Global Information Assurance Certification Paper

The Automotive Top 5: Applying the Critical Controls to the Modern Automobile

GIAC (GCCC) Gold Certification

Author: Roderick Currie, roderick.h.currie@gmail.com
Advisor: Stephen Northcutt
Accepted: February 14, 2016

Abstract

The car of today is an inherently vulnerable platform. At its core is a computing architecture from the 1980s which was designed to be lightweight and efficient, with very little thought given to security. As the modern automobile becomes increasingly connected, its attack surface only continues to grow. In the wake of several recent high-profile car hacking demonstrations, automakers face the daunting task of trying to lock down this insecure platform with bolt-on security fixes. This paper proposes a plausible strategy for securing modern automotive systems which takes into account some of the key limitations of the automobile industry, in addition to presenting a methodology for applying the Critical Controls to the modern automobile platform.

1. Introduction

The modern automobile is a highly vulnerable computing platform. And yet, every day, millions of people trust their safety, and indeed, their lives, to automobiles riddled with security vulnerabilities. The vulnerabilities found in many modern vehicles have been documented and demonstrated by teams of security researchers, often times gaining media attention in the process. However, the automotive industry has generally been slow to improve its security posture or even acknowledge the problem. While there are numerous challenges and constraints unique to the auto industry, it is nonetheless true that there is a lack of consensus in the industry about how to approach the problem of automotive systems security. The Critical Controls – a set of 20 best-practice security controls aimed primarily at enterprise networks – can be adapted to offer a starting point to guide automakers to produce more secure vehicular systems.

The automobile has only grown more complex with the passage of time. This is a trend which should be expected to continue. While cars of the 1980s contained only a handful of computerized parts, the car of today is essentially a rolling computer network.

1.1. An Inherently Vulnerable Platform

Development of the Controller Area Network (CAN) protocol was begun in 1983 by German company Robert Bosch GmbH (Sewell Direct, 2015). After three years of development, CAN bus technology hit the public market in 1986, first showing up in the BMW 850 coupe (Sewell Direct, 2015). The development of the CAN bus eliminated the need for a significant amount of bulky wiring, resulting in a weight reduction of over 100 pounds for the 1986 model year 850 coupe (Sewell Direct, 2015). Another driving force behind CAN was that the National Highway Traffic Safety Administration (NHTSA) and the California Air Resources Board (CARB) sought a way to easily and effectively monitor a vehicle's emission-control systems (Wojdyla, 2012). Because this relied on gathering readings from a wide range of different vehicle sensors, it became necessary to develop a centralized network bus through which these sensors and their respective controllers could all communicate; and so, CAN was born.

While CAN was the perfect solution to the problems it sought to address, it was not designed with security in mind. CAN places the majority of vehicle controllers on the same bus, allowing an attacker to easily pivot around the vehicle. CAN also does not include native support for encryption, as its designers never foresaw a need to protect messages traversing the vehicle's network. Furthermore, CAN lacks the ability to reliably authenticate devices connected to the bus, allowing an attacker to easily connect his own device to the CAN bus and send spoofed messages. Unfortunately, CAN remains the bus standard of choice in the auto industry due to its low cost, light weight, and high interoperability. Therefore, any proposed security solution must incorporate CAN while mitigating as many of its inherent vulnerabilities as possible.

1.2. An Ever-Expanding Attack Surface

What makes the modern automobile particularly challenging from a security perspective is that it attempts to interconnect the 30-year old CAN bus with modern protocols such as Wi-Fi, 3G, Bluetooth, and GPS. Vehicle buyers are increasingly demanding "connected" vehicles which offer features such as hands-free calling, in-vehicle web browsing, navigation with real-time traffic data, real-time weather updates, and more. The result is an outdated CAN platform being exposed to the Internet and public cellular networks. And, as security researchers have already demonstrated, these high-tech interfaces are the perfect entry point through which to gain access to the CAN bus where a vehicle's controls and safety systems can then be manipulated.

2. The Critical Controls

First hosted by SANS in 2008, the Critical Security Controls (CSCs) are a set of cyber security practices that provide specific ways to defend against the real-world threats facing information systems today (Hietala, 2013). Today, the Critical Controls are hosted and maintained by the Center for Internet Security, or CIS. CIS is an organization which utilizes government and private industry partnerships to further the goal of effective cyber defense and enhanced cyber readiness worldwide (Center for Internet Security, 2015). In addition to providing implementation guidance, the Critical Controls

also offer detailed metrics for measuring effectiveness, and suggestions for automation and testing (Hietala, 2013).

The CSCs include 20 specific controls for effective cyber defense. However, it is not necessary for an organization to implement all 20 controls in order to improve its overall security posture. In fact, most successful implementations of the Critical Controls involve selecting a handful of the most high-priority controls to focus on first, and then expanding to incorporate other controls as needed (Hietala, 2013).

2.1. Challenges and Limitations

Applying the Critical Controls to a modern automobile platform does not come without its challenges and limitations. The Critical Controls were, of course, designed to be applied to an organization as a whole. The CSCs can certainly be adapted to fit a single business unit within an organization or perhaps a corporate LAN within a business unit; but applying the CSCs to an entity such as a car requires a more "outside the box" approach.

There are many different makes and models of vehicles on the market today, each with varying levels of connectivity and each with differing security postures. However, in general, the modern automobile is essentially a LAN – a localized network of interconnected computers, each controlling its own vehicular function. Increasingly, as vehicles become more "connected", that LAN finds itself joined to a WAN – a network of other, similar vehicles with the ability to communicate with each other. By looking at the modern automobile in this way, applying the CSCs is not dissimilar to applying them to an enterprise network.

2.1.1. Outdated Architecture

At the heart of any modern vehicle's interconnected systems is the Controller Area Network bus, or CAN bus. The CAN bus is a single, centralized network bus on which all of a vehicle's data traffic is broadcast. The CAN bus carries everything from operator commands such as "roll down the windows" or "apply the brakes", to readouts from sensors reporting engine temperature or tire pressure. The advent of the CAN bus brought about improvements in efficiency and a reduction in complexity while also reducing wiring costs.

The CAN bus is a 30-year old architecture that was developed for various valid reasons, but security certainly was not one of them. Automakers at the time could not possibly envision the risk of cars being hacked decades into the future, nor could the governing bodies that mandated the CAN standard. The CAN architecture was designed to be lightweight and robust, and those qualities it accomplishes very well. However, CAN contains numerous security vulnerabilities that are inherent in its design.

The CAN bus connects a vehicle's many Electronic Control Units (ECUs) together so that they may communicate with each other. A basic conceptual view of CAN architecture is shown below in *Figure 1*.

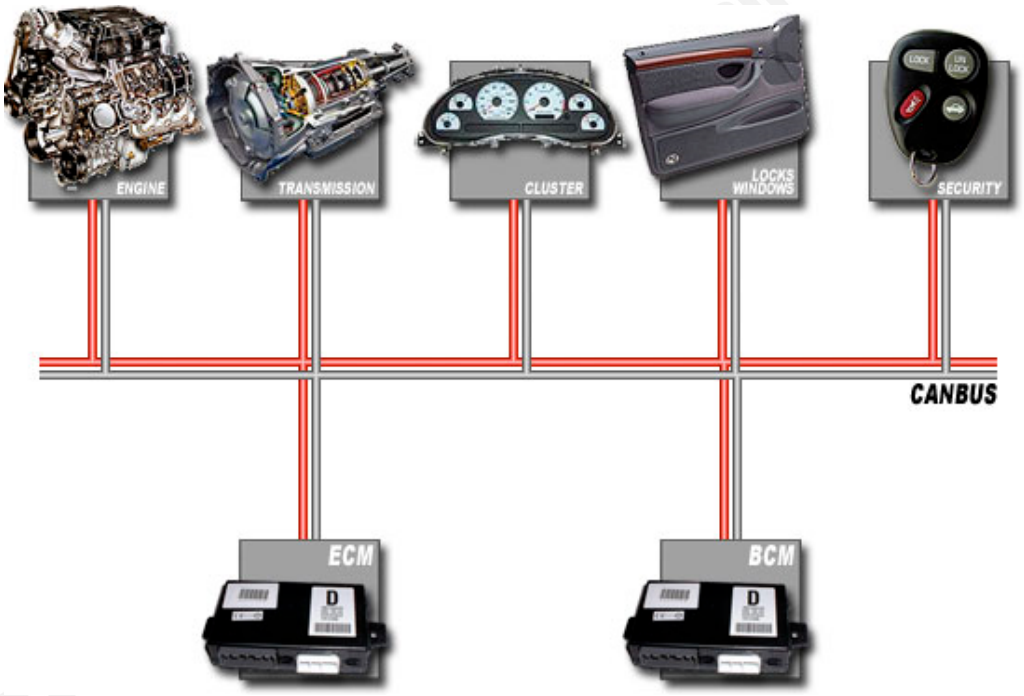

***Figure 1*: CAN Bus Network (Fortin Electronic Systems, 2006)**

The rationale behind the open nature of the CAN bus architecture is that ECUs for different vehicular components often have a need to share messages with one another. Particularly in modern vehicles, it is not uncommon to see even cross-domain communication where an ECU responsible for a vehicle safety function may have a need to communicate with the ECU for the infotainment system, or vice versa. In a recent experiment involving a relatively new model vehicle, a team of researchers from the

University of California, San Diego (UCSD) explored the true depth of CAN interconnectivity.

On the UCSD test vehicle, the research team found that the antilock braking and stability control systems reported the vehicle's speed to other modules, including the speedometer and engine controller (Foster & Koscher, 2015). More surprisingly, they also discovered that the radio also received these same speed messages from the stability control system to dynamically adjust its volume depending on the speed of the vehicle (Foster & Koscher, 2015). The team found that the radio, in fact, needed to receive messages from a whole host of different controllers. On their test vehicle, the "click-clacks" of the turn signals were actually being simulated by the radio, which received the turn signal status from another controller (Foster & Koscher, 2015). Additionally, the telematics system could command the HVAC controller to turn down the vehicle's air-conditioning fans when a phone call was received (Foster & Koscher, 2015).

Because a vehicle's various controllers are so intertwined, almost any component can communicate with the controllers responsible for critical vehicle control and safety functions, resulting in a very serious security vulnerability.

2.1.2. Software Updates

Another area in which the modern car differs from a traditional computer network is in the way it receives software updates. A typical network of desktop computers can be configured to receive software updates and security patches seamlessly via the Internet, almost as soon as they are made available by the software manufacturer. For many vehicle models currently being sold today, however, the option to receive over-the-air (OTA) software updates has yet to be implemented. Therefore, when a critical software update becomes available, vehicle owners must visit their local dealership's service department to have the update performed by a certified technician.

Although vehicles are sometimes recalled due to mechanical defects, it is now estimated that "about 50 percent of recalls are associated with the growing amount of software in cars" (Redbend, 2015). Rather worryingly, a formal study conducted by the U.S. Government Accountability Office (GAO) found that the average recall completion rate for safety related recalls in the United States is approximately 65% (United States

Government Accountability Office, 2011). This means that, on average, about one-third of recalled vehicles go unfixed. This can be attributed not only to the inconvenience to the customer of having to bring a vehicle to the dealer for a recall service, but also the challenge of trying to notify all affected customers by using contact information which is not always kept up to date.

2.1.3. Recall Costs

The cost of performing a recall for a software update is also an important factor which automakers take into consideration. In the traditional computing world, software updates can be pushed out with little effort or expense. Microsoft famously uses its "Patch Tuesday" to roll out a handful of critical security updates for its products each month. Until some form of OTA update architecture becomes commonplace in all new vehicles, the "Patch Tuesday" model is certainly not an option for automakers. Currently, when a critical vehicle security vulnerability is discovered, the vehicle manufacturer must recall hundreds of thousands – sometimes even millions – of vehicles at significant cost to the business.

Following the 2015 hack of a Jeep Cherokee by security researchers Charlie Miller and Chris Valasek, Fiat Chrysler Automobiles (FCA) was forced to recall and patch any vehicles that were vulnerable to Miller and Valasek's exploit. This included some 1.4 million vehicles across the Dodge, Ram, Jeep, and Chrysler brands (Cobb, 2015). By some estimates, the amount which this critical security update cost FCA in labor hours alone was in excess of $10 million (Cobb, 2015). Automakers are also cautiously aware of the reputation hit which can result from such a large recall.

Following the FCA incident, while some vehicle owners chose to have the software patch applied by a certified FCA technician at their local dealership, FCA also gave vehicle owners the option of installing the update manually from a USB device which was sent to affected owners via regular mail. An example of the actual USB device is shown below in *Figure 2*. Going forward, the strategy of mailing USB drives to millions of affected customers each time a vehicle's software needs to be patched is neither practical nor cost effective. It is apparent that this model for software patching is not sustainable.

***Figure 2*: FCA's Software Update Distribution System (Trumpbour, 2015)**

2.1.4. Vehicle Design Constraints

There are several constraints unique to the auto industry which present particular challenges when trying to develop secure automotive systems. Perhaps the single most constraining factor is part cost (Pike et al., 2015). Every ECU, CPU, and their associated wiring all drive up part costs, which in turn eats into the automakers' profits. For the automakers, it sometimes makes more financial sense to leave a vulnerability unmitigated if the cost of mitigation outweighs the perceived financial risk. Therefore, it is extremely important that a secure vehicle platform does not rely on vast amounts of additional costly hardware and software.

The size and weight of parts are also of significant concern to automakers (Pike et al., 2015). The CAN architecture was originally designed to cut down on wiring costs and reduce complexity. Any proposed new architecture which requires a significant increase in the amount of vehicle wiring may not be feasible.

ECU memory and timing constraints must also be considered when developing new security solutions. In the ECUs which make up a vehicle's internal network, memory is often the most expensive component (Pike et al., 2015). Therefore, automakers design

ECUs with the least amount of memory required to perform a given function. This leaves ECUs with very little available memory for security functions such as host-based firewalls. Furthermore, many ECUs perform real-time tasks with safety-critical timing deadlines (Pike et al., 2015). In these cases, any security measures which delay the timing of the ECU will not be feasible.

3. The Automotive Top 5

The Critical Controls are intended to serve as guidance to help an organization improve its overall security posture. While the Critical Controls are primarily designed to be applied to an organization as a whole, they can also be adapted to fit other information security applications. When viewing the modern automobile as a computer network, the Critical Controls provide an effective framework for securing this vulnerable platform.

There are 20 Critical Controls in all, covering areas including Malware Defense, Wireless Access Control, Incident Response, Software Configuration, and many more. However, it is neither necessary nor realistic to implement all 20 controls simultaneously. The most effective implementation of the Critical Controls begins by highlighting the most significant areas of weakness and selecting the controls which best mitigate these vulnerabilities. Given the serious security vulnerabilities which are common across many different makes and models of modern automobile, this paper proposes that there are five key Critical Controls which should be embraced by the auto industry to move towards more secure vehicular platforms. The "Automotive Top 5" is not a fix-all solution, but rather, a recommendation to mitigate the most common vulnerabilities which are likely to be exploited by an attacker.

It is important to recognize that not all makes and models of vehicles share the same vulnerabilities. Indeed, some vehicles are more secure than others. While it is to be expected that each automaker will approach the problem of vehicle systems security in its own way, the "Automotive Top 5" provides a series of fundamental security best practices which should be followed going forward.

3.1. Automotive Control 1: Inventory of Authorized and Unauthorized Devices

Actively manage (inventory, track, and correct) all hardware devices on the network so that only authorized devices are given access, and unauthorized and unmanaged devices are found and prevented from gaining access.
(Center for Internet Security, 2015) (CSC 1).

3.1.1. Why It Matters

Although the Critical Controls need not be implemented in any specific order, the first control is arguably the most important, and serves as a foundation for the other controls that follow. This is true either when applying the control to an enterprise network or when applying the control to the modern automobile platform. Establishing an inventory of authorized devices on the network is a key step in building an understanding of the network environment. The goal, of course, is to keep unauthorized devices out; but this cannot be accomplished without first documenting which devices are authorized to be on the network in the first place. Furthermore, it is futile to apply security controls to some devices on the network if there are still unknown devices residing on the network which remain unsecured.

In 2013, security researchers Charlie Miller and Chris Valasek demonstrated attacks against a 2010 Toyota Prius and a 2010 Ford Escape. In each case, Miller and Valasek's method involved using a laptop PC running Windows XP hooked into the vehicle's OBD-II port via a series of cables (Miller & Valasek, 2014a, p. 23). The OBD-II port is traditionally used by mechanics and repair shops to retrieve fault codes and diagnose problems with a vehicle, but it is also an attractive point of entry for a vehicle security researcher or an attacker. In the case of Miller and Valasek's attack, the Windows laptop represents an unauthorized device. There is no legitimate reason for a laptop computer to be connected to the vehicle's internal network where it can then send messages to critical vehicular control systems. It is essential, therefore, that a secure automobile is able to detect when an unauthorized device has been connected to the network. When an unauthorized device is detected, it should either be dropped from the network, or at the very least, its messages should be ignored by other controllers on the

bus. How unauthorized devices are handled depends greatly on the specific architecture and bus types used in the vehicle.

3.1.2. Implementation

Establishing a comprehensive inventory of the controllers on the vehicle's network is the first step in implementing CSC 1. This inventory should include all local controllers responsible for vehicular functions, but would not be expected to include externally connected devices such as cellular phones or diagnostic devices. Because a modern automobile can include up to 70 Electronic Control Units (ECUs) (Guo, Ang & Wu, 2009), the task of inventorying the controllers is not something which should be undertaken on a finished automobile; rather, the controllers must be inventoried during initial development of the platform to establish a clean hardware baseline for the vehicle.

With the baseline of CAN controllers established, the next challenge is to perform enforcement – keeping unauthorized devices out while allowing authorized devices to communicate on the network. On a traditional Ethernet network, every device connected to the network has a unique identifier – the Media Access Control address, or MAC address. On a CAN system, however, individual controllers do not have MAC addresses to identify themselves. Instead, every CAN message broadcast by a controller includes an 11-bit identifier field which identifies the sending controller. The diagram provided below in *Figure 3* shows the standard CAN frame format. The identifier field can be seen near the beginning of the frame.

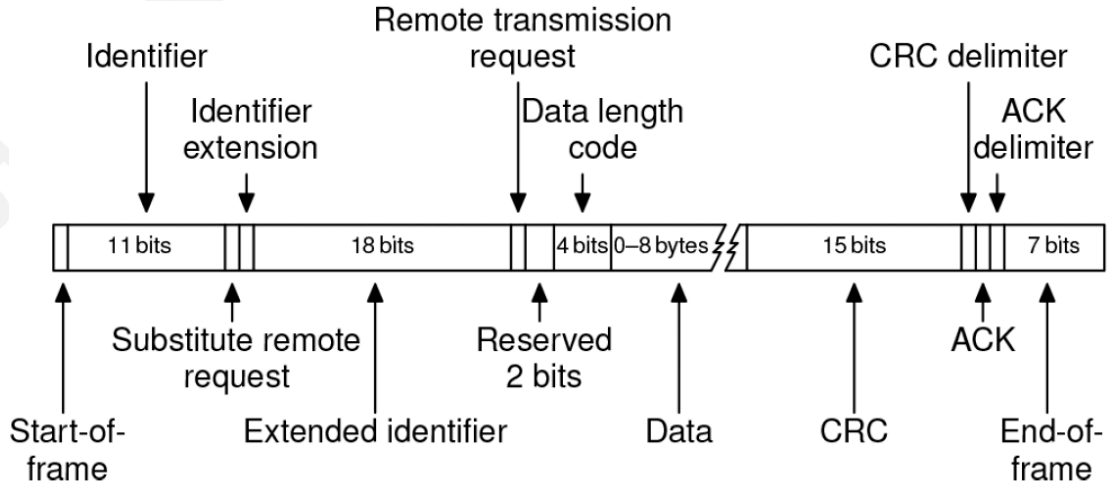

***Figure 3*: The CAN Frame Format (Foster & Koscher, 2015)**

Conceptually, CAN device authentication can be accomplished by preprogramming CAN controllers with a whitelist of CAN identifiers for known good devices. For example, the vehicle's steering controller should know to only trust commands coming from the controller associated with the vehicle's steering wheel – and not from some external source. If an attacker attempts to connect a laptop or mobile device to the CAN bus in an attempt to manipulate the vehicle, any commands sent from the rogue device should be ignored by other controllers if they do not recognize the sending device's identifier. Of course, this opens the door to CAN identifier spoofing, whereby an attacker spoofs messages to make them appear as if they are coming from a legitimate source. And so, for device authentication to work effectively, the CAN identifier field must be encrypted when in transit. Encryption of the CAN identifier field prevents an attacker from being able to intercept and spoof a legitimate CAN device's identifier field.

3.2. Automotive Control 2: Secure Configurations for Hardware and Software

Establish, implement, and actively manage (track, report on, correct) the security configuration of laptops, servers, and workstations using a rigorous configuration management and change control process in order to prevent attackers from exploiting vulnerable services and settings.
(Center for Internet Security, 2015) (CSC 3).

3.2.1. Why It Matters

While controller whitelisting combined with encryption of the CAN identifier field allows for some degree of control over the hardware being connected to the network, a longer-term goal for the auto industry should be to move away from the inherently insecure and open-plan CAN architecture in favor of a more centrally-managed model. Because the CAN controllers themselves cannot be effectively secured, it is necessary to employ a securely configured central control unit to oversee and manage CAN bus communications.

Most automotive attacks that have been demonstrated to date have relied on a lack of secure configuration, allowing for remote access and exploitation of the vehicle

(Miller & Valasek, 2015). CAN controllers – by design – will receive and process any messages they consider to be legitimate. A CAN controller generally lacks the processing power to screen messages effectively. Therefore, it is necessary to devise a secure vehicle architecture in which the security of the different controllers is centrally managed. Rather than lock down each individual controller, a centrally-located computer can be positioned to oversee all traffic on the network and make real-time security decisions.

3.2.2. Implementation

Ideally, a central control unit should manage device authentication and control all communications being passed across the network. One such architecture has recently been developed by scientists and researchers at Germany's Technical University of Munich (TUM, 2015). Known as the "Automotive Service Bus", a diagram of this new model for vehicle networking is shown below in *Figure 4*.

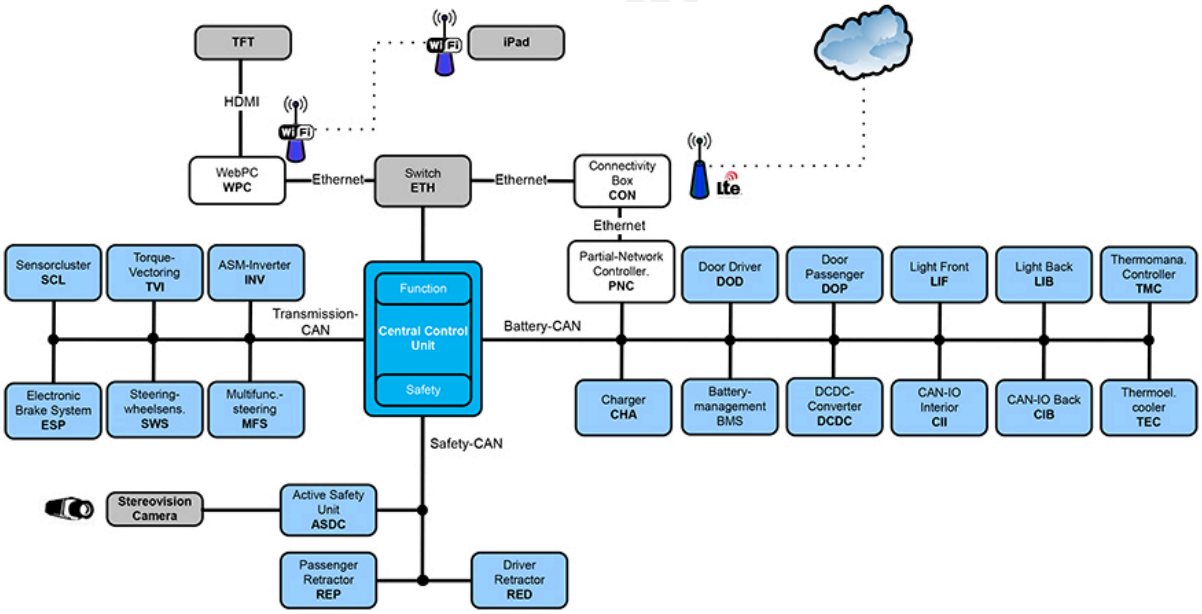

***Figure 4*: The Automotive Service Bus (TUM, 2015)**

The Automotive Service Bus, for which the software has been released publicly under an open-source license, was intended to greatly reduce the complexity of existing in-vehicle networks while also offering enhanced security (TUM, 2015). At the core of the Automotive Service Bus is a Central Control Unit which manages all communications between different buses. This architecture features three separate CAN buses – one

dedicated to vehicle control functions, one dedicated to safety functions, and another dedicated to electronic functions. Most importantly, external communication interfaces run over an Ethernet bus which is separated from the CAN buses by the Central Control Unit. There is, therefore, no way for a connected mobile device to communicate directly with a critical vehicle controller without the permission of the Central Control Unit (TUM, 2015).

Under most typical CAN architectures, a controller only has the logical capability to either process or ignore a CAN message based on the 11-bit identifier created by the originating controller. The Automotive Service Bus architecture adds a lot of new logical capabilities by overseeing all controllers rather than relying on the controller to "think" for itself. The Automotive Service Bus brings about a new syntax which all controllers on the network must adhere to. Messages are broken down into three different types: "events", which provide information such as speed or position; "commands", which allow components to interact with one another; and "preferences", which include driver-specific information such as music settings, addresses, or seat position (TUM, 2015). Because all controllers must adhere to this syntax, a device which would be expected only to generate "events" – such as a Tire Pressure Monitoring Sensor (TPMS) would not be allowed to issue "commands" to control the vehicle. This significantly reduces the attack surface of the vehicle. Furthermore, the Automotive Service Bus' Central Control Unit places all controllers in a read-only state by default, only issuing write capabilities to a specific controller when clearly defined cases arise (TUM, 2015). In this way, even if an attacker is somehow able to subvert controller whitelisting and gain unauthorized access to the network, the attacker would still be unable to issue command messages.

3.3. Automotive Control 3: Limitation and Control of Network Ports, Protocols, and Services

Manage (track/control/correct) the ongoing operational use of ports, protocols, and services on networked devices in order to minimize windows of vulnerability available to attackers.

(Center for Internet Security, 2015) (CSC 9).

3.3.1. Why It Matters

An open port and unsecured service represents an attractive entry point for an attacker. Some of the most serious data breaches began with the exploitation of a default port left open or a service running with default security parameters. Failing to lock down ports and services can leave a system dangerously exposed. This is true not only for enterprise networks, but also for modern vehicle platforms.

In 2014, Chris Valasek and Charlie Miller published a white paper in which they analyzed the remote attack surfaces of various popular cars, including many from the 2014 model year. During their analysis of the 2014 Jeep Cherokee, they targeted the vehicle's built-in Wi-Fi hotspot as a possible entry point. The output of an nmap scan the duo ran against the Wi-Fi hotspot is shown below in *Figure 5*. The nmap output shows multiple open ports, including port 6667 (commonly associated with IRC).

```
Starting Nmap 6.01 ( http://nmap.org ) at 2014-07-27 21:57 CDT
Nmap scan report for 192.168.5.1
Host is up (0.0022s latency).
Not shown: 998 closed ports
PORT     STATE SERVICE   VERSION
2021/tcp open  servexec?
6667/tcp open  irc?
```

***Figure 5*: Nmap Scan of a 2014 Jeep Cherokee's Wi-Fi Hotspot (Miller & Valasek, 2014b, p. 19)**

Miller and Valasek went a step further in 2015, demonstrating a full cycle attack on the 2014 Jeep Cherokee. In their white paper entitled "Remote Exploitation of an Unaltered Passenger Vehicle", the pair highlighted how this open port played a pivotal role in allowing them to gain access and compromise the vehicle.

As the pair performed further analysis of the vehicle, they found some nine different listening ports, as shown in the netstat output in *Figure 6* below.

```
# netstat -n | grep LISTEN
tcp        0      0  *.6010                 *.*                    LISTEN
tcp        0      0  *.2011                 *.*                    LISTEN
tcp        0      0  *.6020                 *.*                    LISTEN
tcp        0      0  *.2021                 *.*                    LISTEN
tcp        0      0  127.0.0.1.3128         *.*                    LISTEN
tcp        0      0  *.51500                *.*                    LISTEN
tcp        0      0  *.65200                *.*                    LISTEN
tcp        0      0  *.4400                 *.*                    LISTEN
tcp        0      0  *.6667                 *.*                    LISTEN
```

***Figure 6*: Netstat Output of Listening Ports (Miller & Valasek, 2015, p. 27)**

Through further research of the port numbers, Miller and Valasek were then able to determine which services were listening, as shown below in *Figure 7*.

Port	Service
2011	NATP
2021	MonitorService
3128	3proxy service
4400	HmiGateway
6010	Wicome
6020	SASService
6667	D-BUS session bus
51500	3proxy admin web server
65200	dev-mv2trace

***Figure 7*: Listening Services on Wi-Fi Hotspot (Miller & Valasek, 2015, p. 27)**

With nine listening services running on the vehicle's Wi-Fi hotspot, the pair were confident that they would be able to find a vulnerability. Focusing in on the D-Bus service running on port 6667, Miller and Valasek attempted to connect and found that the D-Bus service was configured to allow anonymous authentication (Miller & Valasek, 2015, p. 29). D-Bus is an Inter-Process Communication (IPC) mechanism which is used for communication between different processes, in this case pertaining to the vehicle's Uconnect head unit (Miller & Valasek, 2015, p. 28).

By using a GUI-based tool to debug the D-Bus service, the pair were then able to examine and decipher the methods and TCP procedure calls generated by the D-Bus service. They found several TCP method calls which could be used to directly interact with and manipulate the head unit (Miller & Valasek, 2015, p. 31). Within D-Bus, Miller and Valasek found a service named "NavTrailService" which included an "execute" method designed to execute arbitrary code (Miller & Valasek, 2015, p. 38). By exploiting

this service, the pair were able to execute their own arbitrary commands on the head unit, resulting in manipulation of the radio, display, GPS, and HVAC. Finally, the pair demonstrated that once access had been gained on the head unit, it was possible to pivot to the vehicle's CAN bus and issue commands which could impact the vehicle's engine, transmission, steering, and braking (Miller & Valasek, 2015, p. 57).

Miller and Valasek's work highlights a classic example of how a seemingly insignificant open port or unsecured service can provide the foothold an attacker needs to gain entry to the system before pivoting to more critical internal components. In the case of the 2014 Jeep Cherokee, the D-Bus service was most likely left unsecured because it was never intended to be user-accessible. D-Bus was only intended to communicate between processes and should never have been open to user interaction. But to have such a vulnerable service exposed via the Wi-Fi hotspot represents a serious security oversight.

3.3.2. Implementation

Locking down access to ports, protocols, and services first requires a clear understanding of what is running on the system. All running services, and all ports and protocols being utilized on the vehicle must be documented.

It may be found that on a finished automobile platform, many ports and services only exist for debugging and diagnostic purposes. In such cases, these ports should be closed and the services disabled prior to the vehicle being sold to a customer. These diagnostic services should only be accessible to service technicians or mechanics when the vehicle is placed into a special diagnostic mode. The diagnostic services and ports should never be open or accessible when a vehicle is being driven on the road.

Additionally, services and their associated ports which are intended only for internal communication between ECUs should not be user-accessible. In Miller and Valasek's Jeep Cherokee exploit, the D-Bus service was found to be running and port 6667 was found to be open by default. The fact that Miller and Valasek were able to authenticate to D-Bus anonymously via the vehicle's Wi-Fi hotspot represents a glaring security hole. D-Bus, or any other services responsible for internal vehicle

communication, should not be accessible to users connected to the Wi-Fi hotspot or any other external interface.

An appropriately positioned software firewall should be utilized on the vehicle's internal network to filter messages according to their source and destination. Messages originating from the vehicle's Wi-Fi, Bluetooth, or cellular services certainly should not be allowed to pass through to controllers on the internal network. The firewall should utilize a whitelist of approved services and ports when deciding which traffic to allow through. Firewalls and Intrusion Prevention Systems (IPS) are discussed more extensively below in Boundary Defense.

3.4. Automotive Control 4: Boundary Defense

Detect/prevent/correct the flow of information transferring networks of different trust levels with a focus on security-damaging data.
(Center for Internet Security, 2015) (CSC 12).

3.4.1. Why It Matters

The network boundary represents the first line of defense against an attacker. While no network should rely solely on boundary defenses to keep an attacker out, having boundary controls in place is an important part of a defense-in-depth strategy. Typically, an external attacker will focus on Internet-facing systems when seeking to gain a foothold on the target system. Attacking a system via the Internet allows an attacker to potentially gain entry through to the internal network without requiring physical access to the system.

In the last decade, the attack surface of the modern automobile has grown significantly. Ten years ago, hacking into the CAN bus of an average car would have required physical access to the vehicle's OBD-II port. Today, however, a typical modern car features a host of external interfaces including Wi-Fi, Bluetooth, GPS, and cellular 3G/4G connections. The boundary of the modern vehicle platform has become far more complex and more challenging to secure than ever before.

When Charlie Miller and Chris Valasek performed their attack against a 2014 Jeep Cherokee, they gained remote access to the vehicle via a vulnerability in Sprint's 3G

cellular network (Miller & Valasek, 2015, p. 43). Although the responsibility for this specific cellular network vulnerability lay with Sprint, the vehicle nonetheless failed to detect or block malicious code being sent to the vehicle via 3G. Furthermore, the duo pointed out that the Jeep Cherokee's Wi-Fi hotspot was also vulnerable to remote exploitation. The Wi-Fi hotspot would have been their fallback entry point had they not been able to exploit the cellular system (Miller & Valasek, 2015, p. 43).

The secure car, therefore, should be designed to include multiple layers of boundary defense. Ideally, the externally facing interfaces should be protected against incoming malicious traffic through the use of a firewall. An automotive firewall technology may be either hardware-based, residing on the network in front of the vehicle's ECUs, or software-based, running on the vehicle's central processing unit. Once again, the specific implementation of this control depends on the architecture of the vehicle. The best security solution is one which has been designed from the ground up, rather than being bolted on.

In addition to protecting the external interfaces, Boundary Defense must also consider the various boundaries which exist internally. Network segmentation should be utilized, whereby the vehicular network is broken down into different segments which are each logically and physically separated from one another. A real-time Intrusion Detection and Prevention System (IDS/IPS) should be employed wherever two network segments are bridged together. The IDS/IPS, which may either run on an existing ECU or on a dedicated piece of hardware, must be able to recognize malicious commands and prevent them from being broadcast on the CAN bus.

Through a combination of firewalls, IDS/IPS, and effective network segmentation, the modern, secure vehicle will be protected against the majority of known cyber-attacks which have been demonstrated against vehicular platforms to date.

3.4.2. Implementation

Effective boundary defense must be incorporated into the modern vehicle in the design phase. Any attempt at logical segmentation on an existing vehicle would be futile. Automakers must, therefore, embrace the concepts of secure network design when creating future automotive platforms.

The Automotive Service Bus, created by researchers at the Technical University of Munich (TUM, 2015), does an effective job of segmenting the automotive network. A diagram of the Automotive Service Bus, discussed earlier in this paper, is provided again in *Figure 8* below.

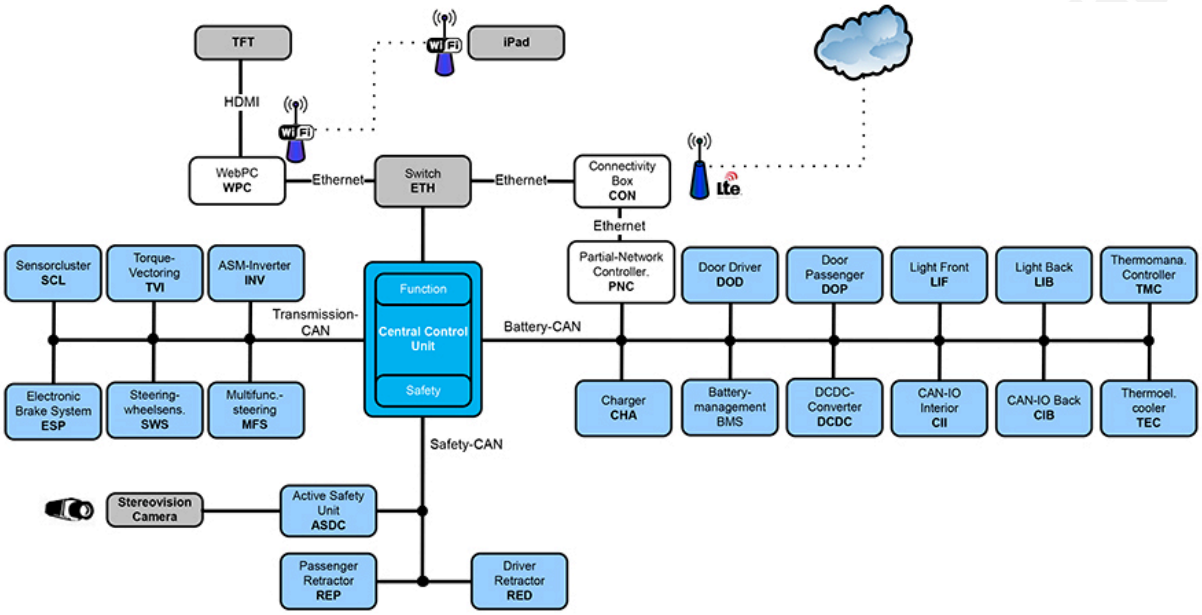

***Figure 8*: The Automotive Service Bus (TUM, 2015)**

As can be seen in the diagram, the CAN portion of the network is split across three separate segments: safety, vehicle control, and electronic functions. Components can still communicate across different buses, but they must go through the Central Control Unit in order to do so. The Central Control Unit, therefore, should be running a software-based firewall and software-based IDS/IPS. This ensures that even if one CAN segment is compromised, the attacker cannot pivot to the other CAN segments to cause more damage. Most importantly, however, the Automotive Service Bus architecture places all external interfaces on their own dedicated network segment, preventing direct access to the CAN bus from an external device.

In addition to network segmentation, automakers should incorporate firewalls and IDS/IPS systems as part of secure automotive system design. Following the publication of several high-profile car hacking demonstrations and white papers in recent years, a number of different third-party vehicle security solutions have been brought to market.

One such offering is the "Security Agent" from Arilou Technologies Ltd (Arilou Technologies, 2012). The Security Agent is essentially a plug-and-play CAN bus firewall component which can be placed on an automotive network at points where messages need to be filtered. The Security Agent can either be placed in front of an existing component on the bus to protect that specific ECU, or it can be placed between network segments as a gateway or bridge device (Cross Border Technologies, 2015). The device works by inspecting CAN messages, focusing in particular on the CAN packet's identifier field and the data field. The filtering logic is whitelist-based (Cross Border Technologies, 2015), meaning that the firewall can be configured to allow only approved message payloads to pass and only approved CAN identifiers to communicate on the network. By utilizing CAN identifier whitelisting, unauthorized devices connected to the bus will not be allowed to communicate with other vehicular components. The Security Agent can also monitor both inbound and outbound traffic, to ensure that neither externally-based attackers nor internal pivot attacks are allowed to succeed (Cross Border Technologies, 2015).

A similar solution being offered by Argus Cyber Security is the Argus IPS (Argus Cyber Security, 2014). The Argus IPS is an Intrusion Prevention System which actively prevents a vehicle from being hacked in real-time, while also offering reporting and alerts for remote monitoring (Argus Cyber Security, 2014). Argus has patented a Deep Packet Inspection (DPI) algorithm (Argus Cyber Security, 2014) which scans all traffic on a vehicle's network and engages in real-time response against abnormal traffic. As with other similar offerings, the Argus IPS can be integrated into existing vehicle platforms without the need for a major change in architecture. Argus also sends regular over-the-air (OTA) updates to its IPS units to counter the most current cyber threats (Argus Cyber Security, 2014).

With the availability of the solutions outlined above and many other comparative offerings available on the market, the technology already exists for automakers to implement boundary defense technologies on new automobile platforms.

3.5. Automotive Control 5: Controlled Access Based on the Need to Know

The processes and tools used to track/control/prevent/correct secure access to critical assets (e.g., information, resources, systems) according to the formal determination of which persons, computers, and applications have a need and right to access these critical assets based on an approved classification. (Center for Internet Security, 2015) (CSC 14).

3.5.1. Why It Matters

The concept of "need to know" is based on the fundamental idea that, in any information system, some pieces of information should be more accessible and obtainable than others. In the military world, a classification system is used to label data according to its sensitivity level; military data may be labeled as *Unclassified, Restricted, Confidential, Secret, or Top Secret*. Similarly, many corporate enterprises develop their own classification schemes to protect sensitive data. Under such a system, data of a higher sensitivity level is protected with a greater level of security controls. And the internal network of a modern automobile should be handled no differently.

Multiple teams of security researchers have proven that gaining remote access to an automobile is not only possibly, but is sometimes even trivial. When designing a secure vehicle, automakers should assume that an attacker *will* be able to gain remote access to the vehicle. Once an attacker has gained access, his attention then shifts to what specific data he has access to and what capabilities he can leverage. In a well-secured vehicle, the attacker may find that although he is able to join the vehicle's network, he ideally should not be able to access any of the vehicle's ECUs or manipulate the functions of the vehicle in any way. Unfortunately, however, the reality is that in most modern vehicles on the road today, an attacker can easily gain access to even the most sensitive components of the vehicle with little effort.

3.5.2. Implementation

The first step in implementing CSC 14 on a modern automobile platform is to classify controllers and messages based on their sensitivity level. This step should be carried out in conjunction with CSC 1 (Inventory of Assets). It also makes more sense to

implement this control during the network design phase, so that controllers of the same classification level may be logically grouped together.

For example, the controller responsible for the driver's seating position should be considered to have a low level of sensitivity and criticality. If this controller were to be compromised, an attacker may be able to remotely move the driver's seat. While unnerving, this would not immediately endanger the driver, and would allow sufficient time for the driver to stop the vehicle safely and shut the vehicle down. The ECU that controls the seat position should share a CAN bus with other non-sensitive ECUs such as those responsible for the power windows, the HVAC, the radio, and so on. However, these controllers should have absolutely no means of direct access to the ECUs responsible for the engine, brakes, transmission, or safety systems. This is where the enforcement of "need to know" comes into play; if a given controller does not have a need to manipulate the vehicle's steering, for example, then it should not reside on the same bus as the steering controller and should not have an unsecured path through to the steering controller.

Similarly, "need to know" should be considered when planning how to implement encryption on the CAN bus. For sensitive components, encryption of the CAN identifier field can prevent an attacker from being able to decipher the controller ID and spoof the controller. In the case of commands being sent to the steering controller, it certainly makes sense to encrypt the CAN identifier. If an attacker were able to intercept and spoof steering commands, they could essentially run a vehicle off the road with disastrous results. However, an automaker might decide that commands being sent to the HVAC controller need not be encrypted. After all, if an attacker managed to decipher and then spoof an HVAC command, the worst they could do might be to lower the A/C fan or turn up the heat. This assumes, of course, that the HVAC controller has been segmented in such a way that the attacker cannot pivot to more sensitive controllers.

While it may seem tempting to designate all controllers and all messages as highly sensitive and highly critical, it is important to remember that the auto industry is very cost-sensitive, and that weight and complexity are also important factors in the design of a modern automobile. Therefore, classifying controllers and messages allows

automakers to focus security controls where they are needed the most. Although the critical vehicle control systems should reside behind a firewall and be protected by an IDS/IPS, it might not make financial sense to place these same security controls on the non-critical bus responsible for comfort and convenience functions. This is why it is necessary to consider the "need to know" when designing a secure vehicle from the ground up.

3.6. Extra Credit: Penetration Tests and Red Team Exercises

> *Test the overall strength of an organization's defenses (the technology, the processes, and the people) by simulating the objectives and actions of an attacker.* (Center for Internet Security, 2015) (CSC 20).

It is a fact that the security controls automakers put in place are going to be tested by attackers. And in some cases, new vulnerabilities may be discovered in the process. Automakers have the choice of either testing their own security measures and discovering new vulnerabilities internally, or allowing external attackers to breach their vehicles in the production world. The consequences of failing to conduct internal penetration tests can be severe. If a vulnerability in a vehicle platform is discovered by a real-world attacker with malicious intent, the consequences could be potentially deadly for the occupants of the vehicle and catastrophic for the reputation of the manufacturer.

When Charlie Miller and Chris Valasek discovered a series of critical vulnerabilities in the 2014 Jeep Cherokee, Fiat Chrysler Automobiles (FCA) got lucky. That incident prompted FCA to recall some 1.4 million vehicles to perform a critical security update, costing in excess of $10 million in labor hours alone (Cobb, 2015). Nonetheless, FCA was lucky that the vulnerabilities had been discovered by security researchers and not by a real attacker. FCA was lucky that the disabling of a vehicle's transmission and brakes at highway speeds was only part of a controlled demonstration, and did not result in serious injury or death for the vehicle's occupants (Greenberg, 2015a). While the recall costs and reputation hit from the public release of this compromise were painful, the outcome could have potentially been so much worse. It is, therefore, in automakers' best interests to discover vulnerabilities in-house before a product is publicly released.

A penetration test against a vehicle platform should mimic a real-world attack scenario, using much the same methodology as was demonstrated by Miller and Valasek in their recent research. Penetration testers should begin by probing for remote access points, which could include vulnerable ports or services running on the Wi-Fi, Bluetooth, or cellular interfaces. Regardless of whether a remote access vulnerability is actually found, the test should also focus on internal vulnerabilities. Testers should operate under the assumption that a real attacker would be able to breach the perimeter of the vehicle's network, and should then focus on probing the internal network for vulnerable services and pivot opportunities. When vulnerabilities are discovered and patched before a product is publicly released, automakers can save significant amounts of money and avoid the reputation damage associated with vehicle recalls.

Automakers would do well to recruit security researchers to participate in red team exercises. There has, to date, been a rather icy relationship between automakers and the security research community, with automakers tending to respond with litigation rather than embracing the efforts of researchers. In one recent case, Volkswagen engaged a team of European security researchers in a 2-year long legal battle to prevent the group from presenting its research paper on a vulnerability they had found in Volkswagen's remote keyless entry system (Cimpanu, 2015).

Recently, some of the more progressive automakers have realized the importance of partnering with the security community in an effort to join forces and create more secure vehicles. Last year, Tesla Motors hired Internet-famous hackers Kristin Paget and Chris Evans to assist in locking down its own vehicular systems (Lambert, 2015). Tesla's "bug bounty" program also resembles those of tech giants like Facebook and Google, in which researchers who find exploits are financially rewarded (Greenberg, 2015b). And Charlie Miller and Chris Valasek, long the bane of the auto industry, now find themselves as part of it. Uber, the ride-sharing company, recently hired the pair to help develop the company's future fleet of self-driving cars and keep them secure from attackers (Greenberg, 2015c). Automakers are slowly beginning to realize the importance of having an in-house penetration testing team to discover and mitigate vulnerabilities before products go live.

4. Conclusion

The modern automobile is a highly vulnerable computing platform. But it can be secured. Through a concerted effort between the auto industry and the security research community, the car of tomorrow can be safer and more secure than ever before. Although the automobile platform presents many unique challenges for security professionals, many of the same principles which apply to securing enterprise networks can also be applied to vehicles.

The Critical Controls offer proven security guidance based on current, real-world cyber threats. This makes them well-suited to being applied to the modern vehicle platform. This paper outlined the "Automotive Top 5" Critical Controls which should be applied to create a secure baseline automotive platform. However, there are many additional controls which can also be applied to further improve the security posture of the modern automobile. The specific implementation of the Critical Controls described in this paper promotes a defense-in-depth approach which forms multiple layers of security to thwart even the most determined attacker. The ultimate goal of this approach is to create an automobile platform which vehicle owners can trust, knowing they are safe from cyber-attack.

References

Argus Cyber Security. (2014). *Argus Solutions*. Retrieved February 10, 2016, from http://argus-sec.com/solutions/

Arilou Technologies. (2012). *Car Hacking*. Retrieved February 10, 2016, from http://www.ariloutech.com/#!car-cyber-security/c1qhz

Center for Internet Security. (2015). *The CIS Critical Security Controls for Effective Cyber Defense, Version 6.0*. Retrieved January 17, 2016, from http://www.cisecurity.org/critical-controls.cfm

Cimpanu, C. (2015). *Volkswagen Sued Researchers for 2 Years to Prevent Them from Publishing a Security Flaw.* Retrieved November 11, 2015, from http://news.softpedia.com/news/volkswagen-sued-researchers-for-2-years-to-prevent-them-from-publishing-a-security-flaw-489347.shtml

Cobb, S. (2015). *Cybersecurity and manufacturers: what the costly Chrysler Jeep hack reveals.* Retrieved November 15, 2015, from http://www.welivesecurity.com/2015/07/29/cybersecurity-manufacturing-chrysler-jeep-hack/

Cross Border Technologies. (2015). *CAN Bus Firewall – Cyber Security for Automobiles*. Retrieved February 10, 2016, from http://www.crossborder-technologies.com/node/85

Fortin Electronic Systems. (2006). *What is CAN Bus?* Retrieved November 12, 2015, from http://canbuskit.com/what.php

Foster, I., & Koscher, K. (2015). *Exploring Controller Area Networks*. Retrieved February 8, 2016, from http://cseweb.ucsd.edu/~idfoster/papers/login_dec15_02_foster.pdf

Greenberg, A. (2015a). *Hackers Remotely Kill a Jeep on the Highway—With Me in It.* Retrieved November 15, 2015, from http://www.wired.com/2015/07/hackers-remotely-kill-jeep-highway/

Greenberg, A. (2015b). *5 Lessons from the Summer of Epic Car Hacks*. Retrieved November 16, 2015, from http://www.wired.com/2015/10/five-car-hacking-lessons-we-learned-this-summer/

Greenberg, A. (2015c). *Uber Hires the Hackers Who Wirelessly Hijacked a Jeep*. Retrieved February 11, 2016, from http://www.wired.com/2015/08/uber-hires-hackers-wirelessly-hijacked-jeep/

Guo, H., Ang, J., & Wu, Y. (2009). *Extracting Controller Area Network Data for Reliable Car Communications*. Retrieved February 7, 2016, from http://www1.i2r.a-star.edu.sg/~guohq/PDF/383.pdf

Hietala, J. (2013). *Implementing the Critical Security Controls*. Retrieved January 21, 2016, from http://www.sans.org/reading-room/whitepapers/analyst/implementing-critical-security-controls-35125

Lambert, F. (2015). *Tesla hired Chris Evans from Google's Project Zero to lead the company's security team*. Retrieved November 16, 2015, from http://electrek.co/2015/08/06/tesla-hired-chris-evans-from-googles-project-zero-to-lead-the-companys-security-team/

Miller, C., & Valasek, C. (2014a). *Adventures in Automotive Networks and Control Units.* Retrieved November 11, 2015, from http://www.ioactive.com/pdfs/IOActive_Adventures_in_Automotive_Networks_and_Control_Units.pdf

Miller, C., & Valasek, C. (2014b). *A Survey of Remote Automotive Attack Surfaces*. Retrieved February 8, 2016, from http://www.ioactive.com/pdfs/IOActive_Remote_Attack_Surfaces.pdf

Miller, C., & Valasek, C. (2015). *Remote Exploitation of an Unaltered Passenger Vehicle.* Retrieved November 15, 2015, from http://illmatics.com/Remote%20Car%20Hacking.pdf

National Instruments. (2014). *Controller Area Network (CAN) Overview*. Retrieved November 27, 2015, from http://www.ni.com/white-paper/2732/en/

Pike, L., Sharp, J., Tullsen, M., Hickey, P., & Bielman, J. (2015). *Securing the Automobile: A Comprehensive Approach*. Retrieved January 25, 2016, from http://www.galois.com/~leepike/pike-car-security.pdf

Redbend. (2015). *Cost Effective Updating of Software in Cars From IVIs, TCUs and Domain Controllers to the Entire Vehicle*. Retrieved January 20, 2016, from http://www.redbend.com/data/upl/whitepapers/Cost%20Effective%20Updating%20of%20Software%20in%20Cars%20Whitepaper.pdf

Sewell Direct. (2015). *A Brief Explanation of CAN Bus.* Retrieved November 12, 2015, from https://sewelldirect.com/learning-center/canbus-technology

Trumpbour, M. (2015). *My sister-in-law's husband just got this USB in the mail for his Jeep*. Retrieved January 21, 2016, from https://twitter.com/mtrumpbour/status/639477023371624448

TUM. (2015). *Visio.M Automotive Service Bus goes open source*. Retrieved February 7, 2016, from http://phys.org/news/2015-03-visiom-automotive-bus-source.html

United States Government Accountability Office. (2011). *NHTSA Has Options to Improve the Safety Defect Recall Process*. Retrieved January 20, 2016, from http://www.gao.gov/assets/320/319698.pdf

Wojdyla, B. (2012). *How it Works: The Computer Inside Your Car.* Retrieved November 12, 2015, from http://www.popularmechanics.com/cars/how-to/a7386/how-it-works-the-computer-inside-your-car/