Automotive Security in the Digital Age: Vulnerabilities of the CAN Bus and Consideration of Future Architectures

Kyle Kelly

M.S. Scholarly Paper Requirement

Department of Electrical and Computer Engineering

University of Maryland

First Reader: Professor Robert Newcomb

Second Reader: Professor Rance Cleaveland

19 April, 2017

Abstract

Current automobiles exist in a nexus between the mechanical past and the digital future. Although cars were once purely mechanical, the modern driver now thinks of car actions and events through layers of abstraction, and may have little or no understanding of what actually occurs on the press of a pedal or the push of a button. The addition of new vehicle features to improve safety, efficiency and convenience requires the integration of Electronic Control Units (ECUs) into the vehicle, and as a result the driver has become more removed from the mechanics. This paper examines communication between those controllers through the Controller Area Network (CAN Bus) protocol, the current standard for in-vehicle communication. It goes on to summarize security vulnerabilities through the CAN bus that have been previously outlined and discussed in several white papers. Finally, the paper briefly discusses white papers on the future of automotive security as we transition into the era of autonomous vehicles.

1. Introduction

Automobiles are no longer the mechanical devices that they were just a generation ago. With every new model year vehicle pushed from the factory floor, modern automobiles relinquish more operational control to internal digital networks.

A typical modern mechanic can no longer diagnose car troubles solely through experience and intuition, but instead relies heavily on digital scanners that pull Diagnostic Trouble Codes (DTCs) from the vehicle to determine the root cause of problems. A 2009 report estimated that a then-current, high-end automobile contained close to 100 million lines of code running on 70 to 100 embedded microprocessors known as Electronic Control Units (ECUs), and predicted that the number of lines of code would increase to between 200 and 300 million in the near future [1].

This paper sets out explain the current state of automotive security for the vast majority of vehicles on the road today. It discusses how digital integration has left many in-vehicle control systems vulnerable to attack, and discusses the seriousness of those vulnerabilities. Lastly, it considers future embedded architectures to ensure passenger and vehicle safety despite an increasing digitalization of the automobile.

To understand the control systems within a vehicle, the reader must first understand the digital network by which they communicate. Therefore, Section 2 of this paper deals with the Controller Area Network (CAN bus), the standard communication protocol for in-vehicle networks, which connects the ECU nodes. The CAN packet structure is detailed and security properties are discussed.

Up until recently, these ECUs were added to vehicles in an ad-hoc manner to extend feature sets, and it was unclear that any big picture strategy was applied to security as it related to passenger safety. Two papers [4, 6], published in 2010 and 2011 as part of a collaboration between the University of Washington and the University of California at San Diego, attempted to examine exactly this. These papers were pivotal is exposing security vulnerabilities of modern automobiles and propelled a breadth of research in the field. Those papers, and their findings, are summarized in Section 3. Additional exploits that resulted from that research are also mentioned at some depth.

Finally, as the era of the autonomous vehicle looms just before us, automotive security has never been more critical. Section 4 examines new methods of automotive security and discusses strategies for implementation.

2. Controller Area Network

The CAN bus was developed by BOSCH in the 1980s as a multi-master, message broadcast system [2] and is used in automotive applications for the communication between ECUs within a vehicle. Starting in 2008, all cars in the U.S. were required to implement a CAN bus for diagnostics [4].

An ECU refers to an embedded system within a vehicle that controls one or more subsystems of that vehicle. An ECU may be connected to the CAN bus to send and receive signals to assist in its control operation [3]. Examples of such ECUs would be the Engine Control Module (ECM or ECU), the Powertrain Control Module (PCM) and the Anti-lock Breaking System (ABS).

2.1 CAN Frames, Arbitration and Buses

The standard CAN frame format has several bitfields, but two fields, arbitration ID and data, are most crucial to understanding a CAN message at a basic level.

The data portion of the message consists of at most 8 bytes of ECU data. The length of the data is designated by the 4-bit Data Length Code (DLC), which gives the number of bytes being transmitted [2]. Each byte in the data will contain information pertaining to the ECU transmitting on the bus. Figure 1 shows the basic format of a CAN frame.

The arbitration ID is a unique 11-bit ID (for standard CAN) manually assigned to an ECU that arbitrates for priority on the bus. One ECU may transmit from multiple IDs but no two ECUs will have the same ID. If two ECUs attempt to send a CAN message at the same time, the one transmitting with the lower arbitration ID will be successful [2]. Therefore, ECUs with with the most critical functions are generally assigned a lower arbitration ID.

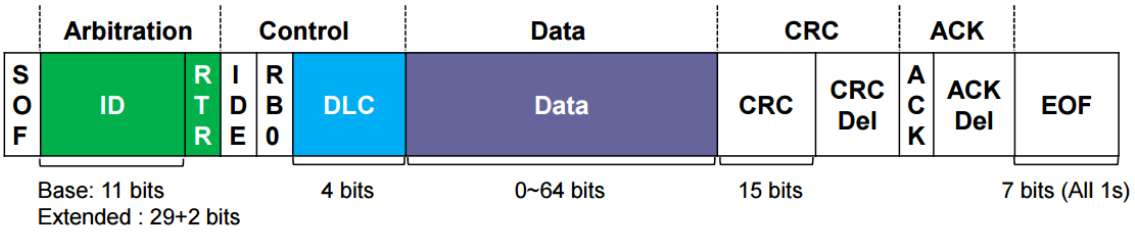

Figure 1: The basic format of a CAN frame [5]. From left to right, the bit fields are: start of frame (SOF), arbitration ID, remote transmission request (RTR), identifier extension (IDE), data length code (DLC), data, cyclic redundancy check (CRC), acknowledgment (ACK), and end of frame (EOF) [2].

Modern-day automobiles tend to employ two or more CAN buses, commonly referred to as high-speed (HS) CAN and mid-speed (MS) CAN. The HS CAN transmits at 500 kbps and is primarily used by the powertrain systems, while the MS CAN typically transmits at 125 kbps and connects less-demanding components [4]. Each bus consists of two wires that act as a differential pair and are terminated with a 120Ω resistor at each end [2].

The CAN is a shared bus, and most ECUs within a vehicle will have access to either the HS CAN, the MS CAN or both. The buses also run to the on-board diagnostics (OBD-II) connector which is present in all modern day automobiles.This port allows for direct access to all CAN messages generated by the various ECUs within the vehicle. A high-level representation of the CAN bus is shown in Figure 2, below.

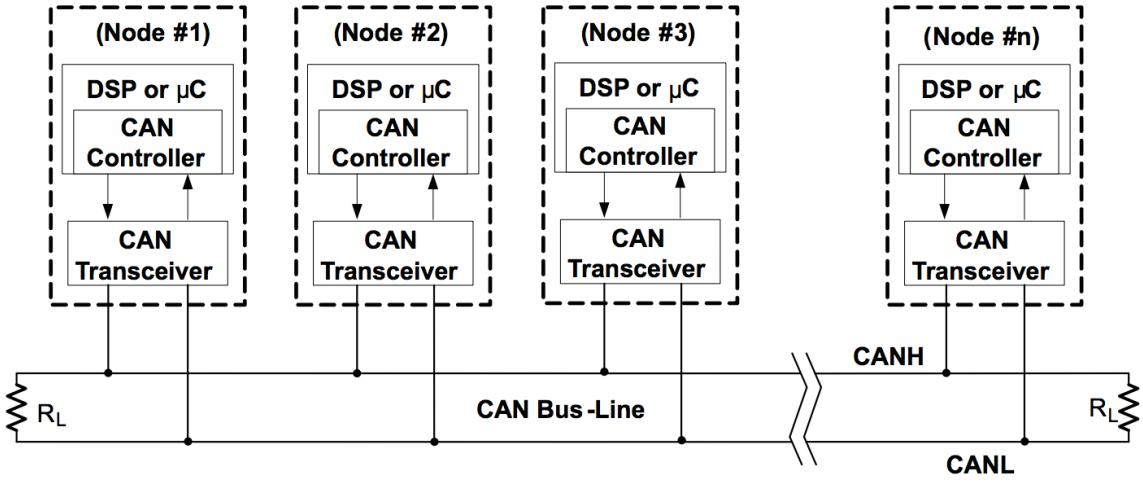

Figure 2: A high-level representation of the CAN bus [2]. Each node represents a complete ECU or part of an ECU. The differential pair can be seen connecting to each node, with the ends terminated with the specified impedance.

2.2 Inherent Security Weaknesses

Several inherent weaknesses in the CAN protocol make vehicle security a challenge. Chief among these are the broadcast nature, no authenticator fields [4] and the lack of encryption.

It is important to realize that all ECUs connected to a CAN bus receive all CAN messages broadcast on that bus. Each ECU will then filter messages pertinent to its functionality. However, this makes for easy network snooping, a fact that is leveraged by most vehicle analyzer tools [4].

Additionally, there is no authenticator field in the CAN frame to provide validation that a CAN message with a certain arbitration ID was actually generated by the ECU holding that ID. Therefore, any component connected to the bus can send a packet to any ECU without it being able to verify the validity of the message.

The CAN protocol currently does not support any security or encryption features. This makes the CAN network an inherent security vulnerability.

3. Security Analysis of a Modern Automobile and their Attack Surfaces

The 2010 and 2011 papers *Experimental Security Analysis of a Modern Automobile* [4] and *Comprehensive Experimental Analysis of Automotive Attack Surfaces* [6], respectively, are responsible for modern day research into automotive security [7]. Before these papers, very little was known about the field and the extent to which an attacker could compromise a vehicle.

The former set out to demonstrate the “fragility of the underlying system structure” of a modern automobile [4]. It worked under the assumption that an attacker had physical access to a car and set out to explore what an attacker with malicious intent could accomplish.

The latter was in response to criticism of the former, in that “attackers with physical access can easily mount non-computerized attacks as well (e.g., cutting the brake lines)” [6]. Therefore, it focused on remote attacks, and in particular the extent to which are possible and practical and the level of risk they represent.

3.1 Internal Access

The first analysis [4] employed two 2009 automobiles of the same make and model and acted in three experimental environments: on a test bench in a lab, on a stationary car on jack stands and on a closed course at speed. The attack methodology was composed of packet sniffing and targeted probing, fuzzing and reverse engineering.

Packet sniffing and targeted probing took advantage of the broadcast vulnerability discussed in Section 2.2. CAN traffic was observed while performing some action (such as turing in the headlights) and then messages were played back on the bus while observing the

vehicle in the hopes of recreating the event. It was employed to map packets to simple functions such as radio control, the Instrument Panel Cluster (IPC) and Body Control Module (BCM).

Fuzzing refers to the “iterative testing of random or partially random packets” to disrupt functionality of some component in the vehicle [4]. It relies on the lack of verification in the protocol, also mentioned in Section 2.2. The researchers discovered that fuzzing in itself proved to be an effective attack because the range of valid CAN packets was relatively small. Through fuzzing, controls for the ECM, BCM, brakes (EBCM) and HVAC systems were discovered.

Reverse engineering involved dumping hardware code into a debugger to gain a detailed understanding of the functionality. This was used for attacks that required new functionality such as bridging the HS and MS buses.

Incredibly, the research showed that an infiltration of nearly any ECU would allow an attacker to “adversarially control a wide range of automotive functions and completely ignore driver input — including disabling the brakes, selectively braking individual wheels on demand, stopping the engine, and so on” [4].

These attacks were performed while the the vehicles were on jack stands and with the vehicles driving on a closed course at speeds of around 40 MPH. Figure 3 displays two tables from the report, with the partial CAN packet that was injected into the vehicle and the physical result of the car.

Packet	Result	Manual Override	At Speed	Need to Unlock	Tested on Runway
07 AE ... E5 EA	Initiate Crankshaft Re-learn; Disturb Timing	Yes	Yes	Yes	
07 AE ... CE 32	Temporary RPM Increase	No	Yes	Yes	✓
07 AE ... 5E BD	Disable Cylinders, Power Steering/Brakes	Yes	Yes	Yes	
07 AE ... 95 DC	Kill Engine, Cause Knocking on Restart	Yes	Yes	Yes	✓
07 AE ... 8D C8	Grind Starter	No	Yes	Yes	
07 AE ... 00 00	Increase Idle RPM	No	Yes	Yes	✓

Table III. Engine Control Module (ECM) DeviceControl Packet Analysis. This table is similar to Table II.

Packet	Result	Manual Override	At Speed	Need to Unlock†	Tested on Runway
07 AE ... 25 2B	Engages Front Left Brake	No	Yes	Yes	✓
07 AE ... 20 88	Engages Front Right Brake/Unlocks Front Left	No	Yes	Yes	✓
07 AE ... 86 07	Unevenly Engages Right Brakes	No	Yes	Yes	✓
07 AE ... FF FF	Releases Brakes, Prevents Braking	No	Yes	Yes	✓

Table IV. Electronic Brake Control Module (EBCM) DeviceControl Packet Analysis. This table is similar to Table II.
†The EBCM did not need to be unlocked with its DeviceControl key when the car was on jack stands. Later, when we tested these packets on the runway, we discovered that the EBCM rejected these commands when the speed of the car exceeded 5 MPH without being unlocked.

Figure 3: Some of the results from the 2010 paper *Experimental Security Analysis of a Modern Automobile* detailing portions of the CAN packet injected into the vehicle and the physical result. Here, one can see successful attacks on the engine and brakes which were tested at speed on a closed course [4].

In addition to single CAN packet attacks, the researchers also detail several composite attack scenarios. One attack set the speedometer 10 MPH less than the actual vehicle speed (after halving the speed up to 20 MPH). Another turned off all of the car's lights when the car was traveling 40 MPH. This included headlights, brake lights and the Instrument Panel Cluster (IPC). A "self-destruct" attack was also created which displayed a 60 second countdown on the dash, honking the horn in the final seconds, and finally killing the engine and activating the door lock relay.

Another surprising result was that attacks to the lower-priority bus could adversely affect the HS bus due to bridging. Although good security practice would call for each bus to be physically isolated, in reality they are bridged to support functionality requirements. An example would be that the vehicle locking system would operate on the lower-priority bus, however it must also be interconnected with the HS bus to monitor safety critical systems. This led to the researchers to demonstrate that any device connected to the lower-priority bus could "influence the operation of the safety-critical components" [4].

While a majority of the vulnerabilities were rooted in the CAN protocol, the researchers also discovered several cases where the car's protocol deviated from the standard and did not follow specifications. Additionally, they found that firmware access control was implemented less broadly than expected. At the conclusion of the experiment the researchers informed the vehicle manufacturers and any affected third-party vendors of the results.

3.2 External Access

Addressing criticism of the initial study, the 2011 report analyzed the external attack surfaces of a late model mass-production sedan. Through empirical analysis, the researchers attempted to make four principal contributions into the research of automotive attack surfaces: threat model characterization, vulnerability analysis, threat assessment and synthesis.

The threat model assumed that the attacker had advanced technical capabilities and knowledge of the target vehicle. They also considered qualitative differences in accessing different I/O channels within the vehicle, which were classified into three categories: indirect physical access, short-range wireless access and long-range wireless access. Figure 4 exhibits many digital I/O channels that can be found in a modern car.

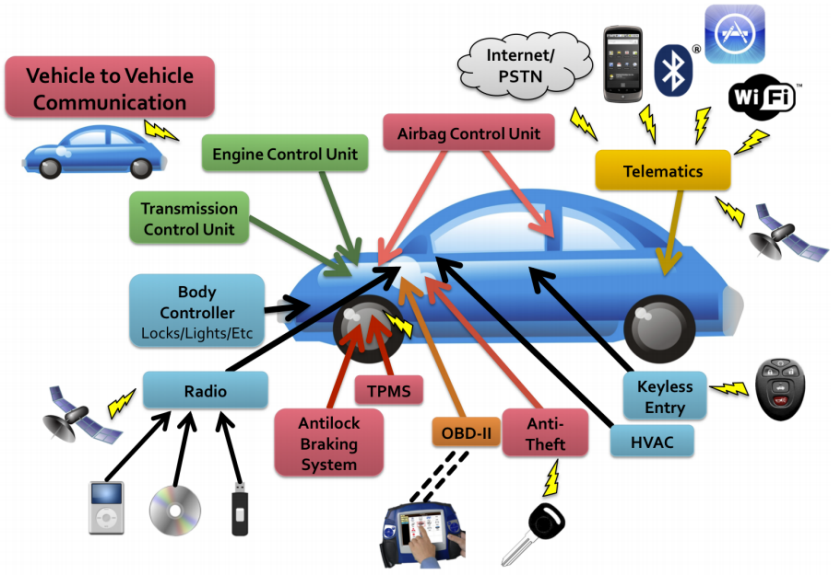

Figure 4: Digital I/O channels appearing on a modern car. Colors indicate a rough grouping of ECUs by function [6].

Examples of indirect physical access are the OBD-II port, CD player and USB port. Short-range wireless refers to Bluetooth, remote keyless entry, tire pressure and RFID keys, while long-range wireless would be broadcast channels (e.g. GPS, satellite radio, FM radio) and addressable channels (e.g. Ford's Sync).

The researchers found vulnerabilities in all categories of external access. One exploit, through the media player, allowed the researchers to modify an audio file that could be burned onto a CD. The CD would play normally in the car while simultaneously issuing CAN packets from the attacker. Through reverse engineering, they also found that any paired Bluetooth device could inject arbitrary code into the telematics unit. Another telematics unit vulnerability found that a similar attack could be conducted with only a cellular connection. The attacker would continuously call the vehicle until a connection was established and then force the telematics unit to download and execute a payload of code from the internet.

The threat assessment envisioned two main scenarios: theft and surveillance. To prove their theft assessment was accurate they created an attack that "directs the car's compromised telematics unit to unlock the doors, start the engine, disengage the shift lock solenoid (which normally prevents the car from shifting out of park without the key present), and spoof packets used in the car's startup protocol (thereby bypassing the existing immobilizer anti-theft measures)" [6]. In the surveillance scenario, they envisioned an attacker exploiting the telematics unit to record in-cabin audio and track the vehicle's location in real-time.

3.3 Ensuing Research

Employing the attack surfaces and methodologies discussed in the 2011 paper [6], computer researchers Dr. Charlie Miller and Chris Valasek went on to execute the most notorious automotive hack to date: taking complete remote control of a 2014 Chrysler Jeep Cherokee [7].

The hack was performed through the vehicle's cellular network and the entertainment system Uconnect. Therefore, it could be executed anywhere within the U.S. where the attacker had an internet connection and the vehicle had a Sprint cellular connection. The researchers demonstrated that the hack could affect a vehicle moving at speeds in excess of 70 MPH and could take control of the brakes, acceleration and a large range of internal controls and settings.

As a result, Chrysler voluntarily recalled 1.4 million vehicles. Miller and Valasek went on to detail in the hack in their paper *Remote Exploitation of an Unaltered Passenger Vehicle* in August of 2015 [7].

A 2016 publication *Error Handling of In-Vehicle Networks Makes Them Vulnerable* unveiled a new CAN vulnerability called the bus-off attack. According to the CAN protocol, when an error is generated by an ECU, it keeps an internal count using the Transmit Error Counter

(TEC). When the TEC for a particular ECU becomes greater than 255, the ECU enters bus-off mode and terminates bus interaction. An attacker with access to the CAN network can inject targeted messages onto the CAN which generate this error and force an ECU to go offline, which could be used to attack safety-critical functions of the vehicle. This was demonstrated on two cars and exposed yet another inherent vulnerability of the protocol.

4. Future Architectures

Advanced driver assist features are already becoming standard in most new cars, and companies such as Ford, Tesla, Nissan and others plan to have fully autonomous commercial vehicles on the road by 2020. Clearly automotive security has never been more important, and a many white papers have been published on the topic.

In a 2015 white paper, Intel laid out best practices for automotive security [9]. Hardware security focuses on ECUs and buses and mentions tamper protection and burned-in device identity to prevent unapproved devices from accessing systems. Software security practices include the enforcement of approved behavior to detect malicious threats while network security practices refer to message and device authentication and access controls. Beyond hardware, software and in-vehicle network security, the paper also considers cloud security services. This is a new aspect of automotive security but is just as critical because autonomous vehicles will most likely require real-time cloud data to operate.

AUTOSAR (AUtomotive Open System ARchitecture) is a development partnership between Original Equipment Manufacturers (OEMs) and suppliers, including core partners Bosch, Ford, the BMW Group and more [8]. AUTOSAR aims to standardize the software architecture of ECUs and “increase reuse and exchangeability of software between OEMs and

suppliers” [8]. The architecture includes a cryptographic interface to provide security solutions for hardware and software.

5. Conclusion

In this paper we have discussed the CAN bus, the standard for in-vehicle communication, and its inherent weaknesses. We then saw how these vulnerabilities were exploited in ground-breaking research, culminating in a hack that resulted in the recall of 1.4 million cars. Finally, we briefly discussed good security practices and future architectures as cars progress into to age of autonomy.

With the automotive industry advancing at breakneck speed, engineers and researchers must be vigilant to ensure that it does not outpace the security needed to ensure safe and private vehicle operation. Treating vehicle security as safety-critical rather than a theft prevention feature is paramount moving forward. It does, however, appear that the industry has recognized the importance of security and it will be exciting to see its implementation in the future.

References

[1] R. Charette. This Car Runs on Code. Online: http://spectrum.ieee.org/transportation/systems/this-car-runs-on-code, February 2009.

[2] Corrigan, S. (2008). Introduction to the Controller Area Network (CAN) (Application Report No. SLOA101A). Retrieved from Texas Instruments website: http://www.ti.com/lit/an/sloa101a/sloa101a.pdf

[3] National Instrument Whitepaper: ECU Designing and Testing using National Instrument Products. Online: http://www.ni.com/white-paper/3312/en/

[4] K. Koscher, A. Czeskis, F. Roesner, S. Patel, T. Kohno, S. Checkoway, D. McCoy, B. Kantor, D. Anderson, H. Shacham, and S. Savage. Experimental security analysis of a modern automobile. In IEEE S&P, May 2010. http://www.autosec.org/pubs/cars-oakland2010.pdf

[5] K. Cho and K. Shin. Error Handling of In-vehicle Networks Makes Them Vulnerable. CCS 2016. https://kabru.eecs.umich.edu/wordpress/wp-content/uploads/ktcho_busoff_CCS_16.pdf

[6] S. Checkoway, D. McCoy, B. Kantor, D. Anderson, H. Shacham, S. Savage, K. Koscher, A. Czeskis, F. Roesner, and T. Kohno. Comprehensive Experimental Analyses of Automotive Attack Surfaces. In USENIX Security, 2011. http://www.autosec.org/pubs/cars-usenixsec2011.pdf

[7] C. Miller and C. Valasek. Remote exploitation of an unaltered passenger vehicle. Black Hat USA, 2015. http://illmatics.com/Remote%20Car%20Hacking.pdf

[8] S. Rathgeber and M. Niklas. Automotive Software Kongress. Online: http://www.autosar.org/fileadmin/files/presentations/2016_09_21_Automotive_Software_Kongress_FH_Landshut.pdf

[9] Intel Security and McAfee White Paper. Automotive Security Best Practices. Online: https://www.mcafee.com/hk/resources/white-papers/wp-automotive-security.pdf