Remote Exploitation of An Unaltered Passenger Vehicle

Charlie Miller and Chris Valasek

Presented by
Samantha McDonald

Background

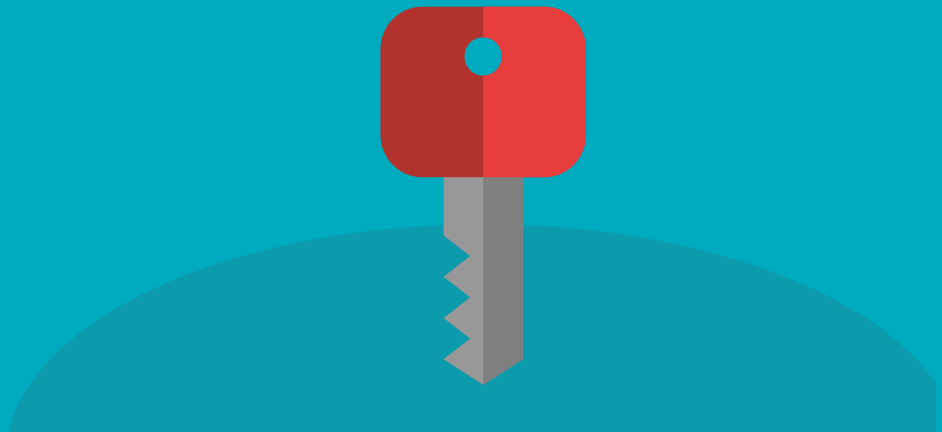
- ★ Charlie Miller – Twitter Security Engineer
- ★ Chris Valasek – Vehicle Security Research Director at IO Active
- ★ Findings presented in the 2015 Black Hat Conference

So, How do you hack a car?

1. **Find a car to attack**
2. **Analyze remote compromise surfaces, focusing on lateralization**
3. **Compromise the messaging system**
4. **Send messages to control the car!**

01

The Target: 2014 Jeep Cherokee



Why a Jeep?

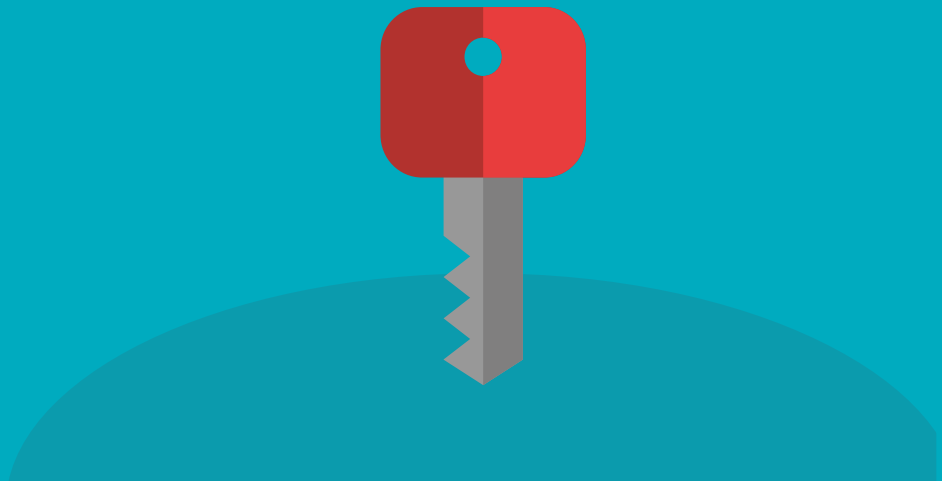
Large Attack Surface

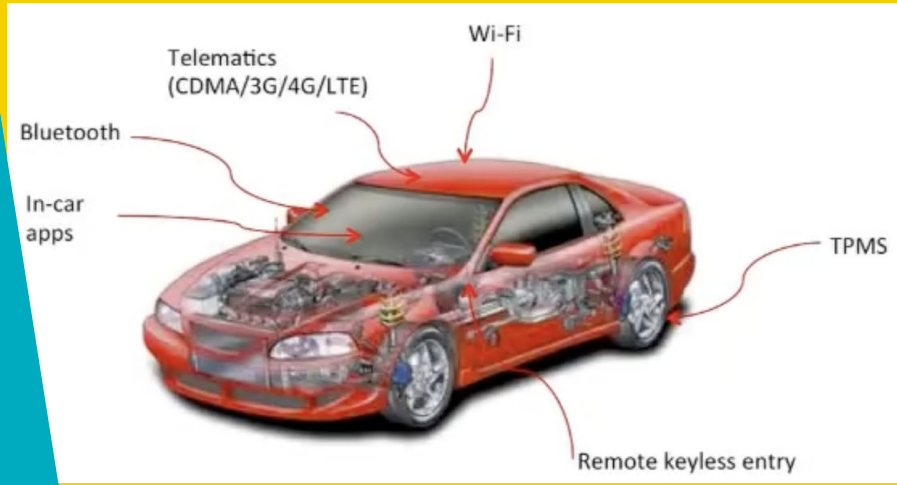
Simple Architecture

Advanced Physical Features

02

Attack Surfaces





Ways to “Get In”

WIFI

Enabled by the owner if desired (Monthly Cost)

BLUETOOTH

Large stack, representing many attack points

CELLULAR

Connection via Sprint’s network (Telematics)

TPMS

Tire pressure monitoring system – Past research conducted

RKE

Remote keyless entry – A radio transmitter that communicates w/ ECU

IN-CAR APPS

Additional vulnerabilities, varies by application

UConnect System

- ★ **UConnect 8.4 AN / RA4 Radio by Harman Kardon**
  - **Wifi , Navigation , Apps , and Cellular Connections**
- ★ **Used in Dodge , Jeep , Ram , and Chrysler**
  - **Vulnerabilities found are translatable**
- ★ **Connects to the CAN - IHS data bus**
  - **Controlled Area Network - Interior High Speed**
  - **Can communicate with other systems!**

Connect to UConnect? - WiFi

- ★ Not on by default
- ★ Password randomly generated
  - ○ Many possibilities
  - ○ Needed for access to network

```
char *get_password(){
        int c_max = 12;
        int c_min = 8;

        unsigned int t = time(NULL);
        srand (t);
        unsigned int len = (rand() % (c_max - c_min + 1)) + c_min;
        char *password = malloc(len);
        int v9 = 0;
        do{
               unsigned int v10 = rand();
                int v11 = convert_byte_to_ascii_letter(v10 % 62);
                password[v9] = v11;
                v9++;
        } while (len > v9);
return password;
```

- ★ Password generator → Found from disassembling the “WifiSvc” binary from the OMAP chip, as a function of time (when the vehicle first started)
  - ○ 180 million possibilities for a given year
- ★ If time isn’t known (not connected to cellular on start), it’s set to 1/1/2013 00:00:00
  - ○ Test vehicle → Password is based on 1/1/2013 00:00:32
  - ○ Likely only a handful of realistic passwords to test!

*Jailbreaking UConnect was used to understand this process, but isn't necessary for the exploits

Connect to UConnect?

- ★ Option 2 → Cellular
- ★ Doesn't have to know the Wifi password, or be within 32 meters
- ★ UConnect uses a Sierra Wireless AirPrime AR5550 (Qualcomm 3G // Sprint)
- ★ Via Jailbreaking, IPs were 68.28.89.85 (public) and 21.28.103.144 (Sprint internal)
  - IPs vary every restart. 21.x.x.x and 25.x.x.x are Sprint's internal Class A IPs for this
- ★ Port 6667 – IRC (Internet Relay Chat) is open internally!
  - Anytime 6667 is open on Sprint network, it is a direct connection to the D-BUS
  - This port was also discovered and open via WiFi connection

*All open ports were exposed via a jailbroken console shell, running netstat

Open Ports (Via WiFi + Sprint)

2021

MonitorService, delivers debug/trace information from runtime system into file or over TCP/IP

3128

Free low-footprint proxy service

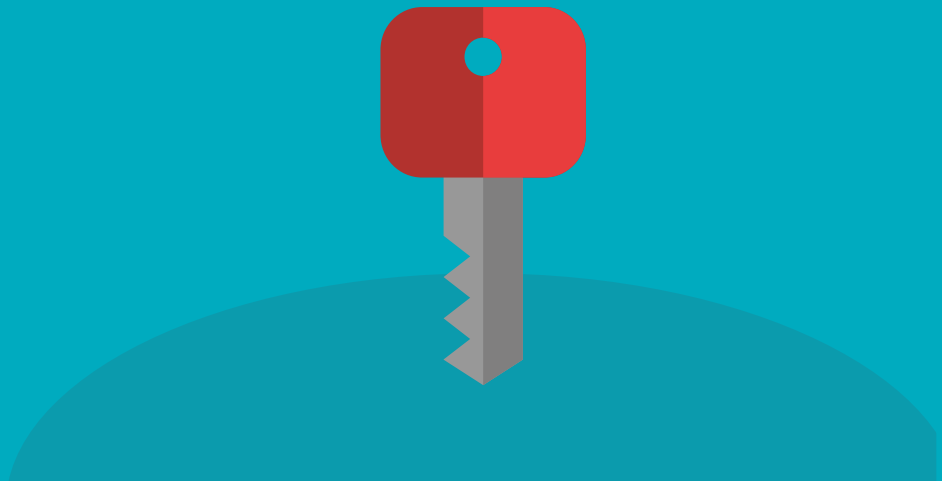
6667

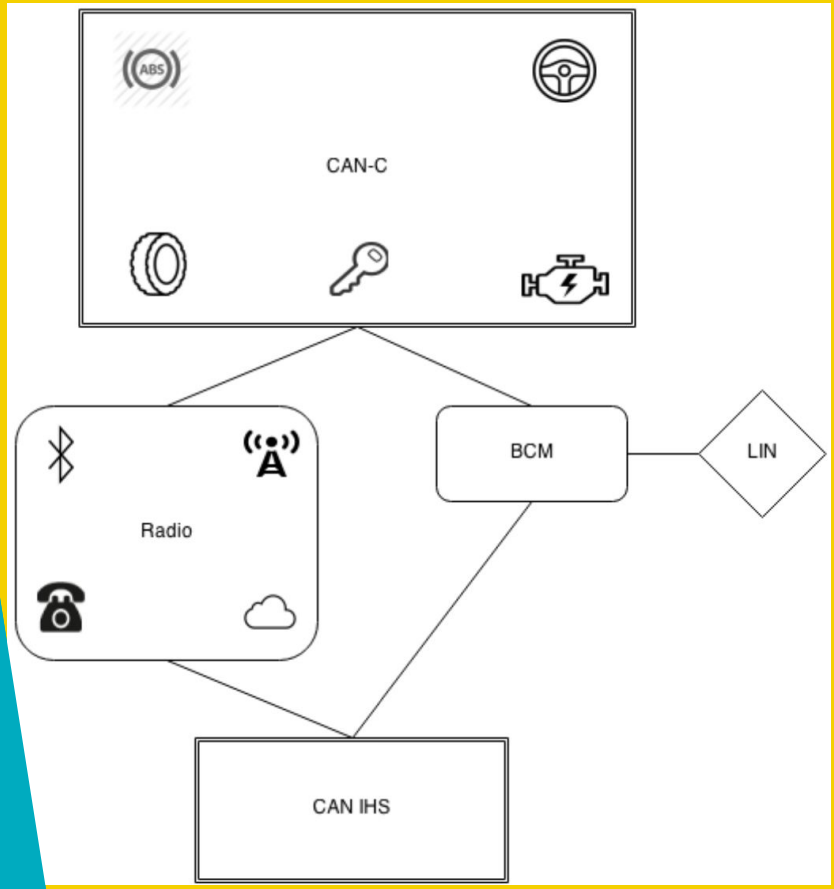
D-BUS // IRC
An inter-process communication (IPC) and remote procedure call (RPC) mechanism

… and more!

03

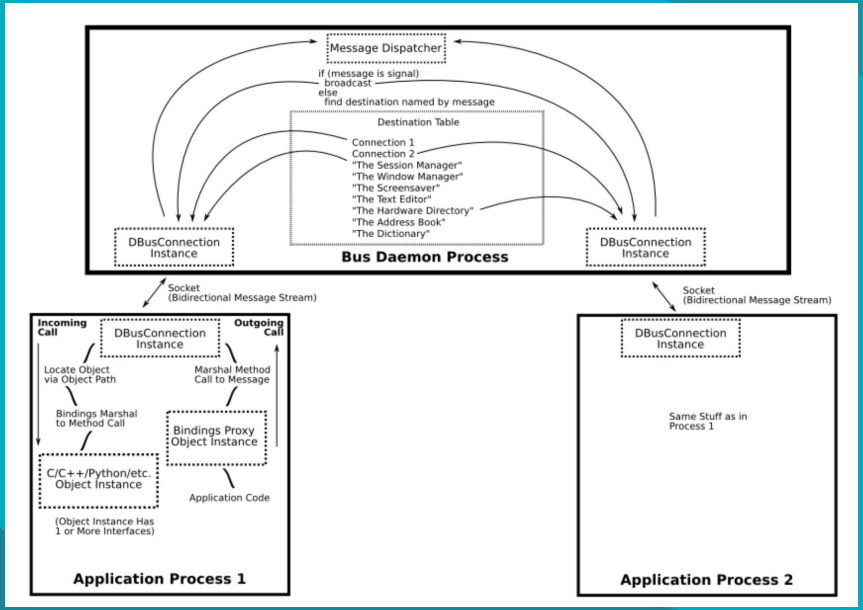
Messaging Compromises





D-BUS

- ★ Auth can be required *but wasn't*
- ★ Directly from D-BUS, authors could:
  - ○ Adjust volume + disable dials
  - ○ Change display image
  - ○ Access PPS data (eg. VIN / GPS)
  - ○ Change HVAC + fan speeds
- ★ Code execution! Via the "execute" command, run arbitrary shell commands as root



However, the command injection is not necessary because the 'NavTrailService' service actually provides an 'execute' method which is designed to execute arbitrary shell commands! Hey, it's a feature, not a bug! Below is a listing of all the services available

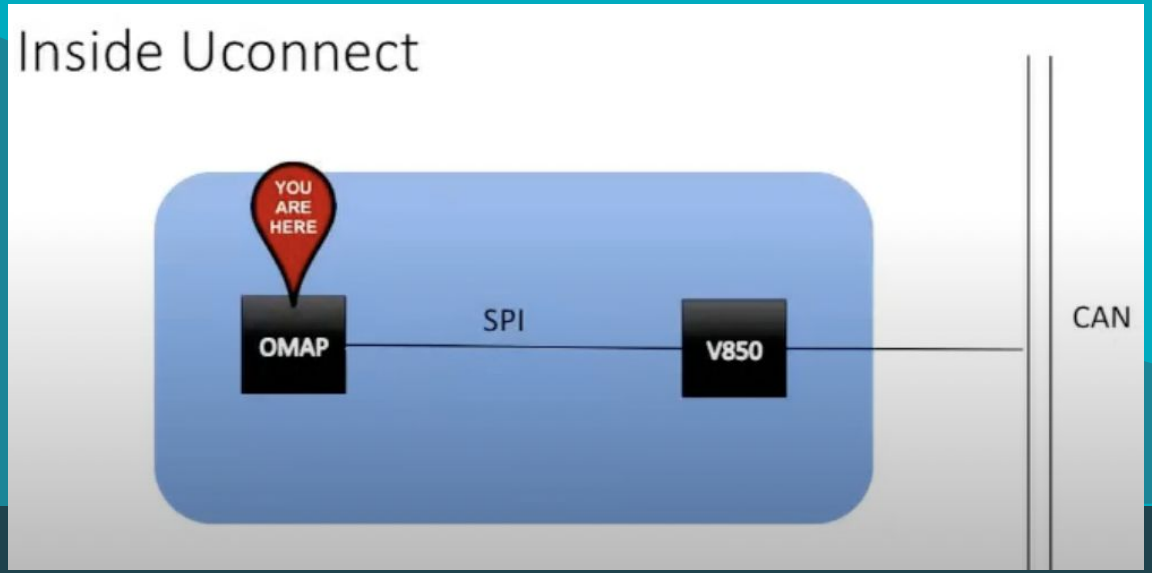
Cellular Reach

- ★ Not limited to WiFi customers
- ★ Not limited to distance (*anywhere in the Sprint Network)
- ★ 1.4 Million vehicles recalled – any could be controlled remotely

Since a vehicle can scan for other vulnerable vehicles and the exploit doesn't require any user interaction, it would be possible to write a worm. This worm would scan for vulnerable vehicles, exploit them with their payload which would scan for other vulnerable vehicles, etc. This is really interesting and scary. Please don't do this. Please.

Further Connections

- ★ OMAP Chip currently compromised, doesn't have reach to the CAN bus
- ★ Via SPI, messages can be sent to the V850 chip
  - Connects to CAN bus, ideal for controlling the whole car
- ★ V850 can be updated by the head unit (USB)
- ★ Via 'update' command, new shell code can be re-flashed to unused ROM (read-only)
- ★ Understanding of CAN messages needed, was reverse engineered



Using CAN

- ★ Serial Peripheral Interface (SPI) and a proprietary protocol connected compromised OMAP to newly re-flashed V850
- ★ SPI used as a “trampoline” to access V850, and sending CAN messages
- ★ Which CAN messages to send?
  - ○ wiTECH products allowed for decrypting Java + Python + JavaScript bytecode
  - ○ In de-obfuscating the message constructor, decryption keys like “G3n3r@ti0n” were found
  - ○ Monitoring CAN traffic led to deriving Speed, Braking Percentages, and checksum authentication (*proprietary XOR, found by decompiling C)

While some of the research could proceed without the diagnostic equipment, many active tests and ECU unlocking require an analysis of the mechanic’s tools. After both authors of this paper sold plasma for several weeks, we were finally able to afford the system

We first reverse engineered the disassembly to C because one of the authors of this paper is a complete psychopath. From there, the C function was ported to Python for testing. The following code is the Python code derived from the disassembly.

04

Remote Control of Car

With CAN Compromise ...

Normal CAN messages can be sent to do the following:

TURN SIGNAL

Both left and right signals can be enabled / disabled via a simple script

CAR LOCKS

All locks can be engaged / disengaged via a similar script

RPMS

Tachometer can be set to any arbitrary value

With CAN Compromise ...

Diagnostic CAN messages can be sent, when speed is below 5-10 mph to do the following:

KILL ENGINE

StartRoutineByLocalIdentifier(15) can be called to stop the engine

BRAKES

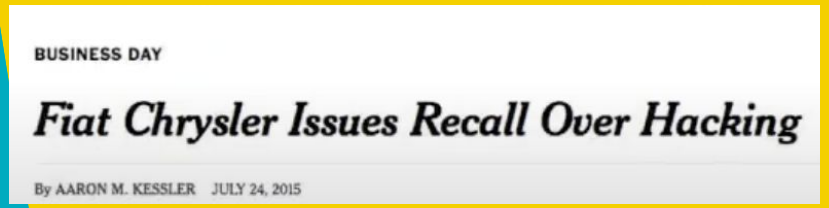
Brakes can either be disengaged, or remotely engaged

STEERING

Through using park assist, the wheel can be turned

05

Moving Forward



Vulnerability Disclosure

- ★ 10/2014: D-BUS exposure via WiFi disclosed
- ★ 03/2015: Reprogramming of V850 disclosed
- ★ 05/2015: D-BUS exposure via Cellular disclosed
- ★ 07/16/2015: Chrysler released a patch for the issue
- ★ 07/21/2015: Wired article about vulnerability released
- ★ 07/24/2015: Chrysler recalls 1.4m vehicles voluntarily

Legislation

Sens. Markey, Blumenthal Introduce Legislation to Protect Drivers from Auto Security, Privacy Risks with Standards & "Cyber Dashboard" Rating System

Tuesday, July 21, 2015

Washington (July 21, 2015) – As both Congress and the federal government struggle to develop a strategy for the Internet of

- ★ **"The SPY Act would direct the National Highway Traffic Safety Administration and the Federal Trade Commission to establish federal standards to secure vehicles and protect drivers' privacy"**
- ★ **"Only two of the 16 car companies examined had developed any capability to detect and respond to a hacking attack in real time"**
- ★ **Since 2015, the SPY Act was read in Congress twice and never passed into Law**
  - **Similar legislation was introduced in 2015 and 2016, and none was passed.**

Paper Critique

01 Length

02 Technical Content

03 Personal Narrative

04 Future Research


Thanks for Listening!
Any Questions?

Works Used

https://archive.org/details/youtube-MAcHkASmXEc

https://ioactive.com/wp-content/uploads/2018/05/IOActive_Remote_Car_Hacking-1.pdf


slidesgo