Research paper

The Connected Car

Ways to get unauthorized access and potential implications

Computest

Table of contents

1. Introduction

Our world is becoming more and more digital, and the devices we use daily are becoming connected more and more. Thinking of IoT products in domotics and healthcare, it's easy to find countless examples of how this interconnectedness improves our quality of life. However, these rapid changes also pose risks. In the big rush forward, we as a society aren't always too concerned with these risks until they manifest themselves. This is where the hacker community has taken an important role in the past decades: using curiosity and skills to demonstrate that the changes in the name of progress sometimes have a downside that we need to consider.

At Computest, our mission is to improve the quality of the software that surrounds us. While we normally do so through services and consulting to our customers, R&D projects play an important role as well. In 2017 we put significant R&D effort in vehicle security. We chose this topic for a number of reasons, besides it being an interesting topic from a technical point of view. For one because we saw more and more cars in our car park with internet connectivity, often without a convenient mechanism to receive or apply security updates. This ecosystem reminded us of other IoT systems, where we face similar problems concerning remote maintenance. We were interested to see how these effect the current state of security in the automotive vehicle industry. We also felt that this research topic would not only be of interest to us, but would also make the world a little bit safer in an industry that affects us all. Lastly this topic would of course allow us to demonstrate our expertise in the field. This paper describes our research approach, the research itself and its findings, the disclosure process with the manufacturer and finally our conclusions.

We are not the first to investigate the current state of security in automotive vehicles, the research of Charlie Miller and Chris Valasek[1] being the most prominent example. They found that the IVI (In-Vehicle Infotainment) system in their car suffered from a trivial vulnerability, which could be reached via the cellular connection because it was unfirewalled. We wanted to see if anything had changed since then, or if the same attack strategy might also succeed to other cars as well.

For this research, we looked at different cars from different models and brands, with the similarity that all cars had internet connectivity. In the end, we focused our research on one specific in-vehicle infotainment (IVI) system, that is used in most cars from the Volkswagen Auto Group and often referred to as MIB. More specifically, in our research we used a Volkswagen Golf GTE and an Audi A3 e-tron.

This research paper is structured as follows:

- Chapter 2 gives an introduction on the anatomy of a modern car that is needed to understand the rest of the paper, feel free to skip this chapter if you are already familiar with this topic.
- Chapter 3 and 4 give an outline of the research goal and approach.
- Chapter 5 describes our findings more in depth.
- We disclosed our findings to the Volkswagen Auto Group (VAG) - this process and timeline are outlined in chapter 6.
- Chapter 7 describes future work that can be done.
- We end this paper with a conclusion in chapter 8.

[1] http://illmatics.com/Remote%20Car%20Hacking.pdf

At Computest we believe in public disclosure of identified vulnerabilities as users should be aware of the risks related to use a specific product or service. But at all times we also consider it our responsibility that nobody is put at unnecessary risk and also no unnecessary damage is caused by such a disclosure.

The vulnerabilities we identified are all software-based, and therefore could be mitigated via a firmware upgrade. However, this cannot be done remotely, but must be done by an official dealer which makes upgrading the entire fleet at once difficult.

Based on above we decided to not provide a full disclosure of our findings in this paper. We describe the process we followed, our attack strategy and the system internals, but not the full details on the remote exploitable vulnerability as we would consider that being irresponsible. This might disappoint some readers, but we are fully committed to a responsible disclosure policy and are not willing to compromise on that.

This is also why we first informed the manufacturer about the vulnerability and disclosed all our findings to them, gave them the chance to review this research paper and also provide a related statement which we would incorporate into this document. We have received feedback on the research paper beginning of February 2018. Prior to release of this research paper, Volkswagen sent us the letter that is attached to this paper confirming the vulnerabilities. In this letter they also state that the vulnerabilities have been fixed in an update to the infotainment system, which means that new cars produced since the update are not affected by the vulnerabilities we found.

If you have any questions based on this research, feel free to contact the authors Daan Keuper and Thijs Alkemade via research@computest.nl.

2. Car anatomy

A modern-day vehicle is much more connected than meets the eye. In the old days, cars were mostly mechanical vehicles that relied on mechanics for functionality like steering and braking to operate. Modern vehicles mostly rely on electronics to control these systems. This is often referred to by drive by wire, and has several advantages over the traditional mechanical approach. Several safety features are possible because components are computer controlled. For example, some cars can and will brake automatically if the front radar detects an obstacle ahead and thinks collision is inevitable. Drive by wire is also used for some luxury functionalities such as automatic parking, by electronically taking over the steering wheel based on radar/camera footage.

All these new functionalities are possible because every component in a modern car is hooked up to a central bus, which is used by components to exchanges messages. The most common bus system is the CAN (Control Area Network) bus, which is present in all cars built since the nineties. Nowadays it controls everything, from steering to unlocking the doors to the volume knob on the radio.

The CAN protocol itself is relatively straight forward. In basis, each message has an arbitration ID and a payload. There is no authentication, authorization, signing, encryption etc. Once you are on the bus you can send arbitrary messages, which will be received by all parties connected to the same bus. There is also no sender or recipient information, each component can decide for itself if a specific message does apply to them.

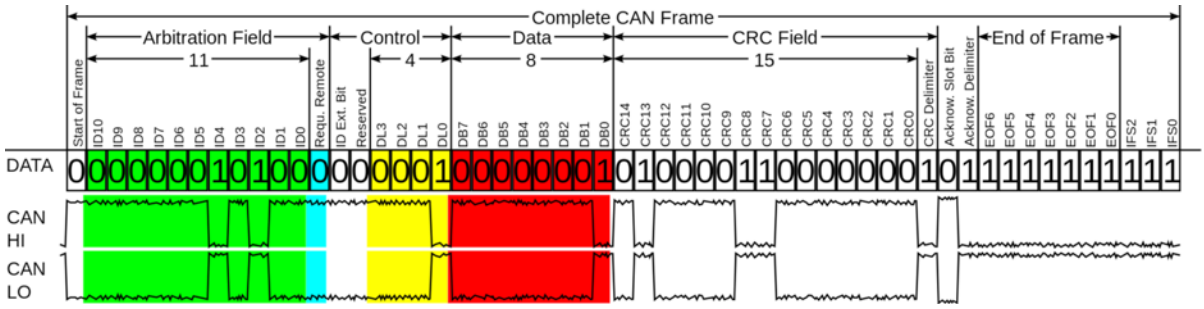


Figure: schematic overview of the CAN protocol, an 11-bit arbitration field with 64 bits of data.

In theory, this means that if an attacker would gain access to the CAN bus of a vehicle, he or she would control the car. They could impersonate the front radar for example to instruct the braking system to make an emergency stop due to a near collision or take over the steering. The attacker only needs to find a way to actually get access to a component that is connected to the CAN bus, without physical access.

The attacker has a lot of remote attack surface to choose from. Some of them require close proximity to the car, while others are reachable from anywhere around the globe. Some of the vectors will require user interaction, whereas others can be attacked unknowingly to its passengers.

For example, modern cars have a system for monitoring tire pressure, called TPMS (Tire Pressure Monitoring System), which will notify the driver if one of the tiers has a low pressure. This is a wireless system, where the tire will communicate its active pressure either via radio signals or Bluetooth to a receiver inside the car. This receiver will, in turn, notify other components via a message on the CAN bus. The instrument cluster will receive this message and as a response turn on the appropriate warning light. Another example is the key fob that will communicate wirelessly with a receiver in your car, which in its turn will communicate with the locks in the door and with the immobilizer in the engine. All these components have two things in common: they are connected to the CAN bus, and have remote attack surface (namely the receiver).

Modern cars have two main ways of protection against malicious CAN messages. The first is the defensive behavior of all components in a car. Each component is designed to always choose the safest option, in order to protect against components that might be malfunctioning. For example, automatic steering for automatic parking might be disabled by default, only to be enabled when the car is in reverse and at a low speed. And if another, malicious, device on the bus impersonates the front-radar to try to trigger an emergency stop, the real front-radar will continue to spam the bus with messages that the road is clear.

The second protection mechanism is that a modern car has more than one CAN bus, separating safety critical devices from convenience devices. The brakes and engine for example are connected to a high-speed CAN bus, while the air conditioning and windscreen wipers are connected to a separated (and most likely low-speed) CAN bus. In theory these buses should be completely separated, in practice however they are often connected via a so-called gateway. This is because there are circumstances were data must flow from the low-speed to the high-speed CAN bus and vice-versa. For example, the door locks must be able to communicate to the engine to enable and disable the immobilizer, and the IVI system receives status information and error codes from the engine to show on the central display. Firewalling these messages is the responsibility of the gateway, it will monitor every message from every bus and decides which messages are allowed to pass through.

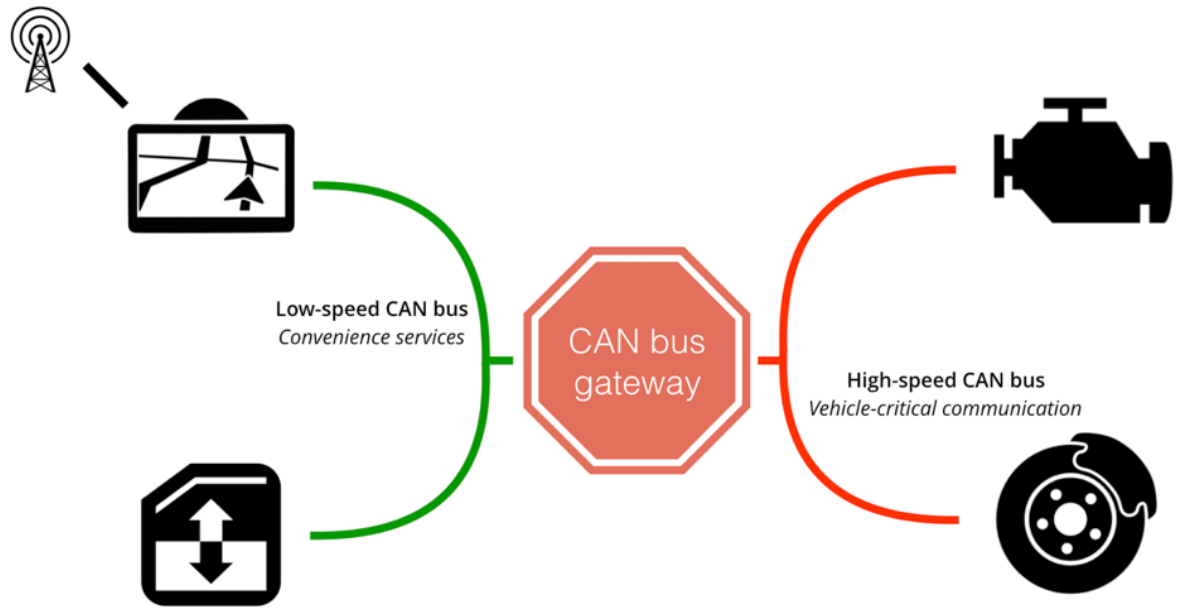


Figure: schematic overview of the typical setup; the IVI system has a cellular connection and is connected to the low-speed CAN bus, the high-speed CAN bus is protected by a gateway.

In the last few years we have seen an increase in cars that feature an internet connection, we even have seen cars that have two cellular connections at once. This connection can for example be used by the IVI system to obtain information, such as maps data, or to offer functionalities like an internet browser or a Wi-Fi hotspot or to give owners the ability to control some features via a mobile app. For example, to remotely start the central heating to preheat the car, or by being able to remotely lock/unlock your car. In all situations, the device that has the cellular connection is also hooked up to the CAN bus, which makes it theoretically possible to remotely compromise a vehicle.

This attack vector is not just theory, there have been researchers in the past that succeeded in this goal[2][3]. Some of these attacks were possible because the cellular connection was not firewalled and had a vulnerable service listening, others relied on the fact that the user would visit an attacker-controlled webpage on the in-car browser and exploited a vulnerability in the rendering engine.

[2] http://illmatics.com/Remote%20Car%20Hacking.pdf

[3] http://keenlab.tencent.com/en/2016/09/19/Keen-Security-Lab-of-Tencent-Car-Hacking-Research-Remote-Attack-to-Tesla-Cars/

3. Research goal

A modern car has many remote vectors, such as Bluetooth, TPMS and the key fob. But most vectors require that the attacker is in close proximity to the victim. However, for this research we specifically focused on attack vectors that could be triggered via the internet and without user interaction. Once we would have found such a vector, our goal was to see if we could use this vector to influence either driving behavior or other critical safety components in the car. In general, this would mean that we wanted to gain access to the high-speed CAN bus, which connects components like the brakes, steering wheel and the engine.

We chose the internet vector above others, because such an attack would further illustrate our point of the risks that are involved with the current eco-system. All other vectors require being physically close to the car, making the impact typically limited to only a handful of cars at a time.

We formulated the following research question:

> "Can we influence the driving behavior or critical security systems of a car via an internet attack vector?"

4. Research approach

We started this research with nine different models, from nine different brands. These were all lease cars belonging to employees of Computest. Since we are not the actual owner of the car we asked permission for conducting this research beforehand from both our lease company and the employee driving the car.

We conducted a preliminary research in which we mapped the possible attack vectors of each car. Determining the attack vectors was done by looking at the architecture, reading public documentation and by a short technical review.

Things we were specifically searching for:

- cars with only a single or few layers between the cellular connection and the high-speed CAN bus;
- cars which allowed us to easily swap SIM cards (since we are not the owner of the cars, soldering, decapping etc. is undesirable);
- cars that offered a lot of services over cellular or Wi-Fi.

From here we choose the car which we thought would give us the highest chance of success. This is of course subjective and does not guarantee success. For some models getting initial access might be easier than others, but this does say nothing about the effort required for lateral movement.

We finally settled for the Volkswagen Golf GTE as our primary target. We later added the Audi A3 e-tron to our research. Both vehicles share the same IVI-system which, on first sight, seemed to have a broad attack surface, increasing the chance of finding an exploitable vulnerability.

Figure: the MIB 2 in a Volkswagen Golf

5. Research findings

5.1. Initial access

We started our research initially with a Volkswagen Golf GTE, from 2015, with the Car-Net app. This car has a IVI system manufactured by Harman, referred to as the modular infotainment platform (MIB). Our model was equipped with the newer version (version 2) of this platform, which had several improvements from the previous version (such as Apple CarPlay). Important to note is that our model did not have a separate SIM card tray. We assumed that the cellular connection used an embedded SIM, inside the IVI system, but this assumption would later turn out to be invalid.

Figure: the MIB system itself, which is accessible via the glove department.

The MIB version installed in the Volkswagen Golf has the possibility to connect to a Wi-Fi network. A quick port scan on this port shows that there are many services listening:

```
$ nmap -sV -vvv -oA gte -Pn -p- 192.168.88.253
Starting Nmap 7.31 ( https://nmap.org ) at 2017-01-05 10:34 CET
Host is up, received user-set (0.0061s latency).
Not shown: 65522 closed ports
Reason: 65522 conn-refused
PORT      STATE    SERVICE         REASON      VERSION
23/tcp    open     telnet          syn-ack     Openwall GNU/*/Linux telnetd
10123/tcp open     unknown         syn-ack
15001/tcp open     unknown         syn-ack
21002/tcp open     unknown         syn-ack
21200/tcp open     unknown         syn-ack
22111/tcp open     tcpwrapped      syn-ack
22222/tcp open     easyengine?     syn-ack
23100/tcp open     unknown         syn-ack
23101/tcp open     unknown         syn-ack
25010/tcp open     unknown         syn-ack
30001/tcp open     pago-services1? syn-ack
32111/tcp open     unknown         syn-ack
49152/tcp open     unknown         syn-ack

Nmap done: 1 IP address (1 host up) scanned in 259.12 seconds
```

There is a fully functional telnet service listening, but without valid credentials, this seemed like a dead end. An initial scan did not return any valid credentials, and as it later turned out they use passwords of eight random characters. Some of the other ports seemed to be used for sending debug information to the client, like the current radio station and current GPS coordinates.

Port 49152 has a UPnP service listening and after some research it was clear that they use PlutinoSoft Platinum UPnP[4], which is open source. This service piqued our interest because this exact service was also found on the Audi A3 (also from 2015). This car however had only two open ports:

```
$ nmap -p- -sV -vvv -oA a3 -Pn 192.168.1.1
Starting Nmap 7.31 ( https://nmap.org ) at 2017-01-04 11:09 CET
Nmap scan report for 192.168.1.1
Host is up, received user-set (0.013s latency).
Not shown: 65533 filtered ports
Reason: 65533 no-responses
PORT      STATE  SERVICE REASON       VERSION
53/tcp    open   domain  syn-ack      dnsmasq 2.66
49152/tcp open   unknown syn-ack

Nmap done: 1 IP address (1 host up) scanned in 235.22 seconds
```

We spent some time reviewing the UPnP source code (but by no means was this a full audit) but didn't find an exploitable vulnerability.

We initially picked the Golf as primary target because it had more attack surface, but this at least showed that the two systems were built upon the same platform.

[4] https://www.plutinosoft.com/platinum/

After further research, we found a service on the Golf with an exploitable vulnerability. Initially we could use this vulnerability to read arbitrary files from disk, but quickly could expand our possibilities into full remote code execution. This attack only worked via the Wi-Fi hotspot, so the impact was limited. You have to be near the car and it must connect with the Wi-Fi network of the attacker. But we did have initial access:

```
$ ./exploit 192.168.88.253
[+] going to exploit 192.168.88.253
[+] system seems vulnerable...
[+] enjoy your shell:
uname -a
QNX mmx 6.5.0 2014/12/18-14:41:09EST nVidia_Tegra2(T30)_Boards armle
```

Because there is no mechanism to update this type of IVI remotely, we made the decision not to disclose the exact vulnerability we used to gain initial access. We think that giving full disclosure could put people at risk, while not adding much to this paper.

5.2. MMX

The system we had access to identified itself as MMX. It runs on the ARMv7a architecture and uses the QNX operating system, version 6.5.0. It is the main processor in the MIB system and is responsible for things like screen compositing, multimedia decoding, satnav etc.

We noticed that the MMX unit was responsible for the Wi-Fi hotspot functionality, but not for the cellular connection that was used for the Car-Net app. However, we did find an internal network. Finding out what was on the other end was the next step in our research.

```
# ifconfig mmx0
mmx0: flags=8843<UP,BROADCAST,RUNNING,SIMPLEX,MULTICAST> mtu 1500
 address: 00:05:04:03:02:01
 media: <unknown type> autoselect
 inet 10.0.0.15 netmask 0xffffff00 broadcast 10.0.0.255
 inet6 fe80::205:4ff:fe03:201%mmx0 prefixlen 64 scopeid 0x3
```

One of the problems we faced was the lack of tools on the system and the lack of a build chain to compile our own. For example, we couldn't get a socket or process list due this. The QNX operating system and build-chain is commercial software, for which we didn't have a license. At first, we tried to work with the tools that were already present. For example, we relied on a broadcast ping for host discovery, and used the included ftp client for a portscan (which took ages). While cumbersome, we found one other host alive on this network. Eventually we applied for a trial version of QNX. Not expecting much of this we continued our research. But, after a few weeks our application got through, and we received a demo license. Which meant we had access to standard tools like telnet and ps, as well as a build chain.

The device on the other end identified itself as RCC, and also had a telnet service running. We tried logging in using the same credentials, but this initially failed. After further investigating MMX's configuration it became apparent that MMX and RCC share their filesystems, using Qnet[5]; a QNX proprietary protocol. MMX and RCC are allowed to spawn processes on each other and read files (such as the shadow file). It even turned out that the shadow file on RCC was just a symlink to the shadow file on MMX. It seemed that the original telnet binary did not fully function, causing the password reject message. After some rewriting everything worked as expected.

[5] http://www.qnx.com/developers/docs/6.5.0/index.jsp?topic=%2Fcom.qnx.doc.neutrino_sys_arch%2Fqnet.html

```
# /tmp/telnet 10.0.0.16
Trying 10.0.0.16...
Connected to 10.0.0.16.
Escape character is '^]'.


QNX Neutrino (rcc) (ttyp0)

login: root
Password:

     ___              _ _    __   __ ___ _____
    /   |_    _  __| (_) |  \/  |_ _|  _  \
   / /| | |  | |/ _  | | | |\/| || || |_)_/
  / __  | |_| | (_| | | | |  | || || |_) \
 /_/  |_|__,__|\__,_|_|_|_|  |_|___|_____/

/ > ls -la
total 37812
lrwxrwxrwx  1 root      root             17 Jan 01 00:49 HBpersistence -> /mnt/efs-persist/
drwxrwxrwx  2 root      root             30 Jan 01 00:00 bin
lrwxrwxrwx  1 root      root             29 Jan 01 00:49 config -> /mnt/ifs-root/usr/apps/
config
drwxrwxrwx  2 root      root             10 Feb 16  2015 dev
dr-xr-xr-x  2 root      root              0 Jan 01 00:49 eso
drwxrwxrwx  2 root      root             10 Jan 01 00:00 etc
dr-xr-xr-x  2 root      root              0 Jan 01 00:49 hbsystem
lrwxrwxrwx  1 root      root             20 Jan 01 00:49 irc -> /mnt/efs-persist/irc
drwxrwxrwx  2 root      root             20 Jan 01 00:00 lib
drwxrwxrwx  2 root      root             10 Feb 16  2015 mnt
dr-xr-xr-x  1 root      root              0 Jan 01 00:37 net
drwxrwxrwx  2 root      root             10 Jan 01 00:00 opt
dr-xr-xr-x  2 root      root       19353600 Jan 01 00:49 proc
drwxrwxrwx  2 root      root             10 Jan 01 00:00 sbin
dr-xr-xr-x  2 root      root              0 Jan 01 00:49 scripts
dr-xr-xr-x  2 root      root              0 Jan 01 00:49 srv
lrwxrwxrwx  1 root      root             10 Feb 16  2015 tmp -> /dev/shmem
drwxr-xr-x  2 root      root             10 Jan 01 00:00 usr
dr-xr-xr-x  2 root      root              0 Jan 01 00:49 var
/ >
```

5.3. RCC

The RCC unit is a separate chip on the MIB system. The MIB IVI is a modular platform, were they separated all the multimedia handling from the low-level functions. The MMX (multimedia applications unit) processor is responsible for things like the satnav, screen and input control, multimedia handling etc. While the RCC (radio and car control unit) processor handles the low-level communication.

RCC runs on the same version of QNX. It has even fewer tools available, and only a few hundred kilobytes of ram. But because of the Qnet protocol it is possible to run all tools from the MMX unit on RCC.

Communication with the lower level components, like DAB+, CAN, AM/FM decoding etc. are handled via serial connections; either SPI or I2C. The various configuration options can be found under /etc/.

5.4. Car-Net

We expected to find a cellular connection on RCC, but we did not. After further research it turned out that the Car-Net functionality is offered by a completely separate unit, and not the IVI. The cellular connection in the Golf is connected to a box which is located behind the instrument cluster, as is shown below.

Figure: the Car-Net box, behind the instrument cluster

The Car-Net box uses an embedded SIM card. Since this box offered no other interface options, and we couldn't make any physical changes to the car (to see if JTAG was available for example), we did not investigate any further.

5.5. Audi A3

From here we decided to put our effort back into the Audi A3. It uses the same IVI system, but used a higher-end version. This version has a physical SIM card, which is used by the Audi connect service . We of course already did a port scan via the Wi-Fi hotspot, which turned out empty, but it might be that the results would be different via the cellular connection.

To test this, we needed to be able to do a port scan on the remote interface. This can either be done if the ISP assigns a public routable IPv4 address (unfirewalled), allows client-to-client communication or by using a hacked femtocell. We chose the first option by using a functionality offered by one of the largest ISPs in the Netherlands. They will assign a public IPv4 address if you change certain APN settings. A portscan on this public IP address gave completely different results than our earlier portscan on the Wi-Fi interface:

```
$ nmap -p0- -oA md -Pn -vvv -A 89.200.70.122
Starting Nmap 7.31 ( https://nmap.org ) at 2017-04-03 09:14:54 CET
Host is up, received user-set (0.033s latency).
Not shown: 65517 closed ports
Reason: 65517 conn-refused
PORT      STATE    SERVICE    REASON      VERSION
23/tcp    open     telnet     syn-ack     Openwall GNU/*/Linux telnetd
10023/tcp open     unknown    syn-ack
10123/tcp open     unknown    syn-ack
15298/tcp filtered unknown    no-response
21002/tcp open     unknown    syn-ack
22110/tcp open     unknown    syn-ack
22111/tcp open     tcpwrapped syn-ack
23000/tcp open     tcpwrapped syn-ack
23059/tcp open     unknown    syn-ack
32111/tcp open     tcpwrapped syn-ack
35334/tcp filtered unknown    no-response
38222/tcp filtered unknown    no-response
49152/tcp open     unknown    syn-ack
49329/tcp filtered unknown    no-response
62694/tcp filtered unknown    no-response
65389/tcp open     tcpwrapped syn-ack
65470/tcp open     tcpwrapped syn-ack
65518/tcp open     unknown    syn-ack

Nmap done: 1 IP address (1 host up) scanned in 464 seconds
```

Most services are the same as those on the Golf. Some things may differ (like port numbers), possibly because the Audi has the older model of the MIB IVI system. But, more importantly: our vulnerable service is also reachable, and suffers from the same vulnerability!

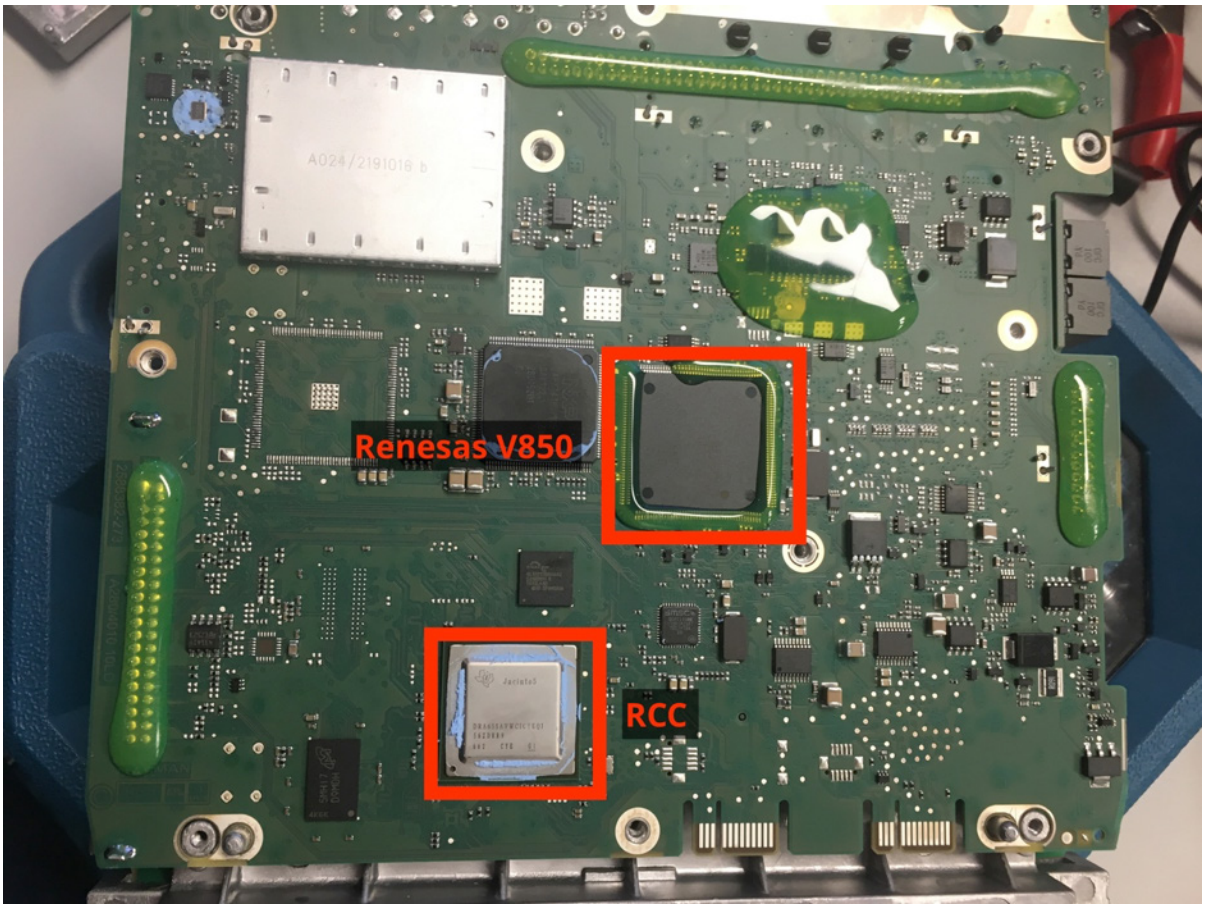
An attacker can only abuse this vulnerability if the owner has the Audi connect service, and the ISP in the country of the owner allows client-to-client communication, or hands out public IPv4 addresses.

To summarize our research up to this point: we have remote code execution, via the internet, on MMX. From here we can control RCC as well. The next step would be to send arbitrary CAN messages over the bus to see if we can reach any safety critical components.

[6] http://www.audi.com/en/innovation/connect.html

5.6. Renesas V850

The RCC unit is not directly connected to the CAN bus, it has a serial connection (SPI) to a separate chip that handles all CAN communication. This chip is manufactured by Renesas and uses the V850 architecture.



The firmware on this chip doesn't allow for arbitrary CAN messages to be sent. It has an API that allows a select number of messages to be sent. Most likely, any vulnerabilities in the gateway would require us to send a message that is not on the list, meaning we need a way to let the Renesas chip send us arbitrary messages. The read functionality on the Renesas chip has been disabled, meaning that it is not possible to extract the firmware from the chip easily.

The MIB system does have a software update feature. For this an SD-card, USB stick or CD must be inserted which holds the new firmware. The update sequence is initiated by the MMX unit, which is responsible for the mounting and handling all removable media. When a new firmware image is found, the update sequence will commence.

The update is signed using RSA, but not encrypted. Signature validation is done by the MMX unit, which will then hand over the appropriate update files for RCC and the Renesas chip. The RCC and Renesas chip will trust that the MMX unit already has performed signature validation, and will thus not revalidate the signature for their new firmware. Updating the Renesas V850 chip can be initiated by the RCC unit (using mib2_ioc_flash).

Firmware images are hard to come by. They are only available for official dealers and not for end-users. However, if one can get a hold of the firmware image, it is theoretically possible to backdoor the original firmware image for the Renesas chip, to allow sending arbitrary CAN messages, and flash this new firmware from the RCC unit.

The figure below shows the attack chain up until this point:

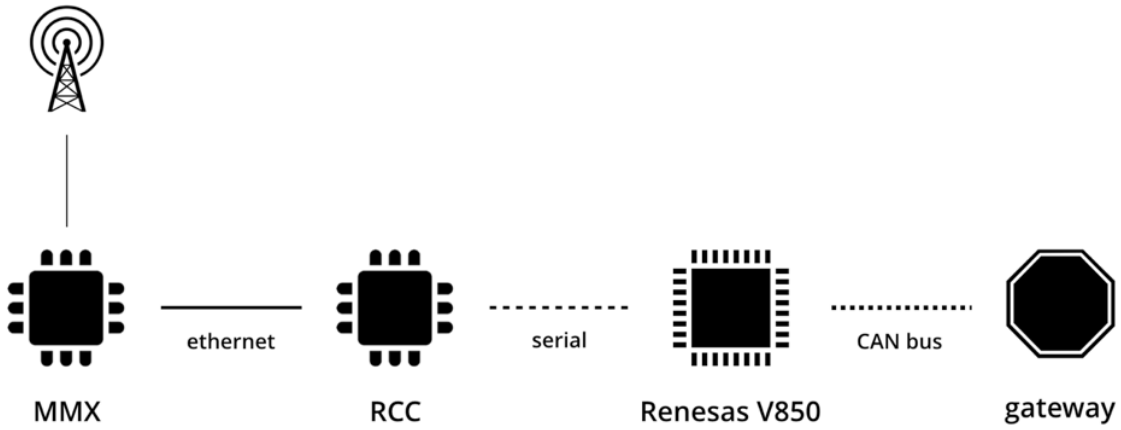


Figure: the current attack chain.

5.7. Gateway

By backdooring the Renesas chip we are able to send arbitrary CAN messages on the CAN bus. However, the CAN bus we are connected to is dedicated to the IVI system. It is directly connected to a CAN gateway; a physical device used to firewall/filter CAN messages between the different CAN buses.

The gateway is located behind the steering column and is connected with a single connector which has all the different buses connected.

Figure: the CAN bus gateway from the Golf

The firmware for the gateway is signed, so backdooring this chip won't work as it will invalidate the signature. Furthermore, reflashing the firmware is only possible from the debug bus (ODB-II port) and not from the IVI CAN bus. If we want to bypass this chip we need to find an exploitable vulnerability in the firmware. Our first step to achieve this would be to try to extract the firmware from the chip using a physical vector. However, after careful consideration we decided to discontinue our research at this point, since this would potentially compromise intellectual property of the manufacturer and potentially break the law.

5.8. USB vector

After finding the remote vector, we discovered a second vector we had not yet explored. For debugging purposes, the MMX unit recognizes a few USB-to-Ethernet dongles as debug interfaces, which will create an extra networking interface. It seems that this network interface will also serve the vulnerable service.

/etc/usblauncher.lua:

```
-- D-Link DUB-E100 USB Dongles
device(0x2001, 0x3c05) {
    driver"/etc/scripts/extnet.sh -oname=en,lan=0,busnum=$(busno),devnum=$(devno),-
phy_88772=0,phy_check,wait=60,speed=100,duplex=1,ign_remove,path=$(USB_PATH) /lib/dll/
devnp-asix.so /dev/io-net/en0";
    removal"ifconfig en0 destroy";
};

device(0x2001, 0x1a02) {
    driver"/etc/scripts/extnet.sh -oname=en,lan=0,busnum=$(busno),devnum=$(devno),-
phy_88772=0,phy_check,wait=60,speed=100,duplex=1,ign_remove,path=$(USB_PATH) /lib/dll/
devnp-asix.so /dev/io-net/en0";
    removal"ifconfig en0 destroy";
};

-- SMSC9500
device(0x0424, 0x9500) {
    -- the extnet.sh script does an exec dhcp.client at the bottom, then usblauncher can
slay the dhcp.client when the dongle is removed
    driver"/etc/scripts/extnet.sh -olan=0,busnum=$(busno),devnum=$(devno),path=$(USB_PATH)
/lib/dll/devn-smsc9500.so /dev/io-net/en0";
    removal"ifconfig en0 destroy";
};

-- Germaneers LAN9514
device(0x2721, 0xec00) {
        -- the extnet.sh script does an exec dhcp.client at the bottom, then usblauncher
can slay the dhcp.client when the dongle is removed
        driver"/etc/scripts/extnet.sh -olan=0,busnum=$(busno),devnum=$(devno),path=$(USB_
PATH) /lib/dll/devn-smsc9500.so /dev/io-net/en0";
        removal"ifconfig en0 destroy";
};
```

But even without this service, telnet is also enabled. The version of QNX that is being used only supports descrypt() for password hashing, which has an upper limit of eight characters. One could use a service like crack.sh[7] which can search the entire key space in less than three days using FPGA's, for only $ 100,-. We found out that the passwords are changed between models/versions; but we think it is doable, both in time and money, to build a dictionary containing all passwords of all different versions of the MIB IVI.

This vector seems to work on all models that use the MIB IVI system, regardless of the version. Since VAG has multiple car brands, components like the IVI are often reused between brands. This vector will therefore most likely also work on cars from, for example, Seat and Skoda.

We tested this vector by changing some kernel parameters on a Nexus 5 phone. This can be done without the need for reflashing, only root privileges are required. After plugging in the phone, it will be recognized as an Ethernet dongle, and the MMX unit will initialize a debug interface.

[7] https://crack.sh

6. Disclosure process

At Computest we believe in public disclosure of identified vulnerabilities as users should be aware of the risks related to use a specific product or service. But at all times we also consider it our responsibility that nobody is put at unnecessary risk and also no unnecessary damage is caused by such a disclosure. That means we are fully committed to a responsible disclosure policy[8] and are not willing to compromise on that.

As recommended we decided to contact the manufacturer as soon as we had verified and documented our findings. To do so we were looking for a specific Responsible Disclosure Policy (RDP) on the website of the manufacturer to understand how such a disclosure should be handled from their point of view.

As Volkswagen apparently did not have such a RDP in place, we followed the public Whistleblower System of Volkswagen and contacted the mentioned external lawyer they listed. Opposite to a typical whistleblower disclosure we had no interest nor reason to stay anonymous and disclosed our identity from the very beginning.

With the help of the external lawyer we got in contact with the quality assurance department of the Volkswagen Group mid of July 2017. After some initial conference calls we decided together that a face-to-face meeting would be the best format to disclose our findings and Volkswagen invited us to visit their IT center in Wolfsburg which we followed end of August 2017.

Obviously, Volkswagen required some time to investigate the impact and to perform a risks assessment. At the end of October we received their final conclusion, that they were not going to publish a public statement themselves. But were willing to review our research paper to check whether we have stated the facts correctly and we have received the review at the beginning of February 2018. In April 2018, just prior to release of this paper, Volkswagen provided us with a letter that confirms the vulnerabilities, and mentions that they have been fixed in a software update to the infotainment system. This means that cars produced since this update are not affected by the vulnerabilities we found. The letter is attached to this report. But based on our experience, it seems that cars which have been produced before are not automatically updated when being serviced at a dealer, thus are still vulnerable to the described attack.

When writing this paper, we decided to not provide a full disclosure of our findings. We describe the process we followed, our attack strategy and the system internals, but not the full details on the remote exploitable vulnerability as we would consider that being irresponsible. This might disappoint some readers, but we are fully committed to a responsible disclosure policy and are not willing to compromise on that.

In addition to the above we would like to mention that we have consulted an experienced legal advisor early on in this project to make sure our approach and actions are reasonable, and to assess potential (legal) consequences.

[8] https://www.ncsc.nl/binaries/content/documents/ncsc-en/current-topics/news/responsible-disclosure-guideline/1/Responsible%2BDisclosure%2BENG.pdf

7. Future work

The current chain of attack only allows for the sending and receiving of CAN messages on an isolated CAN bus. As this bus is strictly separated from the high-speed CAN bus via a gateway, the current attack vector poses no direct threat to driver safety.

However, if an exploitable vulnerability in the gateway were to be found, the impact would significantly increase. Future research could focus on the security of the gateway, to see if there is any way to either bypass or compromise this device. There are still some attack vectors on the gateway that are definitely worth exploring. However, this should only be explored in cooperation with the manufacturer.

We are also looking into extending our research to other cars. We still have some interesting leads from our preliminary research that we could follow.

8. Conclusions

Internet-connected cars are rapidly becoming the norm. As with many other developments, it's a good idea to sometimes take a step back and evaluate the risks of the path we've taken, and whether course adjustments are needed. That's why we decided to pay attention to the risks related to internet-connected cars. We set out to find a remotely exploitable vulnerability, which required no user interaction, in a modern-day vehicle and from there influence either driving behavior or a safety feature.

With our research, we have shown that at least the first is possible. We can remotely compromise the MIB IVI system and from there send arbitrary CAN messages on the IVI CAN bus. As a result, we can control the central screen, speakers and microphone. This is a level of access that no attacker should be able to achieve. However, it does not directly affect driving behavior or any safety system due to the CAN gateway. The gateway is specifically designed to firewall CAN messages and the bus the IVI is connected to is separated from all other components. Further research on the security of the gateway was consciously not pursued.

We argue that the threat of an adversary with malicious intent was long underestimated. The vulnerability we initially identified should have been found during a proper security test. During our meeting with Volkswagen, we had the impression that the reported vulnerability and especially our approach was still unknown. We understood in our meeting with Volkwagen that, despite it being used in tens of millions of vehicles world-wide, this specific IVI system did not undergo a formal security test and the vulnerability was still unknown to them. However, in their feedback for this paper Volkswagen stated that they already knew about this vulnerability.

Speaking with people within the industry we are under the impression that attention on security and awareness is growing, but with the efforts mainly focusing on the models still in development. Component manufactures producing critical components such as brakes, already had security high up in their quality assurance agenda. This focus was not because of the fear of adversaries on the CAN bus, but mainly to protect against component malfunction, which could otherwise result in situations like unintended acceleration[9].

A remote adversary is new territory for most industrial component manufacturers, which, to be mitigated effectively, requires embedding security in the software development lifecycle. This is a movement that was started years ago in the AppSec world. This is easier in an environment with automatic testing, continuous deployment and possibility to quickly apply updates after release. This is not always possible in the hardware industry, due to local regulations and the ecosystem. It often requires coordination between many vendors. But, if we want to protect future cars, these are problems we have to solve. However, what about the cars of today, or cars that were shipped last week? They often don't have the required capabilities (such as over-the-air updates) but will be on our roads for the next fifteen years. We believe they currently pose the real threat to their owners, having drive by wire technology in cars that are internet-connected without any way to reliably update the entire fleet at once.

We believe that the car industry in general, since it isn't traditionally a software engineering industry, needs to look to other industries and borrow security principles and practices. Looking at mobile phones for instance, the car industry can take valuable lessons regarding trusted execution, isolation and creating an ecosystem for rapid security patching. For instance, components in a car that are remotely accessible, should be able to receive and apply verified security updates without user interaction. We understand that component malfunction is a higher threat in day-to-day operation. We also understand that having an internet-connected car has its advantages, and we also not trying to reverse this trend. However, we can't ignore the threats accompanied with today's internet-connected world.

[9] https://www.nhtsa.gov/staticfiles/nvs/pdf/NASA-UA_report.pdf

9. Recommendations

Based on our findings documented in this research paper and our overall experience in IoT security we would like to conclude this paper with some recommendations to manufacturers, consumers and ethical hackers.

9.1. Recommendations for manufacturers

- The growing number of connected consumer devices is not only providing tremendous opportunities, but also comes along with additional risks which need to be taken care of. The quality of produced goods is not only about mechanical functionality and quality of materials used, but the quality and security of the embedded software is equally important and therefore requires equal attention in terms of quality assurance.

- It is common practice, especially in the field of electronics, to purchase components from a third party. That does not clear the manufacturer from the responsibility for their quality and security; these components need to be included in thorough quality assurance. The company selling the completed product should be prepared to take responsibility for its security and quality.

- Even the best quality control cannot prevent mistakes from being made. In such an event, manufacturers should stand to their responsibility and communicate swiftly and with transparency to affected customers. Hiding cannot only lead to damages on the customer side, but can also have a very negative impact on the manufacturers reputation.

- Ethical hackers should not be considered as a threat, but as a help to identify existing vulnerabilities. These people often have different views and approaches, enabling them to find vulnerabilities which otherwise would remain undiscovered. Such identified vulnerabilities are important to improve the product quality.

- Every manufacturer should have a Responsible Disclosure Policy (RDP) stating clearly how external people can report discovered vulnerability in a safe environment. Ethical hackers should not be threatened but encouraged to disclose findings to the manufacturer. See also 'NCSC Leidraad Responsible Disclosure'[10].

9.2. Recommendations for consumers

- Having an internet-connected car brings a number of advantages mostly for consumers. But be aware that this applied technology is still early in its lifecycle and therefore not fully mature yet in terms of quality and security.

- This can be associated with the possibility to relatively easy get remote access to your car. Although it is very unlikely that this can impact driving behaviour, it might provide access to personal data stored in the car entertainment system and/or your smart phone.

- Become informed: ask about quality and security standards of car you are looking into as much as you do that for aspects like crash tests. Specifically ask about the remote maintenance possibilities and how long the manufacturer would maintain the software used in the car (support period). If you want to protect yourself against remote threats, please ask your dealer to install updates during their normal service schedule.

- Keep the software in your car up to date where you have the possibility. This does not only apply to cars, but to all IoT devices such as baby monitors, smart TV's and home automation.

[10] https://www.ncsc.nl/actueel/Responsible+Disclosure+Leidraad

9.3. Recommendations for ethical hackers

- Identifying and disclosing a vulnerability is not about a personal victory or trophy for the hacker, but a responsibility to contribute to safer and better IT systems.

- In case you have identified a vulnerability, don't go further than necessary and make sure you don't harm anybody.

- Inform the owner / manufacturer of the identified vulnerability first immediately and do not share related information with the press or any other third party. Look for a responsible disclosure policy (RDP) on the website of the manufacturer and follow the policy. In case you can't find such a RDP, contact the manufacturer (anonymously) and ask for such a policy to help protect your integrity. A good alternative way is to look for a whistle-blower policy and contact the manufacturer this way.

- Beware that what may look like a simple fix from your perspective as an engineer, can be something completely different in a manufacturing world when applied to the scale of hundreds of thousands of vehicles. Have some patience and empathy for the situation you're putting the manufacturer in, even though you may be right to do so.

- It is important to understand the legal regulations relevant for potential research and investigation activities. Different national legislations and limited relevant jurisdiction does not make that easy. Keep in mind: having no criminal intention does not give a free ride to break the law. In case of doubt seek legal advice upfront!

10. Letter form Volkswagen

Below the letter from Volkswagen we received on April 17 2018. The letter is from the department we have been in contact with from the start of the disclosure process.

VOLKSWAGEN
AKTIENGESELLSCHAFT

VOLKSWAGEN AKTIENGESELLSCHAFT 38436 WOLFSBURG DEUTSCHLAND
BRIEFFACH 8125

Computest
Postbus 653
2700 AR Zoetermeer
Nederland

IHRE ZEICHEN
IHRE NACHRICHT
UNSERE ZEICHEN
DURCHWAHL - 39554
TELEFAX
E-MAIL Arnd.schaarschmidt@volkswagen.de
DATUM 12. April 2018

IT security vulnerability in cars from Volkswagen Group

Dear Sirs,

Volkswagen is currently in a period of transformation – from just being a car maker to becoming a mobility provider.
Digitalisation of vehicles thus also means we are faced with continuous change. We question existing methods and processes in order to bring software development into the focus of development.

Naturally, data security is a key aspect of this. Security precautions against unwanted data manipulation are one of our highest priorities.
This has always been something of a technological "arms race". However, the dynamic pace of development means that digitalisation has increased even faster.

Informants such as Computest can help us make our products even more secure. We very much appreciate this and have great respect for your technical expertise.
You gave us enough reaction time to analyse and eliminate the problem.
Thank you very much for your professional cooperation.

Your "Research Paper on Car Hacking" describes the attack in detail. The objective of manipulating the steering and brake was not achieved. However, you did succeed in accessing the infotainment system and obtaining "Root" authorisations. These administrator rights of the modular infotainment matrix (MIB) are intended for use during development at Volkswagen and not for other people in a customer vehicle. The open interface on the Golf GTE and Audi A3 was closed by an update to the infotainment software from production week 22/2016 onwards.

Thank you very much, with best regards from Wolfsburg,

i. V.

Arnd Schaarschmidt
Head of Smart Quality

Distributor
Mr Ruijs
Mr Keuper
Mr Riedl

VOLKSWAGEN AKTIENGESELLSCHAFT
38436 WOLFSBURG
DEUTSCHLAND
TELEFON +49 5361 9-0
TELEFAX +49 5361 9-28282
VW@VOLKSWAGEN.DE

EHRENVORSITZENDER DES AUFSICHTSRATS:
KLAUS LIESEN

VORSITZENDER DES AUFSICHTSRATS:
HANS DIETER PÖTSCH

VORSTAND:
MATTHIAS MÜLLER – VORSITZENDER
KARLHEINZ BLESSING
HERBERT DIESS
FRANCISCO J. GARCIA SANZ
JOCHEM HEIZMANN
ANDREAS RENSCHLER
RUPERT STADLER
HILTRUD D. WERNER
FRANK WITTER

VOLKSWAGEN AKTIENGESELLSCHAFT
SITZ: WOLFSBURG
AMTSGERICHT BRAUNSCHWEIG
HRB 100484

Performance

Security

Test automation

Wij hebben slechts één doel voor ogen:
(online) applicaties en websites laten werken. Snel, betrouwbaar en veilig! Daarom bieden wij diensten op het gebied van performancetesting, securitytesting en testautomatisering.

Door de intensieve samenwerking tussen deze disciplines profiteren onze klanten van zowel brede als diepe kennis en ervaring. Wij werken in kleine, agile teams. Zo kunnen we altijd snel inspelen op jouw behoeften.

Signaalrood 25
2718 SH Zoetermeer
088 - 733 13 37

info@computest.nl
www.computest.nl

Computest