Biljeske 19.1.2024

Pitanja:

- dockeriziranje
  - svaki ECU je zasebni container?
  - sve u jednom containeru?
  - mogu li se u docker containerima uopce stvarati vcan sucelja bez privilegiranog nacina?
    - ako mogu, mozemo koristiti kombinaciju can gw-a i vxcana za povezivanje izmedju containera
    - inace nema smisla, bolje je samo stvoriti vcanove na hostu i raditi sve u host network namespaceu?
- sto zapravo korisnik simulatora treba moci vidjeti?
  - ako se koriste vcan sucelja na hostu za CAN sabirnice, korisnik ima pristup svim sabirnicama preko host network namespacea
    - mozda ne bi bilo lose korisnika odvojiti u zasebni container iz kojeg napada ili povezati samo sabirnice kojima ima pristup na vcan sucelje na hostu pomocu vxcana
  - treba imati na umu da se zapravo ne pokusavaju kompromitirati sami containeri nego simulirani ECU-i
    - nije cilj dobiti shell na containeru nego iskoristiti ranjivosti u programima ECU-a putem CAN-a i aplikacijskih protokola povrh CAN-a

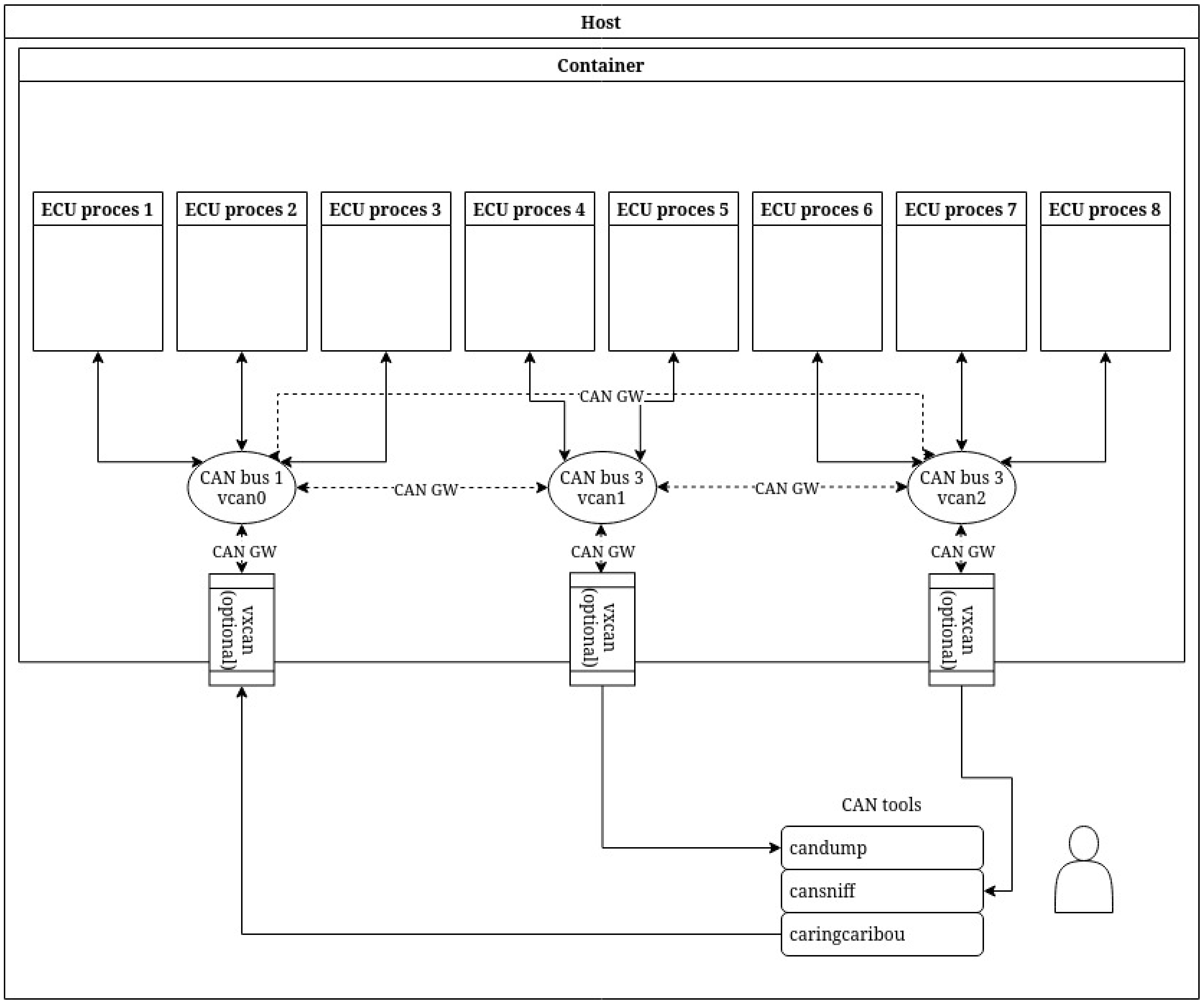
Neke bitne stvari:

- docker compose rjesava networking, ali samo za TCP/UDP
  - https://www.docker.com/blog/how-docker-desktop-networking-works-under-the-hood/
  - tuneliranje preko IP-a
    - https://agill.xyz/tunneling-can
    - docker network driver
      - https://gitlab.com/chgans/can4docker
      - svidja mi se ovaj pristup, ali je vise u PoC fazi
      - ~~u principu opet tuneliranje preko TCP-a, samo napravljeno kao docker network driver~~ GRESKA, nije tuneliranje!
        - nize opisano Docker network plugin za vxcan
  - ako se moze koristiti vcan, nema smisla koristiti CAN tunel preko IP-a
- kernel network namespaces
  - https://blogs.igalia.com/dpino/2016/04/10/network-namespaces/
  - https://blog.scottlowe.org/2013/09/04/introducing-linux-network-namespaces/
  - http://www.opencloudblog.com/?p=66
  - https://marc.info/?l=linux-can&m=149046502301622&w=2
  - https://developers.redhat.com/blog/2018/10/22/introduction-to-linux-interfaces-for-virtual-networking#vxcan
  - network namespaceovi docker containera su imenovani po PID-u containera

Moguce izvedbe drugi dio

Simulator u jednom containeru

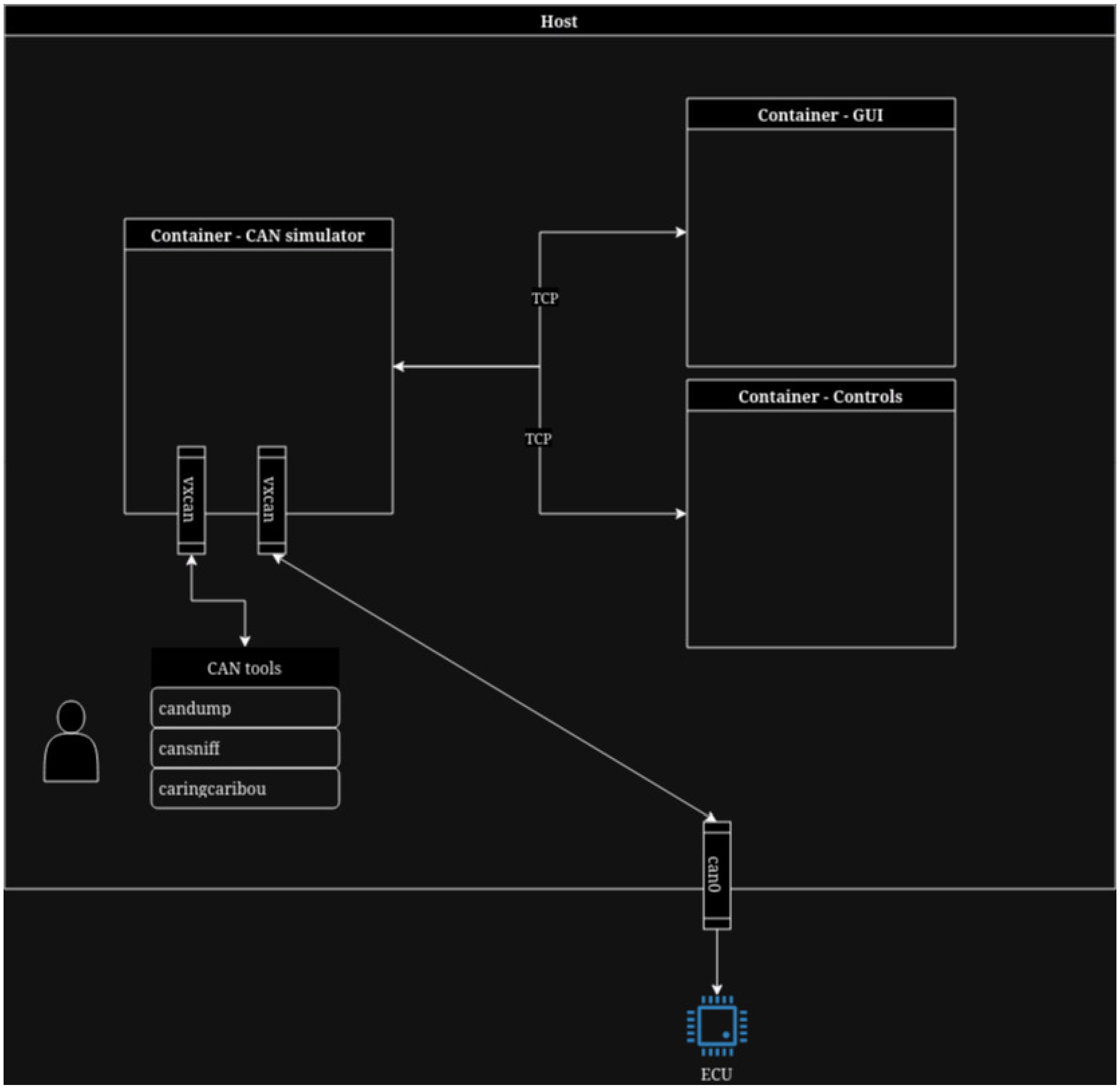
- simulator je unutar jednog containera, kao i vcan sucelja koja koristi kao sabirnice
- sabirnice (vcan sucelja) kojima korisnik treba imati pristup proslijedjena su vxcan-om
- cangw koristi se za povezivanje sabirnica i filtriranje poruka izmedju njih te za povezivanje vxcan i vcan sucelja



Modularno povezivanje GUI-a i kontrola:

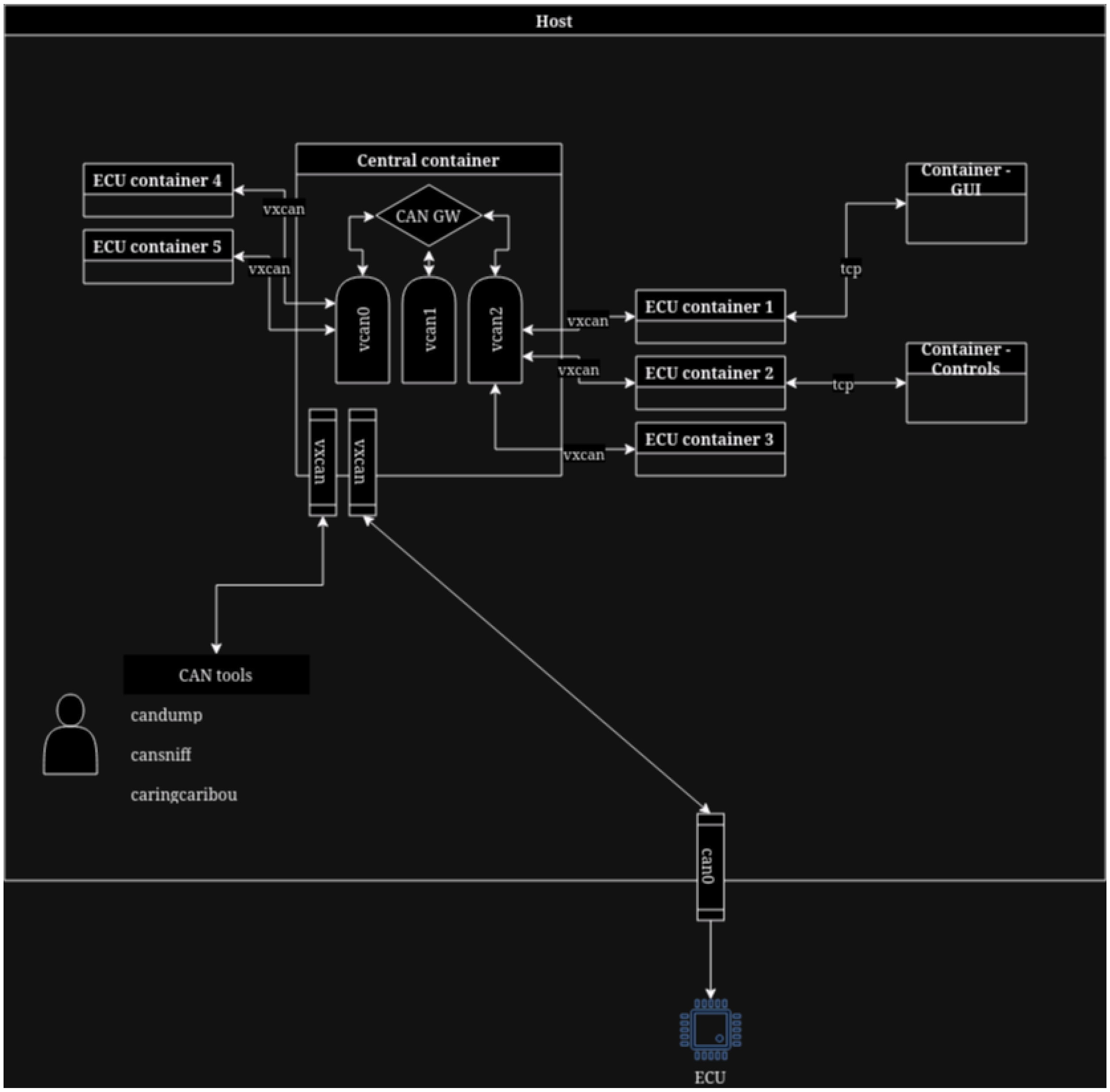
- primjerice AGL ploca s instrumentima
- moze se koristiti docker compose i bridge izmedju containera koji stvara
- svaki container koji zeli komunicirati s GUI-jem moze to dodatno implementirati
    - primjerice, ECU motora moze komunicirati brzinu AGL ploci s instrumentima

Dodatno se moze povezati pravi hardver na simulator ako hardver ima socketcan driver.



Svaki ECU u svom containeru

- svaki ECU u svom containeru, svaki container ima specificiran binary koji treba pokretat
- sve se podize s docker composeom
  - CAN komunikacija zasebnom skriptom/programom prema konfiguraciji (podizanje sucelja, can gw)
  - tcp komunikaciju podize docker compose

- 

Testiranje docker + vcan + vxcan

https://stackoverflow.com/questions/42964839/what-is-the-most-light-weight-base-image-i-can-use-to-build-a-dockerfile

Alpine: This is a minimal distribution, based on busybox, but with the `apk` package manager. The small size comes at a cost, things like glibc are not included, preferring the musl libc implementation instead. You will find that many of the official images are based on Alpine, so inside of the container ecosystem, this is a very popular option.

https://pkgs.alpinelinux.org/packages?name=can-utils

- alpine ima can-utils preko apk-a i lightweight je

```
sudo docker pull alpine
docker run -dit alpine
docker attach <ime-containera>
```

```
apk add can-utils
```

vcan u containeru

Stvaranje i podizanje virtualnog can sucelja putem naredbenog retka:

```
ip link add dev vcan0 type vcan
ip link set dev vcan0 up
```

Spustanje i brisanje virtualnog can sucelja putem naredbenog retka:

```
ip link set dev vcan0 up
ip link delete dev vcan0
```

Dodavanje vcan sucelja ne radi po defaultu:

```
b535ea53f8b0:/# sudo ip link add dev vcan0 type vcan
ip: RTNETLINK answers: Operation not permitted
```

Potrebno je dodati NET_ADMIN capability

- https://docs.docker.com/engine/reference/run/#runtime-privilege-and-linux-capabilities
- For interacting with the network stack, instead of using `--privileged` they should use `--cap-add=NET_ADMIN` to modify the network interfaces.

i onda sve radi

stvoreno vcan0 sucelje se ne vidi na hostu, kao sto se ni vcan sucelja hosta ne vide u containeru

popisivanje network namespaceova:

```
lsns --type=net
```

Isprobavanje mreze - svaki ECU zasebni container

Stvaranje i podizanje vxcan sucelja putem naredbenog retka:

```
sudo ip link add vxcan2 type vxcan peer name vxcan3
sudo ip link set vxcan2 up
sudo ip link set vxcan3 up
```

Spustanje i brisanje virtualnog can sucelja putem naredbenog retka:

```
ip link set vxcan2 down
ip link delete vxcan2
```

Premicanje sucelja izmedju namespaceova:

```
sudo ip link set vxcan3 netns <namespace_id>
```

Izvodjenje ip naredbe u drugom namespaceu:

```
sudo nsenter -t <namespace_id> -n ip link set vxcan3 up
```

setup_gw.sh skripta postavlja "sabirnice", odnosno podize tri vcan sucelja

Postavljanje komunikacije izmedju containera pomocu vxcan-a

Pokusaj stvaranja vxcan para sucelja unutar ecu1 kako bi se povezala vcan sucelje izmedju gw i ecu containera:

```
❯ docker attach ecu_containers_poc-ecu1-1
362481e92ae3:/# ip link add vxcan2 type vxcan peer name vxcan3
362481e92ae3:/# ip link
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
2: vxcan0@vxcan2: <NOARP,M-DOWN> mtu 72 qdisc noop state DOWN qlen 1000
    link/[280]
3: vxcan2@vxcan0: <NOARP,M-DOWN> mtu 72 qdisc noop state DOWN qlen 1000
    link/[280]
76: eth0@if77: <BROADCAST,MULTICAST,UP,LOWER_UP,M-DOWN> mtu 1500 qdisc noqueue state UP
    link/ether 02:42:ac:16:00:04 brd ff:ff:ff:ff:ff:ff
362481e92ae3:/# ip link set vxcan2 up
362481e92ae3:/# ip link
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
2: vxcan0@vxcan2: <NOARP> mtu 72 qdisc noop state DOWN qlen 1000
    link/[280]
3: vxcan2@vxcan0: <NO-CARRIER,NOARP,UP,M-DOWN> mtu 72 qdisc noqueue state LOWERLAYERDOWN qlen 1000
    link/[280]
76: eth0@if77: <BROADCAST,MULTICAST,UP,LOWER_UP,M-DOWN> mtu 1500 qdisc noqueue state UP
    link/ether 02:42:ac:16:00:04 brd ff:ff:ff:ff:ff:ff
362481e92ae3:/# ip link set vxcan3 up
ip: ioctl 0x8913 failed: No such device
362481e92ae3:/# ip link set up vxcan0
362481e92ae3:/# ip link
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
2: vxcan0@vxcan2: <NOARP,UP,LOWER_UP> mtu 72 qdisc noqueue state UP qlen 1000
    link/[280]
3: vxcan2@vxcan0: <NOARP,UP,LOWER_UP> mtu 72 qdisc noqueue state UP qlen 1000
    link/[280]
76: eth0@if77: <BROADCAST,MULTICAST,UP,LOWER_UP,M-DOWN> mtu 1500 qdisc noqueue state UP
    link/ether 02:42:ac:16:00:04 brd ff:ff:ff:ff:ff:ff
362481e92ae3:/# ip link set vxcan2 netns 148991
ip: RTNETLINK answers: No such process
362481e92ae3:/# ps ux
PID   USER     TIME  COMMAND
    1 root      0:00 /bin/bash
   15 root      0:00 ps ux
362481e92ae3:/#
```

Vise problema:

1. `ip link add vxcan2 type vxcan peer name vxcan3` komanda se cudno ponasa
    1. vxcan3 nije stvoren
    2. stvorena su dva sucelja vxcan0@vxcan2 i vxcan2@vxcan0
2. nemoguce premicanje jednog kraja vxcan tunela u namespace drugog containera jer se medjusobno "ne vide"
    1. ip link set vxcan2 netns 148991
        - 148991 je PID gw containera kojeg ecu-1 container ne vidi

---

Uspjesno postavljanje vxcan-a s host racunala:

Na host racunala:

```
$ sudo ip link add gw_vxcan type vxcan peer name ecu_vxcan
$ sudo ip link set ecu_vxcan netns 236978 // pid ecu containera
$ sudo ip link set gw_vxcan netns 236955 // pid gateway containera
$ sudo nsenter -t 236955 -n ip link set host_vxcan up
$ sudo nsenter -t 236978 -n ip link set ecu_vxcan up
```

Problem kod koristenja cangw-a za spajanje vxcana sa vcan "sabirnicom" unutar gw containera:

```
99344ffdde4c:/# cangw -A -X -s gw_vxcan1 -d vcan1
netlink error -95 (Not supported)
```

~~Pokusao modprobe, ali nemam can-gw modul u kernelu~~

```
❯ modprobe can-gw
modprobe: FATAL: Module can-gw not found in directory /lib/modules/6.7.3-arch1-2
```

~~Pokrenut cu sve unutar virtualke da izbjegnem rekompajliranje kernela...~~

Test #1

cangen na ecu1, candump na gw
cangw pravila na ecu:

```
35f0d2b85988:/# cangw -L
cangw -A -s vcan0 -d ecu_vxcan # 0 handled 0 dropped 0 deleted
```

cangw pravila na gw:

```
cangw -A -s gw_vxcan1 -d vcan1 # 0 handled 0 dropped 0 deleted
```

nakon pokretanja cangen na vcan0 na ecu 1, nista ne dobivam na cangw na vcan1

ostavio sam za sad to, idem probat can4docke plugin, odnosno njegovu bugfixanu verziju docker-vxcan Docker network plugin:
https://gitlab.com/chgans/can4docker
https://github.com/wsovalle/docker-vxcan/tree/master

Test #2

cangen na ecu1, candump na gw
(cangen -> vcan0 -> cangw -> ecu_vxcan) -> (gw_vxcan1 -> cangw -> vcan1 -> candump)

Dodavanje -i zastavice -> proradila komunikacija

- objasnjenje -i zastavice
  - https://github.com/linux-can/can-utils/issues/337
  cangw pravila na ecu:

```
35f0d2b85988:/# cangw -L
cangw -A -s vcan0 -d ecu_vxcan -X -e -i# 0 handled 0 dropped 0 deleted
```

cangw pravila na gw:

```
cangw -A -s gw_vxcan1 -d vcan1 -X -e -i # 0 handled 0 dropped 0 deleted
```

Radi i dvosmjerna komunikacija izmedju gw i ecu:
ecu pravila:

```
^Cfecef03ba967:/# cangw -L
cangw -A -s ecu_vxcan -d vcan0 -X -e -i # 19 handled 0 dropped 113 deleted
cangw -A -s vcan0 -d ecu_vxcan -X -e -i # 338 handled 0 dropped 19 deleted
```

gw pravila:

```
^Cad4387e1acd5:/# cangw -L
cangw -A -s vcan1 -d gw_vxcan1 -X -e -i # 0 handled 0 dropped 113 deleted
cangw -A -s gw_vxcan1 -d vcan1 -X -e -i # 420 handled 0 dropped 0 deleted
```

Test #3

komunikacija izmedju dva ecu containera preko zajednicke sabirnice

komunikacija prema gw-u radi, ali se brisu kada trebaju biti poslani prema drugom containeru:

```
ad4387e1acd5:/# cangw -L
cangw -A -s gw_vxcan1 -d vcan1 -X -e -i # 0 handled 0 dropped 0 deleted
cangw -A -s vcan1 -d gw_vxcan1 -X -e -i # 0 handled 0 dropped 569 deleted
cangw -A -s vcan1 -d gw_vxcan2 -X -e -i # 0 handled 0 dropped 569 deleted
cangw -A -s gw_vxcan2 -d vcan1 -X -e -i # 569 handled 0 dropped 0 deleted

```

Zadnje pravilo pokazuje da paketi dolaze do vcan1 (i tamo se mogu procitati), ali forward na gw_vxcan1 po drugom pravilu prikazuje 569 deleted frameova.

Nakon duljeg istrazivanja naisao sam na max_hops postavku can_gw modula:

- https://marc.info/?l=linux-can&m=137752245213501&w=2
  sto se vidi i u modinfo ispisu:

```
❯ modinfo can-gw
filename:       /lib/modules/6.7.4-arch1-1/kernel/net/can/can-gw.ko.zst
alias:          can-gw
author:         Oliver Hartkopp <oliver.hartkopp@volkswagen.de>

...

parm:           max_hops:maximum can-gw routing hops for CAN frames (valid values: 1-6 hops, default: 1)
(uint)

```

Default je 1 hop.

Promjena broja hopova:

```
sudo modprobe can-gw max_hops=6
```

Ali sada zbog zastavice -i i veceg broja hopova dolazi do cirkulranih veza tako da se poruke ponavljaju.

treba nekako izbjeci kruzno proslijedjivanje

Docker network plugin

- docker omogucava pisanje network drivera u obliku lokalnog http servera koji odgovara na predefinirane api callove
  - https://www.inovex.de/de/blog/docker-plugins/
  - https://docs.docker.com/engine/extend/plugins_network/
  - tehnicka dokumentacija
    - https://github.com/moby/moby/blob/master/libnetwork/docs/remote.md
    - https://github.com/moby/moby/blob/master/libnetwork/docs/design.md
    - https://test-dockerrr.readthedocs.io/en/latest/userguide/networking/dockernetworks/
    - https://test-dockerrr.readthedocs.io/en/latest/extend/plugins_network/
  - https://github.com/docker/go-plugins-helpers/blob/master/network/api.go driver go template sa dockerovim sdk-om
  - https://github.com/tugbadm/docker-network-plugin/tree/master/example
- primjerice na create network http request od docker daemona, driver bi mogao stvoriti vcan sucelje kao, na attach endpoint request mogao bi povezati ecu container pomocu vxcan-a
  - netko se tog vec sjetio te je napravljen can4docker plugin
    - https://agill.xyz/tunneling-can - clanak gdje sam ga pronasao
    - linux-can mailing lista
      - https://www.spinics.net/lists/linux-can/msg00297.html
      - https://www.spinics.net/lists/linux-can/msg00306.html
    - izvorni kod
      - https://gitlab.com/chgans/can4docker
      - https://github.com/wsovalle/docker-vxcan/tree/master - doradjena verzija

docker-vxcan/can4docker

https://github.com/wsovalle/docker-vxcan/tree/master
intalacija
`$ docker plugin install wsovalle/vxcan`

radi slicno kao ono sto mi treba za simulator:

```
        +-------+
        |       |                          +-------+
        |       |>vxcan0.1----vxcan0.0<| cont1 |
vcan0<| CANGW |                          +-------+
        |       |                          +-------+
        |       |>vxcan1.1----vxcan1.0<| cont2 |
        +-------+                          +-------+
```

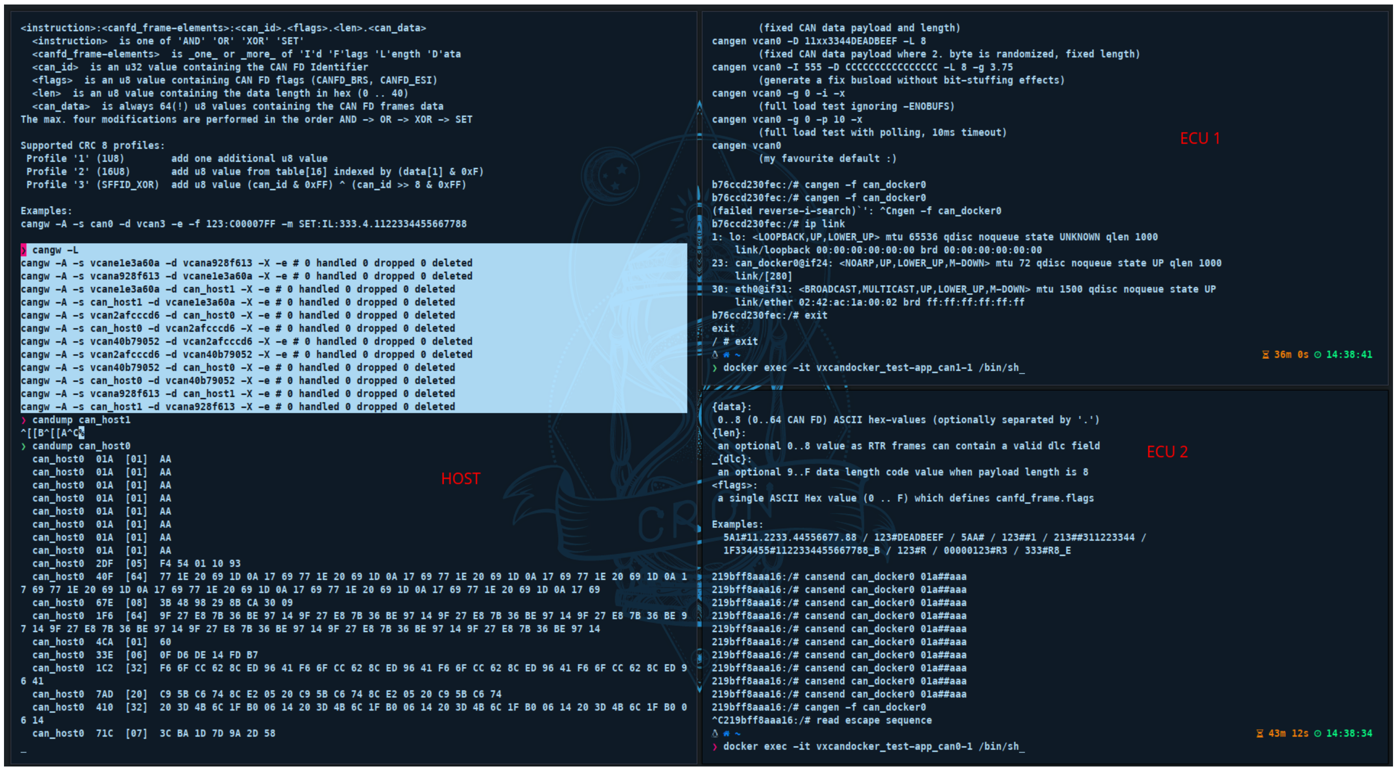
ali bez dodatnog gw ecu containera

- stvara N(N-1) veza izmedju containera i hosta povezanih na istu sabirnicu koristeci cangw pravila:

```
         ❯ cangw -L
cangw -A -s vcane1e3a60a -d vcana928f613 -X -e # 0 handled 0 dropped 0 deleted
cangw -A -s vcana928f613 -d vcane1e3a60a -X -e # 0 handled 0 dropped 0 deleted
cangw -A -s vcane1e3a60a -d can_host1 -X -e # 0 handled 0 dropped 0 deleted
cangw -A -s can_host1 -d vcane1e3a60a -X -e # 0 handled 0 dropped 0 deleted
cangw -A -s vcan2afcccd6 -d can_host0 -X -e # 0 handled 0 dropped 0 deleted
cangw -A -s can_host0 -d vcan2afcccd6 -X -e # 0 handled 0 dropped 0 deleted
cangw -A -s vcan40b79052 -d vcan2afcccd6 -X -e # 0 handled 0 dropped 0 deleted
cangw -A -s vcan2afcccd6 -d vcan40b79052 -X -e # 0 handled 0 dropped 0 deleted
cangw -A -s vcan40b79052 -d can_host0 -X -e # 0 handled 0 dropped 0 deleted
cangw -A -s can_host0 -d vcan40b79052 -X -e # 0 handled 0 dropped 0 deleted
cangw -A -s vcana928f613 -d can_host1 -X -e # 0 handled 0 dropped 0 deleted
cangw -A -s can_host1 -d vcana928f613 -X -e # 0 handled 0 dropped 0 deleted
```

- sva pravila su na hostu i konfigurirana iz hosta, u containerima su samo vxcan sucelja
    - problemi
        - izolacija sucelja za ctf:
            - sva sucelja su vidljiva na hostu
        - nema komunikacije izmedju sabirnica kao sto je to u automobilima
        - u specifičnim uvjetima poruke se duplaju

testiranje:



```
<instruction>:<canfd_frame-elements>:<can_id>.<flags>.<len>.<can_data>
  <instruction>  is one of 'AND' 'OR' 'XOR' 'SET'
  <canfd_frame-elements>  is _one_ or _more_ of 'I'd 'F'lags 'L'ength 'D'ata
  <can_id>  is an u32 value containing the CAN FD Identifier
  <flags>  is an u8 value containing CAN FD flags (CANFD_BRS, CANFD_ESI)
  <len>  is an u8 value containing the data length in hex (0 .. 40)
  <can_data>  is always 64(!) u8 values containing the CAN FD frames data
The max. four modifications are performed in the order AND -> OR -> XOR -> SET

Supported CRC 8 profiles:
 Profile '1' (1U8)        add one additional u8 value
 Profile '2' (16U8)       add u8 value from table[16] indexed by (data[1] & 0xF)
 Profile '3' (SFFID_XOR)  add u8 value (can_id & 0xFF) ^ (can_id >> 8 & 0xFF)

Examples:
cangw -A -s can0 -d vcan3 -e -f 123:C00007FF -m SET:IL:333.4.1122334455667788

❯ cangw -L
cangw -A -s vcane1e3a60a -d vcana928f613 -X -e # 0 handled 0 dropped 0 deleted
cangw -A -s vcana928f613 -d vcane1e3a60a -X -e # 0 handled 0 dropped 0 deleted
cangw -A -s vcane1e3a60a -d can_host1 -X -e # 0 handled 0 dropped 0 deleted
cangw -A -s can_host1 -d vcane1e3a60a -X -e # 0 handled 0 dropped 0 deleted
cangw -A -s vcan2afcccd6 -d can_host0 -X -e # 0 handled 0 dropped 0 deleted
cangw -A -s can_host0 -d vcan2afcccd6 -X -e # 0 handled 0 dropped 0 deleted
cangw -A -s vcan40b79052 -d vcan2afcccd6 -X -e # 0 handled 0 dropped 0 deleted
cangw -A -s vcan2afcccd6 -d vcan40b79052 -X -e # 0 handled 0 dropped 0 deleted
cangw -A -s vcan40b79052 -d can_host0 -X -e # 0 handled 0 dropped 0 deleted
cangw -A -s can_host0 -d vcan40b79052 -X -e # 0 handled 0 dropped 0 deleted
cangw -A -s vcana928f613 -d can_host1 -X -e # 0 handled 0 dropped 0 deleted
cangw -A -s can_host1 -d vcana928f613 -X -e # 0 handled 0 dropped 0 deleted
❯ candump can_host1
^[[B^[[A^C%
❯ candump can_host0
  can_host0  01A  [01]  AA
  can_host0  01A  [01]  AA
  can_host0  01A  [01]  AA
  can_host0  01A  [01]  AA
  can_host0  01A  [01]  AA
  can_host0  01A  [01]  AA
  can_host0  01A  [01]  AA
  can_host0  01A  [01]  AA
  can_host0  2DF  [05]  F4 54 01 10 93
  can_host0  40F  [64]  77 1E 20 69 1D 0A 17 69 77 1E 20 69 1D 0A 17 69 77 1E 20 69 1D 0A 17 69 77 1E 20 69 1D 0A 1
7 69 77 1E 20 69 1D 0A 17 69 77 1E 20 69 1D 0A 17 69 77 1E 20 69 1D 0A 17 69 77 1E 20 69 1D 0A 17 69
  can_host0  67E  [08]  3B 48 98 29 8B CA 30 09
  can_host0  1F6  [64]  9F 27 E8 7B 36 BE 97 14 9F 27 E8 7B 36 BE 97 14 9F 27 E8 7B 36 BE 97 14 9F 27 E8 7B 36 BE 9
7 14 9F 27 E8 7B 36 BE 97 14 9F 27 E8 7B 36 BE 97 14 9F 27 E8 7B 36 BE 97 14 9F 27 E8 7B 36 BE 97 14
  can_host0  4CA  [01]  60
  can_host0  33E  [06]  0F D6 DE 14 FD B7
  can_host0  1C2  [32]  F6 6F CC 62 8C ED 96 41 F6 6F CC 62 8C ED 96 41 F6 6F CC 62 8C ED 96 41 F6 6F CC 62 8C ED 9
6 41
  can_host0  7AD  [20]  C9 5B C6 74 8C E2 05 20 C9 5B C6 74 8C E2 05 20 C9 5B C6 74
  can_host0  410  [32]  20 3D 4B 6C 1F B0 06 14 20 3D 4B 6C 1F B0 06 14 20 3D 4B 6C 1F B0 06 14 20 3D 4B 6C 1F B0 0
6 14
  can_host0  71C  [07]  3C BA 1D 7D 9A 2D 58
_
```

```
        (fixed CAN data payload and length)
cangen vcan0 -D 11xx3344DEADBEEF -L 8
        (fixed CAN data payload where 2. byte is randomized, fixed length)
cangen vcan0 -I 555 -D CCCCCCCCCCCCCCCC -L 8 -g 3.75
        (generate a fix busload without bit-stuffing effects)
cangen vcan0 -g 0 -i -x
        (full load test ignoring -ENOBUFS)
cangen vcan0 -g 0 -p 10 -x
        (full load test with polling, 10ms timeout)
cangen vcan0
        (my favourite default :)

b76ccd230fec:/# cangen -f can_docker0
b76ccd230fec:/# cangen -f can_docker0
(failed reverse-i-search)`': ^Cngen -f can_docker0
b76ccd230fec:/# ip link
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
23: can_docker0@if24: <NOARP,UP,LOWER_UP,M-DOWN> mtu 72 qdisc noqueue state UP qlen 1000
    link/[280]
30: eth0@if31: <BROADCAST,MULTICAST,UP,LOWER_UP,M-DOWN> mtu 1500 qdisc noqueue state UP
    link/ether 02:42:ac:1a:00:02 brd ff:ff:ff:ff:ff:ff
b76ccd230fec:/# exit
exit
/ # exit
                                                                              36m 0s  14:38:41
❯ docker exec -it vxcandocker_test-app_can1-1 /bin/sh_
```

```
{data}:
 0..8 (0..64 CAN FD) ASCII hex-values (optionally separated by '.')
{len}:
 an optional 0..8 value as RTR frames can contain a valid dlc field
_{dlc}:
 an optional 9..F data length code value when payload length is 8
<flags>:
 a single ASCII Hex value (0 .. F) which defines canfd_frame.flags

Examples:
  5A1#11.2233.44556677.88 / 123#DEADBEEF / 5AA# / 123##1 / 213##311223344 /
  1F334455#1122334455667788_B / 123#R / 00000123#R3 / 333#R8_E

219bff8aaa16:/# cansend can_docker0 01a##aaa
219bff8aaa16:/# cansend can_docker0 01a##aaa
219bff8aaa16:/# cansend can_docker0 01a##aaa
219bff8aaa16:/# cansend can_docker0 01a##aaa
219bff8aaa16:/# cansend can_docker0 01a##aaa
219bff8aaa16:/# cansend can_docker0 01a##aaa
219bff8aaa16:/# cansend can_docker0 01a##aaa
219bff8aaa16:/# cansend can_docker0 01a##aaa
219bff8aaa16:/# cansend can_docker0 01a##aaa
219bff8aaa16:/# cansend can_docker0 01a##aaa
219bff8aaa16:/# cangen -f can_docker0
^C219bff8aaa16:/# read escape sequence
                                                                              43m 12s  14:38:34
❯ docker exec -it vxcandocker_test-app_can0-1 /bin/sh_
```

- tijekom testiranja skuzio sam da cangen alat automatski salje can classic, a ne can fd poruke
- **u svakom slucaju mozda bi bilo bolje moju skriptu pretvoriti u docker network plugin po uzoru na ovaj**