

6.3.2024

Uspjesna konfiguracija za komunikaciju containera preko jedne vcan sabirnice

- sudo modprobe can-gw max_hops=2
GW container cangw pravila:

```
b3044cfe20df:/# cangw -L
cangw -A -s gw_vxcan2 -d vcan1 -X -e # 0 handled 0 dropped 39 deleted
cangw -A -s vcan1 -d gw_vxcan2 -X -e # 51 handled 0 dropped 0 deleted
cangw -A -s vcan1 -d gw_vxcan1 -X -e # 51 handled 0 dropped 0 deleted
cangw -A -s gw_vxcan1 -d vcan1 -X -e # 39 handled 0 dropped 39 deleted
```

ECU containeri imaju samo vxcan sučelje prema gw containeru.

Ali dodavanje ECU-a na drugu vcan sabirnicu i postavljanje ove gw konfiguracije:

```
^Cb3044cfe20df:/# cangw -L
cangw -A -s vcan1 -d vcan2 -X -e # 0 handled 0 dropped 15 deleted
cangw -A -s vcan2 -d vcan1 -X -e # 78 handled 0 dropped 0 deleted
cangw -A -s vcan2 -d gw_vxcan4 -X -e # 78 handled 0 dropped 0 deleted
cangw -A -s gw_vxcan4 -d vcan2 -X -e # 15 handled 0 dropped 15 deleted
cangw -A -s gw_vxcan3 -d vcan1 -X -e # 0 handled 0 dropped 63 deleted
cangw -A -s vcan1 -d gw_vxcan3 -X -e # 75 handled 0 dropped 15 deleted
cangw -A -s gw_vxcan2 -d vcan1 -X -e # 0 handled 0 dropped 102 deleted
cangw -A -s vcan1 -d gw_vxcan2 -X -e # 126 handled 0 dropped 15 deleted
cangw -A -s vcan1 -d gw_vxcan1 -X -e # 126 handled 0 dropped 15 deleted
cangw -A -s gw_vxcan1 -d vcan1 -X -e # 39 handled 0 dropped 102 deleted
```

Ne radi. ECU na vcan1 ne može komunicirati s vcan2. (odnosno gw_vxcan4 povezan na vcan2 ne dolazi do gw_vxcan1, gw_vxcan2, gw_vxcan3 povezanih na vcan1)

S **max_hops=3** komunikacija opet radi, ali dolazi do dupliciranja poruka na primarnoj sabirnici ECU s kojeg se šalje (npr. ecu4 šalje na vcan2, ecu-i na vcan1 i vcan1 dobivaju normalnu količinu prometa, ali se na vcan2 promet dupla).

- još je gora situacija ako na sabirnici na koju se šalje ima više ECU-a, primjerice ecu1,ecu2,ecu3 na vcan1, ecu4 na vcan2, ecu1 šalje na vcan1

- ecu2-4 normalno dobivaju poruke, vcan2 normalno, vcan1 cetverostruko:

- prvi hop - ecu1 -> vcan1
- drugi hop - vcan1 -> (ecu2-3 (ecu1 ne dobiva), vcan2)
- treci hop - ecu2-3 -> vcan1 (DUPLICIRANJE!)

Alternative?

- ostavimo max_hops=3 i koristimo can_gw filtriranje,
 - primjerice, samo ecu1 moze slati poruke s 500h arbitration ID-em
 - dodamo pravila koja filtriraju slanje poruka s tim ID-em sa sucelja svih ECU-a osim ecu1
 - nije dobro rjesenje, sto ako svi cvorovi trebaju slati poruku s nekim ID-em
- stavimo max_hops=2 da bi onemogućili dupliciranje poruka, napisati program koji ce prosljedjivati poruke izmedju sucelja kako bi dodali jos taj 1 hop
- umjesto vxcan-a i cangw-a, koristimo ugradjeni docker bridge driver i cannelloni na broadcast adresu
- koristimo can4docker plugin, a u gateway container povezemo na sve can4docker mreze te pokrecemo zasebni program koji ce prosljedjivati poruke izmedju tih mreza

8.3.2024.

Cannelloni i docker bridge

- rad za koji je cannelloni razvijen

<https://ieeexplore.ieee.org/document/7185064>

Mapping CAN-to-ethernet communication channels within virtualized embedded environments

1 na 1 komunikacija, TCP tunel

ECU1:

```
cannelloni/cannelloni -I vcan0 -C s -R ecu2 -r 20000 -l 20000
```

ECU2:

```
cannelloni/cannelloni -I vcan0 -C c -R ecu1 -r 20000 -l 20000
```

-C zastvica određuje radi li se o TCP klijentu ili serveru

```
<value> => fixed value (in hexadecimal for -I and -D)
=> nibbles written as 'x' are randomized (only -D)

The gap value (in milliseconds) may have decimal places, e.g. "-g 4.73"
When incrementing the CAN data the data length code minimum is set to 1.
CAN IDs and data content are given and expected in hexadecimal values.

Examples:
cangen vcan0 -g 4 -I 42A -L 1 -D i -v -v
(fixed CAN ID and length, inc. data)
cangen vcan0 -e -L i -v -v -v
(generate EFF frames, incr. length)
cangen vcan0 -D 11223344DEADBEEF -L 8
(fixed CAN data payload and length)
cangen vcan0 -D 11x3344DEADBEEF -L 8
(fixed CAN data payload where 2. byte is randomized, fixed length)
cangen vcan0 -I 555 -D CCCCCCCCCCCCCC -L 8 -g 3.75
(generate a fix busload without bit-stuffing effects)
cangen vcan0 -g 0 -l -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 :)

011fc0c09c54:/# cangen vcan0
^C011fc0c09c54:/# _

> docker exec -it cannelloni_poc-ecu2-1 /bin/bash
fb615427dd05:/# candump vcan0
vcan0 206 [6] FF 88 0C 15 A8 10 A0 22
vcan0 70E [8] 2C AA 72 14 A6 43 C6 7B
vcan0 22E [1] 18
vcan0 246 [8] 65 9D 42 17 8D 75 17 3F
vcan0 606 [8] 16 AE C7 29 89 58 98 66
vcan0 019 [8] 09 D2 41 18 E5 01 1E 71
vcan0 028 [8] 27 3E 7C 3F A6 B8 4A 01
vcan0 1EC [7] 57 78 89 51 C4 2C 86
vcan0 320 [6] 6F 4A 26 3F 2F E2
vcan0 1A2 [8] 05 25 56 69 01 2B DE 5A
vcan0 31D [8] 2D 59 52 37 83 2D D8 4F
vcan0 207 [3] F0 A2 6F
vcan0 54E [6] E6 18 56 72 68 05
vcan0 61F [8]
vcan0 7CA [5] EE 71 FF 23 87
vcan0 4EC [5] 78 C6 71 7C 0D
vcan0 49C [8] 3C 1A CD 4A 3E 0E FB 27
vcan0 586 [1] 0F
vcan0 0FE [8] 07 96 A3 53 05 58 00 03
vcan0 100 [7] E3 18 BE 0E 1E 27 08
vcan0 1FE [8] 86 2C 3E 30 1D F0 D2 30
vcan0 689 [4] 65 A6 87 53
vcan0 149 [3] 08 20 DF
^Cfb615427dd05:/# _

-d [cubt]          enable debug, can be any of these:
  c : enable debugging of can frames
  u : enable debugging of udp/tcp frames
  b : enable debugging of internal buffer structures
  t : enable debugging of internal timers
  use IPv4 (default)
  use IPv6
  set MTU, default: 1500 bytes
  fork into background / daemon mode
  pid file path (only in daemon mode), default: /var/run/cannelloni.pid
  display this help text

011fc0c09c54:/# cannelloni/cannelloni -I vcan0 -C s -R ecu2 -r 20000 -l 20000
INFO:tcp_server_thread.cpp[73]:attempt connect:Waiting for a client to connect.
INFO:canthread.cpp[188]:run:CANThread up and running
INFO:tcp_server_thread.cpp[73]:attempt connect:Waiting for a client to connect.
INFO:tcp_server_thread.cpp[73]:attempt connect:Waiting for a client to connect.
INFO:tcp_server_thread.cpp[73]:attempt connect:Waiting for a client to connect.
INFO:tcp_server_thread.cpp[73]:attempt connect:Waiting for a client to connect.
INFO:tcp_server_thread.cpp[73]:attempt connect:Waiting for a client to connect.
INFO:tcp_server_thread.cpp[73]:attempt connect:Waiting for a client to connect.
INFO:tcp_server_thread.cpp[73]:attempt connect:Waiting for a client to connect.
INFO:tcp_server_thread.cpp[73]:attempt connect:Waiting for a client to connect.
INFO:tcp_server_thread.cpp[73]:attempt connect:Waiting for a client to connect.
INFO:tcp_server_thread.cpp[73]:attempt connect:Waiting for a client to connect.
INFO:tcp_server_thread.cpp[73]:attempt connect:Waiting for a client to connect.
INFO:tcp_server_thread.cpp[106]:attempt connect:Got a connection from 172.20.0.6:45120
--

> docker attach cannelloni_poc-ecu2-1
fb615427dd05:/# ip link add dev vcan0 type vcan
fb615427dd05:/# ip link set dev vcan0 up
fb615427dd05:/# cannelloni/cannelloni
.dockerenv dev/ lib/ opt/ run/ srv/ usr/
bin/ etc/ media/ proc/ sbin/ sys/ var/
cannelloni/ home/ mnt/ root/ setup_ecu.sh tmp/
fb615427dd05:/# cannelloni/cannelloni -I vcan0 -C c -R ecu1 -r 20000 -l 20000
INFO:tcp_client_thread.cpp[51]:attempt connect:Connecting to 172.20.0.5:20000...
INFO:canthread.cpp[188]:run:CANThread up and running
INFO:tcp_client_thread.cpp[57]:attempt connect:Connected!
--
```

ECU1 cangen

ECU1 cannelloni

ECU2 candump

ECU2 cannelloni

Spajanje vise TCP klijenata nije moguće.

UDP broadcast

Na svakom ECU-u pokrenuti:

```
cannelloni -R 172.23.255.255 -p
```

- radi s manjom količinom prometa
- pokretanje cangen-a s maksimalnom ucestaloscju slanja rezultira u gubitku ili promjeni poretka paketa
 - provereno pokretanjem candumpa na vise containera i cangena na jednom s maksimalnom ucestaloscju slanja
 - potvrdjeno diffom na candump logove