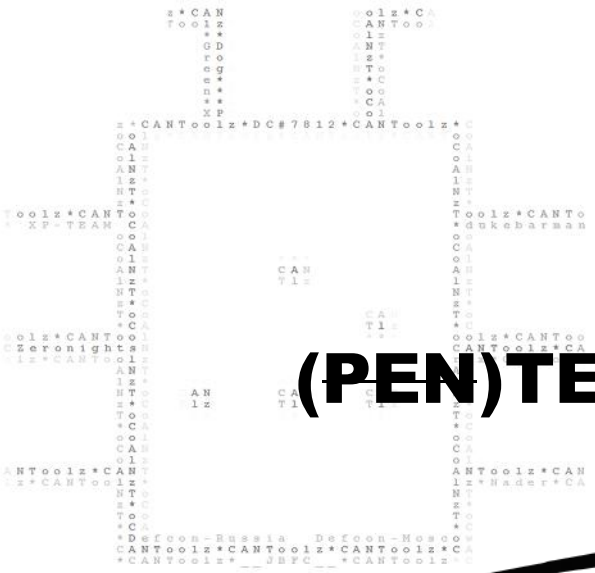


black hat EUROPE 2016

(PEN)TESTING VEHICLES WITH

YACHT - Yet Another Car Hacking Tool

By **ALEXEY SINTSOV** (@asintsov)

#whoami

WORK: Principal Security Engineer at

Community: co-founder of DC group

and

WARNING: I am not a HARDWARE/CAR guy… my past is about JIT-SPRAY, shellcodes, ROP, BoF, UAF and WEB things like SQLi… but now all these things came into automotive world ;)

#CarPWN community from RUSSIA (of course with love)

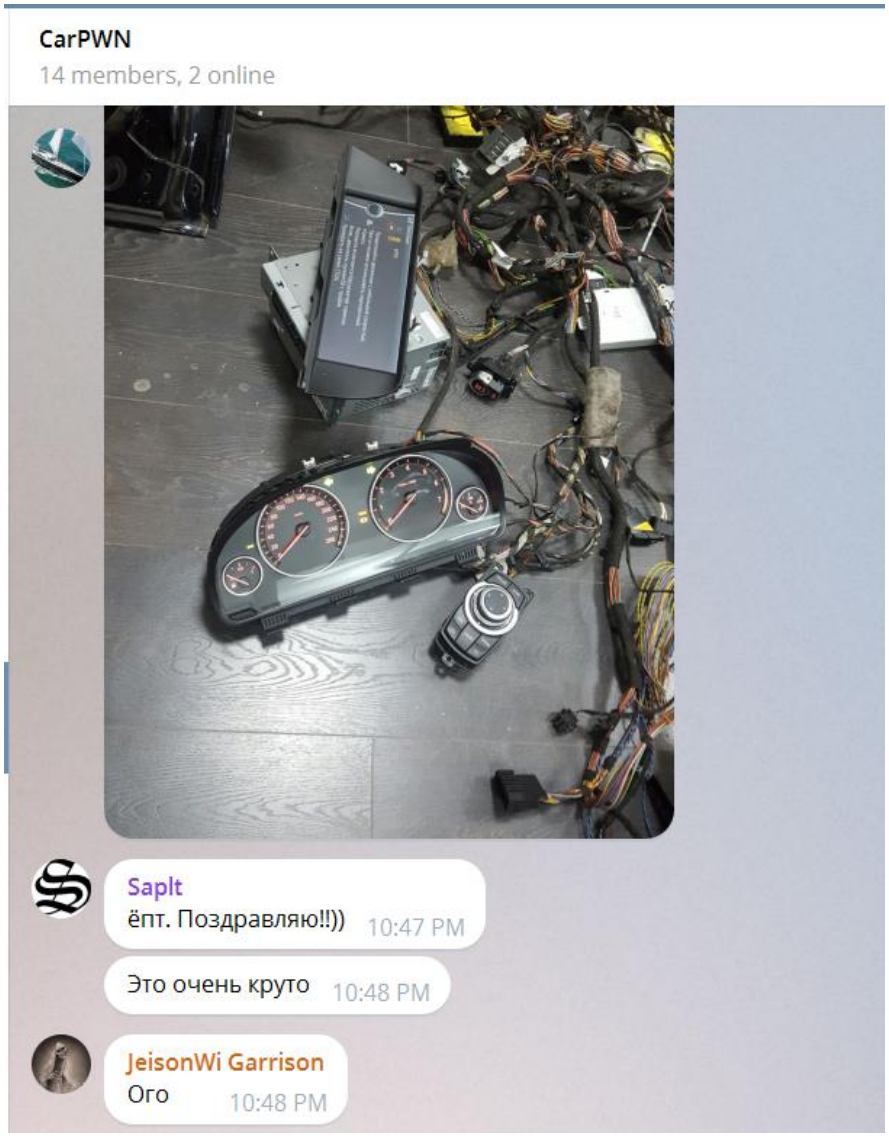
Bunch of CAR hackers/enthusiasts, just a Telegram community who are interested in automotive internals:

- CAN/LIN/Ethernet/Wi-Fi/BT research
- Reverse Engineering of ECU/HU
- Bug hunting
- Creating own tools and hardware modules (DIY)

They are active CANToolz users and my best testers and bug-reporters, thx to them:

- Michael Elizarov
- Sergey Horujenko
- Anyton Sysoev
- Dmitry Nedospasov (@nedos)
- Sergey Kononenko (CANToolz dev)
- Ilya Dinmuhametov

.. and more more more…



#Pentesting?

DISCLAIMER:
This is **not a FUD talk**, **I am not going to 'sell' any devices or services**.
Automotive Security Engineers are doing a good job right now and they are trying to address all issues. So it is not SO bad as you could read in mass-media. There are some challenges and problems – yes. But people are working on making this world a more secure place and **tomorrow is always better than yesterday**.

#Attack surface
Direct attacks

- Local I/O
  - CAN interfaces
  - Ethernet
  - WiFI
  - OBD-II

- Wireless components and ECUs
  - Long Radio:
    - GSM/UMTS
    - Radio/RDS
    - GPS
  - Short Radio:
    - WiFi/Bluetooth
    - TPMS
    - Keyless lock/start
    - Radars/Sensors/Cameras
- HeadUnit
  - Software components
    - WEB Browser
    - MP3/etc
    - RDS
    - Applications
    - Connected Car services

#Attack surface Connected Car

- **CSRF**
- **MITM**
- **Internet Backend services hacking**
- **…**

- Local I/O
  - CAN interfaces
  - Ethernet
  - WiFI
  - OBD-II

- Wireless components and ECUs
  - Long Radio:
    - GSM/UMTS
    - Radio/RDS
    - GPS
  - Short Radio:
    - WiFi/Bluetooth
    - TPMS
    - Keyless lock/start
    - Radars/Sensors/Cameras
- HeadUnit
  - Software components
    - WEB Browser
    - MP3/etc
    - RDS
    - Applications
    - Connected Car services

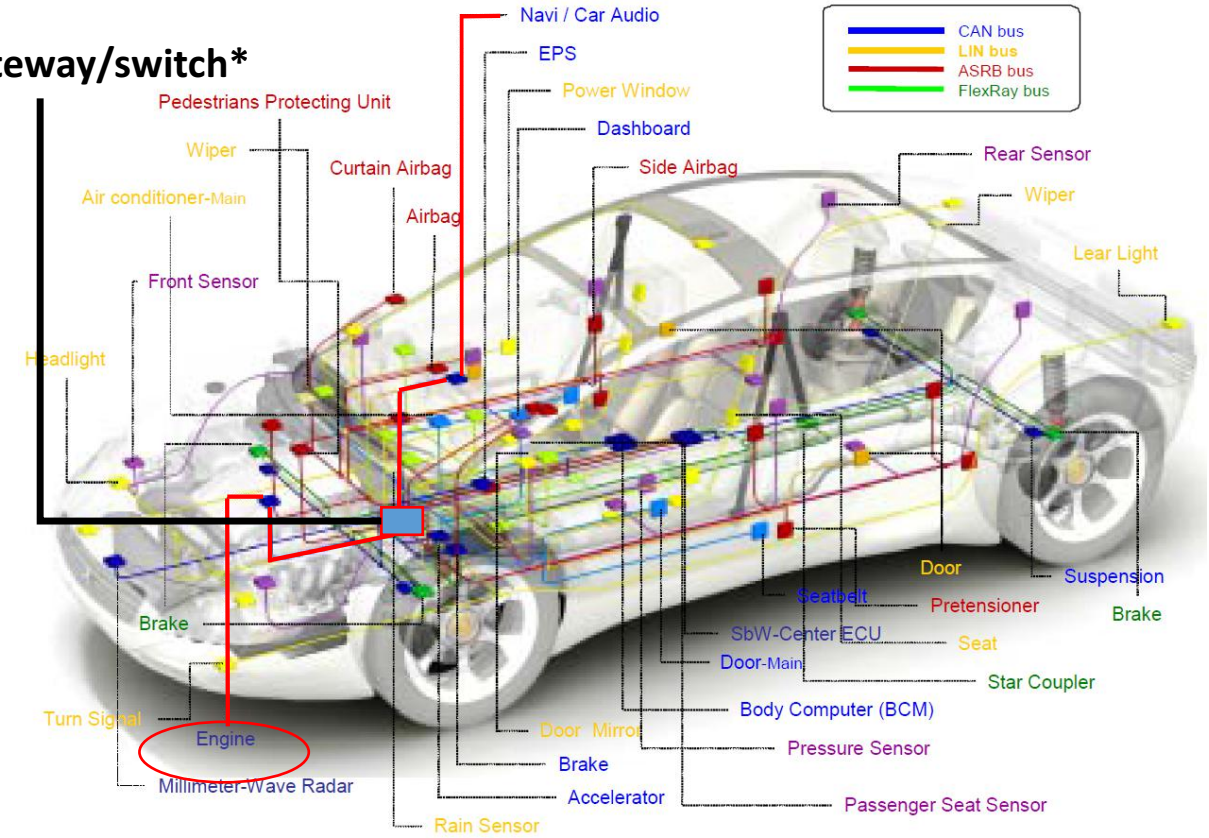
#Attack surface local interfaces

- Local I/O
  - CAN interfaces
  - Ethernet
  - WiFI
  - OBD-II

- Wireless components and ECUs
  - Long Radio:
    - GSM/UMTS
    - Radio/RDS
    - GPS
  - Short Radio:
    - WiFi/Bluetooth
    - TPMS
    - Keyless lock/start
    - Radars/Sensors/Cameras
- HeadUnit
  - Software components
    - WEB Browser
    - MP3/etc
    - RDS
    - Applications
    - Connected Car services

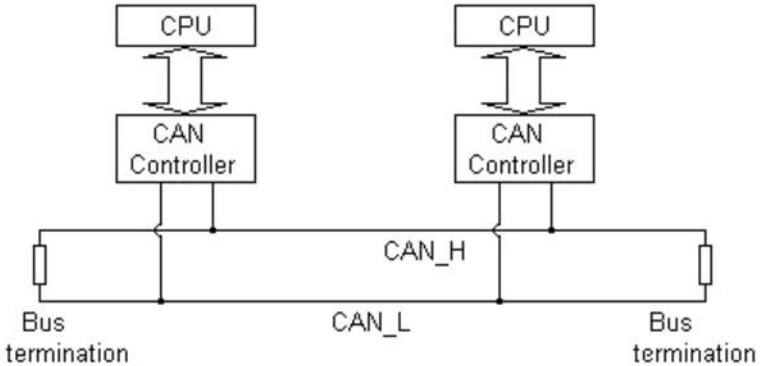
#CAN Bus

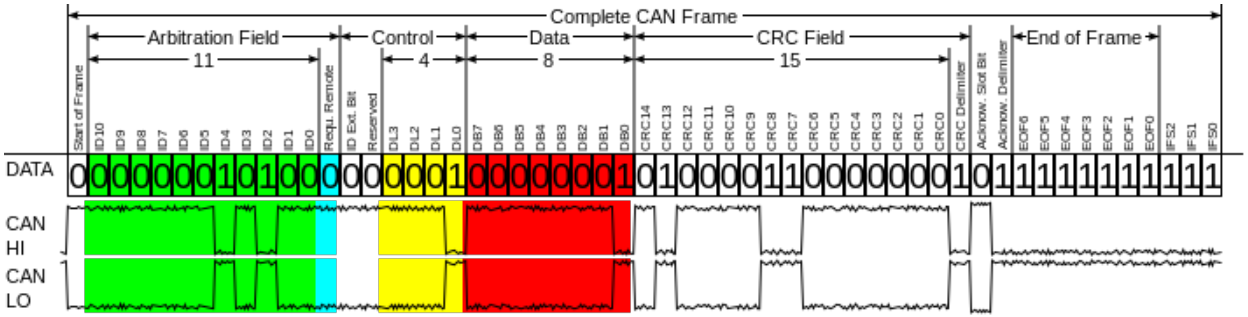
CAN gateway/switch*



* Different topology possible

#CAN Bus

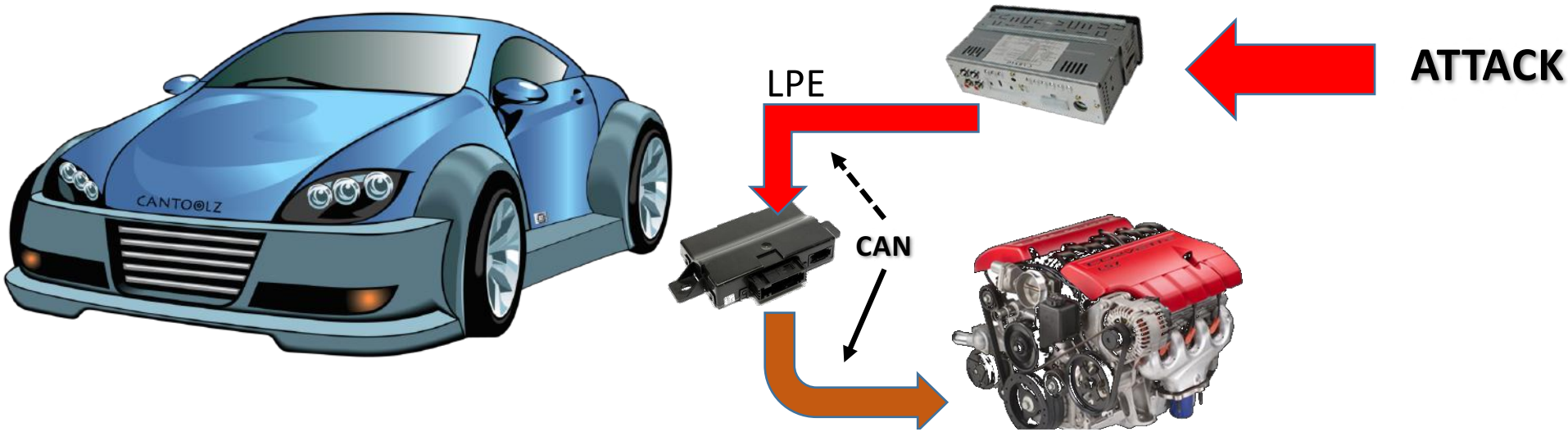




https://en.wikipedia.org/wiki/CAN_bus

#Attack vector (“remote” example)

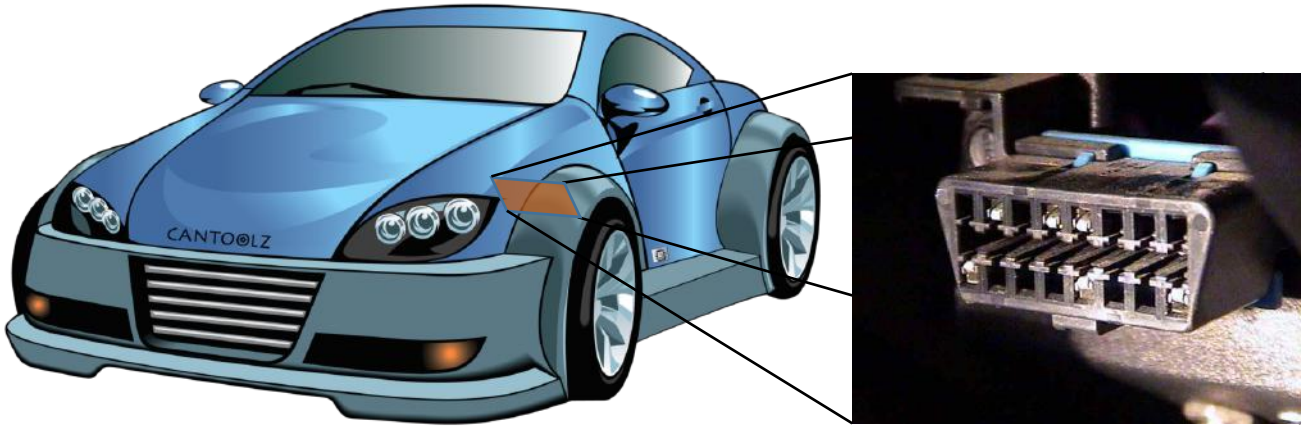
1. From the Internet/GSM/Wi-Fi into HU (RCE)
2. From HU to “intermediate device” like GatewayECU or another computer (for example privileged access to CAN bus)
3. PROFIT



#Backdoors

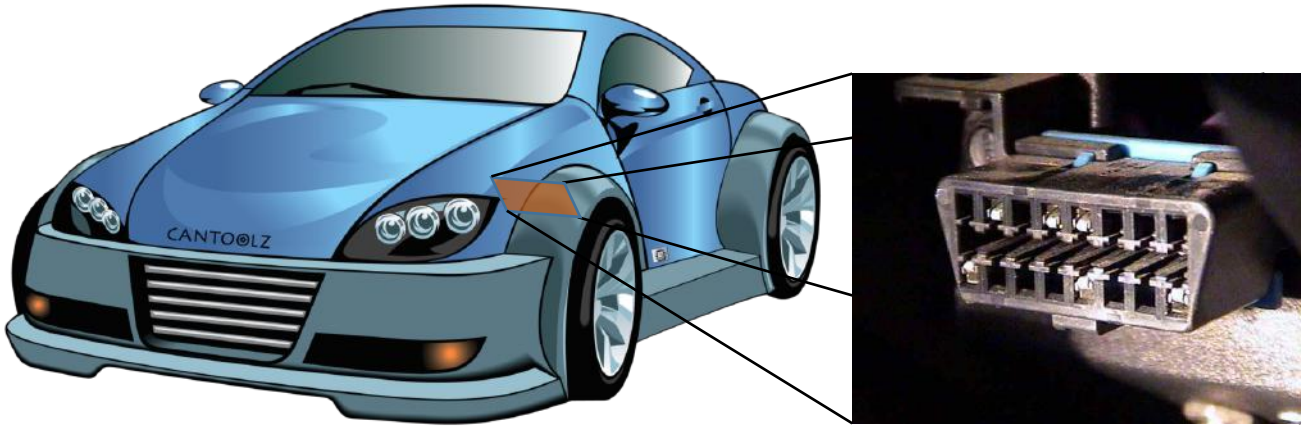
- Backdoor connected to OBD2 (external, not stealth mode)
- Backdoor connected to CAN BUS (internal, stealth)

.. or you could always compromise Internet back-end services (for connected car) and spy remotely.



#Local vector

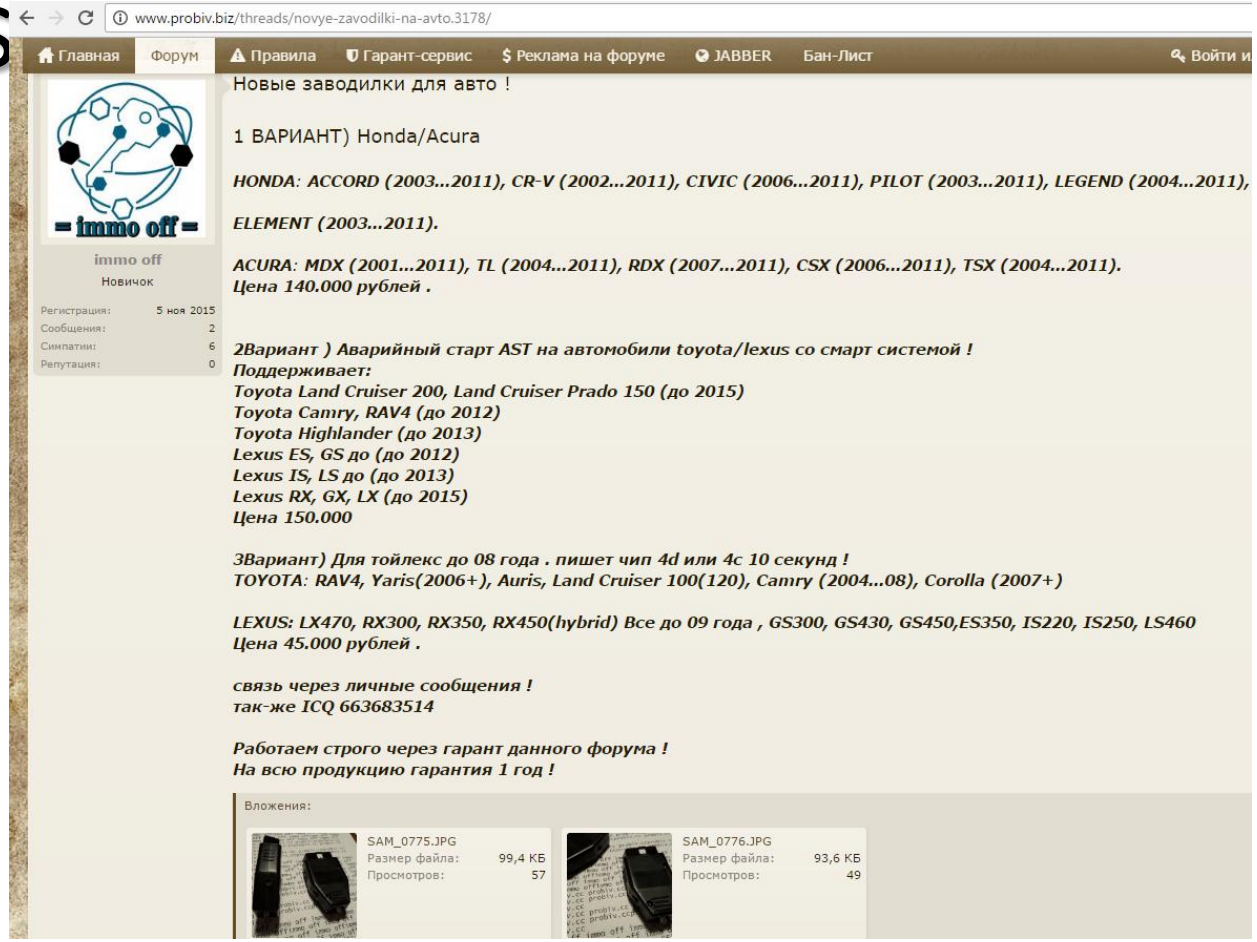
- CAN/K-LINE and UDS (over OBD2 or unauthorized CAN access)
  - “Fake” ECU
  - Firmware “update”
  - Keys reset/rewrite



#Unauthorized CAN access

#OBD-II and UDS

ISO 14229-1



www.probiv.biz/threads/novye-zavodilki-na-avto.3178/

Главная | Форум | Правила | Гарант-сервис | $ Реклама на форуме | JABBER | Бан-Лист | Войти и

immo off
Новичок
Регистрация: 5 ноя 2015
Сообщения: 2
Симпатии: 6
Репутация: 0

Новые заводилки для авто !

1 ВАРИАНТ) Honda/Acura

HONDA: ACCORD (2003...2011), CR-V (2002...2011), CIVIC (2006...2011), PILOT (2003...2011), LEGEND (2004...2011),

ELEMENT (2003...2011).

ACURA: MDX (2001...2011), TL (2004...2011), RDX (2007...2011), CSX (2006...2011), TSX (2004...2011).
Цена 140.000 рублей .

2Вариант) Аварийный старт AST на автомобили toyota/lexus со смарт системой !
Поддерживает:
Toyota Land Cruiser 200, Land Cruiser Prado 150 (до 2015)
Toyota Camry, RAV4 (до 2012)
Toyota Highlander (до 2013)
Lexus ES, GS до (до 2012)
Lexus IS, LS до (до 2013)
Lexus RX, GX, LX (до 2015)
Цена 150.000

3Вариант) Для тойлекс до 08 года . пишет чип 4d или 4c 10 секунд !
TOYOTA: RAV4, Yaris(2006+), Auris, Land Cruiser 100(120), Camry (2004...08), Corolla (2007+)

LEXUS: LX470, RX300, RX350, RX450(hybrid) Все до 09 года , GS300, GS430, GS450,ES350, IS220, IS250, LS460
Цена 45.000 рублей .

связь через личные сообщения !
так-же ICQ 663683514

Работаем строго через гарант данного форума !
На всю продукцию гарантия 1 год !

Вложения:

SAM_0775.JPG
Размер файла: 99,4 КБ
Просмотров: 57

SAM_0776.JPG
Размер файла: 93,6 КБ
Просмотров: 49

#Not only for *BAD* things, like theft and hacking…

UGLY THINGS:

- ‘Paid’ features unlock (illegal)
- Resets: VIN, mileage …

GOOD THINGS (on you own risk):

- Custom anti-theft systems
- Custom/DIY connected-car systems
- MOD’s, custom firmware for ECU…

#Tools for CAN

A lot of really good tools:

http://illmatics.com/content.zip - with examples from Charlie and Chris talk about CAN
https://github.com/ericevenchick/CANard - Abstract CAN lib with UDS/ISO TP support
https://github.com/zombieCraig/UDSim - Fuzzing, traffic simulator and more

Moarrrr: **https://github.com/jaredmichaelsmith/awesome-vehicle-security**

BUT, my needs are differnet:

- **HARDWARE independent** software for CAN bus reverse engineering and black-box analysis
- **Flexible and powerful framework** with multi interface support, for MITM, fuzzing and scanning
- **Module based framework**, where all modules could be used the way I want (like GNURadio design)
- **Features**: like data-type analysis, stats-analyses, UDS detection, CAN network emulator
- **API** interface

Nmap + MetaSploit + BurpSuite + GNURadio + “something like that”, but for CAN network….

YACHT - YET ANOTHER CAR HACKING TOOL

- Open Source Python Framework (Apache 2)
- Works on Windows/Linux/Mac OS
- All code in Python3

- CANToolz core engine:
  - Multi interface/bus support
    - MITIM supported
  - As python lib (dev API)
  - WEB API
  - Console/ WEB GUI (extendable)

Ready-to-use modules:

- CAN firewall
- Simple Fuzzer, and proxy/MITM-fuzzer
- UDS Scanner/sniffer
- Stats analysis features… a lot of
- Dump and replay
- Extendable CAR/CAN emulator (ECU level)

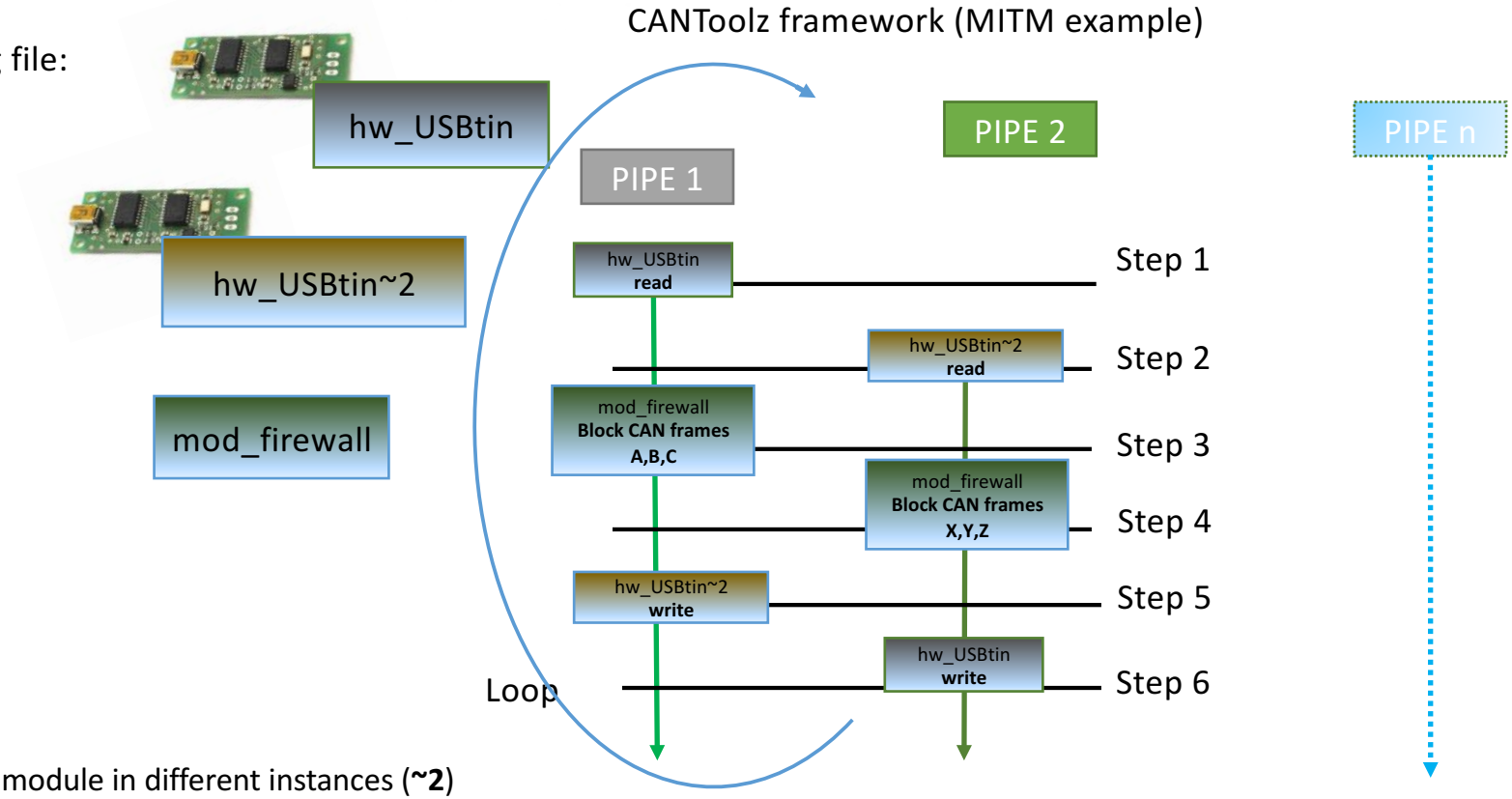
- TCP2CAN I/O module, **tunnels and more!**
- USBTin I/O module
- CANBus Triple I/O module
- CAN Socket I/O module (linux only)

Ready to use for CAN traffic reverse engineering and black-box analysis!

https://github.com/eik00d/CANToolz

#CANToolz design: pipe's concept

In config file:



- You can use module in different instances (**~2**)
  - Different memory, state
- You can use same instance multiple times
  - Same memory/state

https://github.com/eik00d/CANToolz

#CANToolz design: pipe's concept

In config file:

```
#  Load modules
load_modules = {
    'hw_USBtin'       :,
    'hw_USBtin~2'     : ,
    'mod_firewall'    : {},
}


# Scenario with steps

actions = [
    {'hw_USBtin'       : {'pipe': 1, 'action':     'read'}},
    {'hw_USBtin~2': {'pipe': 2, 'action':     'read'}},

    {'mod_firewall'    : {'pipe': 1, 'black_list': [1337, 31337]}},
    {'mod_firewall'    : {'pipe': 2, 'black_list': [0x1122, 0x2211]}},

    {'hw_USBtin~2': {'pipe': 1, 'action':     'write'}},
    {'hw_USBtin'       : {'pipe': 2, 'action':     'write'}}
]
```

https://github.com/eik00d/CANToolz

#DEV API example: one-byte proxy-fuzzer

```python
def counter(self):
    if self._i >= 255:
        self._i = 0
        self._active = False
    self._i += 1
    return self._i - 1

def do_init(self, params):
    self._active = False

def do_start(self, params):
    self._i = 0

# Change one byte to random
def do_fuzz(self, can_msg, byte_to_fuzz):
    if 0 < byte_to_fuzz < 9:
        can_msg.CANFrame.frame_data[byte_to_fuzz - 1] = self.counter()
    return can_msg

# Effect (could be fuzz operation, sniff, filter or whatever)
# can_msg - CANToolz message from the pipe (IN)
def do_effect(self, can_msg, args):
    # can_msg.CANData - boolean, if CANFrame in the Message
    if can_msg.CANData and can_msg.CANFrame.frame_type == CANMessage.DataFrame:
        if can_msg.CANFrame.frame_id in args.get('fuzz', []) and 'byte' in args:
            can_msg = self.do_fuzz(can_msg, args['byte'])
            can_msg.bus = self._bus
        elif 'nfuzz' in args and can_msg.CANFrame.frame_id not in args.get('nfuzz', []) and 'byte' in args:
            can_msg = self.do_fuzz(can_msg, args['byte'])
            can_msg.bus = self._bus
    # can_msg - CANToolz message TO the pipe (out)
    return can_msg
```

#Documentation

GitHub, Inc. [US] | https://github.com/eik00d/CANToolz/wiki

experience of CANToolz user's community!

CANToolz use-cases

1) **Black-Box analyzing**:

- How to find control CAN frames: unlock doors and etc by @Z0ha4, @_Saplt, @_j0hnni3
- UDS Scan: how to find UDS services by Anton Sysoev
- And once again: how to find control/statuses frames - PHDays VI CAN challange
- TBD

2) **Fuzzing and vulnerability analysis**:

- MITM/UDS Tester analyzer, how to find SecurityAccess key by Anton Sysoev
- TBD

3) **Testing and validation in R^D**:

- TBD

4) **Creating OWN modules**:

- TBD

This framework can be used as:

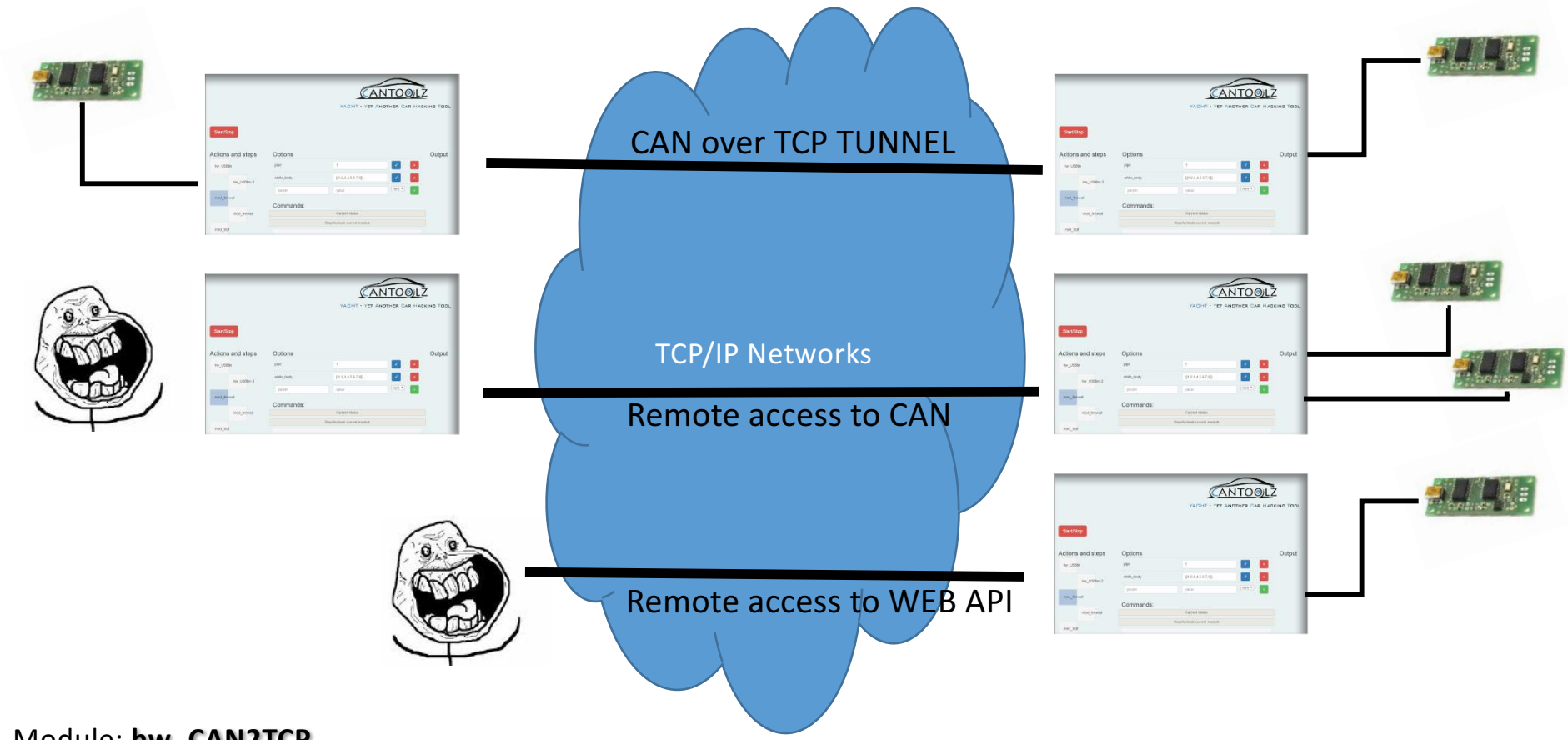
WIKI on GitHub:

- Use-cases and usage examples
- Modules documentation (not for all..eh, outdated sometimes)
- API documentation

Blog: https://asintsov.blogspot.com

- Developer's blog

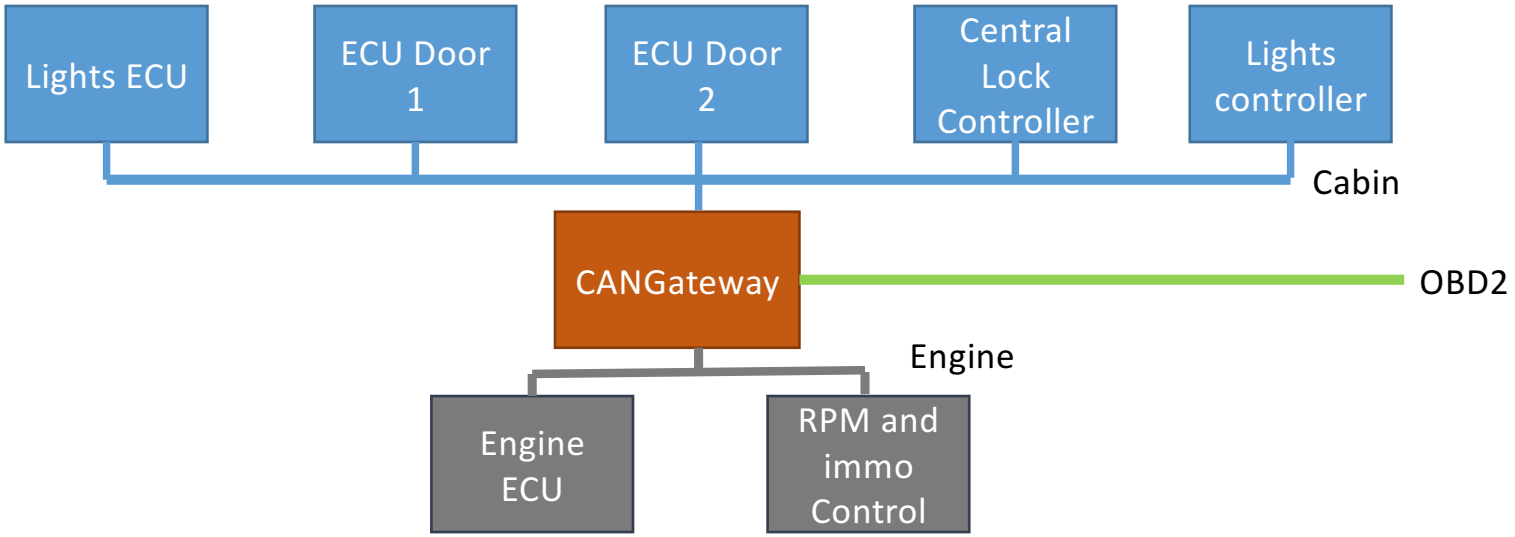
#CANToolz CAN over TCP



Module: **hw_CAN2TCP**

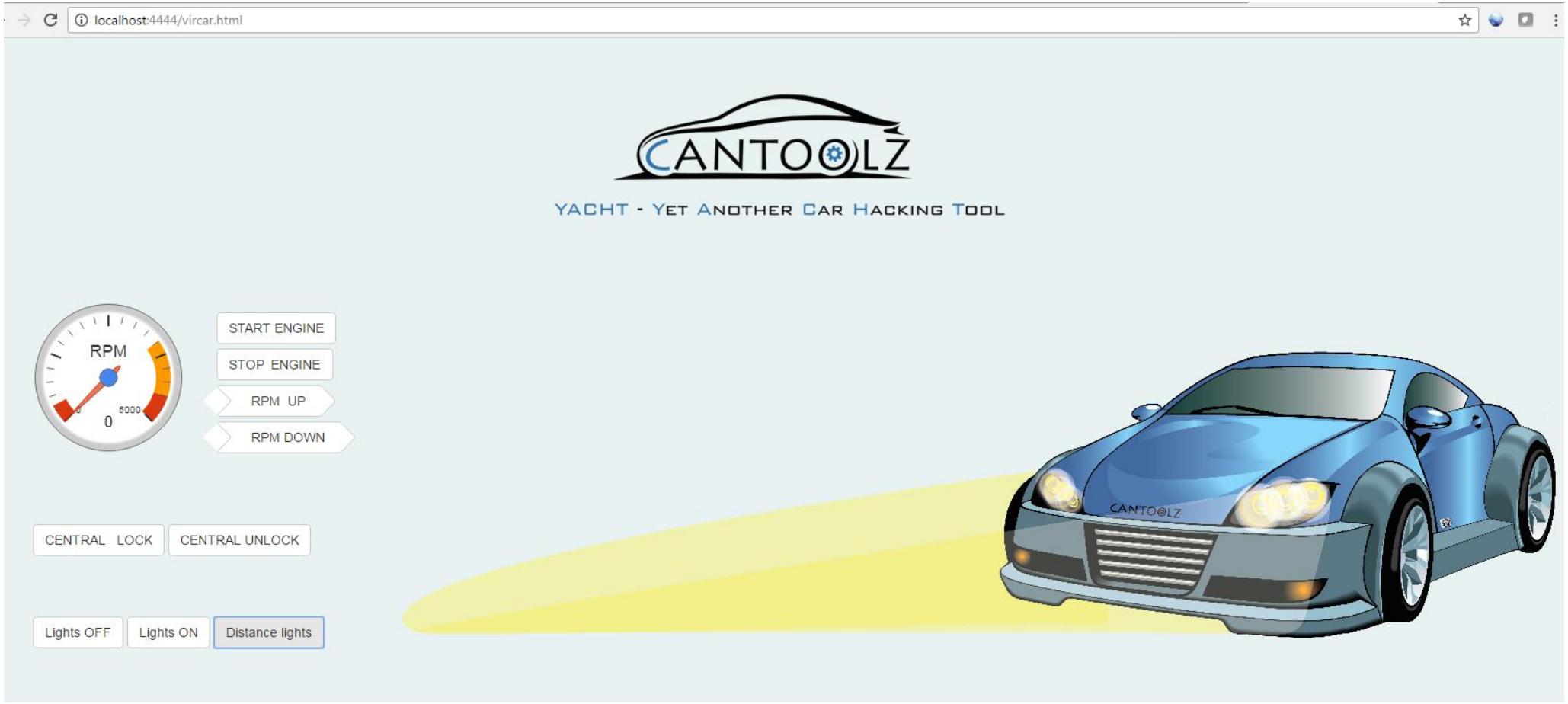
#CANToolz: car emulator

Inspired by **@dn5__** VIRcar emulator (https://github.com/dn5/vircar), I have created ECU modules as **CANtoolz modules** and connect them as they are connected in real car.
So I have ECU devices emulators, worked on different CAN buses and connected via CAN gateway.



- Door sensors
- Lights sensors
- RPM sensors
- CANGateway routing rules
- VIN fingerprint
- UDS support (auth, key)
- Engine start – control auth.

#CANToolz car emulator

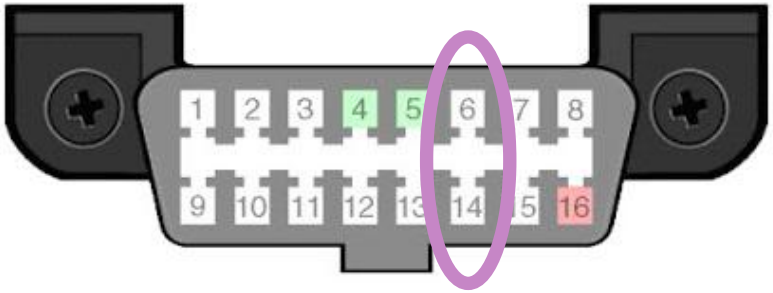


#OBD-II

What can we do over CAN?

- Data analysis (if exists, on some new cars OBD2 interface has no 'live traffic')
- OBD2 (boring)
- UDS
  - MITM – Session Hijacking
  - Proxy - Just sniffing
  - SCAN - Black-Box search (like nmap)

Data Link Connector (vehicle OBDII port)



1 Make/Model Specific
2 SAE J1850-PWM POS(+) or SAE J1850-VPW POS(+)
3 Make/Model Specific
4 Chassis Ground (all protocols)
5 Signal Ground (all protocols)
6 ISO15765-4 CAN-Bus High
7 ISO9141-2 K-Line or ISO14230-4 KWP2000 K-Line
8 Make/Model Specific
9 Make/Model Specific
10 SAE J1850-PWM NEG(-)
11 Make/Model Specific
12 Make/Model Specific
13 Make/Model Specific
14 ISO15765-4 CAN-Bus Low
15 ISO9141-2 L-Line or ISO14230-4 KWP2000 L-Line
16 +12v (always on) (all protocols)

#UDS Scan – black box

CANToolz modules: **gen_ping/mod_stat** (examples/uds_scan.py)

Generate UDS pings with chooses services and subcommands

Analyzing all traffic and detect UDS sessions

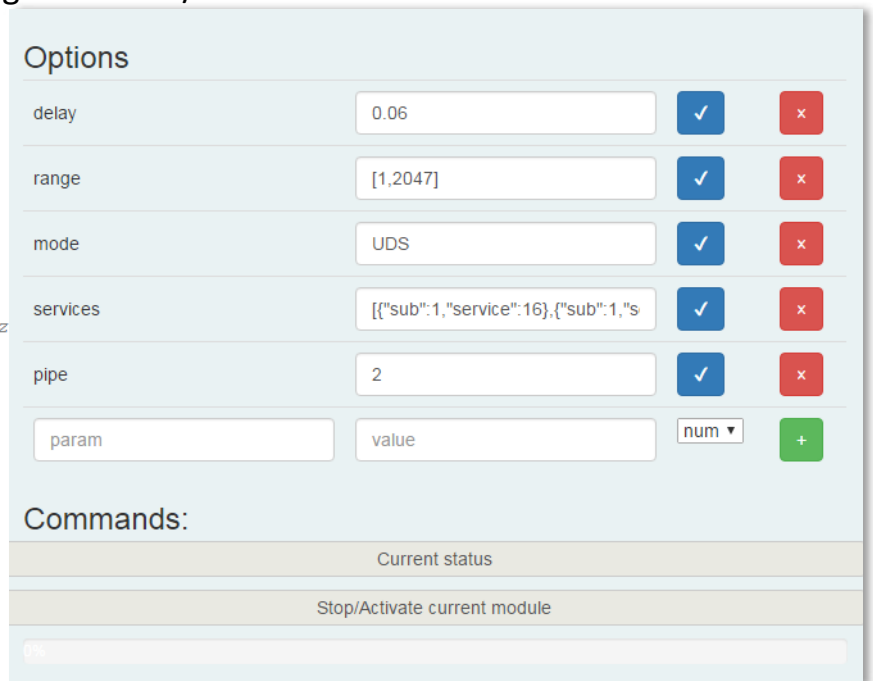
UDS based on ISO TP, sessions, authentication via SecurityAccess

ISO TP: first byte(s) of CAN data used as fragmentation flag and index/counter: **ISO 15765-2**

CANToolz CONFIG:

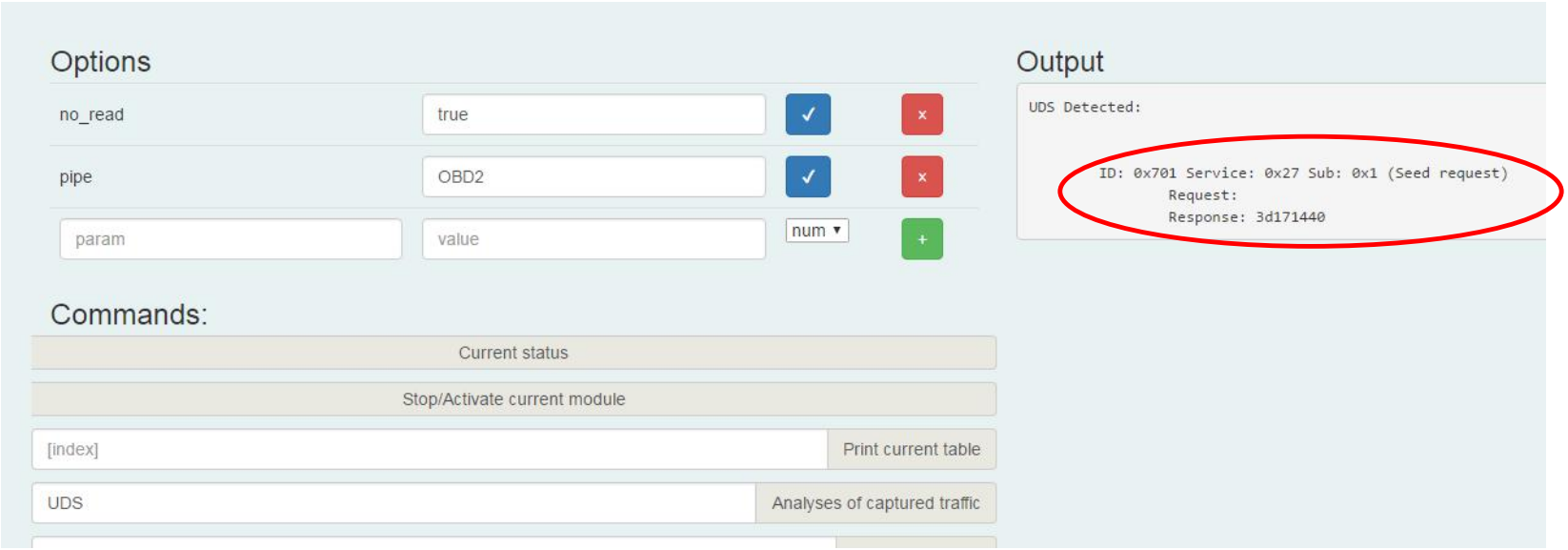
```
# Load needed modules
load_modules = {
    'hw_USBtin':    {'port': 'auto', 'debug': 1, 'speed': 500},
    'gen_ping' :    {},
    'mod_stat':     {}
}

actions = [
    {'hw_USBtin':   {'action': 'read'}}, # Read to PIPE 1
    {'mod_stat':    {}},                 # Mod stat (with CAN traff  yz
    {'gen_ping':    {                    # Generate UDS requests
        'pipe': 2,
        'delay': 0.06,
        'range': [1, 2047],            # ID range (from 1790 to 1794)
        'services':[{'service': 0x10, 'sub': 0x01},
                    {'service': 0x3E, 'sub': 0x01}],
        'mode':'UDS'}
    },
    {'mod_stat':    {'pipe': 2}},
     {'hw_USBtin':   {'action': 'write','pipe':2}}
    ]
```

WEB GUI:



#UDS Scan – DEMO on emulator

Analyzing captured traffic after scan for devices that has UDS SecurityAccess :



#UDS Scan – DEMO on real captured traffic

Sent by **Anton Sysoev**

Output

```
UDS Detected:

        ID: 0x70a Service: 0x10 Sub: 0x1 (Enter diag session)
                Request:
                Response: 003201f4

        ID: 0x70a Service: 0x3e Sub: 0x1 (Tester present)
                Request:
                Error: Subfunction not supported

        ID: 0x70a Service: 0x3e Sub:  (ALTERNATIVE INTERPRETATION if NO SUB)  (Tester)
                Request:
                Error: Incorrect message length or invalid format

        ID: 0x70b Service: 0x10 Sub: 0x1 (Enter diag session)
                Request:
                Response: 003201f4

        ID: 0x70b Service: 0x3e Sub: 0x1 (Tester present)
                Request:
                Error: Subfunction not supported

        ID: 0x70b Service: 0x3e Sub:  (ALTERNATIVE INTERPRETATION if NO SUB)  (Tester)
                Request:
                Error: Incorrect message length or invalid format

        ID: 0x70c Service: 0x10 Sub: 0x1 (Enter diag session)
                Request:
                Response: 003201f4
```

In real life it is not so simple as described in books:

- padding,
- UDS offset is not 8…

And more devices were found…

#UDS tester tools sniffing results

Sent by **Anton Sysoev**

Output

```
UDS Detected:


        ID: 0x70a Service: 0x10 Sub: 0x3 (Extended Diag Session)
                Request:
                Response: 003201f4

        ID: 0x70a Service: 0x22 Sub: 0xf1 (Read Data By Identifier)
                Request: 9e
                Response: 9e45565f4550485641313456573336383030303000
                ASCII: .EV_EPHVA14VW3680000.


        ID: 0x70a Service: 0x22 Sub: 0x6 (Read Data By Identifier)
                Request: 01
                Response: 0103

        ID: 0x70b Service: 0x10 Sub: 0x3 (Extended Diag Session)
                Request:
                Response: 003201f4

        ID: 0x70b Service: 0x22 Sub: 0xf1 (Read Data By Identifier)
                Request: 9e
                Response: 9e45565f52444b42455255333000
                ASCII: .EV_RDKBERU30.


        ID: 0x70b Service: 0x22 Sub:  (ALTERNATIVE INTERPRETATION if NO SUB)  (Read Data By Identifier)
                Request: 0601
                Error: Request out of range

        ID: 0x70c Service: 0x10 Sub: 0x3 (Extended Diag Session)
                Request:
                Response: 003201f4
```

#UDS SecurityAccess auth sniff

Sent by **Anton Sysoev**

Output

```
UDS Detected:


        ID: 0x74a Service: 0x22 Sub: 0x5 (Read Data By Identifier)
                Request: 20
                Response: 207fbe0f34000cc35195f207e484620028

        ID: 0x74a Service: 0x22 Sub:  (ALTERNATIVE INTERPRETATION if NO SUB)  (Read Data By Identifier)
                Request: f199
                Error: Request out of range

        ID: 0x74a Service: 0x2e Sub: 0xf1 (Write Data By Identifier)
                Request: 99160508
                Response: 99

        ID: 0x74a Service: 0x2e Sub: 0x5 (Write Data By Identifier)
                Request: 207fbe0f34000cc35190f207e484620028
                Response: 20

        ID: 0x74a Service: 0x27 Sub: 0x3 (Security Access)
                Request:
                Response: 00000fd1

        ID: 0x74a Service: 0x27 Sub: 0x4 (Security Access)
                Request: 000020d3
                Response:
```

Seed from ECU…

Auth. code from tester to ECU…

Auth.PIN = 0x20d3 – 0xfd1
= **0x1102**

#UDS SecurityAccess and immo bypass on CANToolz emulator

DISCLAMER: this is not a REAL hack, not real data, and even not a real situation. This is just a simulation of what could be in real, So this is just **my imagination and bugs in my VIRTUAL car, only for education purposes!**

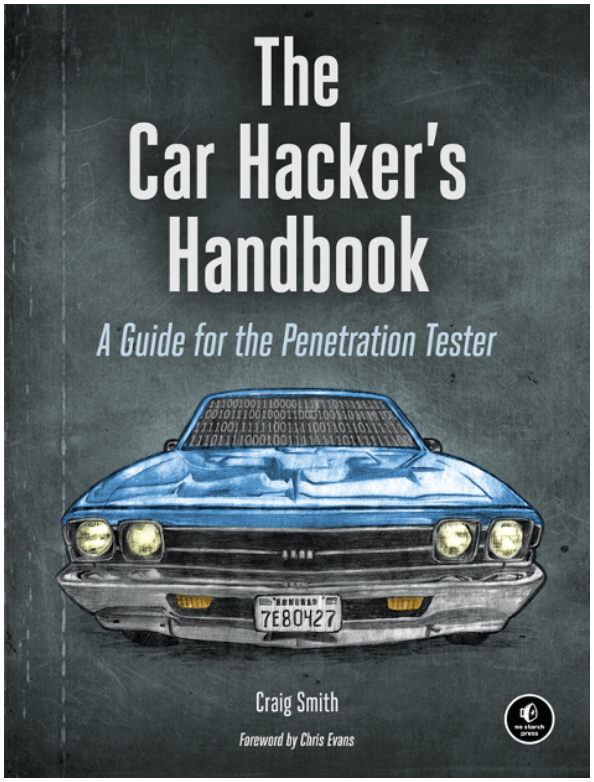
SCENRAIO:

1. Attacker drilled a door and connect to the CABIN bus. Now they can unlock doors. (probably they need to activate bus, but in my emulator let's think we have the bus active all the time)
2. Attacker has no immo keys, which are unique. But he could use UDS to reset these keys to his own (not a real situation right now, I hope). Because Immobilizer keys are unique for each car but SecurityAccess code is not, it could be shared and could be known (by RE). Access to OBD2 is enough for UDS access…
3. Now an attacker can sniff VIN from the same CAN bus and try to start engine with "new keys".

DEMO

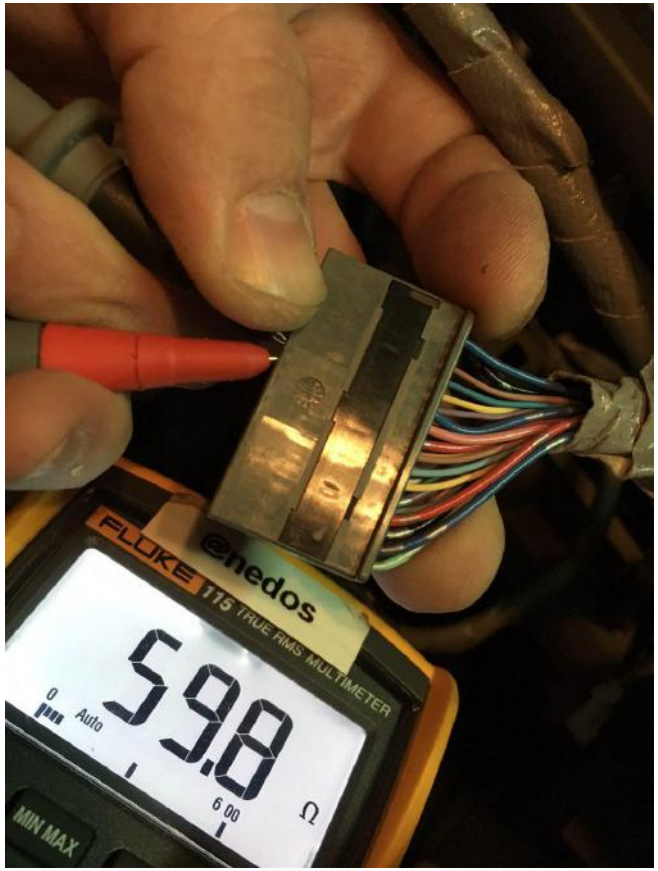
#CAN Reverse Engineering/BlackBox analysis

- With what command I can unlock car?
- What are door status signals?
- How to get RPM?
- And all things like that: data analysis

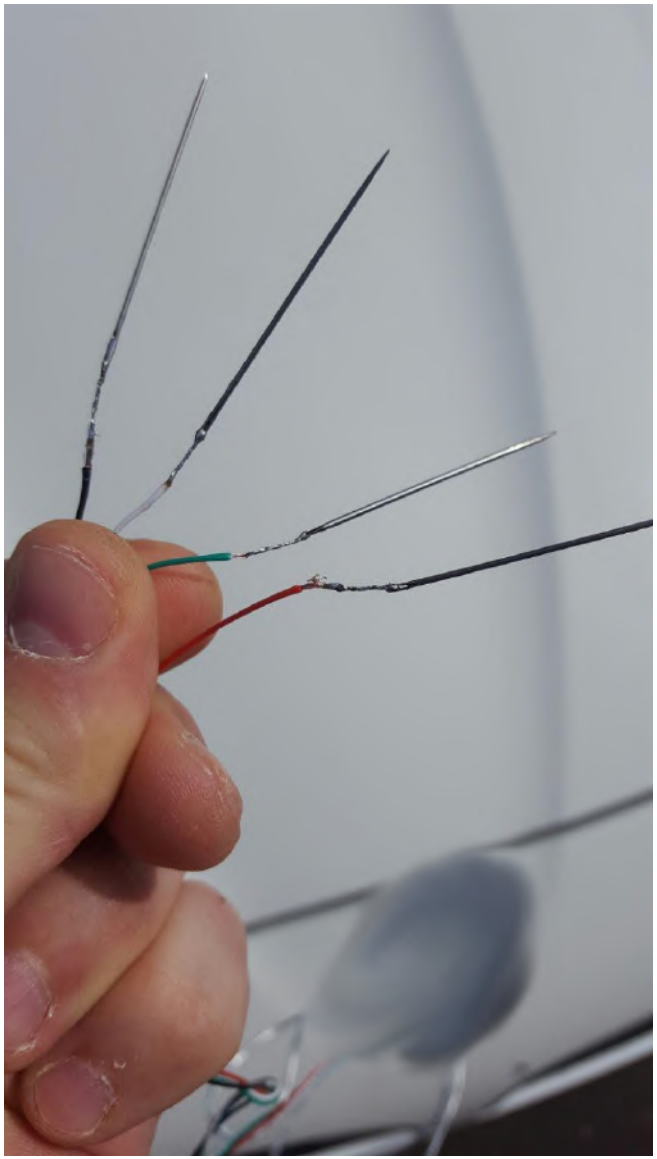


Simple techniques, like DUMP and BINARY-SEARCH
via REPLAY… but then I understood that I could do it better in CANToolz…

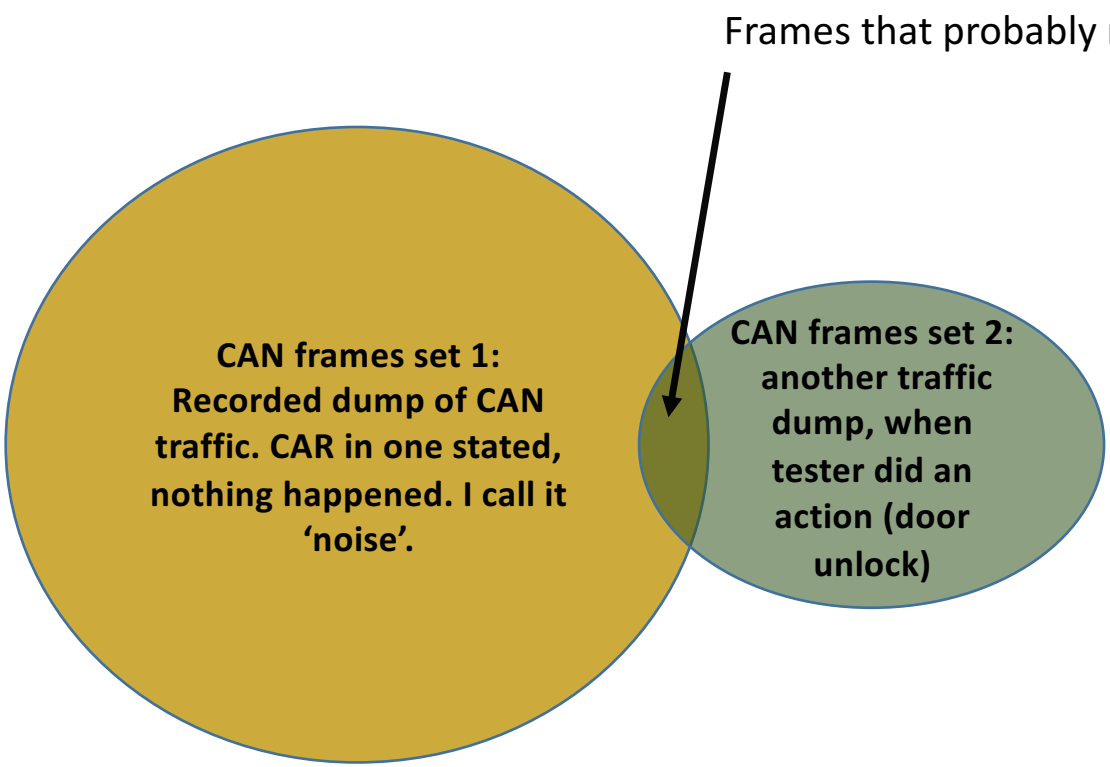
#BUS search


FLUKE 115 TRUE RMS MULTIMETER
@nedos
59.8
Auto
MIN MAX


@nedos
120.0
HOLD
MIN MAX
RANGE
OFF
COM



#CANToolz: DIFF method



- **New ID**
- **New DATA values**

Also we could set-up a filter for amount of frames in the second set with "new values"

*All next examples and demos are search for LOCK/UNLOCK

#CANToolz: DIFF method - demo

[buffer index 1], [buffer index 2], [uniq values max] | Print Diff between two buffers

[buffer index 1], [buffer index 2] | Print Diff between two buffers (new ID only)

DIFF sets between noise.dump.txt_buffer and lock_action.dump.txt_buffer

BUS	ID	LENGTH	MESSAGE	ASCII	DESCR	COUNT
mod_stat	0xef82010	1	80			2
mod_stat	0xef82010	1	40			2
mod_stat	0x12f85050	5	0002ef0000			1
mod_stat	0x12f83110	1	18			1
mod_stat	0x12f83110	1	10			1
mod_stat	0x12f83110	1	14			1
mod_stat	0x12f83110	1	04			4
mod_stat	0x12f83110	1	24			1
mod_stat	0x12f83110	1	28			2
mod_stat	0x12f83210	1	00			5
mod_stat	0x12f84110	1	00			5
mod_stat	0x12f84210	1	00			5
mod_stat	0x12f85351	8	3700000000016380			2
mod_stat	0x12f85851	5	3700003900			6
mod_stat	0x12f85250	4	870085b5			4
mod_stat	0xef81296	2	0100			1

Diff between two sets

DIFF sets between noise.dump.txt_buffer and lock_action.dump.txt_buffer

BUS	ID	LENGTH	MESSAGE	ASCII	DESCR	COUNT
mod_stat	0x12f83210	1	00			5
mod_stat	0x12f84110	1	00			5
mod_stat	0x12f84210	1	00			5
mod_stat	0xef81296	2	0100			1

Diff between two sets, with filter:
only 2 values detected in new set

Lock/Door statuses
CAN command for lock

DIFF sets between noise.dump.txt_buffer and lock_action.dump.txt_buffer

BUS	ID	LENGTH	MESSAGE	ASCII	DESCR	COUNT
mod_stat	0xef81296	2	0100			1

Diff between two sets, with filter:
only 1 value detected in new set

#CANToolz: DIFF method - demo

Output

DIFF sets between bh_dumps/stat_noise_buffer and bh_dumps/stat_lock_buffer

BUS	ID	LENGTH	MESSAGE	ASCII	DESCR	COUNT
mod_stat	0x199	6	fffd4b7fff2f			1
mod_stat	0x1b9	2	3c46			4
mod_stat	0x2bf	3	226223	"b#		8
mod_stat	0x2bf	3	220223			26
mod_stat	0x163	8	e6f0ef24b924ffff			6
mod_stat	0x163	8	8efaef24b924ffff			5
mod_stat	0x163	8	d2f5ef24b924ffff			5
mod_stat	0x163	8	68f5ef24ba24ffff			1
mod_stat	0x163	8	5cf0ef24ba24ffff			1
mod_stat	0x163	8	34faef24ba24ffff			1
mod_stat	0x163	8	69f9ef24b924ffff			7
mod_stat	0x163	8	8ff4ef24b924ffff			5
mod_stat	0x163	8	e7feef24b924ffff			6
mod_stat	0x281	2	adf3			1
mod_stat	0x34c	8	740000008eaaf0ff			4
mod_stat	0x349	5	c726430400			3
mod_stat	0x349	5	ca26430400			2
mod_stat	0x349	5	cb26420400			2
mod_stat	0x349	5	c826420400			4
mod_stat	0x349	5	c626420400			2
mod_stat	0x349	5	c526420400			2
mod_stat	0x349	5	c726420400			5
mod_stat	0xec	8	222200000008fcff			3
mod_stat	0x415	7	687480b2f10080			4
mod_stat	0x32e	8	ffffffa542ffff20			3
mod_stat	0x2ca	2	78b1			2
mod_stat	0x2ca	2	78b2			2
mod_stat	0x330	8	409e011d00391c12			1
mod_stat	0x2fc	7	21000000000000			1
mod_stat	0x2fc	7	22008000000000			4
mod_stat	0x2fc	7	82000000000000			2
mod_stat	0x328	6	d1fba609cf17			1
mod_stat	0x328	6	d2fba609cf17			1
mod_stat	0x328	6	d3fba609cf17			1
mod_stat	0x2a0	8	22222801ffffffff	""(.....		3
mod_stat	0xe2	8	f2ffe7fcffffffff			6
mod_stat	0xe6	8	f2ffe7fcffffffff			3
mod_stat	0xee	8	f2ffe7fcffffffff			7
mod_stat	0xea	8	82ffe7fcffffffff			4
mod_stat	0xf2	8	f2ff87fcffffffff			4

Diff between two sets

DIFF sets between bh_dumps/stat_noise_buffer and bh_dumps/stat_lock_buffer

BUS	ID	LENGTH	MESSAGE	ASCII	DESCR	COUNT
mod_stat	0x34c	8	740000008eaaf0ff			4
mod_stat	0xec	8	222200000008fcff			3
mod_stat	0x415	7	687480b2f10080			4
mod_stat	0x2a0	8	22222801ffffffff	""(.....		3
mod_stat	0xe2	8	f2ffe7fcffffffff			6
mod_stat	0xe6	8	f2ffe7fcffffffff			3
mod_stat	0xee	8	f2ffe7fcffffffff			7
mod_stat	0xea	8	82ffe7fcffffffff			4
mod_stat	0xf2	8	f2ff87fcffffffff			4

Diff between two sets, with filter:
only 2 values detected in new set

DIFF sets between bh_dumps/stat_noise_buffer and bh_dumps/stat_lock_buffer

BUS	ID	LENGTH	MESSAGE	ASCII	DESCR	COUNT

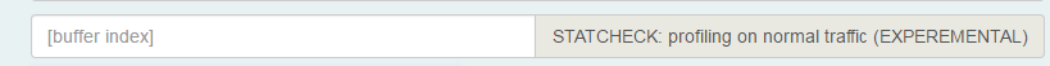
Diff between two sets, with filter:
only 1 value detected in new set

Hmmm… diff works, but you still need more analysis

#CANToolz: STATS. ABNORMALITIES METHOD

Stage 1. Learning.

Each CAN frame for chosen ID has a profile:



- Bits that has been changed (max 64 bits)

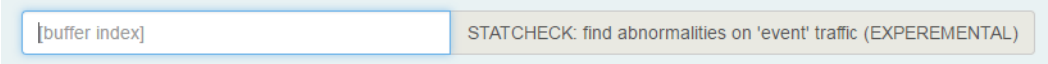
BIT_MASK = BIT_MASK **OR** PREVIOS_FRAME_DATA_BITS **XOR** NEW_FRAME_DATA_BITS

- Minimum time between CAN frames with same ID

Stage 2. Compare.

Each CAN frame for chosen ID has a profile:

- In bit mask of changed bits new bits detected.
- Time between frames less than minimum.



Stage 3. Correlation.

Find dependences on those changes between different ID (correlation). Remove all other changes from the result.

➔ **EVENT/ACTION SESSION selection!**

#CANToolz: STATS. ABNORMALITIES METHOD

DEMO

Extracted event as session

```
SELECTED SESSION(ready to dump into file now and for ACTIVE check)::

TIME      ID       LENGTH    MESSAGE             COMMENT
2.2089    0x2fc    7         21000000000000       first event, changed from 0081000000000000
2.2091    0x2a0    8         22222801ffffffff     first change from 88888801ffffffff, probably because of
2.2125    0xec     8         222200000008fcff     first change from 111100000008fcff, probably because of
2.2675    0x2fc    7         22008000000000       additional changes, probably because of: ['0x2fc', '0x2
2.3077    0xea     8         82ffe7fcffffffff     first change from 81ffe7fcffffffff, probably because of
2.3079    0xe2     8         f2ffe7fcffffffff     first change from f1ffe7fcffffffff, probably because of
2.3081    0xee     8         f2ffe7fcffffffff     first change from f1ffe7fcffffffff, probably because of
2.3522    0x2a0    8         22222801ffffffff     'impulse' rate increased abnormally: EVENT
2.3678    0x2fc    7         22008000000000       'impulse' rate increased abnormally: NEW STAGE
2.3722    0xec     8         222200000008fcff     'impulse' rate increased abnormally: EVENT
2.408     0x2fc    7         22008000000000       'impulse' rate increased abnormally: EVENT
2.4677    0x2fc    7         22008000000000       'impulse' rate increased abnormally: EVENT
2.5228    0x2a0    8         22222801ffffffff     'impulse' rate increased abnormally: EVENT
2.5334    0xec     8         222200000008fcff     'impulse' rate increased abnormally: EVENT
2.6279    0x2a0    8         88888801ffffffff     released value back
2.6313    0xe6     8         f2ffe7fcffffffff     first event, changed from f1ffe7fcffffffff
2.7211    0x2fc    7         82000000000000       'impulse' rate increased abnormally: EVENT
2.8823    0x2fc    7         82000000000000       additional changes, probably because of: ['0xe6']
```

#CANToolz: STATS. ABNORMALITIES METHOD

Stage 4: automatic detection.

STATCHECK: find action frame (EXPEREMENTAL)

Lock button pressed and released… ?

Automatic replay one by one CAN frames from extracted session and then check if OTHER ID have changed BIT_MASK, like on stage 2.

If this happened then it is mean that Our last replayed frame caused those changes and it was our target!

DEMO on emulator

SELECTED SESSION(ready to dump into file now and for ACTIVE check)::

TIME	ID	LENGTH	MESSAGE	COMMENT
2.2089	0x2fc	7	21000000000000	first event, changed from 0081000000000000
2.2091	0x2a0	8	22222801ffffffff	first change from 88888801ffffffff, probably because of
2.2125	0xec	8	222200000008fcff	first change from 111100000008fcff, probably because of
2.2675	0x2fc	7	22008000000000	additional changes, probably because of: ['0x2fc', '0x2
2.3077	0xea	8	82ffe7fcffffffff	first change from 81ffe7fcffffffff, probably because of
2.3079	0xe2	8	f2ffe7fcffffffff	first change from f1ffe7fcffffffff, probably because of
2.3081	0xee	8	f2ffe7fcffffffff	first change from f1ffe7fcffffffff, probably because of
2.3522	0x2a0	8	22222801ffffffff	'impulse' rate increased abnormally: EVENT
2.3678	0x2fc	7	22008000000000	'impulse' rate increased abnormally: NEW STAGE
2.3712	0xec	8	222200000008fcff	'impulse' rate increased abnormally: EVENT
2.408[illegible]	0x2fc	7	22008000000000	'impulse' rate increased abnormally: EVENT
2.4[illegible]	0x2fc	7	22008000000000	'impulse' rate increased abnormally: EVENT
2.[illegible]28	0x2a0	8	22222801ffffffff	'impulse' rate increased abnormally: EVENT
2.[illegible]34	0xec	8	222200000008fcff	'impulse' rate increased abnormally: EVENT
2.[illegible]279	0x2a0	8	88888801ffffffff	released value back
[illegible]6313	0xe6	8	f2ffe7fcffffffff	first event, changed from f1ffe7fcffffffff
[illegible].7211	0x2fc	7	82000000000000	'impulse' rate increased abnormally: EVENT
2.8823	0x2fc	7	82000000000000	additional changes, probably because of: ['0xe6']

Probably STATUS messages

#CANToolz: features in-dev

- Automatic fields extractor based on changed bits rate
- Automatic data-type identification: signal, counter, integers, ASCII

Those features **already in CANToolz** partially and could work for some limited situations. But still not stable and could produce “false positives”.

#CANToolz: fields detection

mod_stat	0x12f85050	5	00029f0000	13
mod_stat	0x12f85050	5	00029b0000	21
mod_stat	0x12f85050	5	0002a50000	11
mod_stat	0x12f85050	5	0002a60000	8
mod_stat	0x12f85050	5	00029e0000	18
mod_stat	0x12f85050	5	0002a30000	15
mod_stat	0x12f85050	5	0002980000	7
mod_stat	0x12f85050	5	0002a00000	27
mod_stat	0x12f85050	5	00029c0000	18
mod_stat	0x12f85050	5	0002a10000	19
mod_stat	0x12f85050	5	0002a70000	7
mod_stat	0x12f85050	5	0002ab0000	1
mod_stat	0x12f85050	5	0002b10000	5
mod_stat	0x12f85050	5	0002b80000	5
mod_stat	0x12f85050	5	0002bd0000	12
mod_stat	0x12f85050	5	0002c50000	11
mod_stat	0x12f85050	5	0002ce0000	1
mod_stat	0x12f85050	5	0002cf0000	2
mod_stat	0x12f85050	5	0002d40000	3
mod_stat	0x12f85050	5	0002d90000	2
mod_stat	0x12f85050	5	0002d10000	4
mod_stat	0x12f85050	5	0002d70000	3
mod_stat	0x12f85050	5	0002d20000	2
mod_stat	0x12f85050	5	0002cb0000	11
mod_stat	0x12f85050	5	0002c70000	3
mod_stat	0x12f85050	5	0002c20000	9
mod_stat	0x12f85050	5	0002c00000	12
mod_stat	0x12f85050	5	0002bb0000	11
mod_stat	0x12f85050	5	0002c10000	6
mod_stat	0x12f85050	5	0002bf0000	6
mod_stat	0x12f85050	5	0002be0000	6
mod_stat	0x12f85050	5	0002c30000	9
mod_stat	0x12f85050	5	0002d30000	2
mod_stat	0x12f85050	5	0002d80000	3

0x12f85050, int | Show values in fields for chosen ECU (EXPEREMENTAL)

Output

```
Data by fields in ECU: 0x12f85050


by length: 5

671    0
667    0
677    0
678    0
670    0
675    0
664    0
672    0
668    0
673    0
679    0
683    0
689    0
696    0
701    0
709    0
718    0
719    0
724    0
729    0
721    0
727    0
722    0
715    0
711    0
706    0
704    0
699    0
705    0
703    0
702    0
707    0
723    0
728    0
```

#CANToolz: meta-data

0x12f85050, .*, RPM data and probably speed	Meta-data: add description for frames
0x12f85050,5, int:8: Unknown, int:24: RPM	Meta-data: bits fields description
<filename>	Load meta-data
<filename>	Save meta-data

0x12f85050	5	Unknown: 0 RPM value: 671	RPM data and probably speed	13
0x12f85050	5	Unknown: 0 RPM value: 667	RPM data and probably speed	21
0x12f85050	5	Unknown: 0 RPM value: 677	RPM data and probably speed	11
0x12f85050	5	Unknown: 0 RPM value: 678	RPM data and probably speed	8
0x12f85050	5	Unknown: 0 RPM value: 670	RPM data and probably speed	18
0x12f85050	5	Unknown: 0 RPM value: 675	RPM data and probably speed	15
0x12f85050	5	Unknown: 0 RPM value: 664	RPM data and probably speed	7
0x12f85050	5	Unknown: 0 RPM value: 672	RPM data and probably speed	27
0x12f85050	5	Unknown: 0 RPM value: 668	RPM data and probably speed	18
0x12f85050	5	Unknown: 0 RPM value: 673	RPM data and probably speed	19
0x12f85050	5	Unknown: 0 RPM value: 679	RPM data and probably speed	7
0x12f85050	5	Unknown: 0 RPM value: 683	RPM data and probably speed	1
0x12f85050	5	Unknown: 0 RPM value: 689	RPM data and probably speed	5
0x12f85050	5	Unknown: 0 RPM value: 696	RPM data and probably speed	5
0x12f85050	5	Unknown: 0 RPM value: 701	RPM data and probably speed	12
0x12f85050	5	Unknown: 0 RPM value: 709	RPM data and probably speed	11
0x12f85050	5	Unknown: 0 RPM value: 718	RPM data and probably speed	1
0x12f85050	5	Unknown: 0 RPM value: 719	RPM data and probably speed	2
0x12f85050	5	Unknown: 0 RPM value: 724	RPM data and probably speed	3
0x12f85050	5	Unknown: 0 RPM value: 729	RPM data and probably speed	2
0x12f85050	5	Unknown: 0 RPM value: 721	RPM data and probably speed	4
0x12f85050	5	Unknown: 0 RPM value: 727	RPM data and probably speed	3
0x12f85050	5	Unknown: 0 RPM value: 722	RPM data and probably speed	2
0x12f85050	5	Unknown: 0 RPM value: 715	RPM data and probably speed	11

Meta-data – project's 'notes'. This gives tester ability to set labels and some data-extraction rules for already known data.

#CANToolz: loops detection with counters and ASCII

```
mod_stat    0x16f86250    8    0000000004000013                    46
mod_stat    0x16f86250    8    014        6423823    .[   FB8#    49
mod_stat    0x16f86250    8    023        1754313a   .6   GT1:    45
mod_stat    0x16f86250    8    033032313633000c    .02163..       44
```

Looks like VIN…

```
De-Fragmented frames (using loop-based detection):

        ID 0x16f86250 and length 28
                Data: 00000004000013        .[...]        04754313a3032313633000c
                ASCII: .......[   3.   )GT1:02163..
```

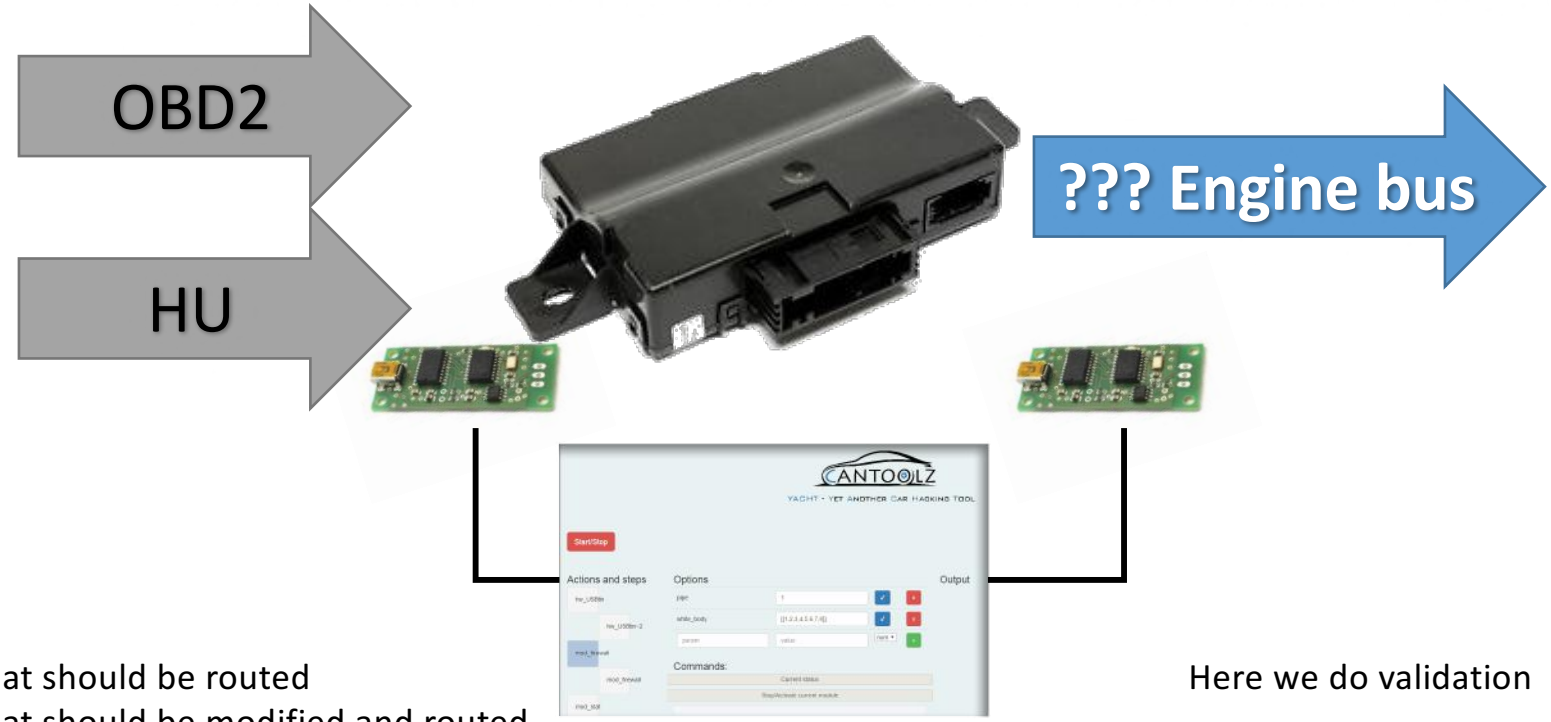
FRAG | Analyses of captured traffic

#Other side

- Validation tests
- Fuzzing
  Also usable by researchers, but **only when you know WHAT are you fuzzing! Do not do it in real CAR's bus without understanding what you are doing!**
- Research and developing (prototyping and tests)

#CANGateway scan

Check/validate routing/filtering rules, in unit-test/silenium style…



Test frames:

- Frames that should be routed
- Frames that should be modified and routed
- Frames that should not be routed
- Random set of frames (that should not be routed)

Here we do validation

DEMO ON EMULATOR

#Fuzzing

WARNING, again: only if you know what are you doing.

- Fuzz known bytes/bits (if “2” – lock, “1” – unlock, then 3 - ?)
- SecurityAccess bruteforce (almost stupid idea, most ECU has anti-bruteforce, but DoS/Reset+Bruteforce maybe could work)
- DoS/Reset combinations…

Problem: No debugger attached, so how can we get ‘feedback’ from the ‘crashed’ ECU?

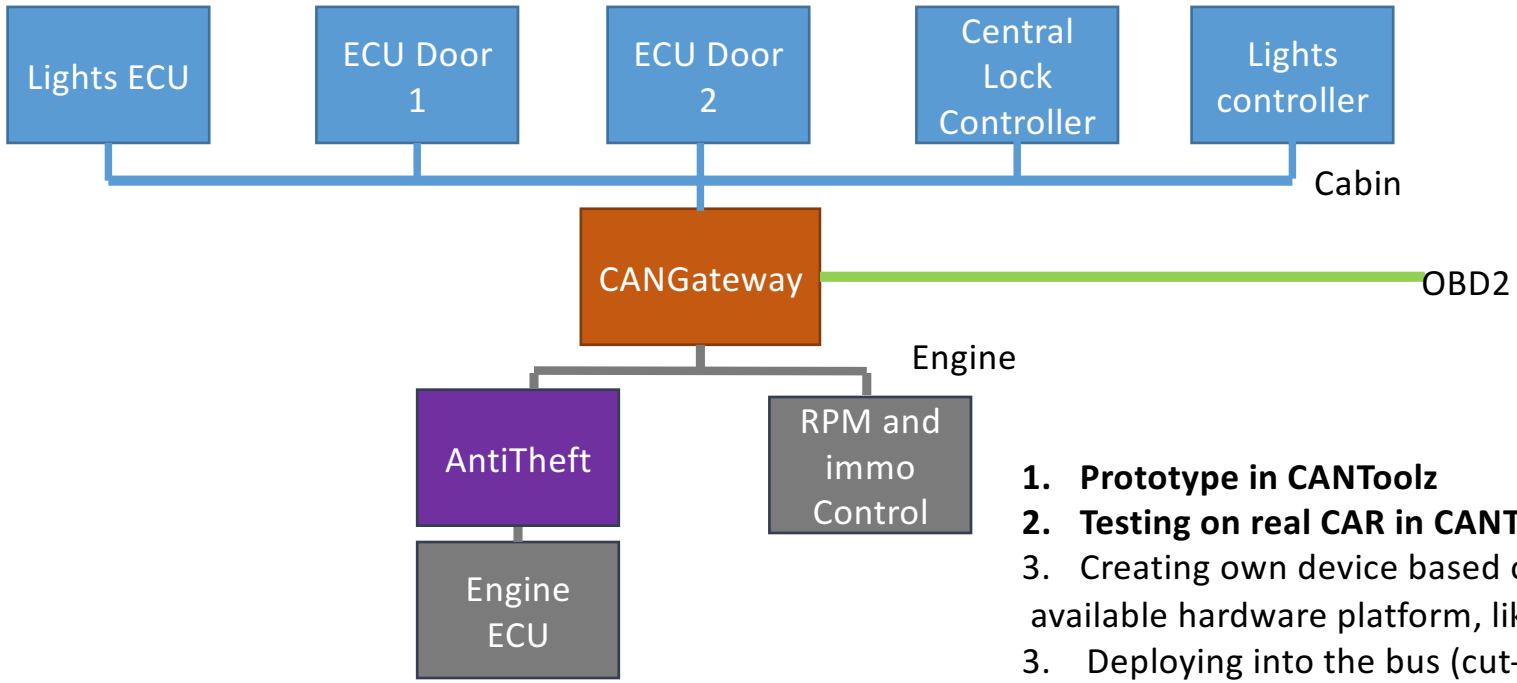
- Attach a debugger (captain!)
- *Check abnormalities in the CAN traffic after Fuzz sample has been sent (slower)*:
  - Signal lost
  - New pattern/mask of changed bits
  - Time delay between signals changed
- Check if normal typical CAN ‘request’ causes the same type of ‘response’ (if applicable)

(all these solutions are not implemented in CANToolz yet, but…actually we can do something… **demo**)

Found interesting paper with close ideas: https://www.escar.info/images/Datastore/2015_escar_EU_Papers/3_escar_2015_Stephanie_Bayer.pdf
Stephanie Bayer, Alexander Ptok ©

#AntiTheft prototype (DIY)

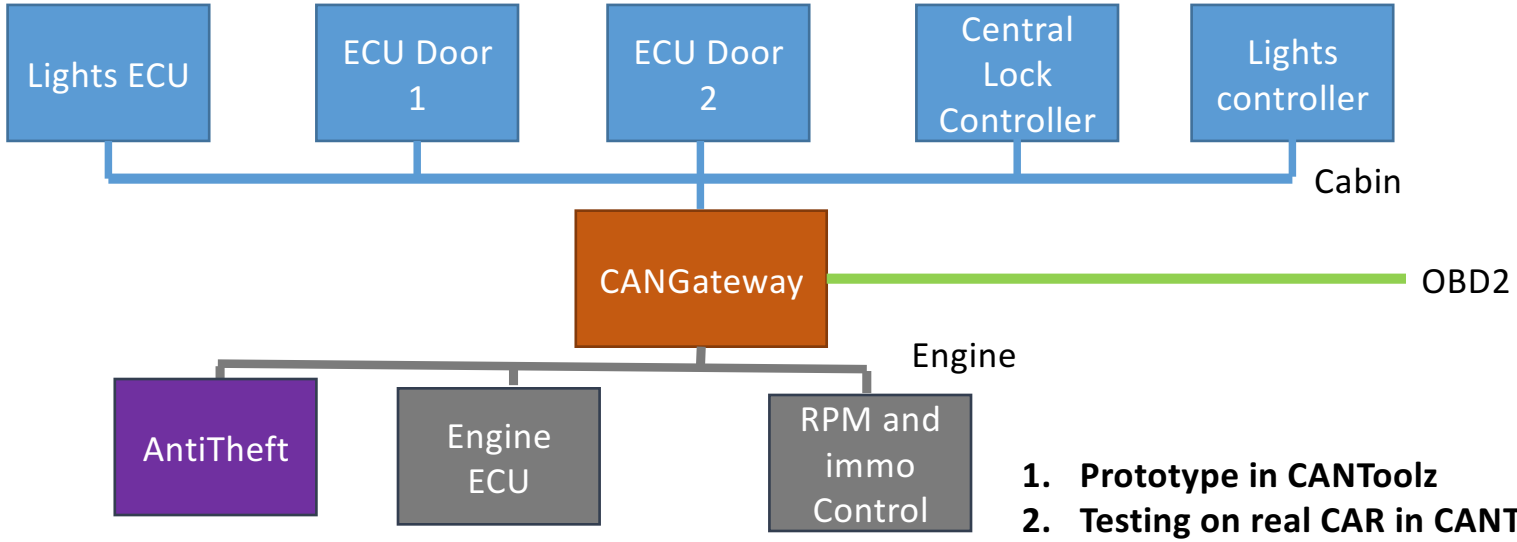
WARNING: only if you know what are you doing. If your 'device' crashes then you will lose your engine connection!



1. **Prototype in CANToolz**
2. **Testing on real CAR in CANToolz (MITM in the bus)**
3. Creating own device based on any available hardware platform, like Arduino.
3. Deploying into the bus (cut-in, MITM)

#AntiTheft prototype (DIY)

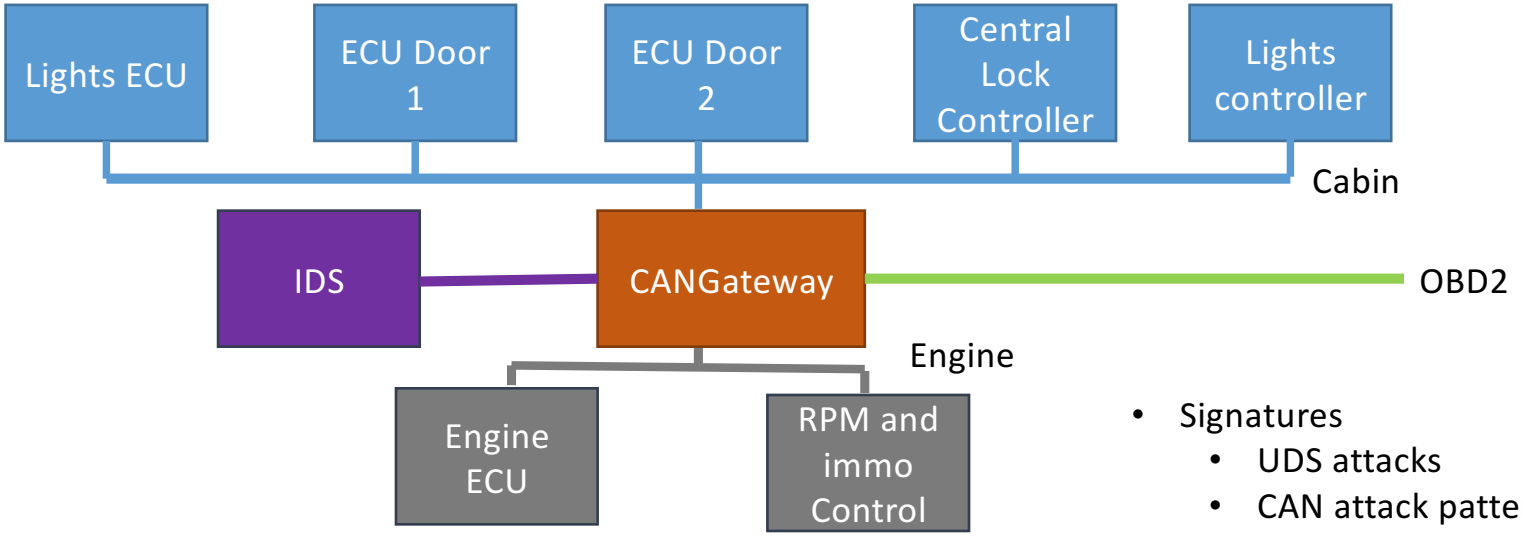
WARNING: only if you know what are you doing.



1. **Prototype in CANToolz**
2. **Testing on real CAR in CANToolz (no MITM)**
3. Crating own device based on any available hardware platform, like Arduino.
3. Deploying into the bus (parallel connection)

* Anyway good antitheft system is not ONLY CAN based… just as PoC

#IDS



- Signatures
  - UDS attacks
  - CAN attack patterns
- Abnormalities

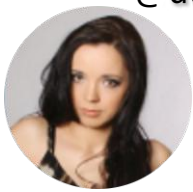
Interesting IDS PoC with abnormalities: https://conference.hitb.org/hitbsecconf2016ams/sessions/cansee-an-automobile-intrusion-detection-system/

#Questions…

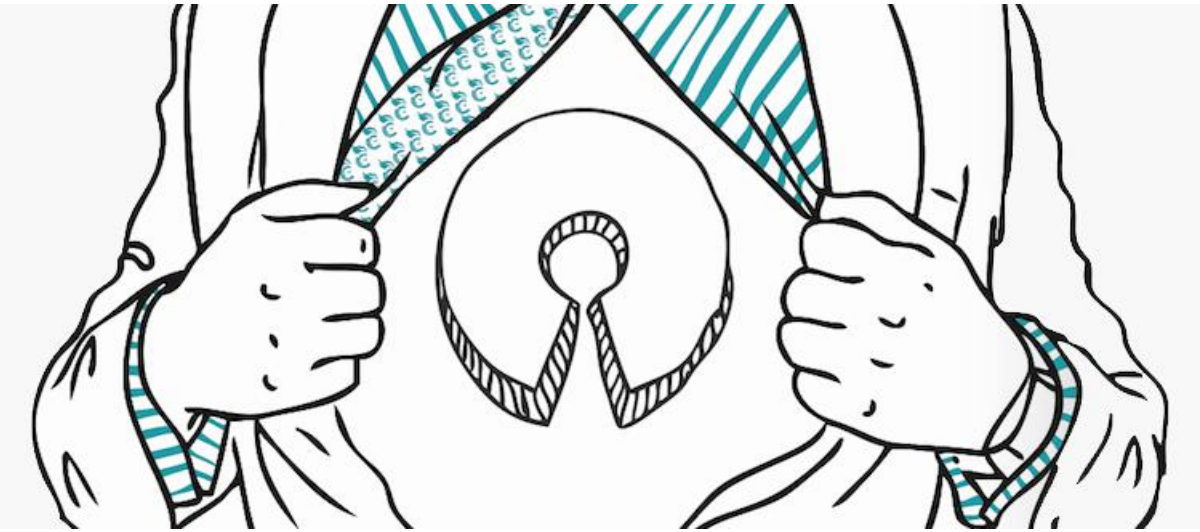
Contributors:

Sergey Kononenko
@kononencheg

Boris Ryutin
@dukebarman



Svetlana Sintcova



If you think this project could be helpful for you:

Contributors are WELCOME!
Testers are WELCOME!
Developers/users/researchers are welcome!