Hardening Docker daemon with Rootless mode

Akihiro Suda

NTT Corporation

@_AkihiroSuda_

LIVE 2020

Rootless Docker

- Run Docker daemon as a non-root user on the host
- Protect the host from potential Docker vulnerabilities and misconfiguration

NTT

LIVE 2020

Why do we need rootless?

- Docker is designed to be safe by default
  - Namespaces, capabilities, cgroups, seccomp, AppArmor, SELinux...
- But there is no such thing as vulnerability-free software
- root-in-container could break out with an exploit

NTT

LIVE 2020

Why do we need rootless?

- **CVE-2019-5736:** A malicious container could replace the `runc` binary via `/proc/self/exe`

- **CVE-2019-14271**: Running `docker cp` against a malicious container could result in loading a malicious library onto the host

NTT

LIVE 2020

Why do we need rootless?

- And people often make misconfiguration!

- *"We found 3,822 Docker hosts with the remote API exposed publicly."*
  -- Vitaly Simonovich and Ori Nakar (March 4, 2019)
  https://www.imperva.com/blog/hundreds-of-vulnerable-docker-hosts-exploited-by-cryptocurrency-miners/

NTT

LIVE 2020

Why do we need rootless?

- Rootless mode per se doesn’t fix vulns and misconfigurations
- But it can mitigate attacks

NTT

LIVE 2020

Why do we need rootless?

- Even if the host gets compromised, the attacker won't be able to:
  - access files owned by other users
  - modify firmware and kernel (→ undetectable malware)
  - ARP spoofing (→ DNS spoofing)

NTT

LIVE 2020

Don’t confuse with...

```
$ sudo docker
```

NTT

LIVE 2020

Don’t confuse with...

```
$ sudo docker
```

```
$ usermod -aG docker <username>
```

NTT

LIVE 2020

Don’t confuse with...

```
$ sudo docker
$ usermod -aG docker <username>
$ docker run --user <uid>
```

NTT

LIVE 2020

Don’t confuse with...

```
$ sudo docker
$ usermod -aG docker <username>
$ docker run --user <uid>
$ dockerd --userns-remap
```

NTT

LIVE 2020

Don’t confuse with...

```
$ sudo docker
$ usermod -aG docker <username>
$ docker run --user <uid>
$ dockerd --userns-remap
```

All of them run the daemon as the root!

NTT

LIVE 2020

Don’t confuse with...

```
$ sudo docker
$ usermod -aG docker <username>
$ docker run --user <uid>
$ dockerd --userns-remap
```

All of them run the daemon as the root!

```
$ docker run -v /:/host
```

NTT

LIVE 2020

Demo

Getting started

Getting started

```
$ curl -fsSL https://get.docker.com/rootless | sh

$ export DOCKER_HOST=unix://$XDG_RUNTIME_DIR/docker.sock

$ docker run hello-world
```

NTT

LIVE 2020

Getting started

- sudo is NOT required
- Binaries are installed under `~/bin`
- The daemon can be start/stopped with
  `systemctl --user <start|stop> docker.service`

NTT

LIVE 2020

Getting started

There are some prerequisites, but the installer shows helpful guide if prerequisites are unsatisfied

- `/etc/subuid` and `/etc/subgid` need to configured
  - Typically configured by default
- Debian and CentOS 7 requires adjusting sysctl values

NTT

LIVE 2020

Katacoda scenario available!

https://www.katacoda.com/courses/docker/rootless

Katacoda

Rootless Docker

Step 2 of 4

Step 2 - Install Rootless Docker

Docker have made available a script which will deploy the required components for the new Rootless version.

Run the following command as *lowprivuser* to execute the script and install the components.

```
curl -sSL https://get.docker.com/rootless | sh ✓
```

After this has finished, proceed to the next step to setup the user environment and start launching containers.

CONTINUE

Terminal

```
lowprivuser@host01:~$ curl -sSL https://get.docker.com/rootless | sh
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100 59.8M  100 59.8M    0     0  1700k      0  0:00:36  0:00:36 --:--:-- 1575k
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
 39 14.0M   39 5675k    0     0   787k      0  0:00:18  0:00:07  0:00:11  787k
```

NTT

LIVE 2020

How it works under the hood

User Namespaces

```
[user@localhost] $ whoami
user
[user@localhost] $ unshare --user --map-root-user
[root@localhost] # whoami
root
```

NTT

LIVE 2020

User Namespaces

- Did I gain the root?

NTT

LIVE 2020

User Namespaces

```
[user@localhost] $ whoami
user
[user@localhost] $ unshare --user --map-root-user
[root@localhost] # whoami
root
[root@localhost] # touch /evil
touch: cannot touch '/evil': Permission denied
```

NTT

LIVE 2020

User Namespaces

- Did I gain the root?
  → **No!**

- It's just a "fake root" environment for emulating root privileges enough to run containers
  - Create other namespaces (mnt, net, uts, ipc, ...)
  - Change hostname
  - Mount bind-mount and tmpfs

NTT

LIVE 2020

User Namespaces

```
[user@localhost] $ whoami
user
[user@localhost] $ unshare --user --map-root-user
[root@localhost] # whoami
root
[root@localhost] # unshare --uts
[root@localhost] # hostname customhost
[root@customhost]# hostname
customhost
```

NTT

LIVE 2020

Snapshotting

- “Rootful” Docker uses OverlayFS for creating containers from an image without duplicating files
- But vanilla kernel doesn’t allow non-root users to use OverlayFS

NTT

LIVE 2020

Snapshotting

- **On Ubuntu kernel and Debian kernel:**
  OverlayFS is used

NEW! (Docker 20.0X)

- **On other distros w/ kernel >= 4.18:**
  FUSE-OverlayFS is used instead (if installed)

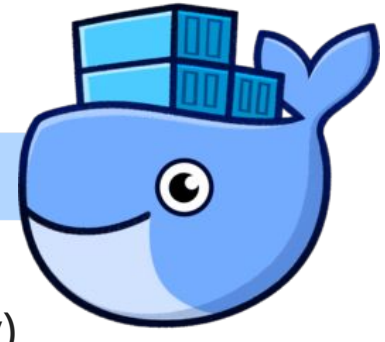
- **On older kernel:**
  files are just duplicated (“vfs” mode; slow and wasteful!)

NTT

LIVE 2020

Unprivileged networking

- Setting vEth interfaces require real root
- User-mode TCP/IP stack is used instead of vEth
  - **VPNKit** (spun out from MirageOS)
    - Also used by Docker for Mac/Win
  - **slirp4netns** (spun out from QEMU)
- SETUID helper (lxc-user-nic) is also experimentally supported for the best performance (sacrificing security)



NTT

LIVE 2020

Cgroup (--cpus, --memory, --pids-limit, ...)

NEW! (Docker 20.0X)

- Now rootless mode supports cgroups for limiting resources such as CPU and memory

- Requires cgroup v2 and systemd
  - **Fedora:** enabled by default since Fedora 31
  - **Others:** require kernel cmdline `systemd.unified_cgroup_hierarchy=1`

NTT

LIVE 2020

Caveats

- Unsupported features:
  - AppArmor
  - `docker checkpoint create`
  - `docker run --net=host`
  - SCTP ports
  - Overlay network (Swarm-mode)

NTT

LIVE 2020

FAQs

Q. Is rootless mode still experimental?

NEW! (Docker 20.0X)

- No, since the next version

NTT

LIVE 2020

Q. Is rootless mode the panacea?

- No
- If Docker had a vuln, attackers still might be able to:
  - Mine cryptocurrencies
  - Springboard-attack to other hosts
- Not effective for potential vulns on kernel / VM / HW side

NTT

LIVE 2020

Q. docker run -p 80:80 doesn't work?

- The port numbers below 1024 are called "privileged ports"

- Use unprivileged numbers (>= 1024) instead
  e.g. `docker run -p 8080:80`

NTT

LIVE 2020

Q. docker run -p 80:80 doesn't work?

- Or write "`0`" to `/proc/sys/net/ipv4/ip_unprivileged_port_start`
  - Default: `1024`
- Or set `CAP_NET_BIND_SERVICE` on `rootlesskit` binary

NTT

LIVE 2020

Q. Rootless Docker vs Rootless Podman?

- The two projects have been mutually exchanging a lot of codes for supporting rootless since 2018
  - **Basis and network:** Docker/Moby → Podman
  - **FUSE and Cgroup:** Docker/Moby ← Podman
- Almost same features
- Almost same performance

NTT

LIVE 2020

Q. Rootless Docker vs Rootless Podman?

- But the life cycles of the NetNS are different

- **Rootless Docker lacks:** `docker run --net=host`
- **Rootless Podman lacks:** `docker network create`

NTT

LIVE 2020

Recap

Recap

- Rootless mode protects the root from vulnerabilities and misconfigurations

NEW! (Docker 20.0X)

- Now out of experimental, with full support for cgroups (`--cpus`, `--memory`, ...)

```
$ curl -fsSL https://get.docker.com/rootless | sh
```

NTT

LIVE 2020

Further information

https://docs.docker.com/engine/security/rootless/

https://rootlesscontaine.rs/

NTT

LIVE 2020