

# Docker learnings

Learnings from using Docker after 3 years of usage

---

Jan Baer

30. April 2019



This talk is not about Kubernetes or Docker Swarm

## Security

---

- You shouldn't run your container as root

- You shouldn't run your container as root
  - Create user in Dockerfile

- You shouldn't run your container as root
  - Create user in Dockerfile
  - Take care about **uid**, map to an existing uid on the host if needed

- You shouldn't run your container as root
  - Create user in Dockerfile
  - Take care about **uid**, map to an existing uid on the host if needed
- Give only the privileges you really need



- You shouldn't run your container as root
  - Create user in Dockerfile
  - Take care about **uid**, map to an existing uid on the host if needed
- Give only the privileges you really need
  - [Docker documentation](#)

```
docker run -d --cap-drop CHOWN alpine
```

- You shouldn't run your container as root
  - Create user in Dockerfile
  - Take care about **uid**, map to an existing uid on the host if needed
- Give only the privileges you really need
  - [Docker documentation](#)

```
docker run -d --cap-drop CHOWN alpine
```

- Using **tmpfs** for sensitive data which shouldn't be saved outside of the container

```
docker run ... \  
--tmpfs /tmp/${CONTAINER_NAME}:uid=1000,gid=1000 \  
...
```

## Building images

---

- Use multistage builds

## How to build smaller images

- Use multistage builds
- Use build cache (copy package.json and yarn.lock in an extra step before yarn install)

- Use multistage builds
- Use build cache (copy package.json and yarn.lock in an extra step before yarn install)
- Remove developer node\_modules before copy

## Maintaining

---

- Remove dangling images with **docker image prune**
- Remove stopped containers with **docker container prune**
- Same for network and volume or all in once with **docker system prune**
- Autoremove a container after it's stopped with **docker -rm...**



Keep an eye on your logfiles

---

## How to grep the logs

```
docker logs {container} | grep {term}
```

This might will not work as expected

## How to grep the logs

```
docker logs {container} | grep {term}
```

This might not work as expected

This can happen if the container is logging to **stderr**. Piping works only for **stdout**. So you have to redirect **stderr** to **stdout** before you can pipe it to **grep**.

## How to grep the logs

```
docker logs {container} | grep {term}
```

This might not work as expected

This can happen if the container is logging to **stderr**. Piping works only for **stdout**. So you have to redirect **stderr** to **stdout** before you can pipe it to **grep**.

```
docker logs {container} 2>&1 | grep {term} | less
```

## Take care about the size of your Docker log files

- When using JSON File logging driver (which is the default)
  - Using `/etc/docker/daemon.json`

```
{  
  "log-driver": "json-file",  
  "log-opts": {  
    "max-size": "10m",  
    "max-file": "3"  
  }  
}
```

## Take care about the size of your Docker log files

- Or using commandline option

```
docker run --rm -it --log-opt max-size=10m alpine
```

- [docs.docker.com](https://docs.docker.com)

To delete all log files, you can use the following command

```
find /var/lib/docker/containers/  
-type f -name "*.log" -delete
```

More tips - Using a UI in the browser  
or terminal

---



## Use portainer locally without a password

```
alias portainer="docker run --rm -d  
-p 9000:9000  
--name=portainer  
-v /var/run/docker.sock:/var/run/docker.sock  
portainer/portainer --no-auth"
```

Hint: call it in an anonymus browser session

# Use sen as terminal ui to inspect the layers of your images

```
alias sen="docker run --rm --name=sen  
-v /var/run/docker.sock:/run/docker.sock  
-ti -e TERM tomastomecek/sen"
```

```
scratch  
→ fa5be2806d4c /bin/sh -c #(nop) MAINTAINER The CentOS Project <cloud-ops@centos.org>  
↳ 2bf4902415e3 /bin/sh -c #(nop) ADD file:c0989f72baa665000706f3a870a88fb4075ac62b77d79b148f7  
↳ 86bcb57631bd /bin/sh -c #(nop) LABEL name=CentOS Base Image vendor=CentOS license=GPLv2 l  
↳ c8a648134623 /bin/sh -c #(nop) CMD ["/bin/bash"]  
↳ 979d3e7d4dc7 /bin/sh -c #(nop) MAINTAINER Pavel Raiskup <praiskup@redhat.com>  
↳ d41e6e6bdfd8 /bin/sh -c #(nop) ENV container=docker  
↳ 235218c0d071 /bin/sh -c #(nop) LABEL INSTALL=docker run -t -i --rm --privileged :  
↳ 3f8341e3ed1b /bin/sh -c yum -y install postgresql-server && yum -y reinsta  
↳ 51ce5a01237d /bin/sh -c #(nop) ADD dir:1543912f127caa2263603d5f3ff11fddddfe4  
↳ 1efd5268689e /bin/sh -c systemctl disable getty.service console-getty.serv  
↳ 9efdb56ef4cc /bin/sh -c #(nop) VOLUME [/var/lib/pgsql/data]  
↳ 9100f952927c /bin/sh -c #(nop) EXPOSE 5432/tcp  
↳ 578145a76f11 /bin/sh -c #(nop) USER [postgres]  
↳ 55c64acbaef1 /bin/sh -c #(nop) ENTRYPOINT &{["/usr/bin/container-e  
↳ 9e6c06b57ed7 docker.io/praiskup/postgresql:latest /bin/sh -c #(nop)  
→ 6888fc827a3f /bin/sh -c #(nop) MAINTAINER Patrick Uiterwijk <puiterwijk@gmail.com>  
↳ 9bdb5101e5fc docker.io/fedora:23 /bin/sh -c #(nop) ADD file:bc5e5cdd4c4d1cac6f05788cf50  
→ 95612a3264fc  
↳ c23bf6e72b30 docker.io/rhel7/rsyslog:latest  
↳ ba3ffbc37ab docker.io/rhel7/sadcc:latest  
#0 [MJ Listing #1 LT
```

# Use ctop for monitoring your local containers

- github ctop

cTop - 10:04:15 AEDT 20 containers						
NAME	CID	CPU	MEM	NET RX/TX		
ultimate_jennifer	9eb1e9a6cb91	45%	136M / 2G	1K / 2K		
luminous_lady	cf754eb3aa09	42%	134M / 2G	2K / 1K		
top_notch	7b5fd634a980	36%	164M / 2G	2K / 2K		
neat_roulette	0677c5437698	20%	59M / 2G	667B / 698B		
exquisite_jackpot	72372078a320	16%	65M / 2G	786B / 780B		
ace_void	292db4fa7c5c	15%	59M / 2G	753B / 706B		
neat_multiple	5f8e5928be03	13%	51M / 2G	1020B / 800B		
peachy_sakura	70bd0664ff8c	12%	53M / 2G	710B / 749B		
legendary_korath	23a40fbbalcb	11%	58M / 2G	792B / 854B		
astounding_nikita	93de62c2b03f	-	-	-		
cats_pajamas	ec19826bd862	-	-	-		
fantabulous_titane...	8e23ebb7e05f	-	-	-		
grand_bebop	c3a231ea8f49	-	-	-		
impressive_sentine...	e154df51f8e6	-	-	-		

Last tipp

---

## You won't need to be sudo always

To prevent to make all Docker related operations with the root user, add yourself to the Docker group

```
sudo usermod -a -G docker $(whoami)
```

Thank you!