|  |  |
| --- | --- |
| Name: | Deekshith KR |
| Lab User ID: | 23SEK3324\_U08 |
| Date: | 10/01/2024 |
| Application Name: | [**Vulnerable Java Web Application**](https://github.com/D33ksh1th/VulnerableJavaWebApplication) – **o2** |

**Follow the below guidelines:**





System Architecture:

(Understand the system and document the physical and logical architecture of the system, use the shapes and icons to capture the system architecture)

AWS

Ubuntu instance

Docker

Web Server

Container

Docker image

Define system’s normal behavior:

(Define the steady state of the system is defined, thereby defining some measurable outputs which can indicate the system’s normal behavior)

This is a Java web application which is deployed using a AWS ubuntu instance and

And configured to run on port 8000.

This web application takes in user input without any abnormal behavior.

This application is used to test website for their availability and performance.

Hypothesis:

(During an experiment, we need a hypothesis for comparing to a stable control group, and the same applies here too. If there is a reasonable expectation for a particular action according to which we will change the steady state of a system, then the first thing to do is to fix the system so that we accommodate for the action that will potentially have that effect on the system. For eg: "If one of our database servers fails, our service will automatically switch to a backup server, and users will not experience any downtime or data loss.")



**Known**

Things we are aware of but don’t understand.

Things we are aware of and understand.

942236

**Unknown**

**Unknown**

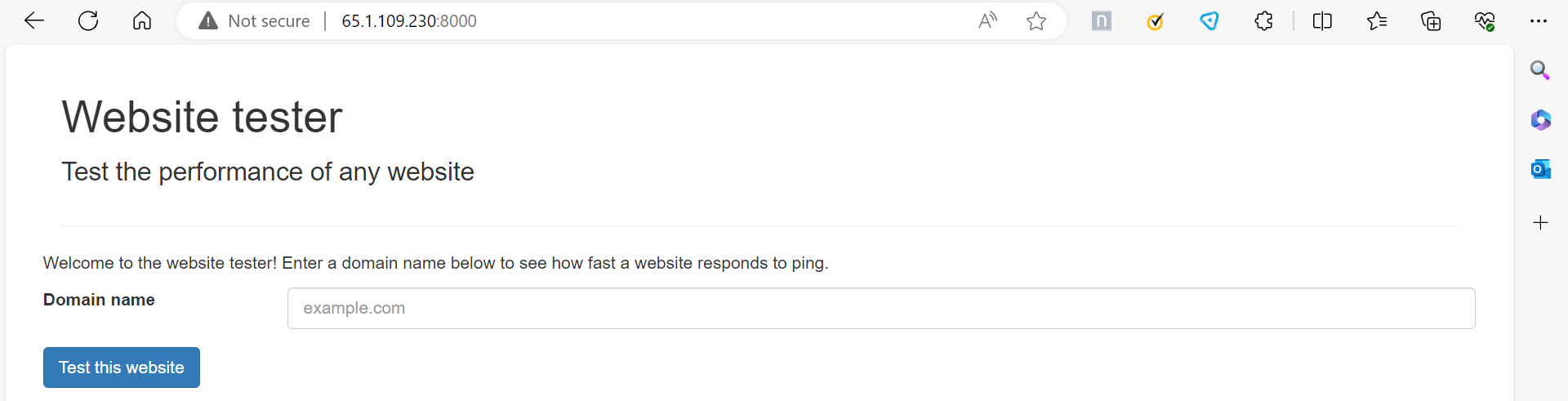
**Known**

Things we are neither aware of nor understand.

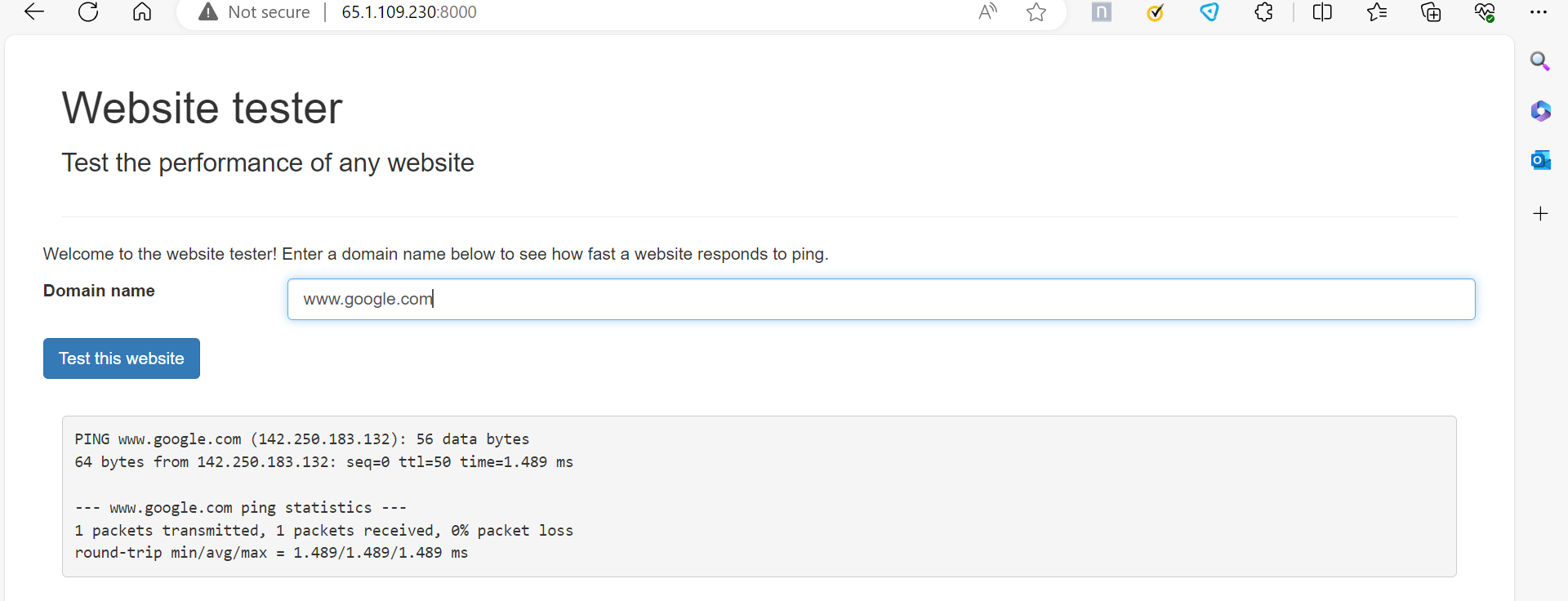
Things we understand but are not aware of.

First we need to live the application , I have built the docker file to create an image and run it exposing to the port

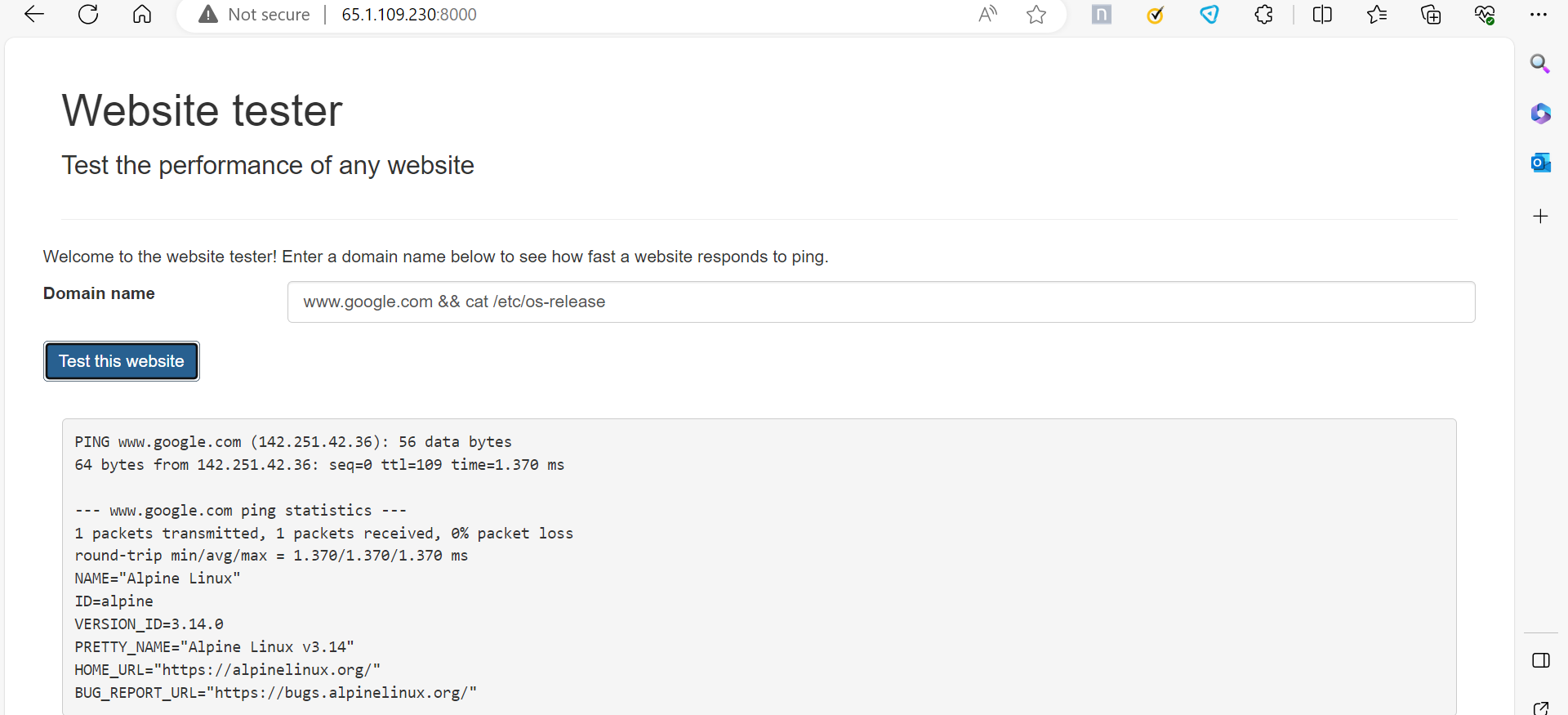
8000 and now we can access it using localhost:8000. The output of the web page is :



User input and the output :



We also see the when given extended commands to expose the website it reveals the content and this web application is vulnerable to face any kind of attacks and expose information :



We are going to test this web application to find out the vulnerabilities.

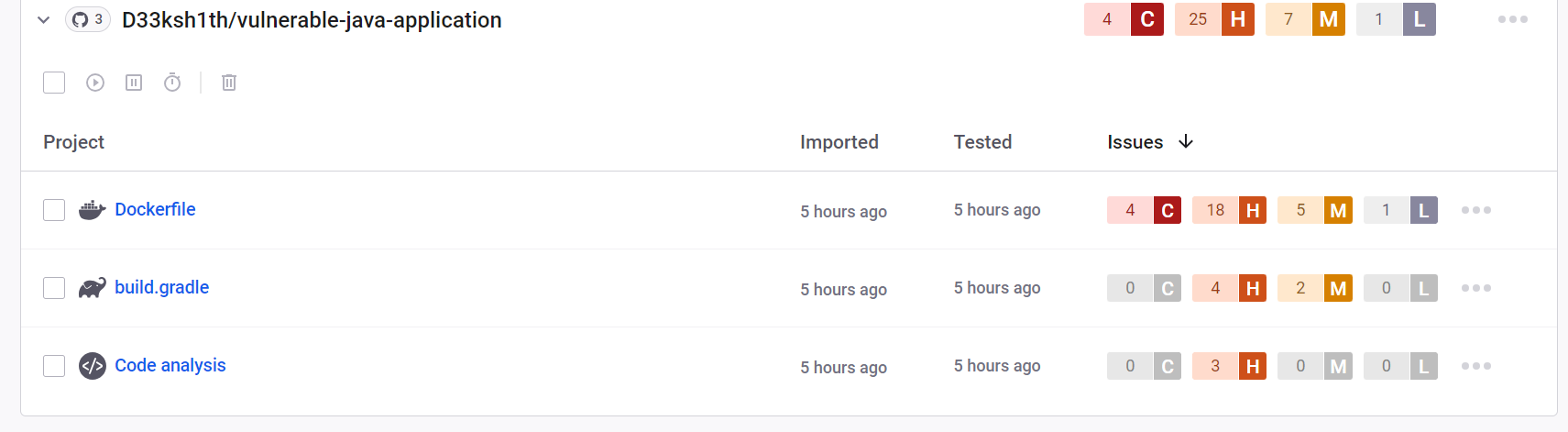
The tests we are using here is :

Snyk – to test the repository file.

Nuclei

1. By using snyk we can scan the vulnerabilities of the application

Below is the output of the issues found using the Synk tool:



Details of the vulnerabilities are:

CWE-78:

Improper Neutralization of Special Elements used in an OS Command ('OS Command Injection')

Description: The application constructs OS commands using input from external sources without proper validation, exposing it to OS Command Injection vulnerabilities.

Mitigation: Utilize parameterized queries or prepared statements to prevent command injection. Validate and sanitize user input before using it in command construction. Employ application firewalls to detect and block malicious inputs.

CWE-120:

Buffer Copy without Checking Size of Input ('Classic Buffer Overflow')

Description: The Docker file contains a vulnerability where input buffers are copied to output buffers without size verification, leading to classic buffer overflow issues.

Mitigation: Ensure all buffer operations have size checks to prevent overflow. Use safer functions for copying, such as strncpy or snprintf. Regularly update and patch software to fix known vulnerabilities.

1. Using Nuclei

To find the vulnerabilities in the live application that has been created using nuclei

The report is :

X-Content-Type-Options Header Missing:

Identified URLs: http://13.232.48.232:8000/, http://13.232.48.232:8000/js/main.js

Description: The X-Content-Type-Options header is missing, which may expose the

application to MIME-sniffing attacks.

Recommendation: Set the X-Content-Type-Options header to 'nosniff' to mitigate MIME-sniffing risks.

Permissions Policy Header Not Set:

Identified URLs: http://13.232.48.232:8000/, http://13.232.48.232:8000/js/main.js

Description: The Permissions Policy header is not set, leaving the application more susceptible to security vulnerabilities.

Recommendation: Configure and set a strong Permissions Policy header to control and limit browser features.

Modern Web Application:

Identified URL: http://13.232.48.232:8000/

Description: The application is identified as a modern web application.

Recommendation: Stay updated with modern security best practices and ensure the application follows secure coding standards.

Experiment:

(Document your Preparation, Implementation, Observation and Analysis )