|  |  |
| --- | --- |
| Name: | Dhwanit Pandya |
| Lab User ID: | 23SEK3324\_U34 |
| Date: | 10-01-2024 |
| Application Name: | **wrongsecrets** |

**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)



Created a Container with jeroenwillemsen/wrongsecrets

as image

Hosted application using node on port: 8080

**Ubuntu 20.04**

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)

Basic functionality such as login, signup works perfectly. Along with these all the features for eg: Add beneficiary, view beneficiary, my balance, Transfer money, Transactions, Approve Benf, Currency Rates and Password reset

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**

If there is any traffic then we have logs to monitor but we don’t know how the functionality will work

Every functionality is working fine

**Unknown**

**Unknown**

**Known**

Any unknown vulnerabilities in code that might affect the availability of the application

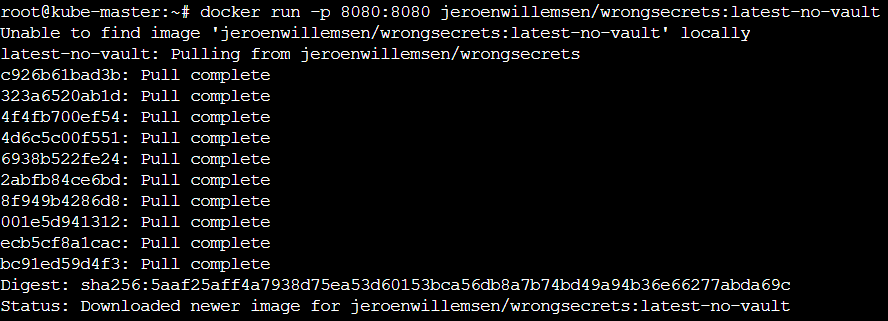
If there is an increase in traffic then it would affect the functionality but don’t know how swiftly would it perform

Experiment:

(Document your Preparation, Implementation, Observation and Analysis )

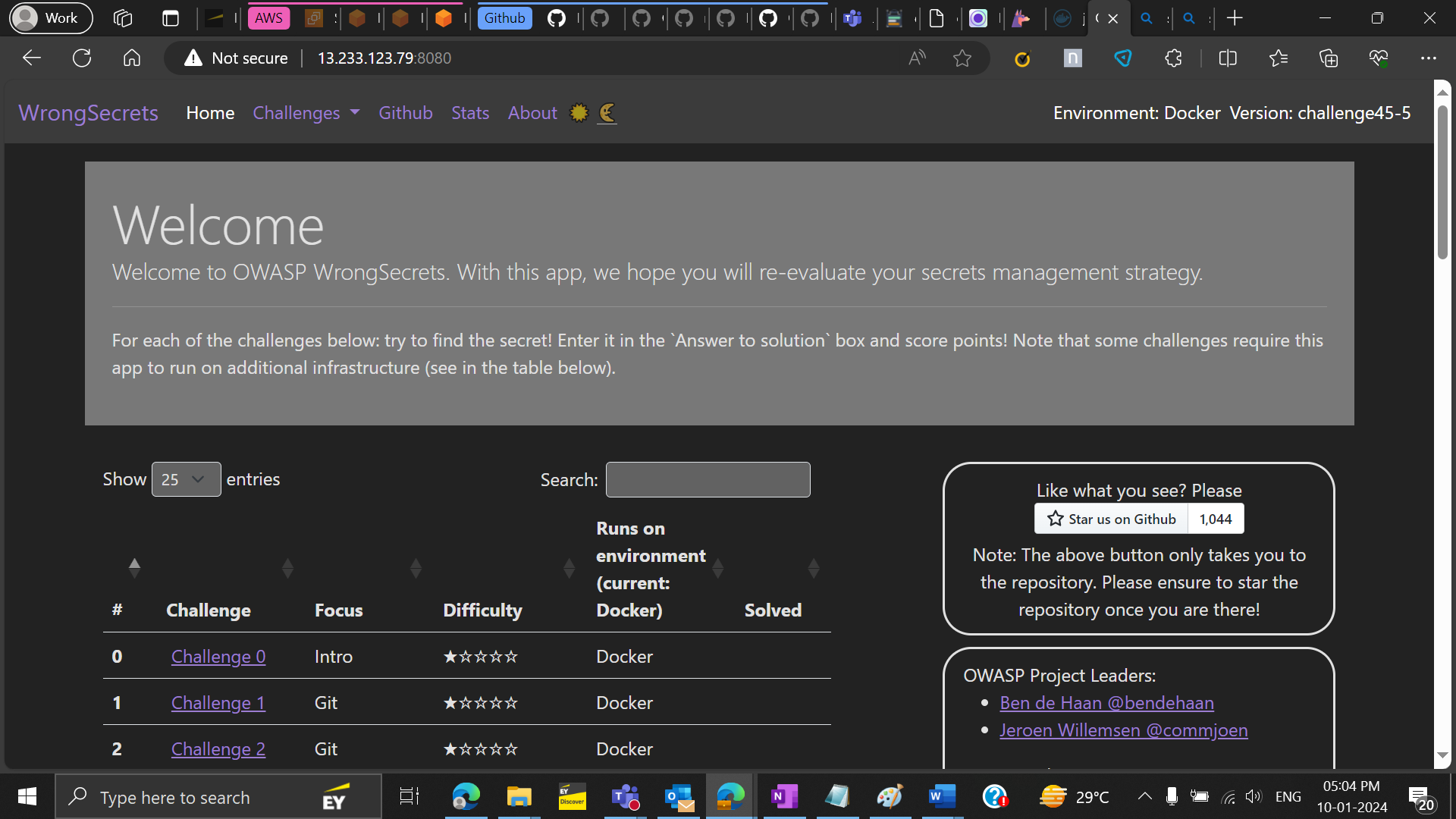
**Preparation:**

* Launched a virtual machine on AWS with Ubuntu 20.04
* updated the machine and installed docker
* Set up a docker container with the image jeroenwillemsen/wrongsecrets



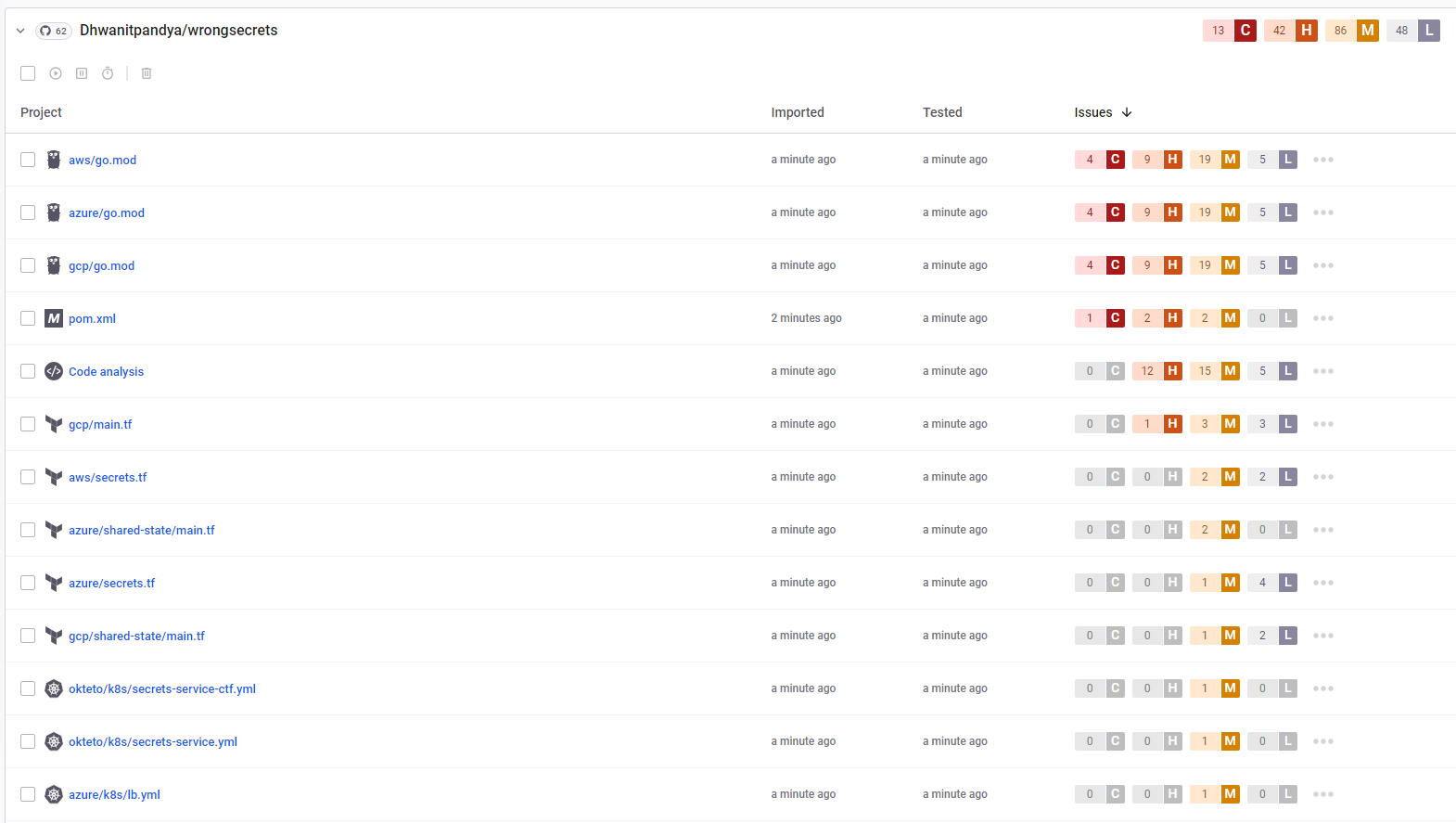
**Implementation:**

* **Now we can see the website gets hosted on localhost:8080**



**Observation and Analysis**

* Performed vulnerability analysis of the following repo using the snyk tool and found:

****

* Out of these Vulnerabilities some critical ones are and their fixes are mentioned below:

1. **Vul : go Improper Authentication**

**Impact:** When the access control of an application is broken, a regular user may be able to access functionality that is meant to be reserved for administrators, or perhaps they can access data that does not belong to them.

**Fix**: Permissions structures still need to be implemented by the developer, because every application has specific, custom requirements.

1. **Vul : go Improper Access Control**

**Impact:** When the access control of an application is broken, a regular user may be able to access functionality that is meant to be reserved for administrators, or perhaps they can access data that does not belong to them.

**Fix**: Permissions structures still need to be implemented by the developer, because every application has specific, custom requirements.

1. **Vul : go Improper Input Validation**

**Impact:** Affected versions of this package are vulnerable to Improper Input Validation due to an insufficient access check, a recovery flow link that is created by an admin (or sent via email by an admin) can be used to set the password for any arbitrary user.

**Fix**: here are some fixes:

* + - Avoid the use of dangerous functions
    - Reconsider the need for dynamic code execution
    - Lock down the interpreter
    - Utilize a static analysis tool

1. **Vul : vm2 Sandbox Bypass**

**Impact:** Affected versions of this package are vulnerable to Sandbox Bypass by abusing an unexpected creation of a host object based on the maliciously crafted specification of Proxy. Exploiting this vulnerability allows an attacker to gain remote code execution rights on the host running the sandbox via the Function constructor.

**Fix**: No fix as such but to prevent sandbox bypass attacks, organizations can take several measures, such as keeping their security tools and sandbox updated with the latest patches and signatures, using multiple layers of defense, such as endpoint protection, network security, and threat intelligence, educating their employees about the risks of opening suspicious emails, links, or attachments, and implementing security policies that restrict access to sensitive data and critical systems.

1. **Vul : socket.io-parser-Denial of Service (DoS)**

**Impact:** ReDoS attack attempts to slow down or even render an application unavailable. processing of the malicious string exhausts the computing power or memory available, thus impacting the application's performance and, in certain circumstances, causing a denial of service (or DoS).

**Fix**: Avoid using regex for user input validation. Closely review and analyze all patterns before implementation to ensure they do not contain any evil regex patterns.