Securing Docker with SELinux

Everything mentioned in this document was extracted from the resources provided below. The objective of this document is to provide you, the reader, with a succinct answer on how to secure Docker containers using SELinux

Sources:

https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux_atomic_host/7/html/container_security_guide/docker_selinux_security_policy

http://jaormx.github.io/2018/selinux-and-docker-notes/

Comprehensive man page:

https://www.mankier.com/8/docker_selinux

Summary

SELinux extends the same libvirt security policy that isolates VMs to containers running on Linux. Through the use of labels we can allow or block containers from accessing processes, files and system resources both on the host and other containers running on the same host.

Below are a few examples. The best way to get started and understand how SELinux works with Docker containers.

Examples

Check if SELinux is enabled

```
$ sestatus
```

Enable SELinux

```
$ sudo setenforce 1
```

Alternatively, you can update `/etc/selinux/config` and then `reboot`

Enable Docker SELinux Policy

```
$ semodule -v -e docker
```

Add the `--selinux-enabled` flag to `/usr/lib/systemd/system/docker.service`

Restart the docker daemon `$ systemctl stop docker`

Check if Docker is now using selinux `$ docker info | grep 'Security Options'`

You should see `Security Options: seccomp selinux`

Let's Put it Into Action

Use `ls -Z` to check the security context of a directory on the root directory of the host. Let's take a look at the security of the `tmp` directory

```
system_u:object_r:tmp_t:s0
```

Docker will get a permission denied if you try to do anything in that directory from the container. To show this, run a docker container and try to create a file in that directory:

```
$ docker run -ti -v /home:/host alpine sh
```

When we try try `echo "test" > /host/hello.txt` we get the following:

```
sh: can't create /host/tmp/text.txt: Permission denied
```

To fix this we can pass in an option when mounting the host directory and it will do the relabeling for us. This is well documented here (same link as above under sources)

So the fix is as follows

```
docker run -ti -v /home:/host-home:Z alpine sh
```

The new policy context is:

```
rw-r--r--. root   root   system_u:object_r:container_file_t:s0:c780,c937 test.txt
```

Multi Category Security (MCS)

From RedHat:

> *MCS - Multi-Category Security - this is similar to Multi-Level Authentication. Each container is given a unique ID at startup, and each file that a container writes carries that unique ID. Although this is an opt-in system, failure to make use of it means that you will have no isolation between containers. If you do not make use of MCS, you will have isolation between containers and the host, but you will not have isolation of containers from one another. That means that one container could access another container's files.*

Whether we leverage depends on the level of isolation we're seeking. However, it is possible to isolate container from host and a container from other containers.

Additional useful information:

> *By default, docker gets access to everything in **/usr** and most things in **/etc**. To give docker access to more than that, relabel content on the host. To restrict access to things in **/usr** or things in **/etc**, relabel them. If you want to restrict access to only one or two containers, then you'll need to use the opt-in MCS system.*

Docker Network Security and Routing:

> *By default, docker creates a virtual ethernet card for each container. Each container has its own routing tables and iptables. When specific ports are forwarded, docker creates certain host iptables rules. The docker daemon itself does some of the proxying. If you map applications to containers, you provide flexibility to yourself by limiting network access on a per-application basis. Because containers have their own routing tables, they can be used to limit incoming and outgoing traffic: use the ip route command in the same way you would use it on a host.*

For additional, in-depth explanations refer to the documentation at the beginning of this document.