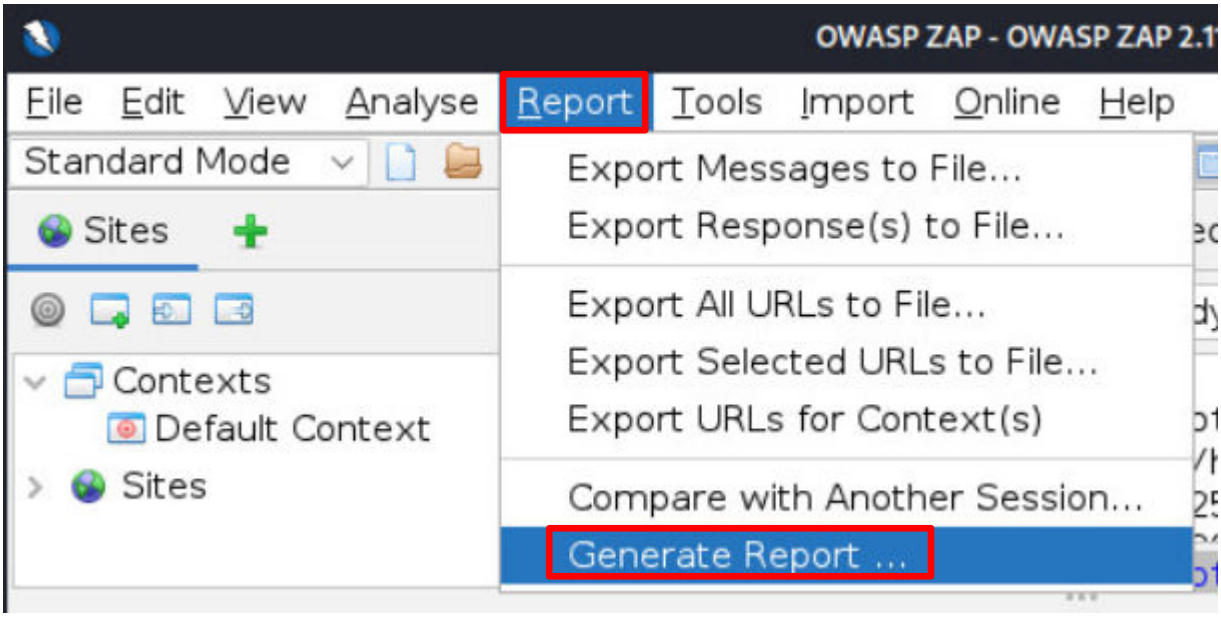


55

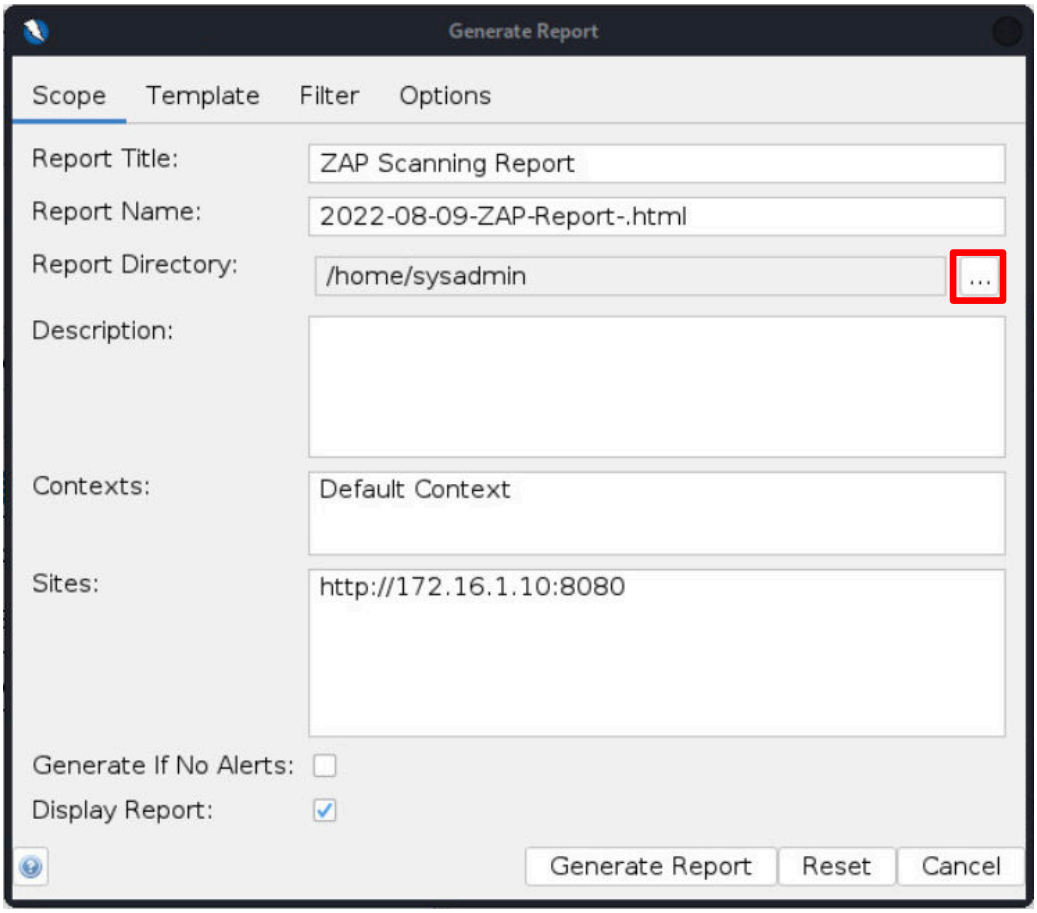
The vulnerability's description can be found in the *Description* section of the lower-right pane.



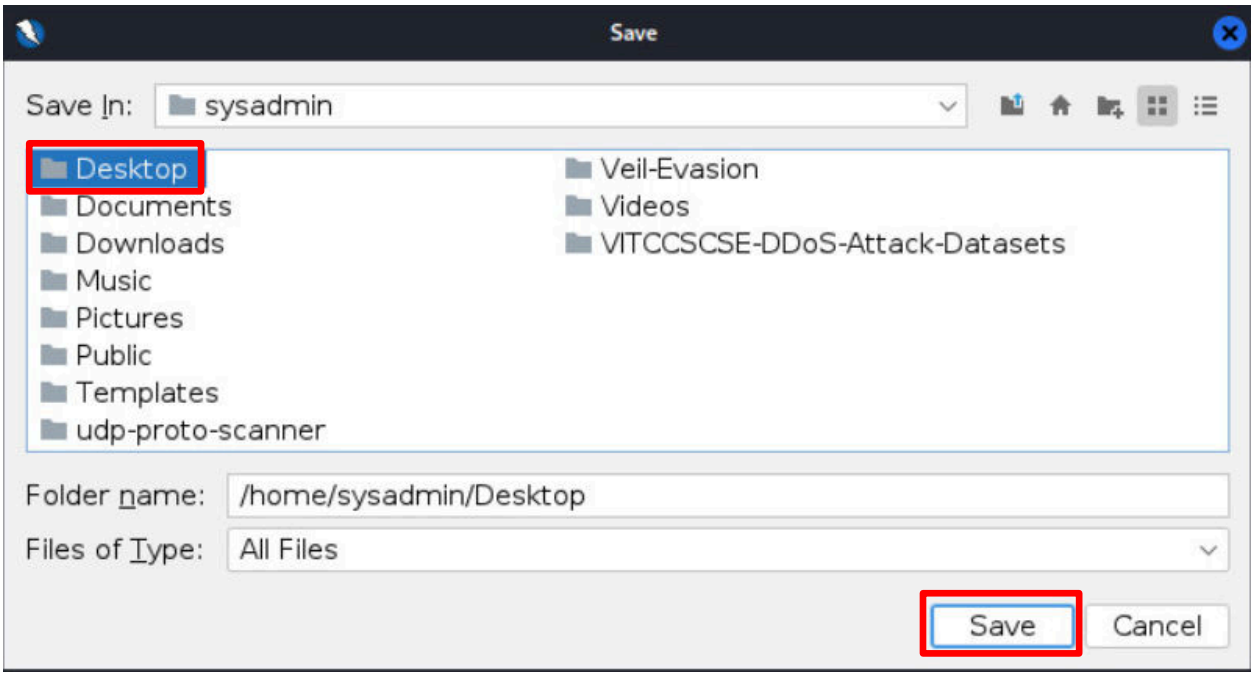
6. To display a report from the scan, on the *OWASP ZAP* menu, click on **Report,** then click on **Generate Report**.



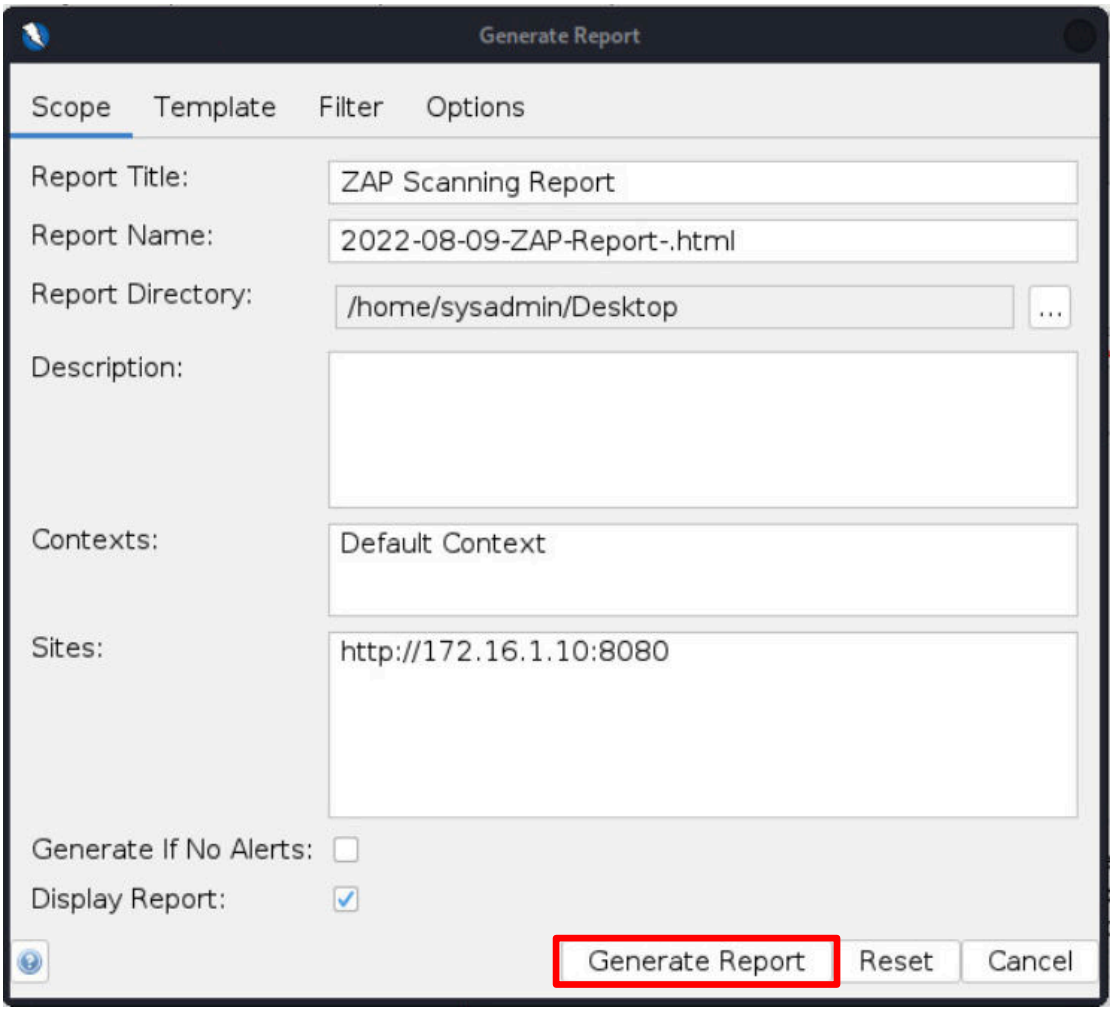
7. On the *Generate Report* window, click on the **Change Directory** icon to the right of the **Report Directory** entry.



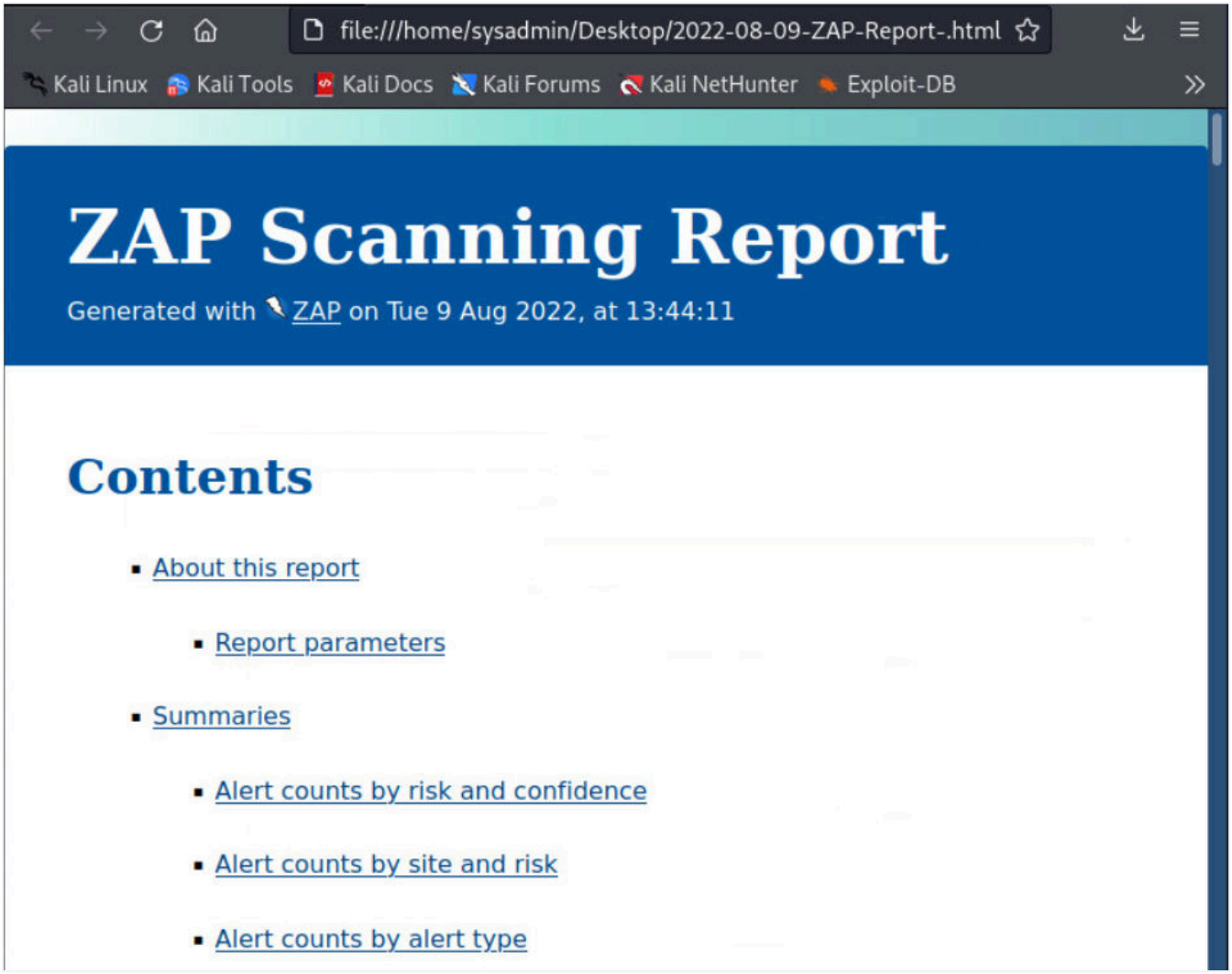
8. On the *Save* window, click the **Desktop** folder, then click the **Save** button.



9. Back on the *Generate Report* window, click the **Generate Report** button.



10. The *ZAP Scan Report* will open in a browser window. From here, the report's HTML file could be printed or saved for later documentation.

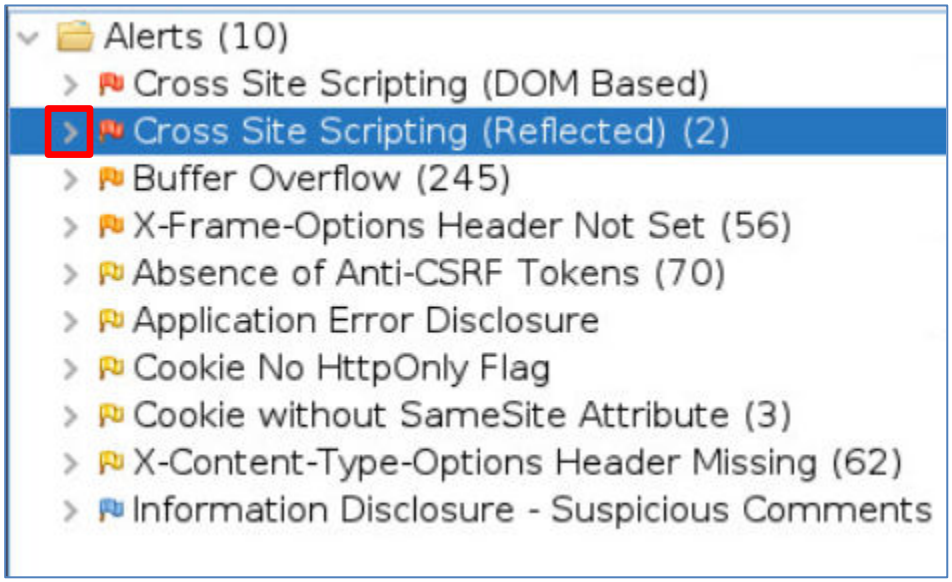


11. After viewing the page, close the browser window.
12. Keep the *OWASP ZAP* application open for the next section.

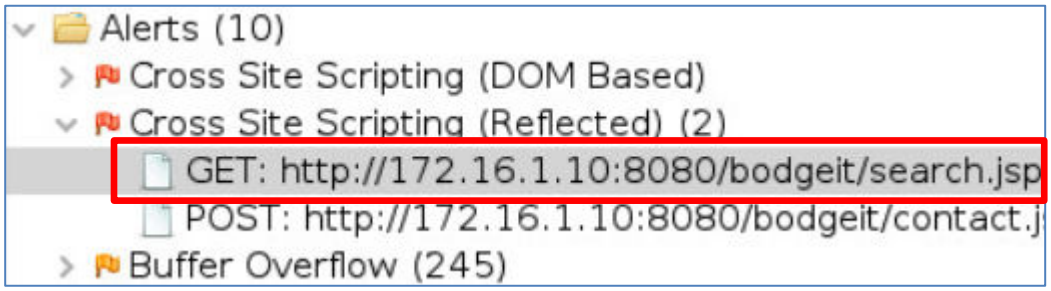
2.3 Exploiting the Vulnerable Website After OWASP ZAP Discovery

Now that you have discovered several web vulnerabilities with *Nikto* and *OWASP ZAP*, you can explore how to exploit these vulnerabilities. This will give better insight into how common coding problems are exploited.

1. In the lower-left panel of the *OWASP ZAP* window, look for the alert in the list, which should say *Cross Site Scripting (Reflected),* and click on the **>** to expand the view of specific vulnerabilities found by the scan.



2. Click on the first entry in the list, **GET: http://172.16.1.10:8080/bodgeit/search.jsp**.



3. On the right pane, you will see the entry describing the *Cross Site Scripting* vulnerability along with a detailed description and a solution.

Cross Site Scripting (Reflected)

URL: http://172.16.1.10:8080/bodgeit/search.jsp?q=%3C%2Ffont%3E%3Cscript%3Ealert%281%29%3B%3C%2Fscript%3E%3Cfont%3E
Risk: High
Confidence: Medium
Parameter: q
Attack: <script>alert(1);</script>
Evidence: <script>alert(1);</script><font>
CWE ID: 79
WASC ID: 8
Source: Active (40012 - Cross Site Scripting (Reflected))

Description:

Cross-site Scripting (XSS) is an attack technique that involves echoing attacker-supplied code into a user's browser instance. A browser instance can be a standard web browser client, or a browser object embedded in a software product such as the browser within

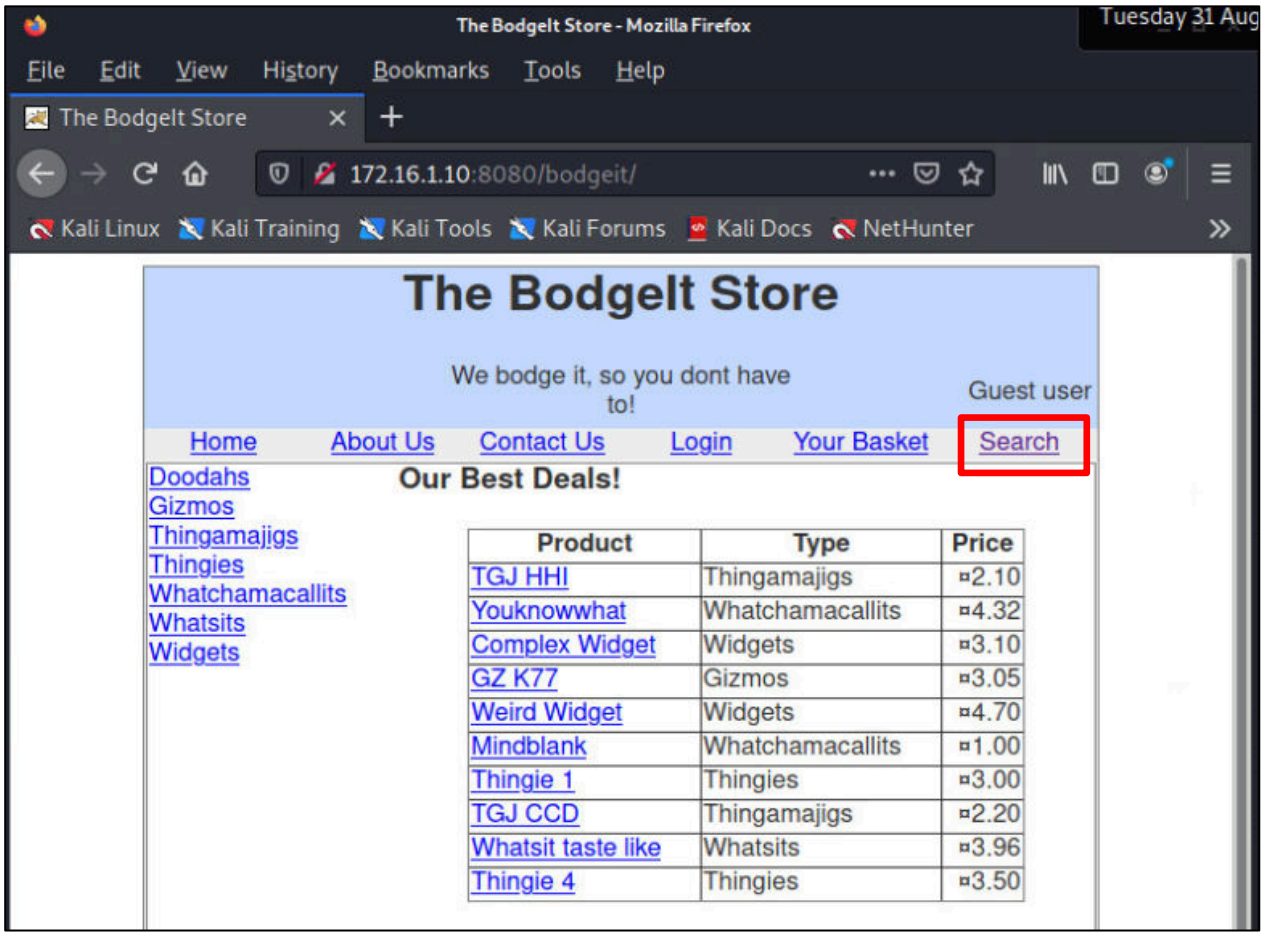
Other Info:

Solution:

Phase: Architecture and Design
Use a vetted library or framework that does not allow this weakness to occur or provides constructs that make this weakness easier to avoid.

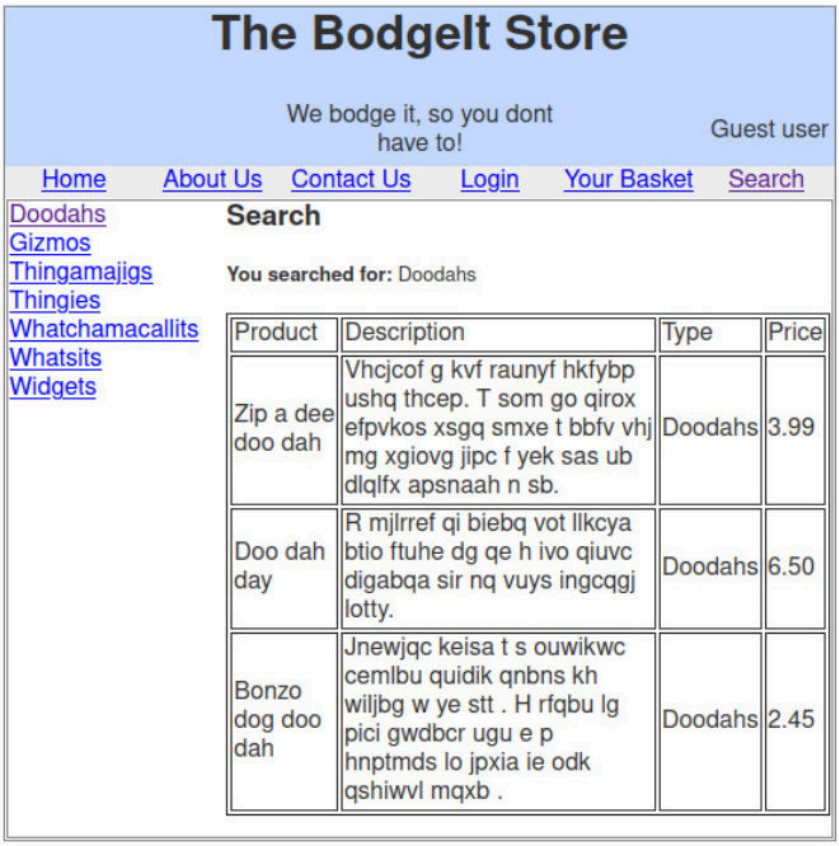
4. Minimize all open windows.

5. Let's see if we can exploit the problem by inserting a simple popup alert into the search function. Open the web browser and type `http://172.16.1.10:8080/bodgeit`.
6. When the *Bodgeit Store* page opens, click on the **Search** menu option.



7. On the left side, you will see a list of all the product types in the *Bodgelt Store*. Each entry will show a product list for the entries. To see the entry for `Doodahs` type it as the search item and click the **Search** button.

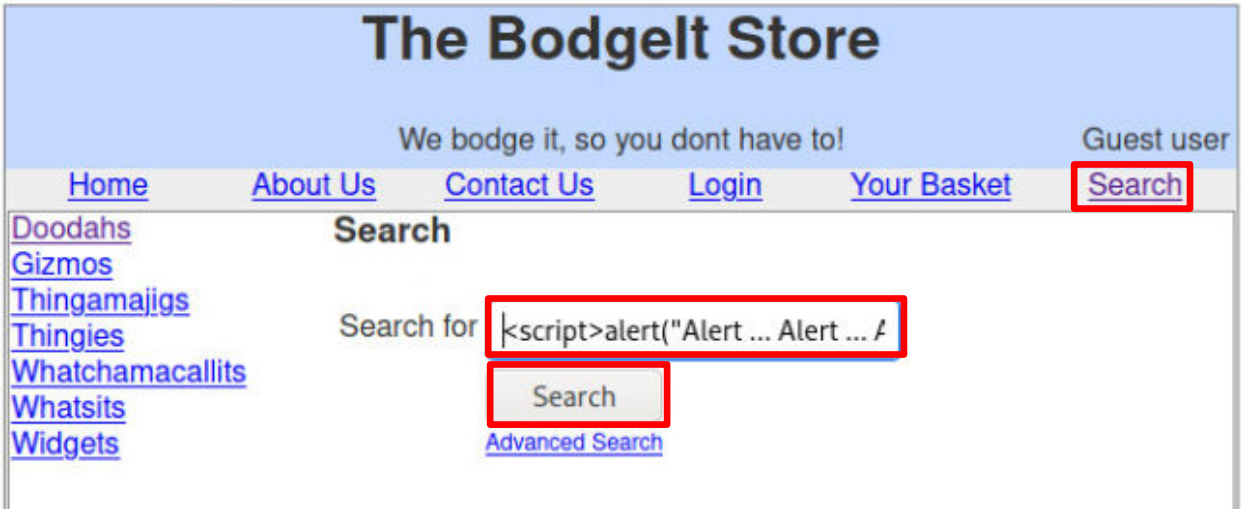
8. You should have a results screen similar to the one below:



9. This time we will inject a Cross-Site Script that will exploit the vulnerability discovered in the previous step. Click the **Search** button in the top-right and type:

```
<script>alert("Alert … Alert … Alert")</script>
```

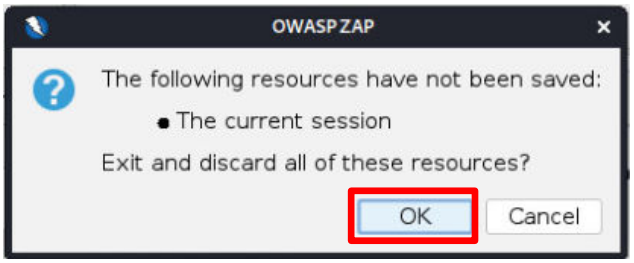
10. Click the **Search** button below the *Search For* box to execute the Cross-Site Script and pop up the alert message box.



11. This time the response to the search is the *Cross-Site Script* popup box.

The takeaway from this example is that user input should never be trusted and should always be validated and checked thoroughly before any processing is done. But, without scanning the website for vulnerabilities, the problem might never have been found.

12. You can close the web browser and *OWASP ZAP*. When asked to *Exit and Discard all of these resources*, click **OK.**



13. The lab is now complete; you may now end the reservation.