|  |  |
| --- | --- |
| Name: | Arpitha S P |
| Lab User ID: | 23SEK3324\_U10 |
| Date: | 10-01-2024 |
| Application Name: | Wrong Secrets |

**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)

EC2 Instance-VM

Docker Engine

Container

13.233.93.0:8080

Docker Image

Physical Architecture :

* EC2 Instance
  + Instance type : t2 medium
  + Platform : ubuntu (Linux/UNIX)
  + Volume size(GiB) : 20
  + Availability zone : ap-south-1a

Logical Architecture :

* Docker Containers : These are isolated instances running within docker engine
* Docker Engine manages and orchestrates these containers

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)

Firstly the web server will listen to the defined ports like your ip:8080. The user will have access to the website which is hosted on the server once they are in the designated ip:port. Monitor and measure how often secrets are used in operations or transactions within the system. Deviations from the established usage metrics could signal potential security incidents.

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.")



Information, secrets or facts that are known both the individuals.groups involved. This includes shared knowledge, agreed-upon facts.

Aspects, facts or knowledge that are known by one party but not yet known to other. Secrets, unshared info, undiscovered insights.

**Known**

Things we are aware of but don’t understand.

Things we are aware of and understand.

Includes disclosures, revelations, or new discoveries.

**Unknown**

Unforeseen events, hidden knowledge, undiscovered areas waiting to be exploited.

**Unknown**

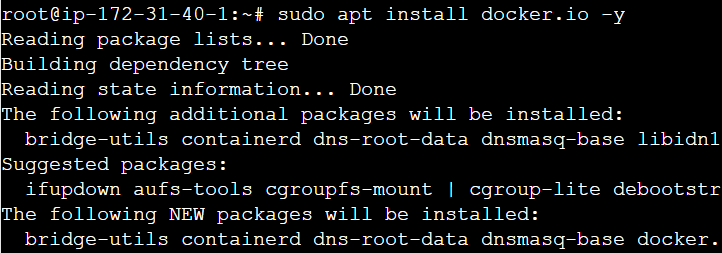
**Known**

Experiment:

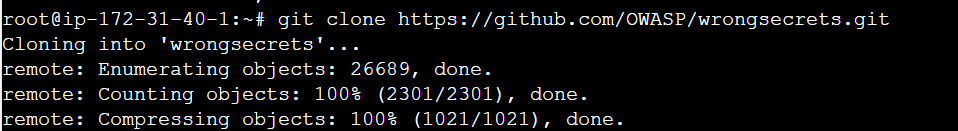
(Document your Preparation, Implementation, Observation and Analysis )

The game is packed with real life examples of how to not store secrets in your software. Each of these examples is captured in a challenge, which you need to solve using various tools and techniques. Solving these challenges will help you recognize common mistakes & can help you to reflect on your own secrets management strategy.

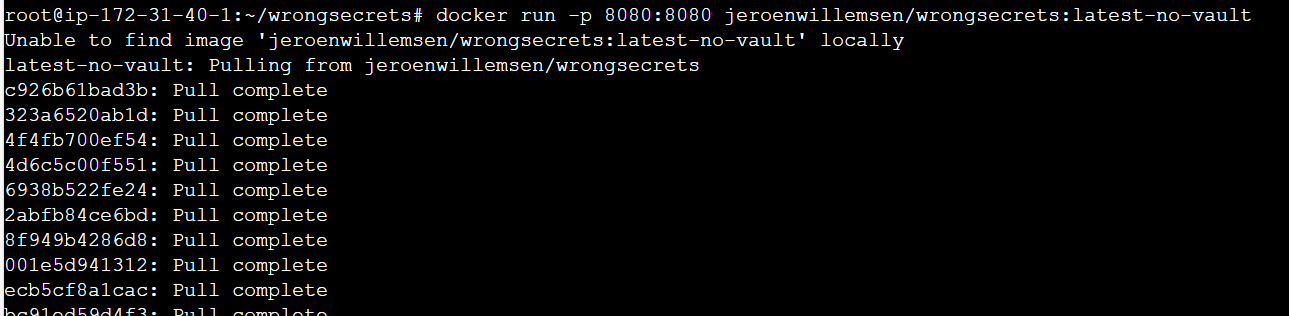
Step 1 : Installing Docker

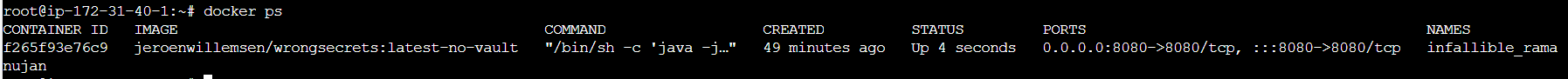


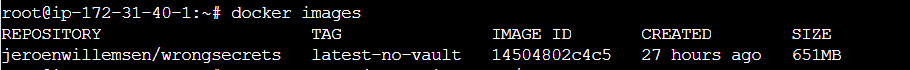
Step 2: Cloning Repository

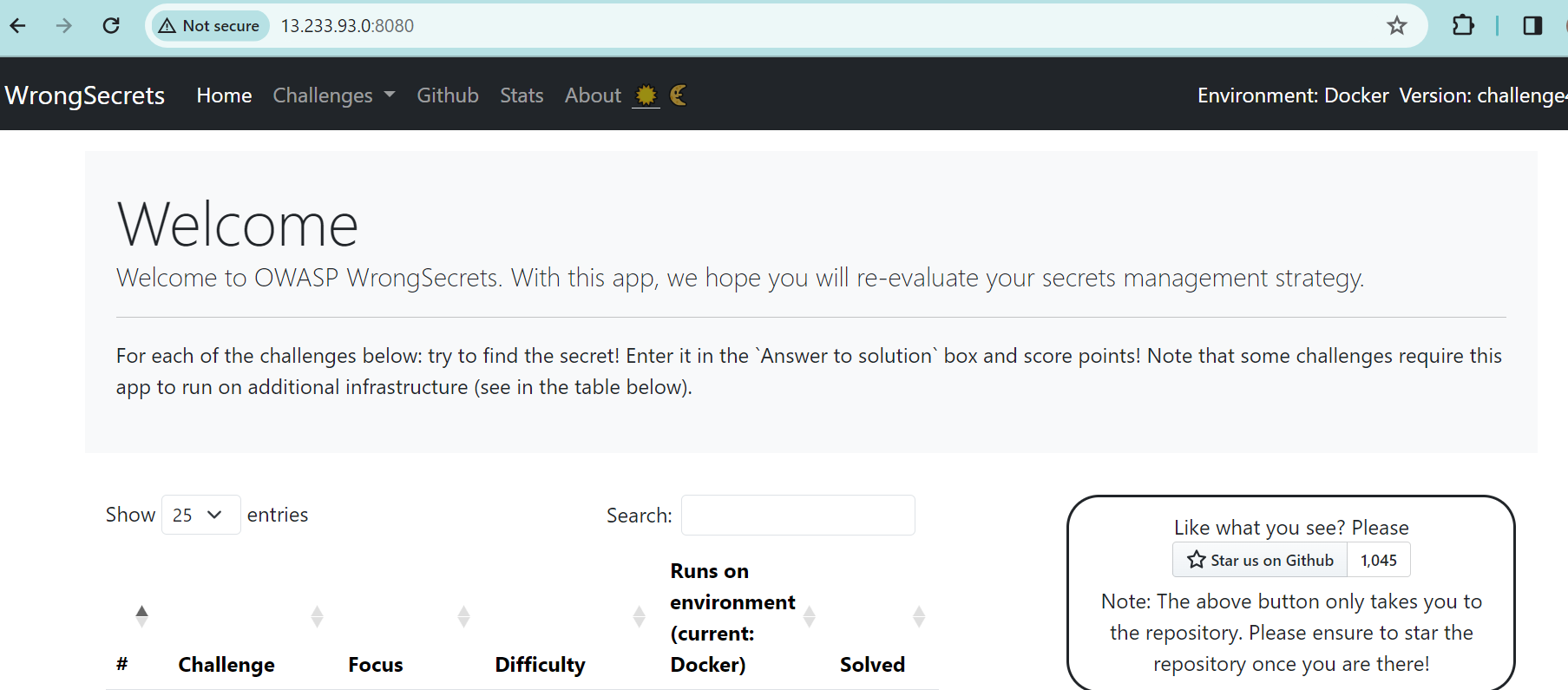


Step 3 : Docker Run

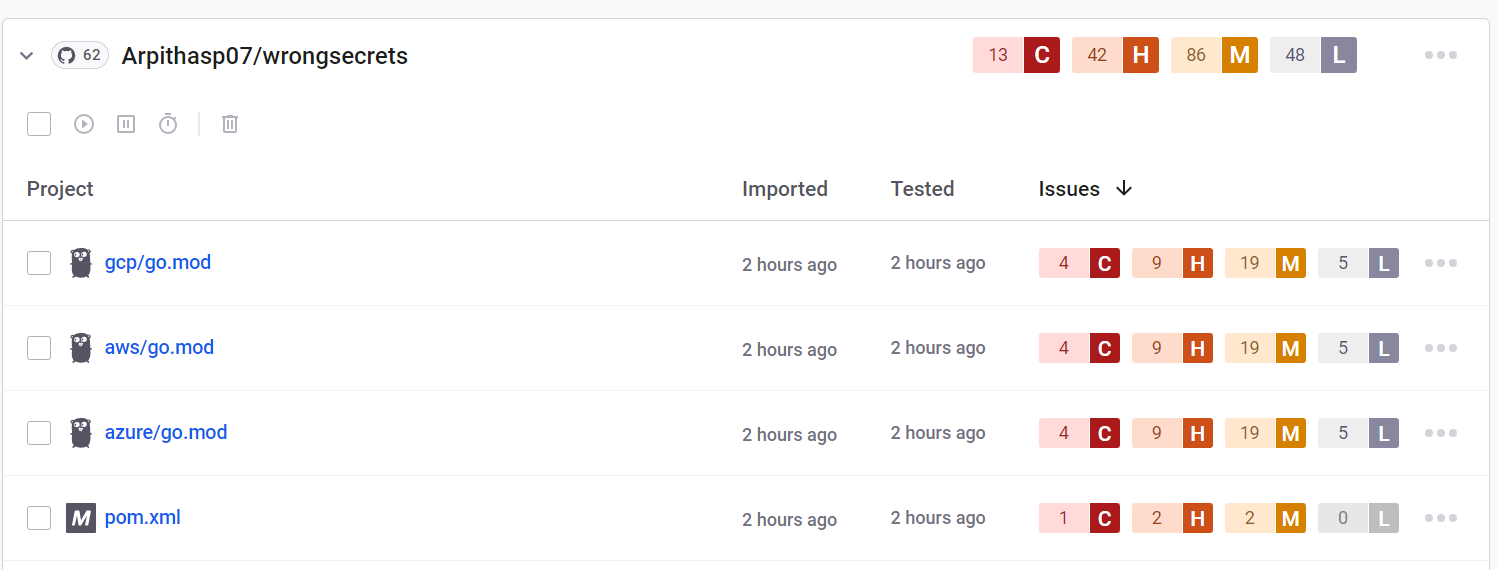








**Vulnerability Check using snyk:**



**Docker :**

Docker is a platform designed to make it easier to create, deploy, and run applications using containers. It provides

tools and platform to manage these containers, making it simpler to build and run applications consistently across

different environments.

**Docker run :** docker run is used to create and start a container based on a specific docker image.

docker run -p 8080:8080 jeroenwillemsen/wrongsecrets:latest-no-vault – docker run initiates the creation and launch of the container.

**‘-p 8080:8080’** : port mapping . It maps port 3000 of the host machine to port 3000 of the container which allows us

to access the application running inside the container via port 3000 on your local machine.

**Snyk :**

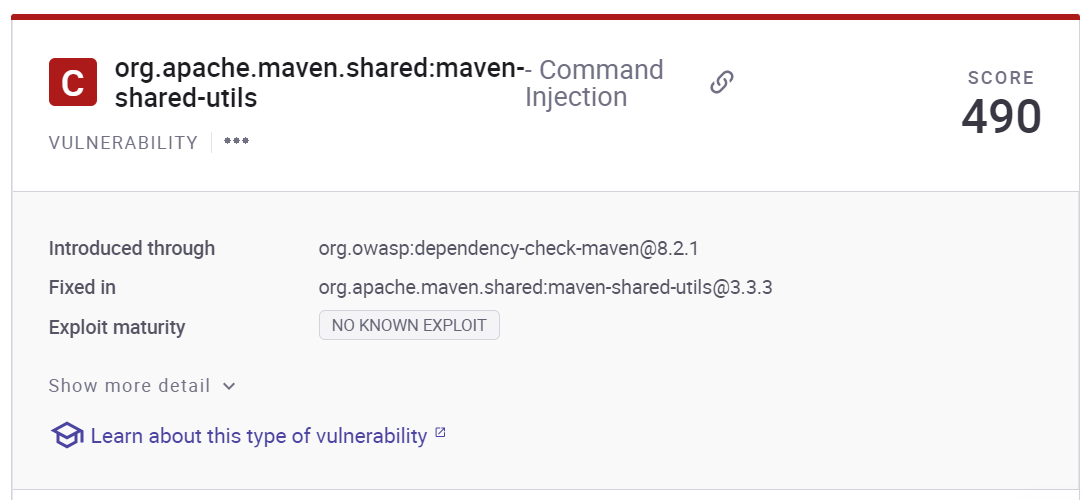
Snyk is a security tool used to help developers identify, fix, and prevent vulnerabilities in their applications.It scans

container images for known vulnerabilities and security issues.

**Observation :**

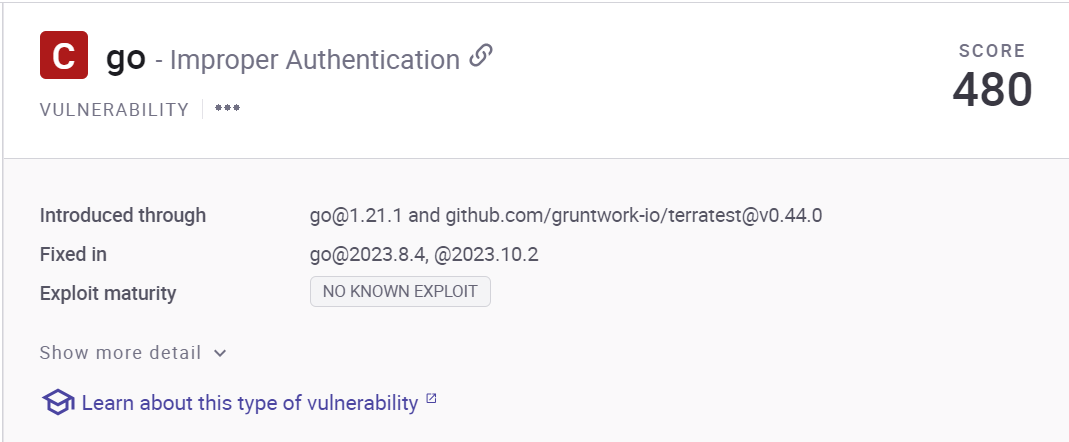
There are 13 Critical, 42 high, 86 medium and 48 Low issues.

CRITICAL :



Code injection is a type of attack that allows an attacker to inject malicious code into an application through a user input field, which is then executed on the fly. org.apache.maven.shared:maven-shared-utils is a functional replacement for plexus-utils in Maven. Affected versions of this package are vulnerable to Command Injection. The Commandline class can emit double-quoted strings without proper escaping, allowing shell injection attacks.

Solution : Avoid the use of dangerous functions In JavaScript, avoid the use of eval(), setTimeout(), setInterval() and the Function constructor, especially when dealing with user input. Perhaps the most straightforward of all prevention techniques is to reconsider the need to evaluate any dynamically generated server-side code.



Affected versions of this package are vulnerable to Improper Authentication such that token reuse in invitation URLs leads to access control bypass via the usage of a different enrollment flow than the one provided. 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.

Solution : In recent years, application frameworks have come a long way. Unfortunately, frameworks do not yet have the capability of automatically implementing permissions structures. Permissions structures still need to be implemented by the developer, because every application has specific, custom requirements.



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.