Docker container-User namespaces

User namespaces in Docker containers provide a way to isolate the user and group IDs of processes running inside a container from those on the host system. This isolation is achieved through the use of mapped user and group IDs between the host and the container. The primary goal is to enhance security by limiting the potential for privilege escalation when applications run within containers.

Basic Concept

Docker's implementation of user namespaces allows the container runtime (e.g., Docker Engine) to manage a separate user namespace for each running container. This means that inside a container, the IDs of users and groups are virtual and do not directly map to the real-world user ID/GID on the host system. However, processes within the container can still use Unix permissions based on these virtual IDs.

How it Works

When you start a Docker container with `--userns-remap`, Docker maps the root user of the container to another UID and GID on the host, preventing the process from gaining privileges outside the container. This is similar in concept to creating a new user and group with a high UID/GID (like 65534) but not directly mapping it.

Example

Let's consider an example to demonstrate how user namespaces work within Docker containers:

1. **Host Configuration**:
   - Ensure you have a recent version of Docker installed. As of my last update, this is Docker 19.03 or higher.
   - You might need to adjust your Docker configuration to enable user namespace remapping. This involves setting `userns-remap` to `'always'` in the Docker daemon configuration (`/etc/docker/daemon.json`).
2. **Dockerfile**:

```
FROM alpine:latest

RUN groupadd -r test && \
    useradd -m -d /home/test -s /bin/bash -r test
```

3. **Building the Image**:

```
docker build -t myimage .
```

4. **Running the Container with User Namespace Remapping**:

```
docker run --rm -it \
   --userns-remap always \
   --cap-add=NET_ADMIN \
   myimage /bin/bash
```

5. **Inside the Container**:
   - You can use `id` to verify your user and group IDs have been remapped.

```
id
```

   This should show you're running with a different UID/GID than on the host.

Conclusion

User namespaces in Docker containers are a powerful tool for improving security by isolating container processes from host system processes. By creating separate virtual user and group IDs within each container, you can limit potential privilege escalations. However, configuring these features requires careful planning to ensure that application functionality isn't impacted unnecessarily.

Always remember to balance security with the operational needs of your applications.