acsd applied cyber security darmstadt

h_da darmstadt university of applied sciences

member of EUT+ EUROPEAN UNIVERSITY OF TECHNOLOGY

Redefining V2G

How to use your vehicle as a game controller

Timm Lauser & Jannis Hamborg

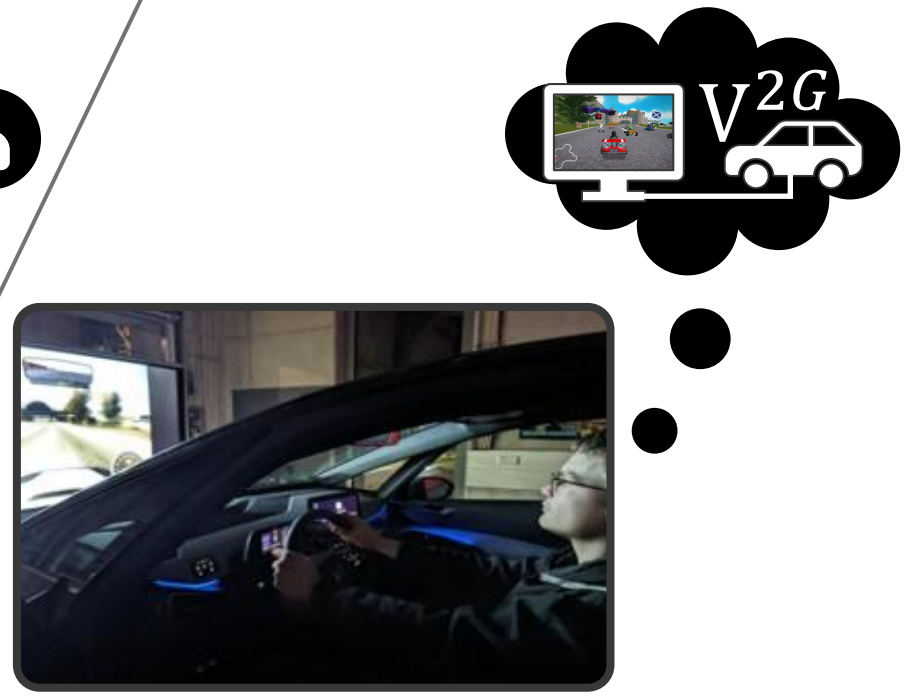
DEF CON 32 - 08/11/2024

Motivation

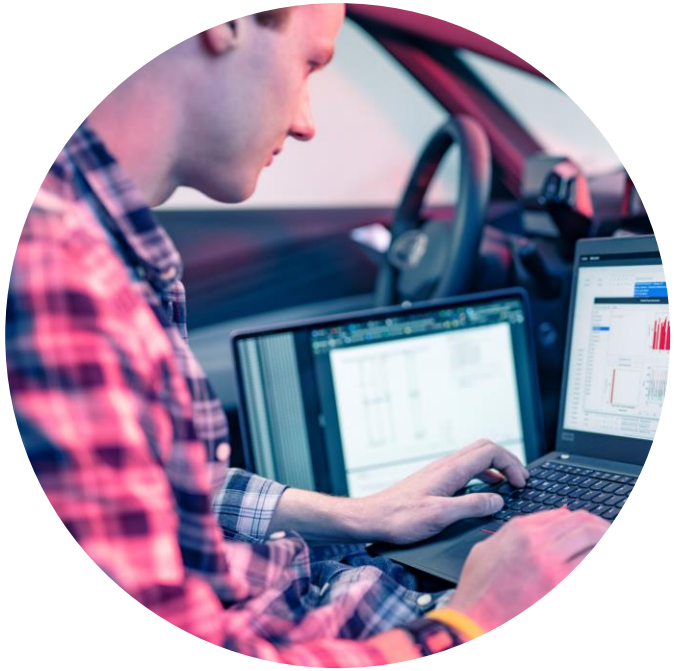
What our professor told us to do

What he thinks we are doing

What we actually do



About Us

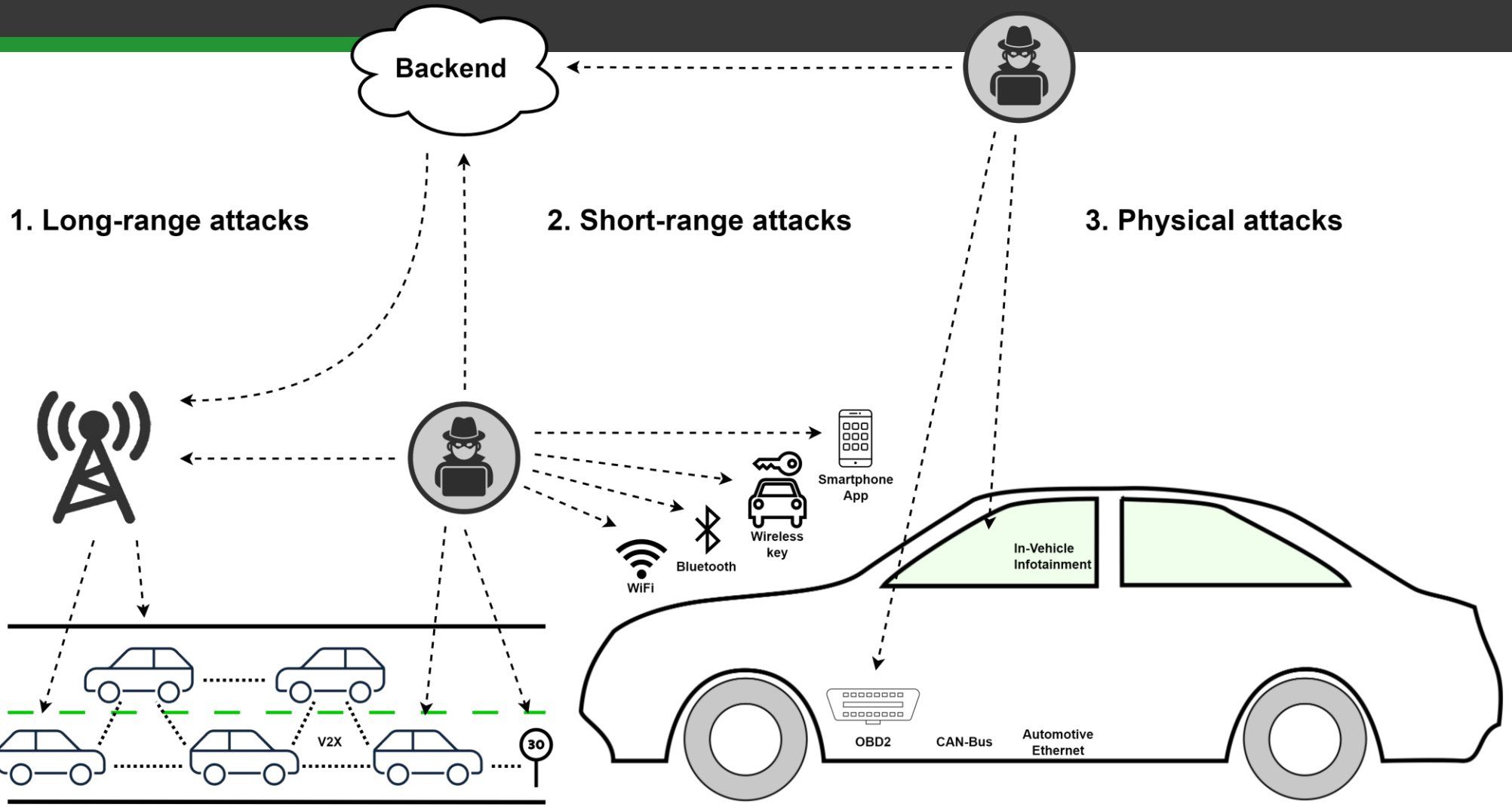


Timm Lauser
Ph.D. Student

Volkswagen ID.3
Research Car

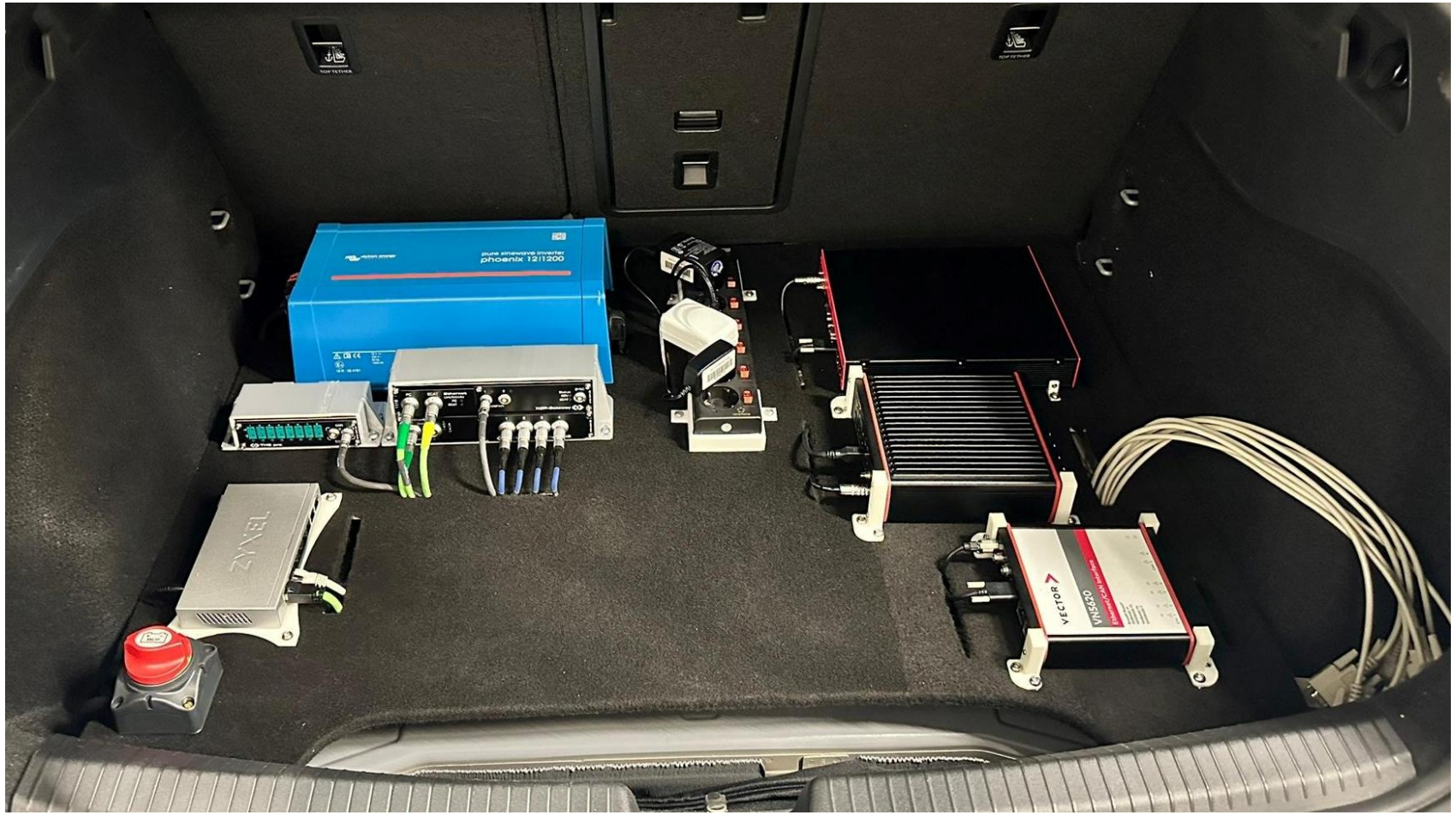
Jannis Hamborg
Ph.D. Student

Theoretical Attack Surface



Our Research Cars

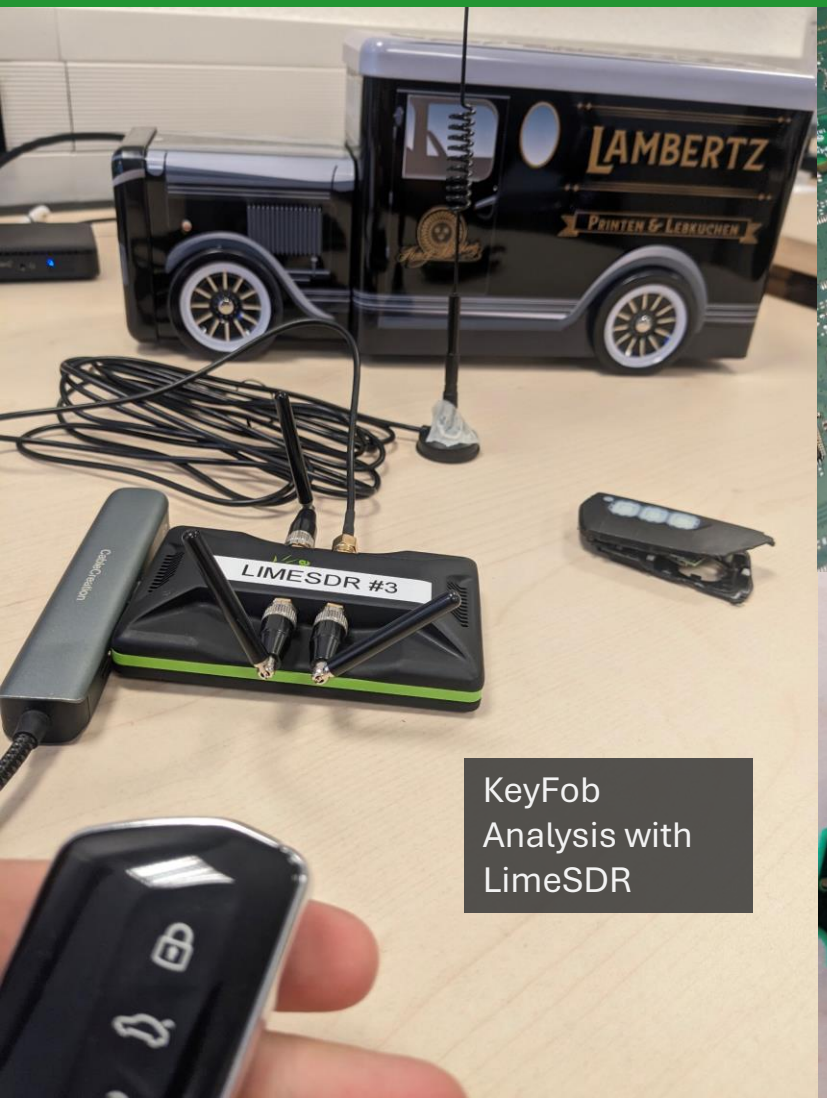
Measurement Technology



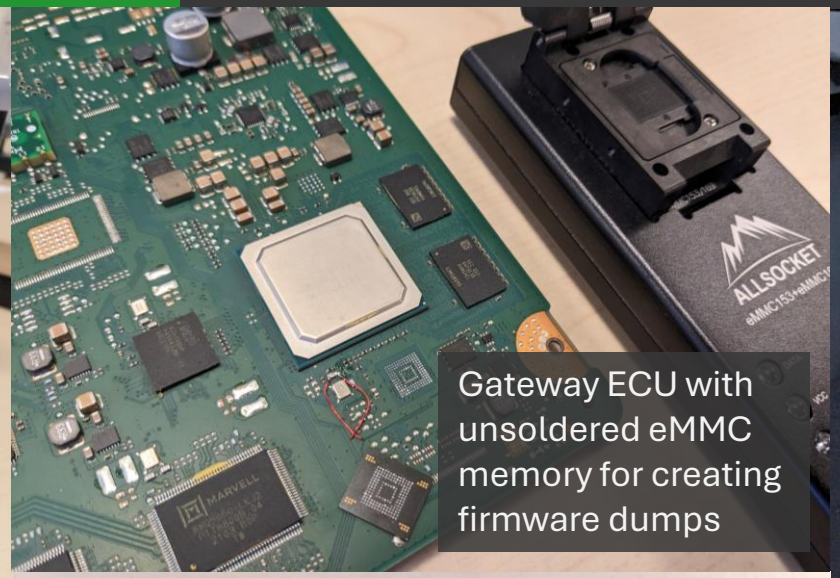
acsd applied cyber security darmstadt

h_da / eur+

Practical Attack Surface

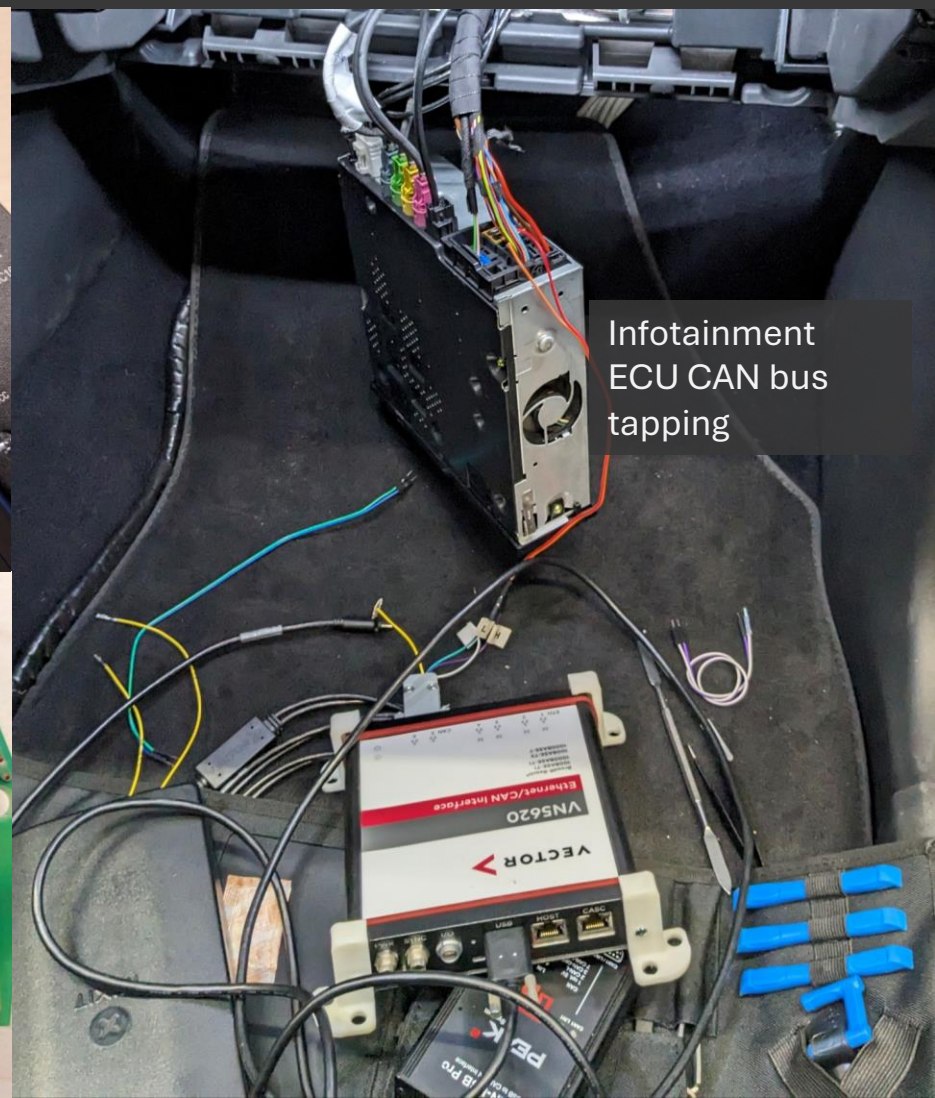


KeyFob Analysis with LimeSDR



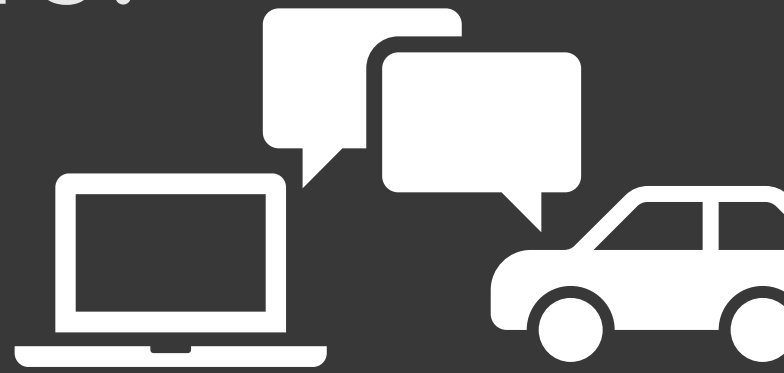
Gateway ECU with unsoldered eMMC memory for creating firmware dumps

KeyFob Circuit Board with unsoldered SoC



Infotainment ECU CAN bus tapping

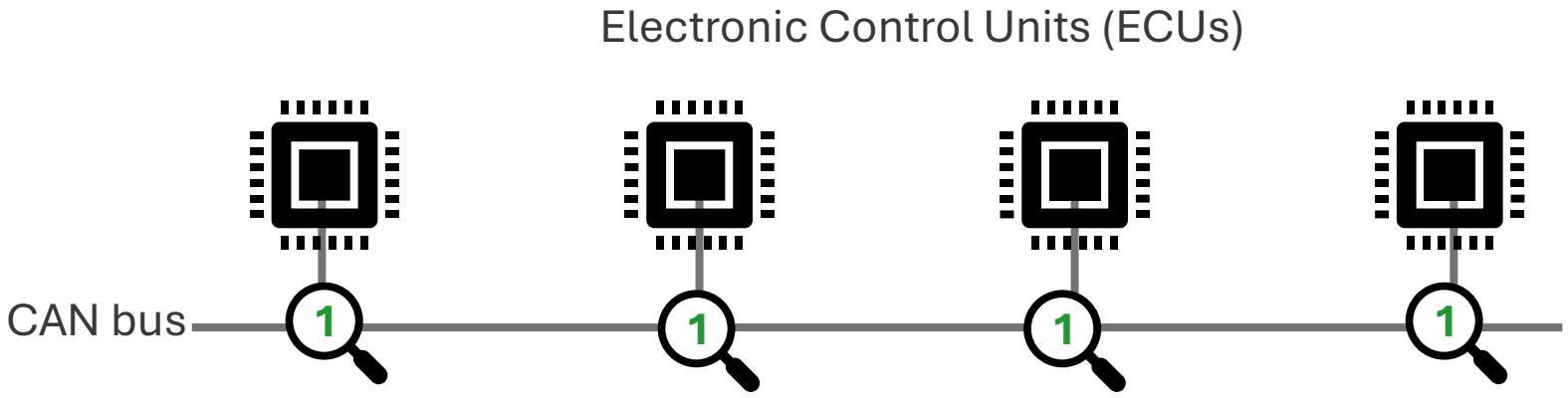
How CAN we talk with cars?



The CAN Bus and Protocol

The CAN Bus and Protocol

- Developed by Bosch in 1986
- Dual wires (twisted pair)
- Bus topology



- Multi-master protocol

Accessing the Internal Communication

CAN adapter installed in the middle console of a Tesla Model 3

- Look at repair manuals/wiring diagrams to find access points
- Look for ready-to-buy adapters
- Otherwise, a contactless CAN reader can be an option

CANCrocodile

Accessing the Internal Communication

PCAN USB Pro

Raspberry Pi Zero W with CAN Hat

Working with CAN Messages

- Commandline Tools / Libraries
  - can-utils
  - python-can
  - cantools
- GUI – Tools
  - SavvyCAN
- Commercial Tools

```
$ candump can0
  can0  13D   [6]  00 00 00 00 FF 00
  can0  419   [1]  00
  can0  54F   [8]  00 00 00 00 00 00 00 00
  can0  439   [8]  02 04 10 00 00 00 00 00
  can0  2E5   [8]  00 00 05 32 04 00 DD 00
  can0  545   [8]  01 00 00 00 00 10 50 AB
  can0  757   [8]  00 40 07 14 C0 17 04 00
  can0  788   [8]  D4 C4 71 83 00 00 40 12
  can0  221   [8]  61 05 55 05 00 00 40 23
  can0  3C0   [8]  02 00 00 00 00 00 00 00
  can0  273   [8]  81 C1 30 00 28 03 30 11
  can0  266   [8]  01 00 03 1E 03 00 49 01
```

Working with CAN Messages

```
$ sudo ip link set can0 up \
        type can bitrate 500000
```

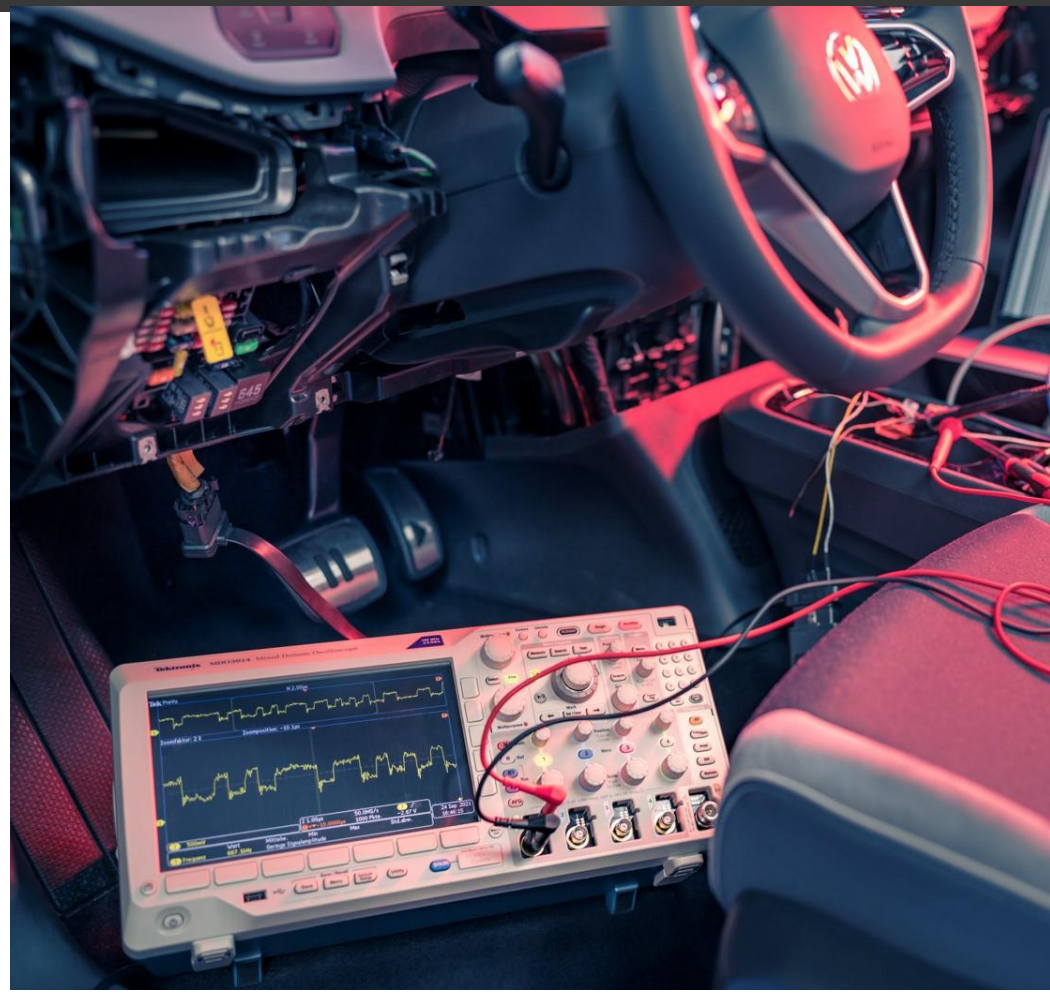
```
$ candump can0
```

...

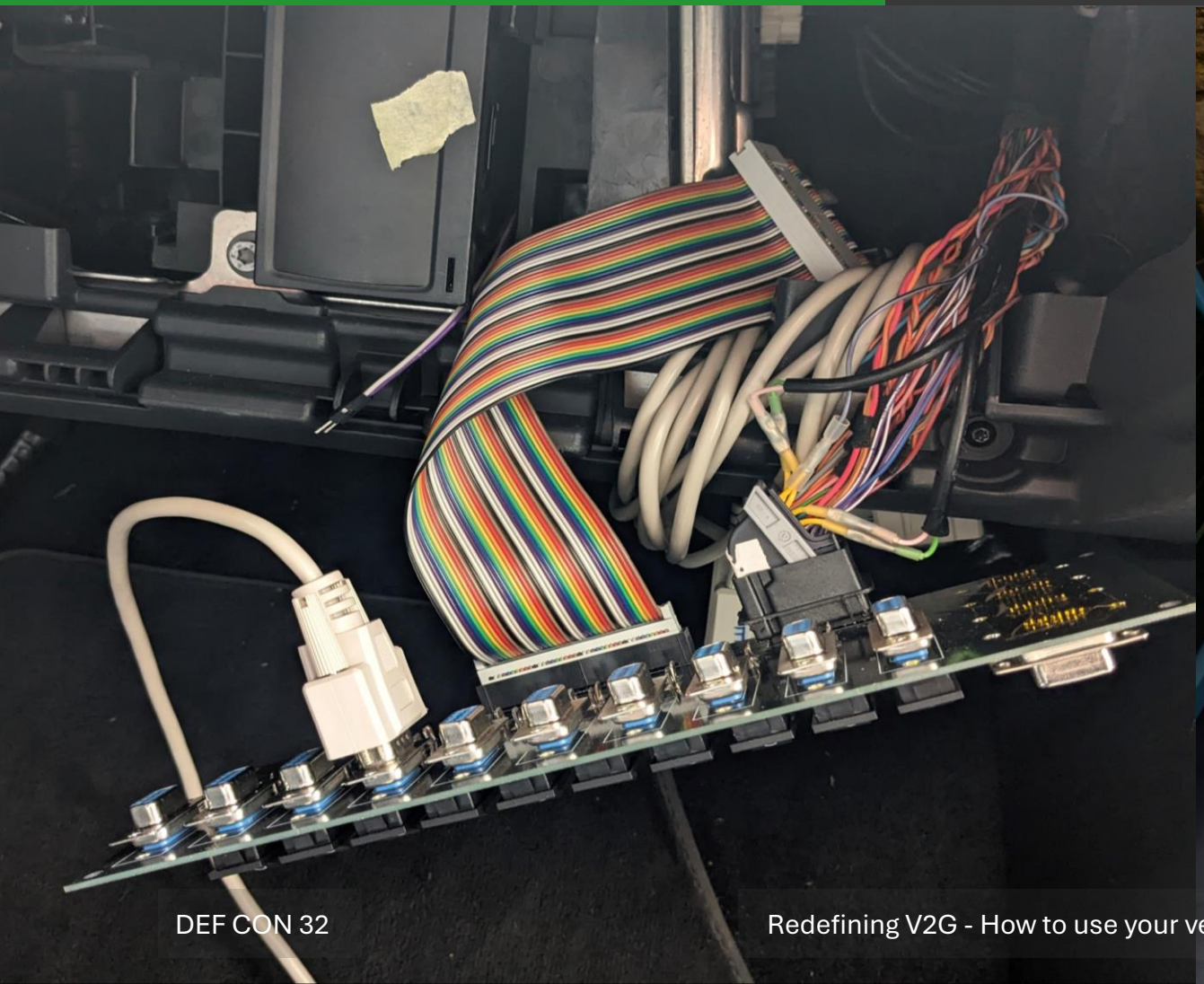
```
can0  757  [8]  00 40 07 14 C0 17 04 00
```

Bus	CAN ID	Data Length	Data
	11 Bit	in Byte	≤ 8 Byte

...



How to Access the CAN Bus?



Vehicle Diagnostics

Vehicle Diagnostics via OBD

OBD-II

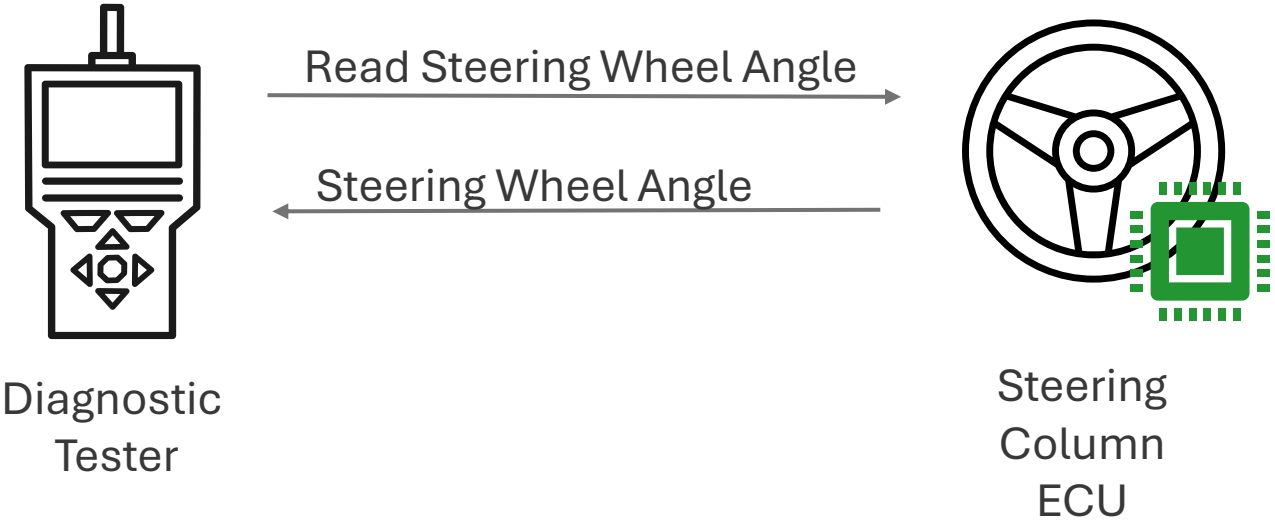
Diagnostic Tester

Hella Gutman Mega Macs VCI

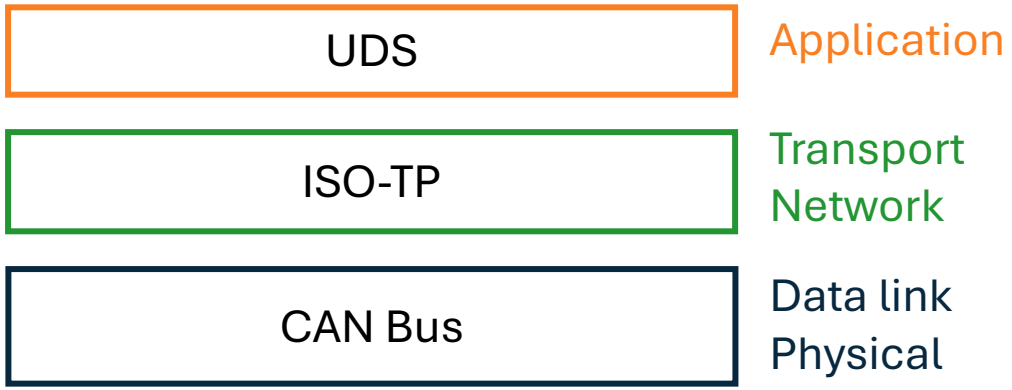
OBDeleven

Unified Diagnostic Services (UDS)

Request/Response Messages



Diagnostic Protocol Stack



! Different data can have the same CAN ID

Data can be split over multiple CAN messages

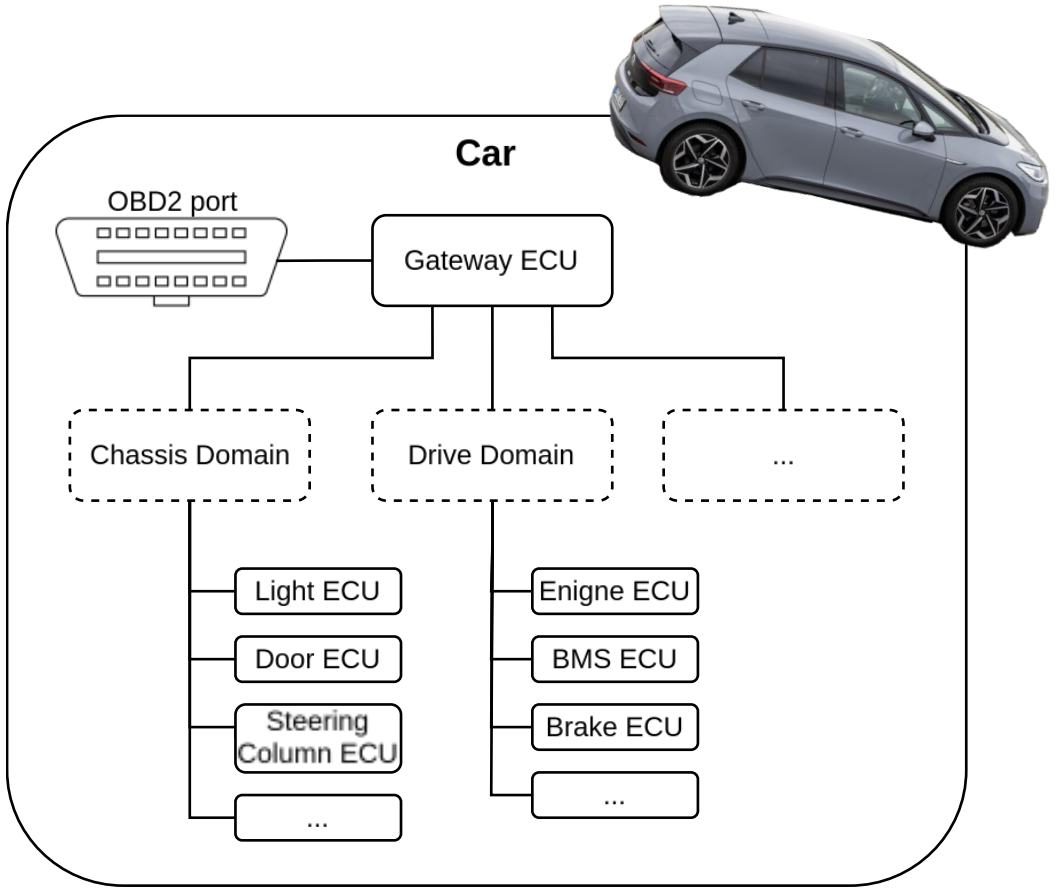
Vehicle-to-Game


AIRBAG
20:55
17,5°C
WH
Willkommen WiMiJa
Nutzerwechsel
Mehr Informationen
OK
Achtung: Deckel vom Handschuhfach fällt heraus

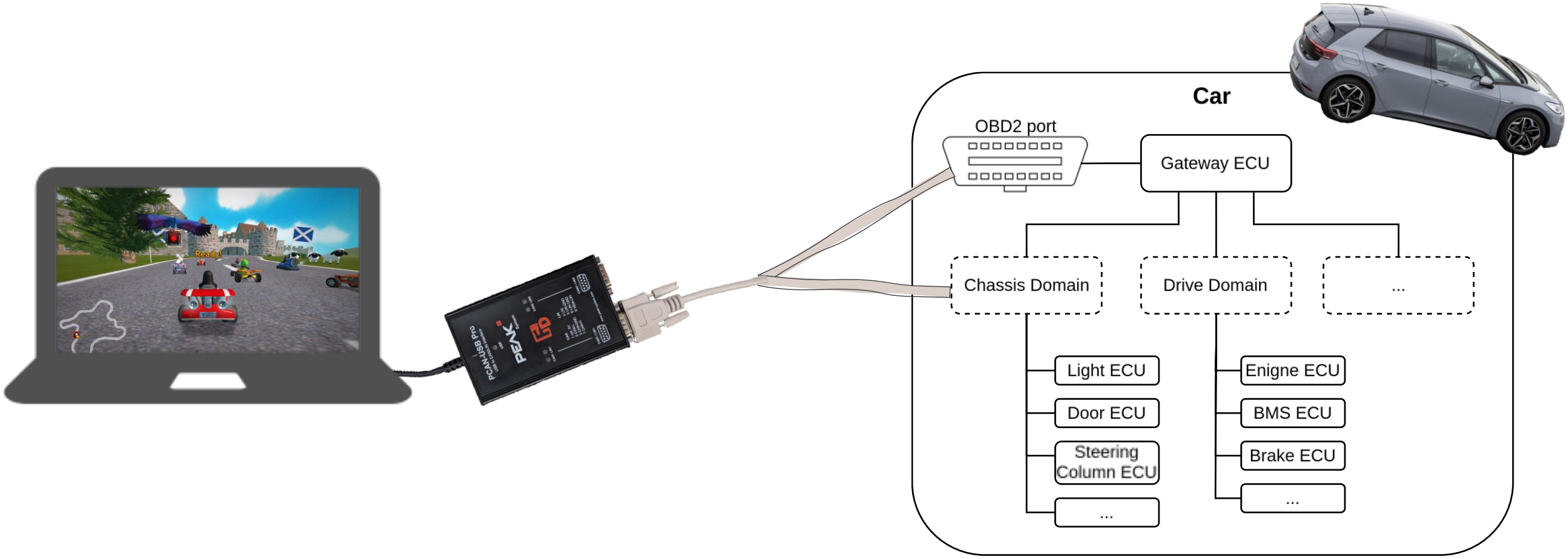
V2G Experience

Modes of Usage

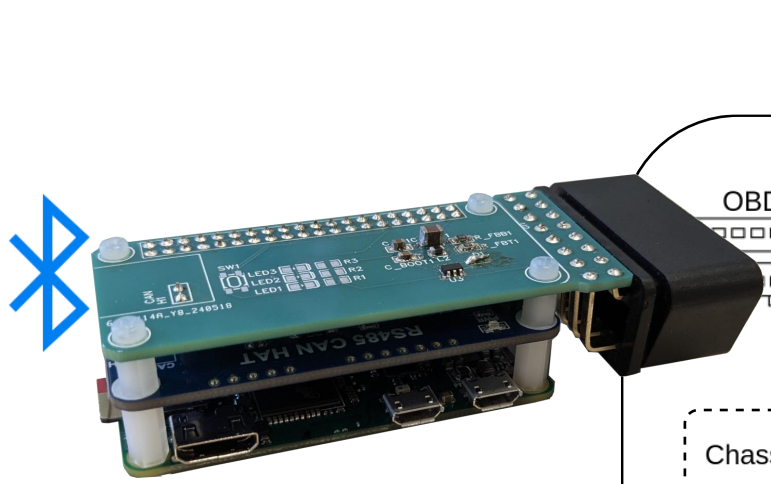
?

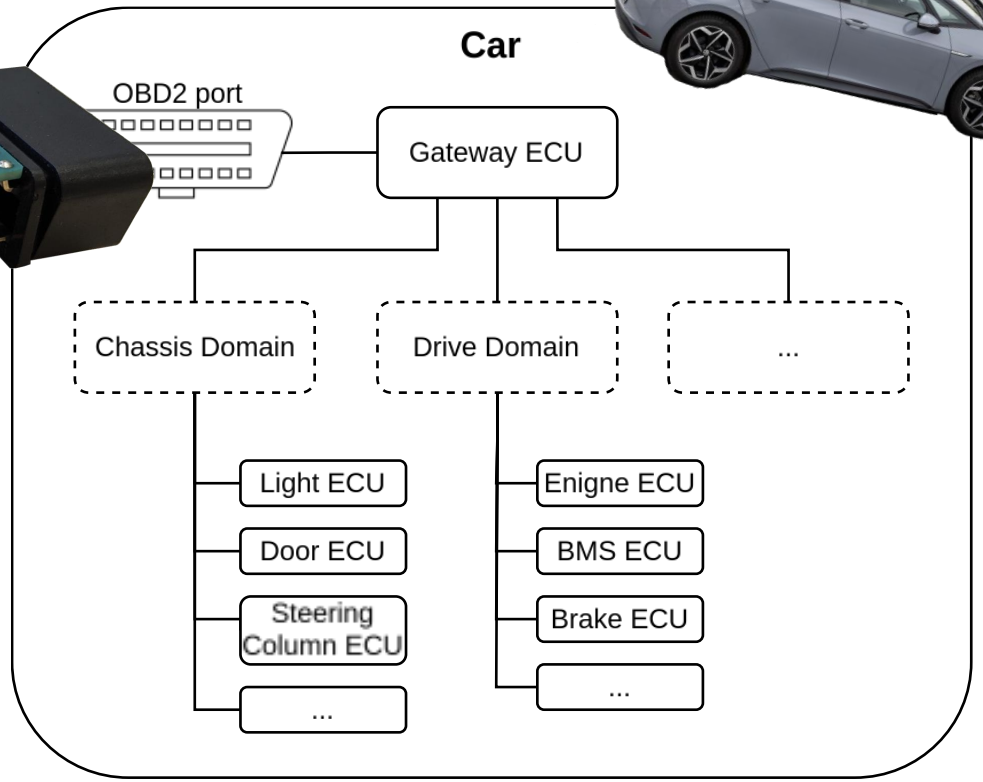


Modes of Usage - USB

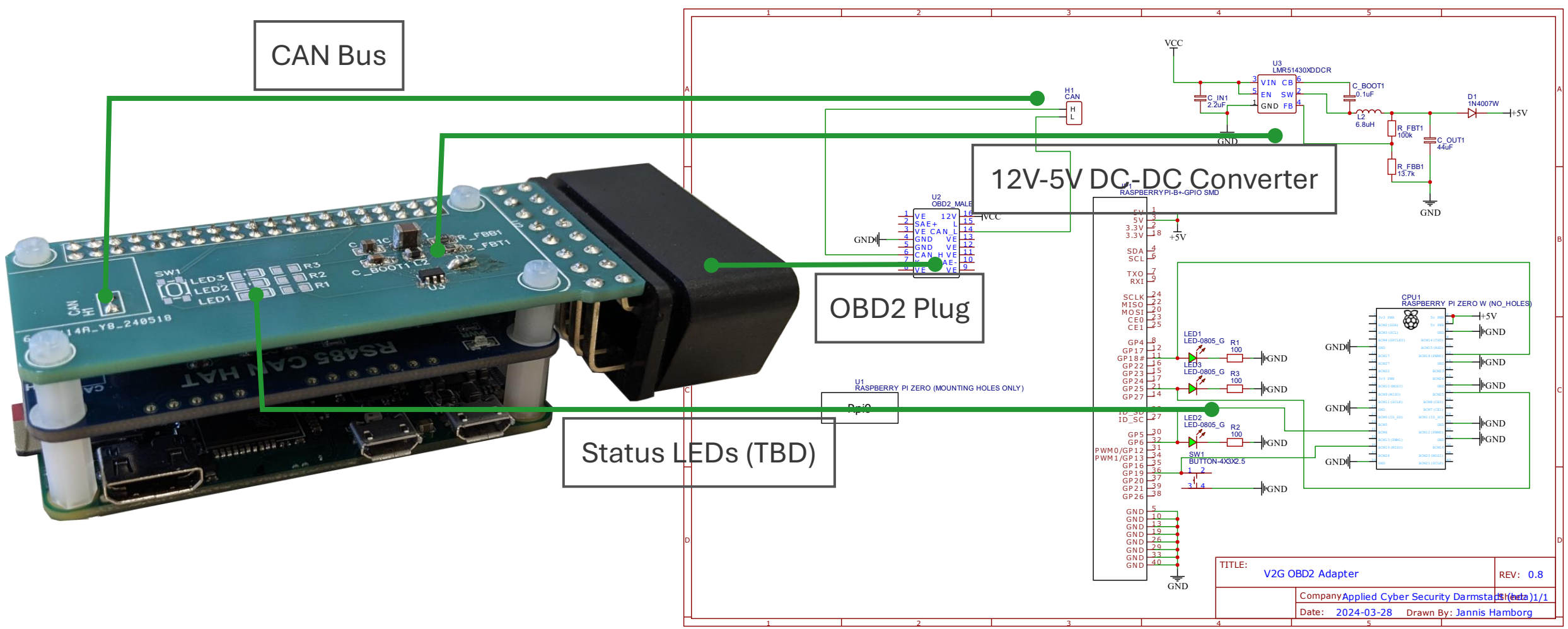


Modes of Usage - Bluetooth

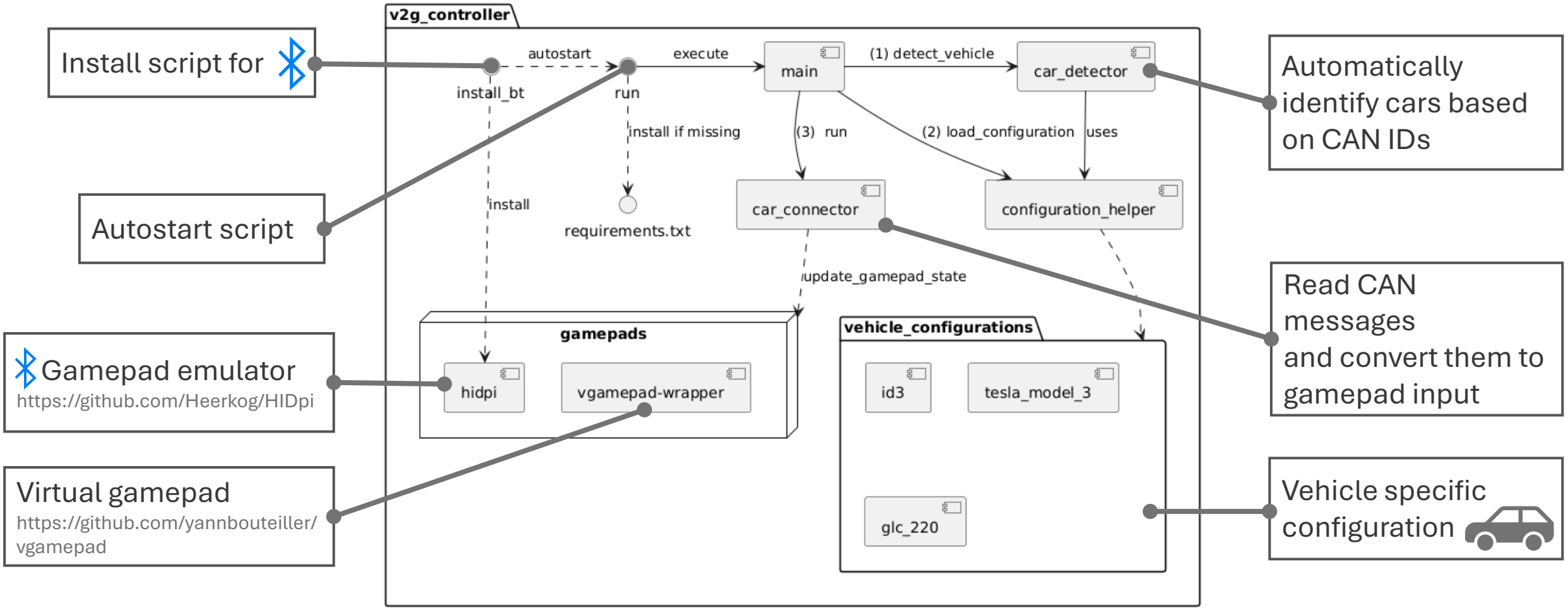




Bluetooth mode – Hardware



Software Architecture



Vehicle Configuration

```
config_uds = VehicleConfiguration(
        vehicle="ID3_UDS",
        operation_mode=OperationMode.UDS,
        can_buses=[CANBus("standard", 500000)],
        auto_detect_ids=[0x77c, 0x77d, 0x7a5,0x776],
        steering_max=0.15,
        steering_deadzone=0.1,
        steering_exponent=1.0,
        configurations=[
            SignalConfiguration(
                id=0,
                name="steering",
                can_signal=CANSignal(
                    id=0x77C,
                    byte=5,
                    length=2,
                    mapping=lambda x: -range_map(
                        ((x[0] << 8) + x[1]), 0x0000, 0x2EDF, -1.0, 1.0
                    ),
                ),
                type=Type.Steering,
            ),
        ...
        ],
```

```
    polling_messages=[
        PollingMessage(
            arbitration_id=0x73B,
            data=[0x03, 0x22, 0x47, 0xD4, 0x55, 0x55, 0x55, 0x55],
            is_extended_id=False,
            polling_interval=POLLING_INTERVAL_FAST,
        ),
        PollingMessage(
            arbitration_id=0x712,
            data=[0x03, 0x22, 0x18, 0x12, 0x55, 0x55, 0x55, 0x55],
            is_extended_id=False,
            polling_interval=POLLING_INTERVAL_FAST,
        ),
        PollingMessage(
            arbitration_id=0x70C,
            data=[0x03, 0x22, 0x1F, 0x00, 0x55, 0x55, 0x55, 0x55],
            is_extended_id=False,
            polling_interval=POLLING_INTERVAL_SLOW,
        ),
    ]
```


03:50.939
3/3
15:26
22.5°C
WH
Willkommen WiMiJa
Nutzerwechsel
Mehr Informationen
OK
km/h
0
PARK
81 %
236 km
OFF
kWh/100km
Ab Start
D / B
N
R
SET
RES
MODE
OK

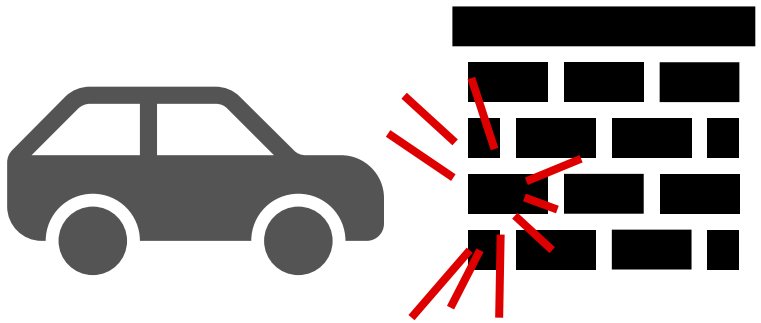

km/h
0
PARK
81 %
236 km
km
kWh/100km
Ab Start
D / B
N
R
OFF
SET
RES
MODE
AIRBAG
OK
VIEW
15:26
Willkommen WHALE
Nutzerwechsel
Mehr Informationen
OK


04:05.433
3/3
km/h
0
km
D / B
N
R
81%
236 km
kWh/100km
Ab Start
OFF
15:26
22.5°C
Willkommen WiMiJa
Nutzerwechsel
Mehr Informationen
OK
SET
RES
MODE
OK
VIEW

How to support your own car

Getting Started with V2G

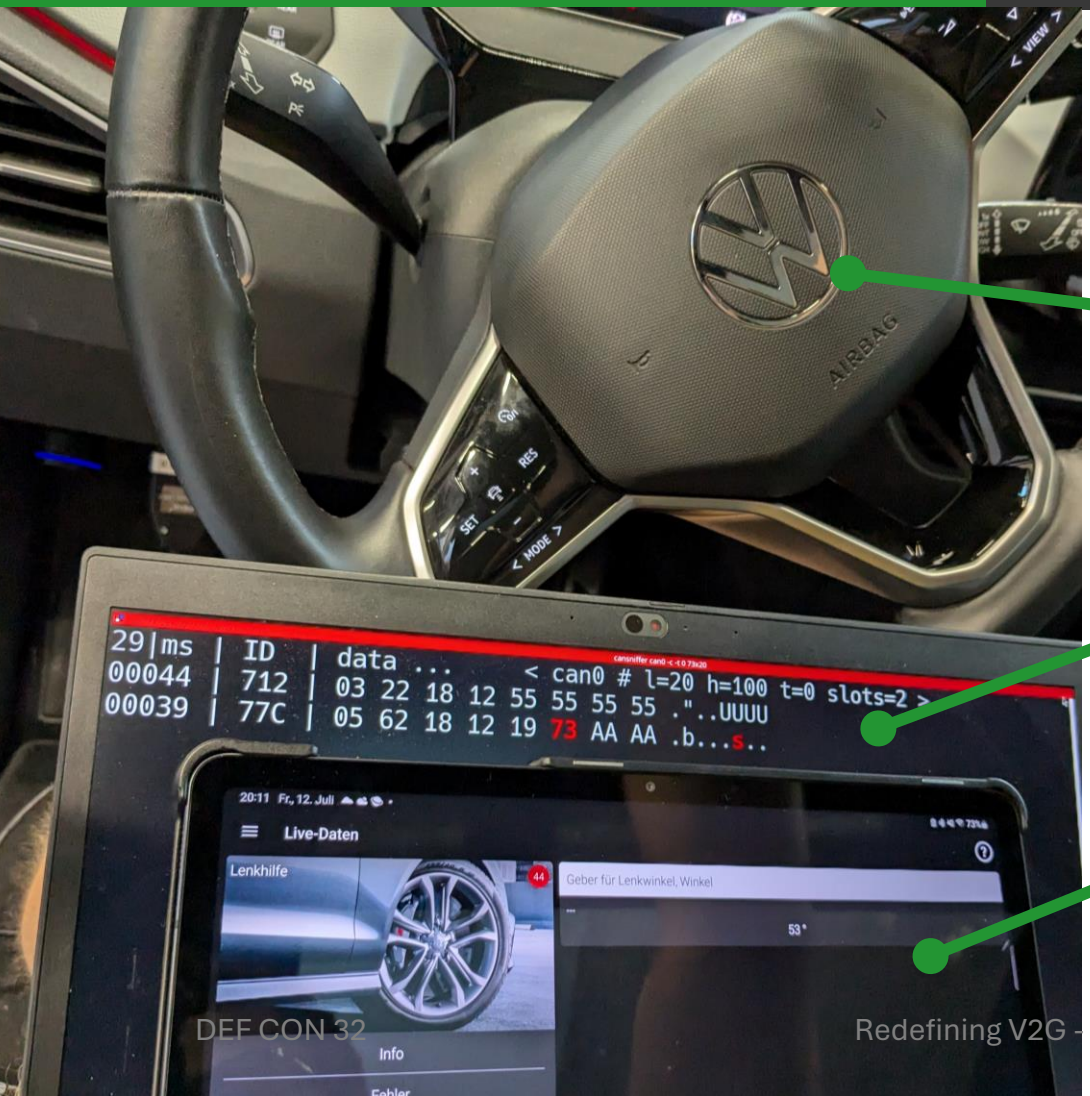
- Safety-first!
  - Make sure you cannot accidentally drive while gaming
  - Make sure you do not drain your 12V battery



- Cost:
  - Raspberry Pi Zero W
  - CAN hat
  - OBD hat or OBD Cable

  < 50 $
- + Diagnostic Tester for UDS communication

Reverse Engineering of CAN Messages



$ cansniffer -c can0

Actuator
Trigger input source to change values

cansniffer
View the raw request and response messages

OBD-II Diagnostic Tester
View the signal of interest

Reverse Engineering of CAN Messages

$ cansniffer -c can0

Actuator
Trigger input source to change values

cansniffer
View the raw request and response messages

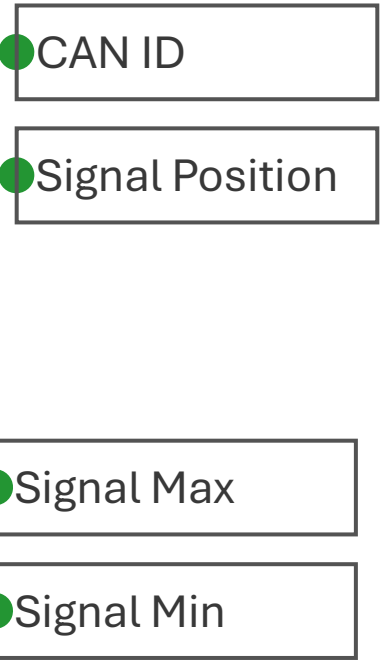
OBD-II Diagnostic Tester
View the signal of interest

Or search for

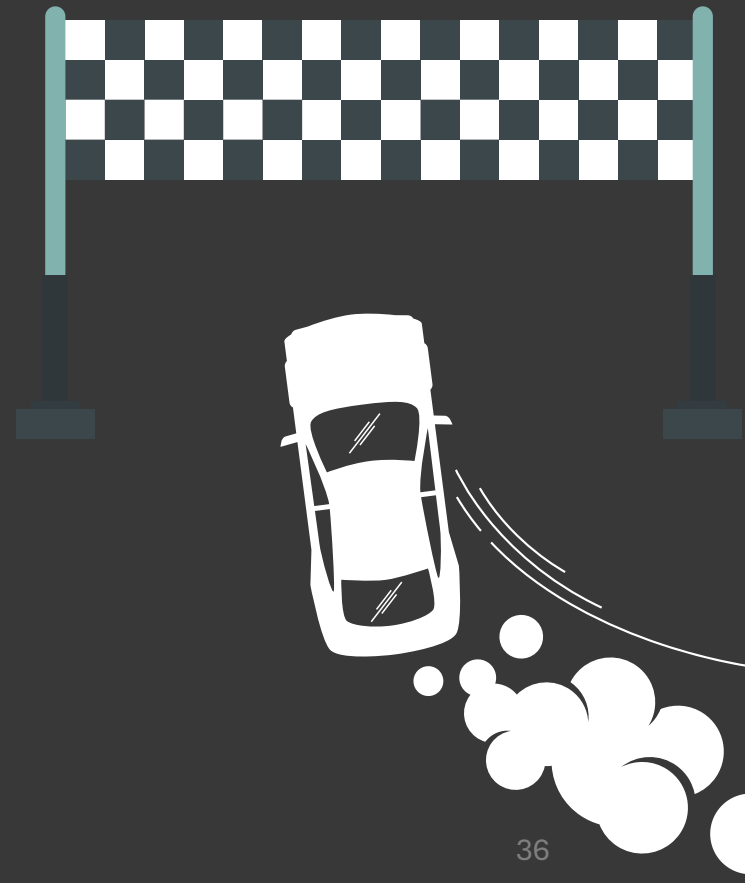
DBC Files

Vehicle Configuration

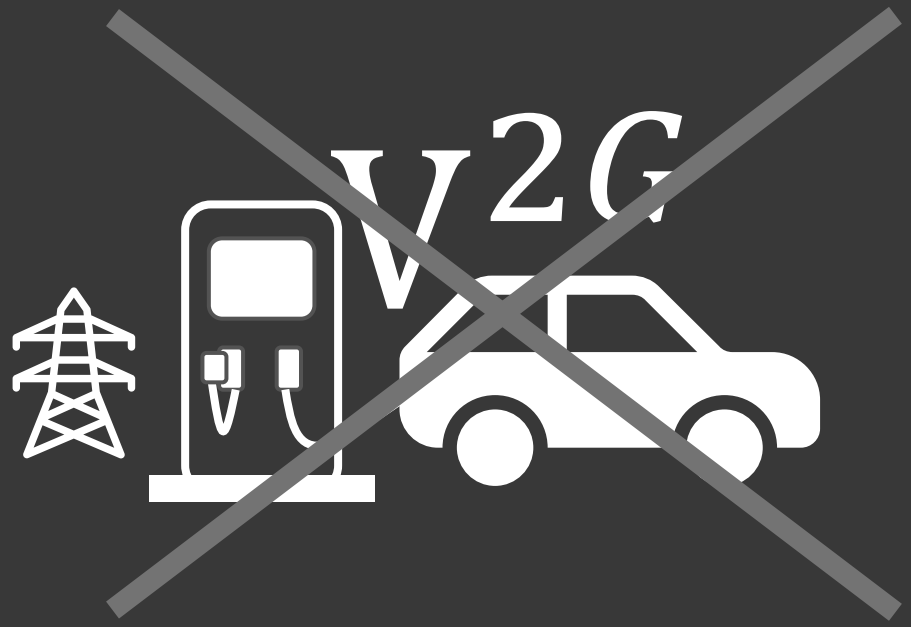
```
SignalConfiguration(
        id=0,
        name="steering",
        can_signal=CANSignal(
                id=0x77C,
                byte=5,
                length=2,
                mapping=lambda x: -range_map(
                        ((x[0] << 8) + x[1]), 0x0000, 0x2EDF, -1.0, 1.0
                ),
        ),
        type=Type.Steering,
),
```



Conclusion



V2G now stands for Vehicle-to-Game



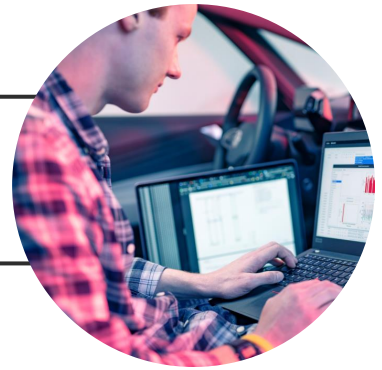
Conclusion

- Car hacking can be fun
- V2G can be a good entry point
- Always be careful!
- We use V2G as a demonstrator
  to get students and visitors enthusiastic about vehicle security
- What will you use it for?

Contact & Links

https://github.com/Vehicle2Game/v2g

Timm Lauser
timm.lauser@h-da.de



Jannis Hamborg
jannis.hamborg@h-da.de

Acknowledgments

We acknowledge the German Research Foundation (DFG) for partially funding our project within the "Resilience in Connected Worlds" (SPP 2378) program - project number 503329135, enabling advancements in our research.

In addition, we would like to thank Heerko Groefsema for the development of HidPi and Yann Bouteiller, JumpyzZ and willRicard for maintaining and contributing to vgamepad.

We also thank Volkswagen AG for their kind permission to publish our reverese engineered configuration for the Volkswagen ID.3.

Finally, we would also like to thank all the members of our research group ACSD, especially Prof. Dr. Christoph Krauß, who supported and motivated us in the realization of this project.

References

- Pictures
  - On slide 4: with kind permission of INCYDE GmbH (Whitepaper, modified)
  - On slide 3, 14, 39: Gregor Schuster for ACSD
  - On slide 5: Thomas Schäfer (previously ACSD)
  - Rest: ACSD
  - The V2G Logo was designed with Canva