17.3.2024.

Jos o namespaceovima:

- https://www.xmodulo.com/remove-network-namespaces-linux.html
- https://learn-docker.it-sziget.hu/en/latest/pages/advanced/kernel-namespaces-network.html#two-containers-using-the-same-network-namespace

18.3.2024

https://docs.docker.com/engine/extend/plugin_api/

Docker discovers plugins by looking for them in the plugin directory whenever a user or container tries to use one by name.

There are three types of files which can be put in the plugin directory.

- `.sock` files are Unix domain sockets.
- `.spec` files are text files containing a URL, such as `unix:///other.sock` or `tcp://localhost:8080`.
- `.json` files are text files containing a full json specification for the plugin.

https://docs.docker.com/engine/extend/#developing-a-plugin

19.3.2024.

Kreiranje docker plugina koji se moze publishat

- trebam imati config.json file u rootfs direktoriju:
  - https://docs.docker.com/engine/extend/config/
- rootfs direktorij koji sadrzi samu plugin aplikaciju:
  - https://docs.docker.com/engine/extend/#the-rootfs-directory

```
git clone <git url plugina>
cd docker-volume-sshfs
docker build -t rootfsimage .
id=$(docker create rootfsimage true) # id was cd851ce43a403 when the image was created
sudo mkdir -p myplugin/rootfs
sudo docker export "$id" | sudo tar -x -C myplugin/rootfs
```

```
docker rm -vf "$id"
docker rmi rootfsimage
```

20.3.2024.

Dva nacina za napravit driver dostupnim docker daemonu:

- Napravit rootfs s gornjim komandama i config.js te upakirat s docker plugin create
- Napravit .spec ili .json s adresom "remote" drivera (localhost:xxxx) te ga pokrenuti zasebno koristeci systemd ili bilo koji drugi nacin (potrebno je pokrenuti ga prije docker daemona ili restartati daemon nakon pokretanja)