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

**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

13.232.110.42:3000

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)

* consistent presence of known vulnerabilities within the wordpress installation, such as outdated plugins, weak passwords, SQL injection.
* Deliberately configured settings or configurations that leave the site exposed.
* A stable and accessible environment purposely designed for users to practice identifying and exploiting vulnerabilities.
* Ongoing availability of info or documentation highlighting the vulnerabilities present in DVWA site for educational reference.

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.")



* Shared vulnerabilities that both site admin and user of DVWA site are aware of.
* Vulnerabilities known to the site admin but not yet discovered or understood by users.

**Known**

Things we are aware of but don’t understand.

Things we are aware of and understand.

* Newly discovered vulnerabilities or exploit methods that were previously unknown but now are identified and shared among users or admin of the DVWA site.

**Unknown**

* Potential yet-to-be discovered vulnerabilities or security weaknesses within the DVWA site that neither the admin nor the users are currently aware of.

**Unknown**

**Known**

Things we are neither aware of nor understand.

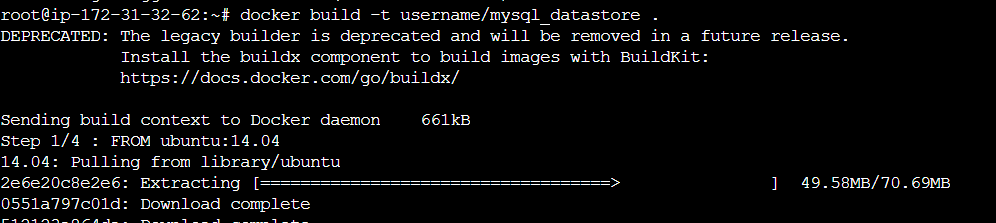
Things we understand but are not aware of.

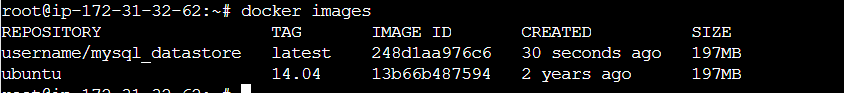
Experiment:

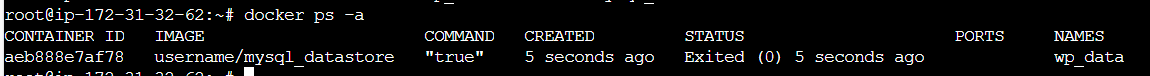
(Document your Preparation, Implementation, Observation and Analysis )

This is simply a Docker container with a vulnerable WordPress version and vulnerable plugins. This project is aimed for people who want to learn about penetration testing by playing with vulnerable WordPress setup.

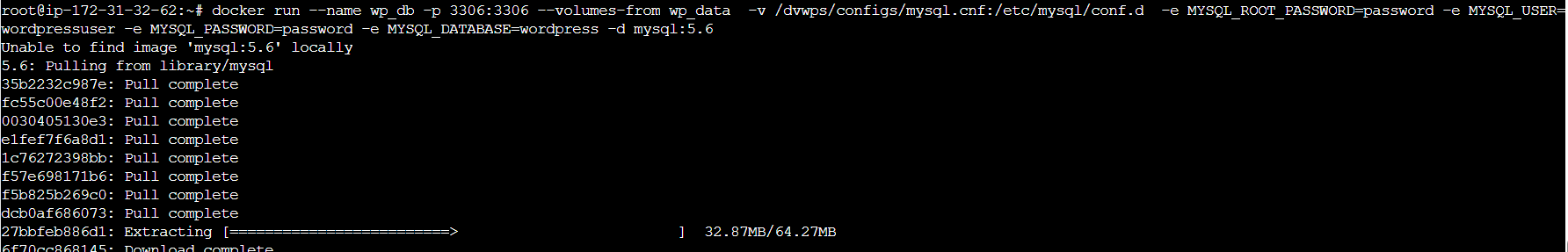
Step 1 : Docker build

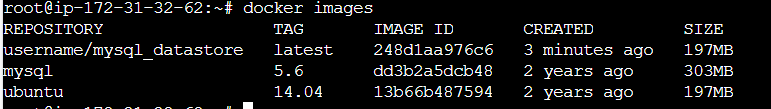


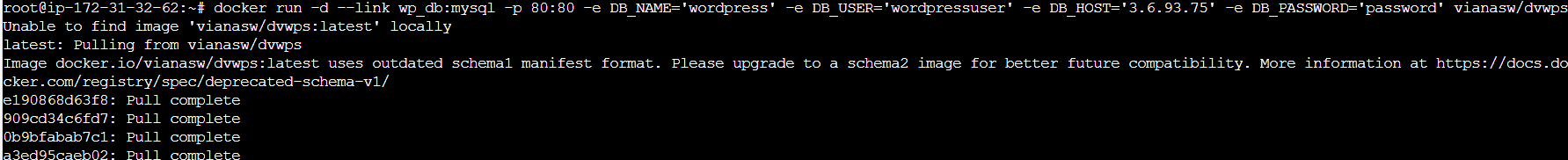


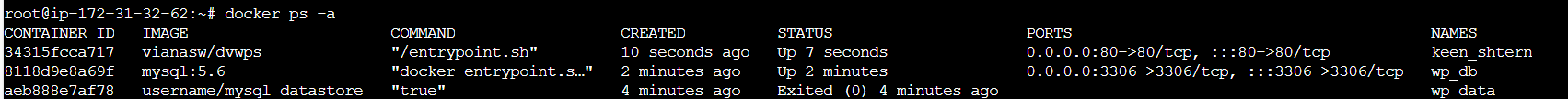


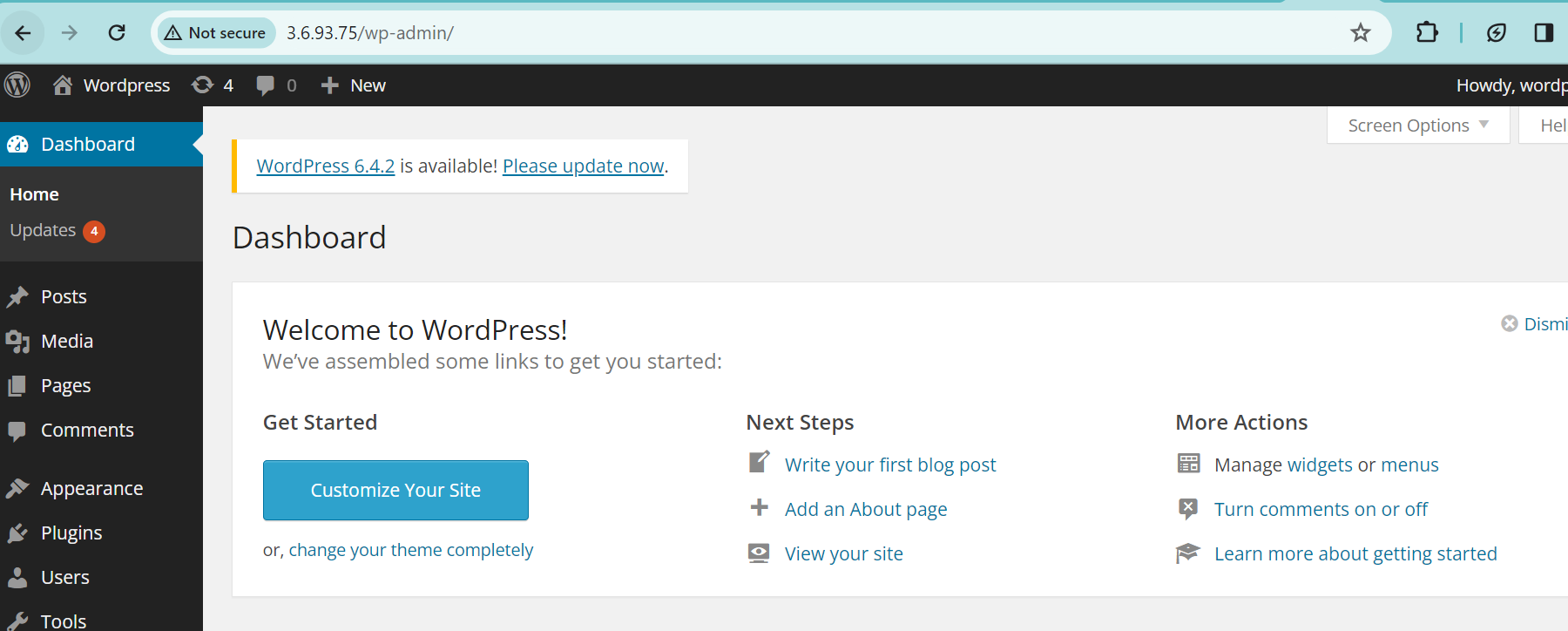
Step 2: Docker run



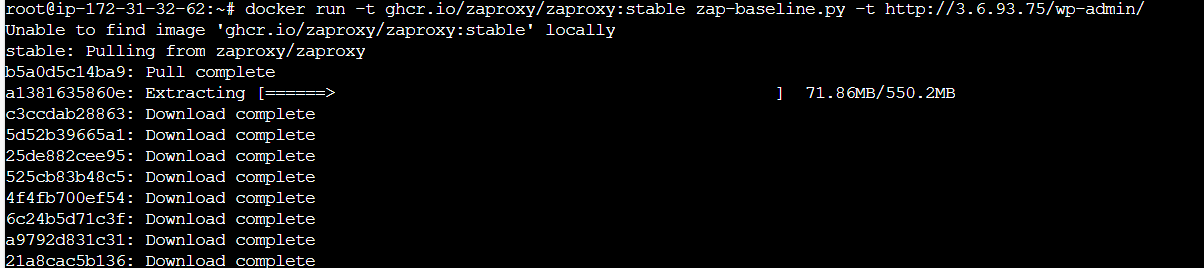


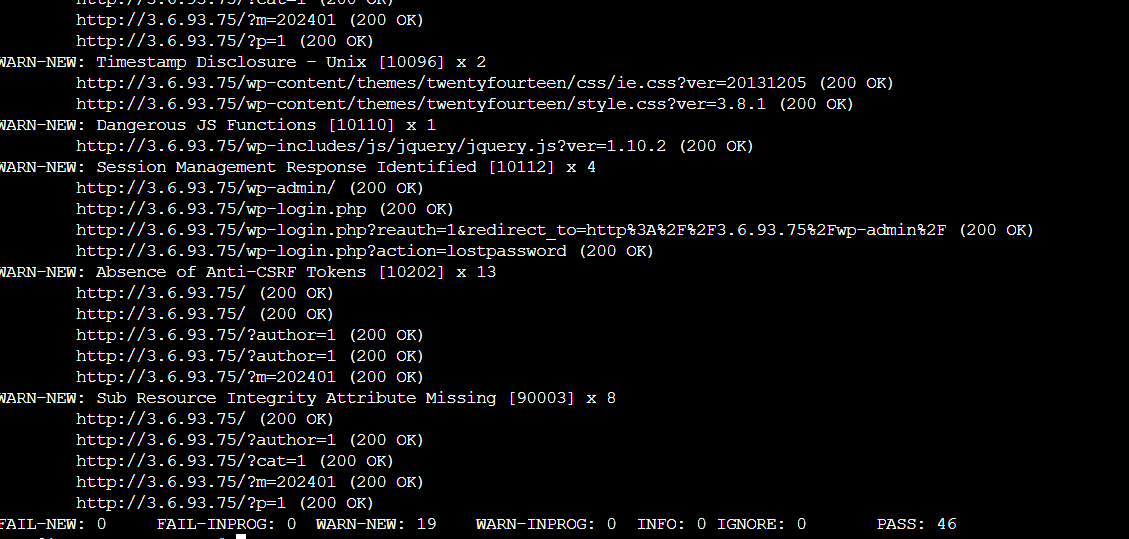




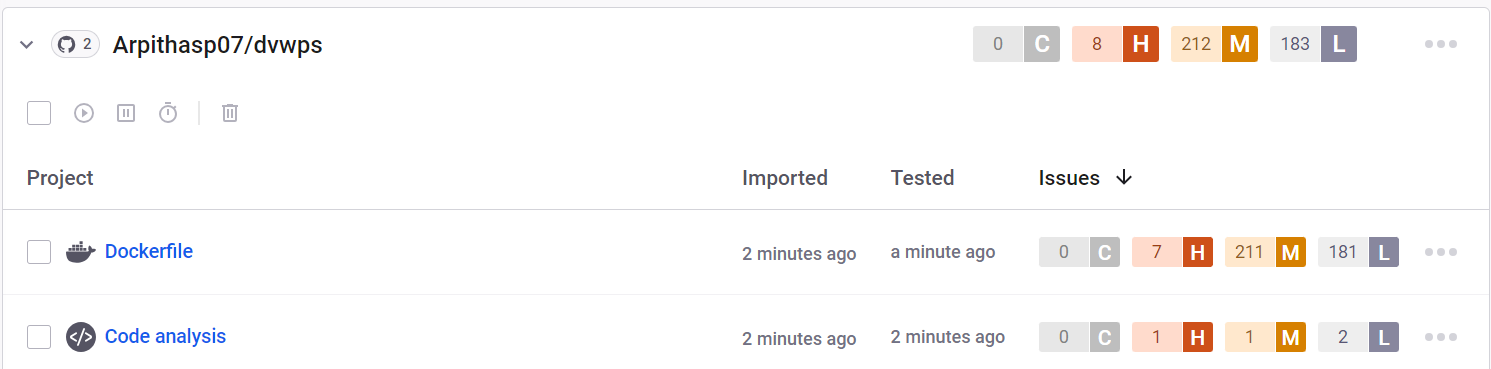


**Vulnerability Check using ZAP:**





**Vulnerability Scan 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.

**ZAP(Zed Attack Proxy) :**

It is an open-source web application scanner developed by OWASP(Open web application Security Project).

ZAP scans web applications to identify security vulnerabilities such as SQL Injection and more.

1. Timestamp Disclosure - 10096

A timestamp was disclosed by the application/web server.

Solution : Manually confirm that the timestamp data is not sensitive, and that the data cannot be aggregated to disclose exploitable patterns.

1. Dangerous JS Functions -10110

A dangerous JS function seems to be in use that would leave the site vulnerable.

Solution : Keep current with the latest Angular library releases. Don't alter your copy of Angular. Avoid Angular APIs marked in the documentation as "Security Risk".

**Snyk :**

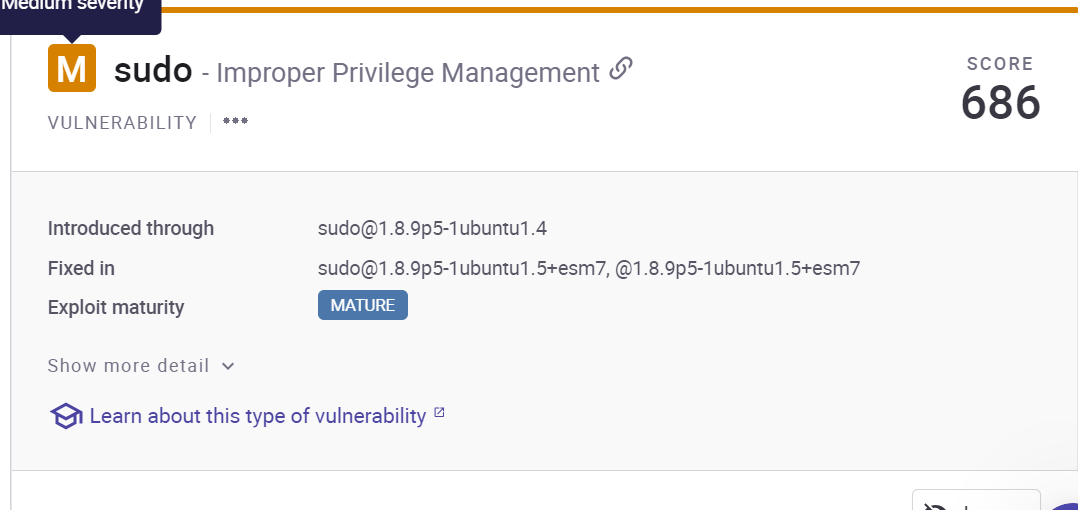
Snyk is a security tool used to help developers identify, fix, and prevent vulnerabilities in their applications.It scans

container images for known vulnerabilities and security issues.

**Observation :**

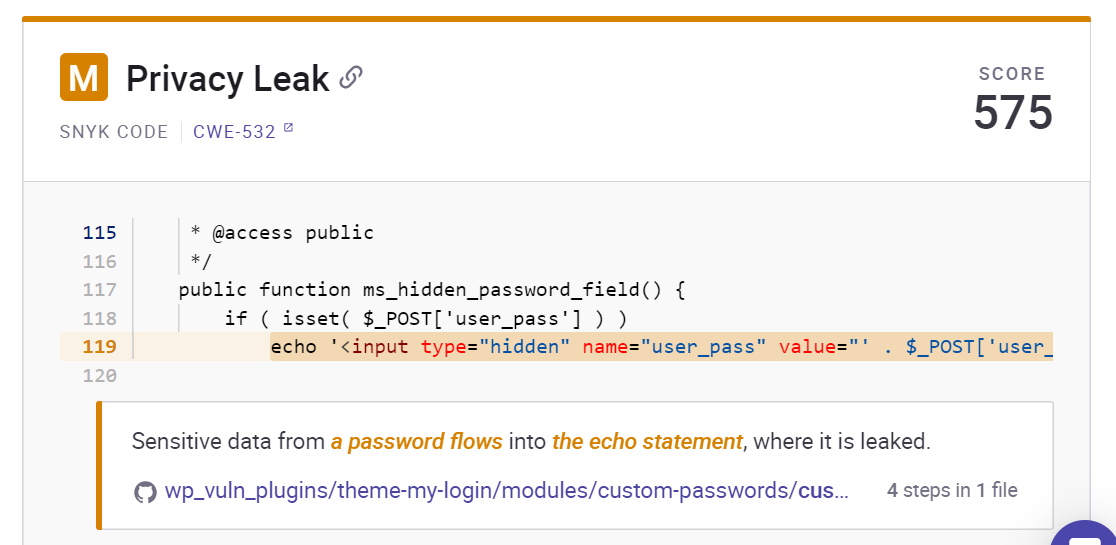
There are 7 High, 211 Meduim and 181 Low issues.

MEDIUM :



Improper Privilege Management : Insecure design is a type of flaw that can sit in the background of everything you do. This vulnerability relates to how you as a developer design your programs, architect solutions, and employ security practices such as threat modeling.

Solution: Have security be an active part of your development. Integrate security tools directly into your SDLC, read about vulnerabilities that could affect you, seek advice from security professionals. Testing your code helps identify general bugs with your programs, which can include issues with security. Having a robust test suite can ensure you minimize silly bugs which can put your application at risk.



Privacy Leak : Logging is the process of creating output logs based on actions within an application or service with the idea to provide information to understand possible issues like crashes or performance problems. Logging vulnerabilities are simply security vulnerabilities that arise from the process of logging.

Solution: Ensure that all sensitive or high-value events are logged and monitored for suspicious activity

Ensure that the method used for logging can not result in log entries being faked, tampered with, or removed

Ensure that logs are not ever accessible except to those who need to see them, like your security team.