|  |  |
| --- | --- |
| Name  : | Muskan Shukla |
| Lab User ID: | 23SEK3324\_U14 |
| Date: | 09-01-2024 |
| Application Name: | Juice-shop Application |

**Follow the below guidelines:**



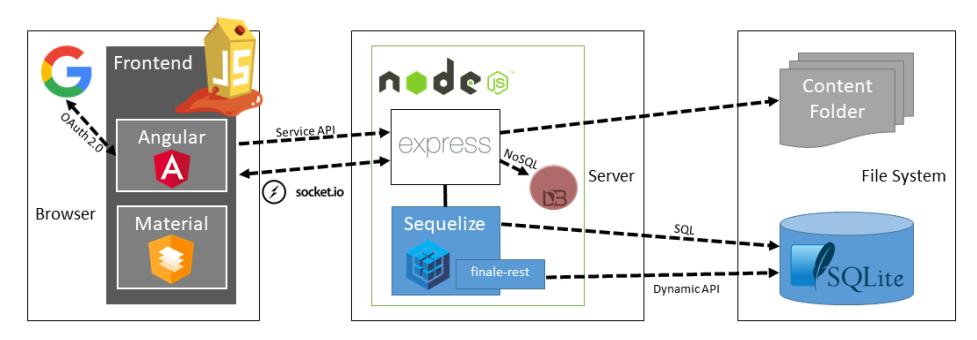


System Architecture:

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







OWASP Juice Shop System Architecture

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)

OWASP Juice Shop is a web application developed by OWASP (Open Web Application Security Project) which acts as a platform for security testing and awareness-training. The normal behavior of this application mirrors a typical online shopping website or e-commerce platform, but it is intentionally designed with security deficiencies for educational purposes.

Normal System Behavior:

1. User Registration: The system allows new users to sign up using an email and password.

2. User Login: Registered users can log in with their credentials.

3. Browse Products: Users can browse through the available products in the online store similar to any ecommerce platform.

4. Product Search: It has a search feature which allows users to locate specific products within the store.

5. View Product: Users can view a specific product to see more details including price, description, reviews and ratings.

6. Add to Basket: Users can add desired products to the shopping cart/basket.

7. Shopping Cart: Users can check, add, or remove items from their basket before proceeding to checkout.

8. Checkout: It allows users to make purchases for the items in their cart.

9. Feedback: Users can provide feedback or complaint about products or the store itself.

 10. User Profile: It allows users to update or change their personal information tied to their account.

 11. Score Board: This is a distinctive functionality of Juice Shop, where users can see the identified vulnerabilities that have been solved along with some hints about the unsolved ones.

 12. Forgotten Password: The application provides a feature to reset the password if the user forgot their password.

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**

OWASP Juice Shop presents a range of security vulnerabilities, understanding both why these vulnerabilities exist and how they can be exploited and defended might not be immediately clear.

When inducing latency into the database, the OWASP Juice Shop application will continue to function and successfully serve users with acceptable performance.

**Unknown**

With the introduction of random resource limitations like CPU and memory, the OWASP Juice Shop application will manage resource consumption efficiently, remain functional, and maintain performance.

The Juice Shop uses much different technology, like AngularJS, NodeJS, SQLite, and others. Some vulnerabilities in these tech stack yet to be ascertained.

**Unknown**

**Known**

Experiment:

(Document your Preparation, Implementation, Observation and Analysis )

**Preparation :-**

Establish an AWS account if you don’t already have one. Log in to the AWS Management Console and access the EC2 Dashboard.

Create an AWS t2.medium instance. Choose your desired OS (e.g. Ubuntu Server 20.04 LTS).

After the instance is successfully launched, connect to your instance via SSH through the command line or use an SSH client.

**Repository :-** [**https://github.com/juice-shop/juice-shop.git**](https://github.com/juice-shop/juice-shop.git)

**Implementation :-**

Update the system: Execute the command `*sudo apt update* && *sudo apt upgrade -y*` to ensure all the system packages are up to date.

Run the installation commands for Docker as ‘*sudo apt install docker.io -y’*

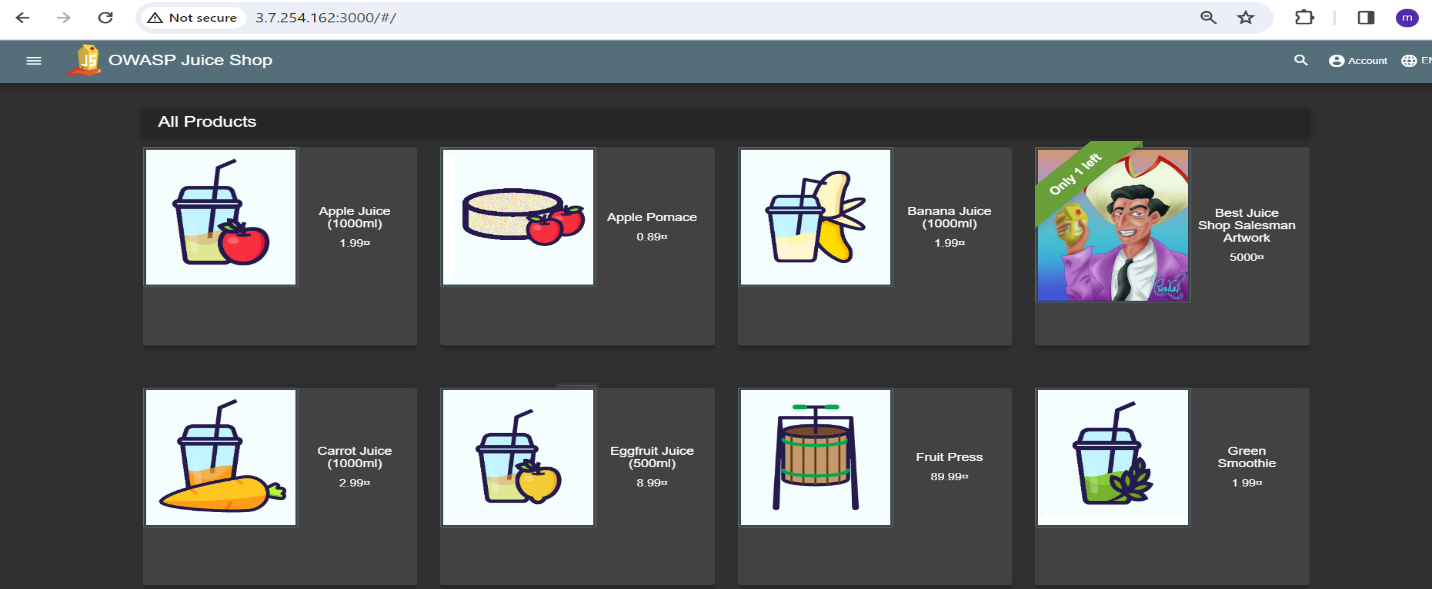
Build Docker image and run it accordingly.

***Command*** :

*docker pull bkimminich/juice-shop*

*docker run -d -p 80:3000 bkimminich/juice-shop*

Live the webserver.



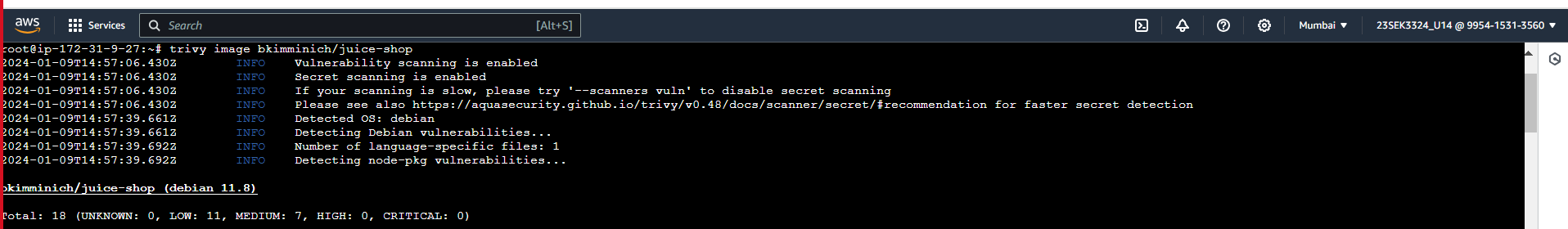
For this repository I am using some tools for security analysis on this Juice shop application.

1. TRIVY

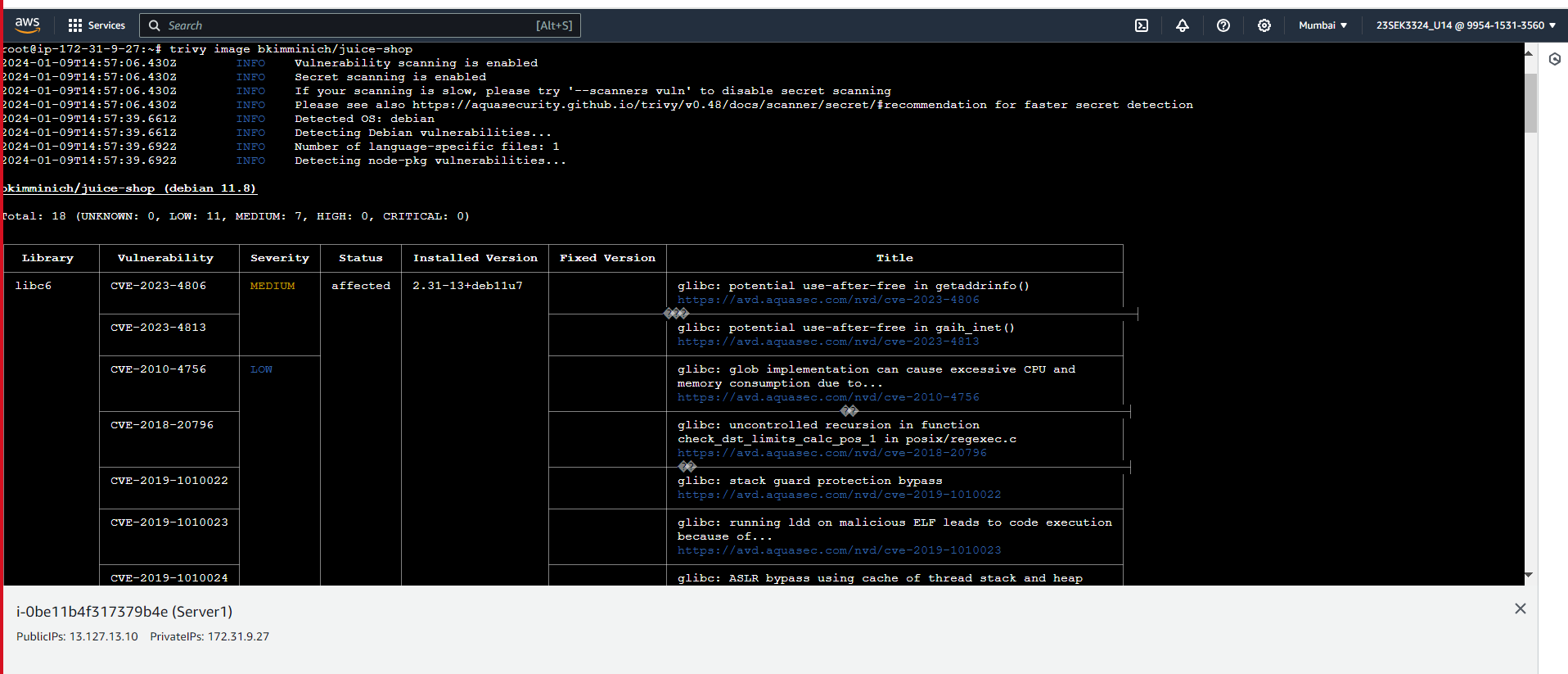
2. SNYK

1. **TRIVY :-**

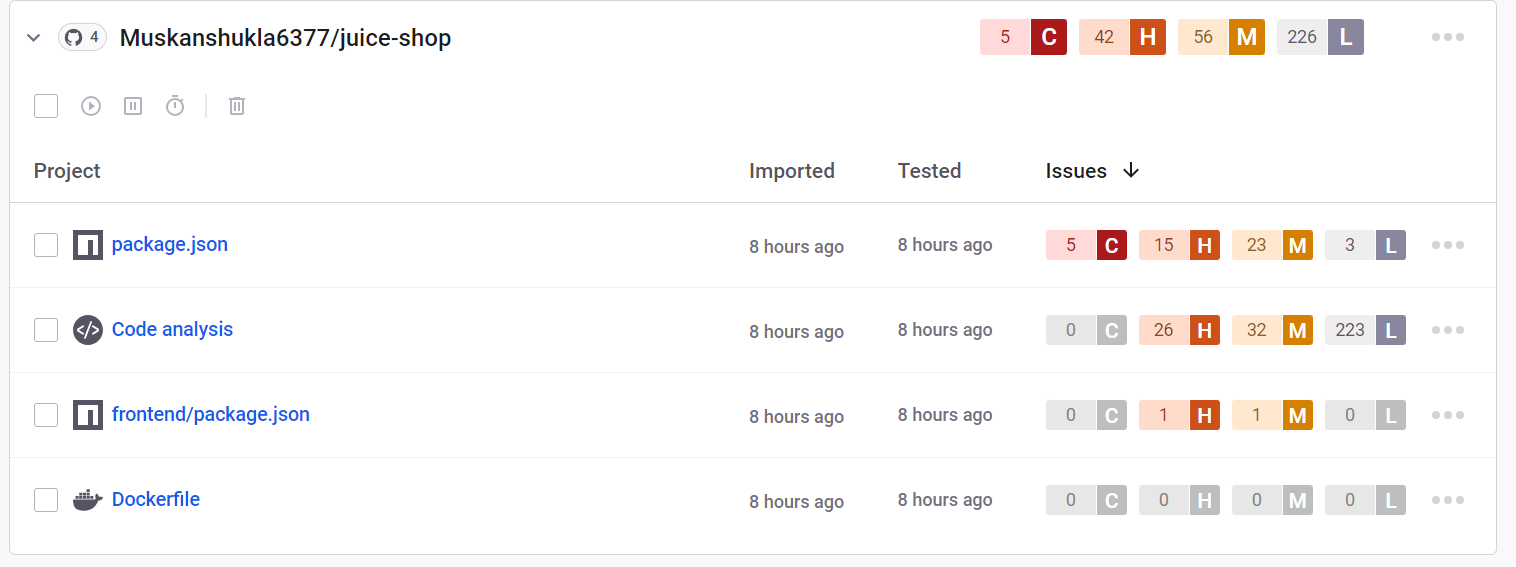
By using this command, we can find the vulnerability using this command ‘*trivy image bkimminich/juice-shop’*



After performing this trivy, I have got several security issues.



1. **SNYK :-** Performing SNYK



**Observation and Analysis :-** After performing SNYK, It has 5 critical, 56 medium, 42 high and 226 low vulnerability in the Juice shop repository. Describing some severity issues with solution-

-->***CVSS Score-9.4 -*** It is a critical severity and vulnerability is Code Injection. A code injection vulnerability has been identified in the application, allowing attackers to execute arbitrary code within the application's context. This vulnerability arises due to improper input validation and sanitization, enabling malicious actors to inject and execute unauthorized commands.

--> ***CVE-20210-3807 –*** It is a moderate vulnerability related to the "ansi-regex" package in the npm (node package manager) ecosystem. The vulnerability lies in the use of regular expressions (regex). The affected versions of this package are vulnerable to Inefficient Regular Expression Complexity attacks, leading to a Regular Expression Denial of Service, or "ReDoS." The vulnerability allows an attacker to cause a denial of service (DoS) condition. While it does not allow unauthorized disclosure of information or unauthorized modification, it can disrupt service and availability.

-->***CVE-2021-26540 -*** It is an Unrestricted Upload of File with Dangerous Type vulnerability. It resides in the "Edit Patient Picture" feature of Iris Clinic Management which does not sufficiently check file types during upload of patient pictures. Consequently, an attacker can exploit this weakness to upload files with malicious content (e.g., scripts or executables) that could be leveraged for further attacks.

-->***CVE-2016-1000223 -*** It is categorized as a catastrophic flaw that causes a cross-site scripting (XSS) vulnerability. The vulnerability may allow an attacker to inject malicious web scripts in the target users' browser. This injected script could then alter the appearance of the web page or steal sensitive information without the user's knowledge. Additionally, the flaw could disrupt the availability, confidentiality, and integrity of the infected environment, contributing to potential data breaches.