Automobile CAN Bus Network Security and Vulnerabilities

Shawn Hartzell and Christopher Stubel
University of Washington
Seattle, Washington

***Abstract* - The modern automobile uses a network of Electronic Control Units (ECUs) to implement systems such as door locks, anti-lock brakes, and engine control. Several bus protocols have been used in the implementation of these systems, with the most popular being the CAN bus. The CAN bus has been designed with security, but its accessible design has many vulnerabilities. The transition from isolated mechanical systems to networked electrical systems has enhanced the performance of automobiles but has made them more vulnerable to network attackers. In this paper, we will discuss the vulnerabilities of the CAN bus. Based on our research we will propose several security concepts, and finally discuss why security in the future is important, and why it should be an upfront requirement while designing vehicles. As vehicles become more connected to wireless networks, CAN bus security becomes more important.**

I. INTRODUCTION

The Controller Area Network (CAN) bus was publicly released in 1986 by the Society of Automotive Engineers (SAE) for in-vehicle networks [1]. As vehicles added more complex electrical systems, point to point electrical wiring became heavy and expensive. The CAN bus provided a simple network to reduce wiring, and allow multiple microcontrollers to communicate on a single bus. In recent years, the number of ECUs has grown exponentially. These ECUs rely on the CAN bus to communicate between microcontrollers and sensors. Many systems and functionality of the common vehicle are being converted from mechanical to electrical systems due to increased reliability and technology advancements. New features of vehicles are being continuously being added for safety or convenience, such as blind spot sensors, remote start, adaptive braking and lane assist, and adaptive cruise control.

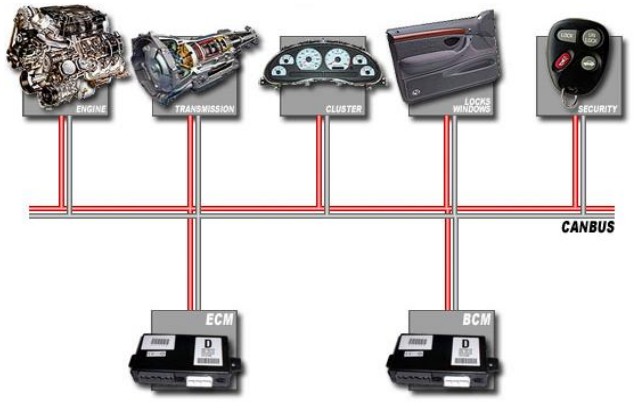

Figure 1. CAN bus use in an automobile [11]

The CAN bus acts as the backbone for many systems in the car, and signals are multicast over the bus. Safety critical vehicle information such as engine control, door locks, anti-lock braking, and cruise control, is passed on the CAN bus. The single bus consolidation of all these systems allows an attacker access to all these critical functions once access to the bus is obtained. This can result in a safety hazard or compromise of personal data or belongings. It is therefore necessary to address security as an upfront requirement while developing systems in vehicles.

In our research, we found that CAN bus networks are vulnerable to malicious or passive attacks. The CAN bus architecture is designed to be a stable and flexible network from a communication and hardware standpoint, but lacks robust security against attacks. Future vehicles are heading to autonomous driving making it critical for vehicle manufacturers to address security upfront in the design. If not the consequences could result in a failure of the critical functions described above compromising the safe operation of the vehicle. After a summary of the vulnerabilities which exist in the CAN bus we will propose high level suggestions on making the CAN bus architecture more secure. Finally, we will give reasons why security in vehicles is important looking forward.

II. CAN BUS ARCHITECTURE

The CAN bus is a multi-master differential communication system. This is a simple architecture which is shown in Fig. 2. The messages are multi-cast, meaning every microcontroller and component connected to the CAN bus receives each message. The messages are sent and received on a regular synchronized time basis. The CAN bus architecture allows for every ECU on the bus to listen to the multicast messages. This design allows for multiple systems, designed by multiple companies to be integrated together. This design is robust from the standpoint that if one node[1] fails all others nodes are still operational. Usually multiple CAN bus networks exist in a car and are linked together by a common gateway. The gateway can be physically accessed by the OBD-II port of a vehicle. Although the CAN bus can support a data rate of up to 1 Mbps [1], most vehicles have a data rate of 500 kbps. The CAN bus also has a cyclic redundancy check (CRC) to ensure the integrity of the message is valid upon receipt at any node.

[1]Node: Any point with an ECU connected to the CAN bus.

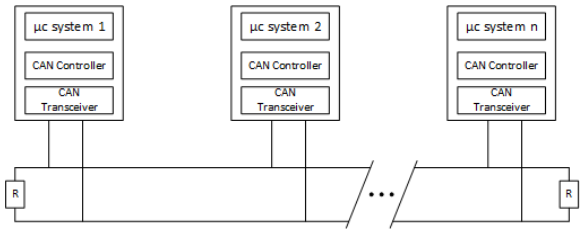

Figure 2. High level CAN bus architecture

A. *Assets and Associated Risks:*

The CAN bus hosts many assets critical to the vehicle's proper and safe operation. Thus, these signals are important to the vehicle's owner and all environments in which the vehicle operates. Usually a car has a specific CAN bus for the operation and communication of the powertrain. This includes transmission, gear shifting, speed, acceleration, and engine control. These are the most critical assets of the car, and should be secure to any outside passive[2] or malicious[3] attackers. Some other common assets on the CAN bus are signals relayed to the instrument cluster. Vehicle operators use this information to get an indication of how the car is performing. The instrument cluster also provides feedback to driver commands. If this data is manipulated it could mislead the driver and cause a safety concern. These assets are absolutely vital for the safe operation of the vehicle. If compromised by an attacker major safety situations could arise. Personal data sent on the CAN bus such as GPS information is also an asset that need to be protected as it can be used by attackers to track people or be used to create a profile of a person's travel habits. We identified three categories of assets.

- *Private Data:* Data private to the car owner. Includes GPS data, personal device uses, and personal device access, data related to driving habits.
- *Safety Critical Data:* Data which is critical to the safe operation of the vehicle. This includes engine management, anti-lock brakes, acceleration, etc.
- *Vehicle Security Data:* Data which could result in a breach of security of the vehicle without the owner's consent. This includes door locking functions, power windows, and remote start.

B. *Adversaries:*

There are many attackers which could want access to the assets which reside on the CAN bus. Passive attackers could monitor the bus giving them access to every message on the bus. This could be just to learn about the driver or this data could be stored, and attackers could use it for blackmail in the future. An example could be an insurance company or a transportation city planner. Insurance companies could use this data to determine a person's driving habits and decide to not insure them or raise their rates. City planners could use GPS data from vehicles to see general traffic flow on streets and determine where congestion should be addressed or to flag roads with consistent speeders. Thieves could use GPS data to know when a person is home and when they are not, making a break in easier. They could also know where the car will typically reside and when it is not being used, if they want to steal it. Malicious attackers could gain access to the CAN bus data to override motor controls, causing a collision resulting in injury or death. This could be an attack on a specific person or it could be an anarchist targeting a mass number of cars simultaneously.

[2]Passive Attack: A network attack in which a system is monitored and sometimes scanned for open ports and vulnerabilities. The purpose is solely to gain information about the target and no data is changed on the target.

[3]Malicious Attack: A network exploit in which a hacker attempts to make changes to data on the target or data en-route to the target, or inject data.

C. *Vulnerabilities:*

Our research shows that the typical CAN bus is extremely vulnerable to attacks in several key areas. In older cars which are not connected to external sources such as a wireless network, these vulnerabilities become harder to exploit due to the physical localization of the systems. In the future, connected cars have to treat these vulnerabilities seriously to protect from remote attackers. The vulnerabilities are listed below:

- *Multicast Messaging*: When a message is sent to the CAN bus, it has no specific destination. Every access point or controller on the bus has access to all messages. Passive attackers could listen in on the communication with ease. Listening could be achieved by tapping into the diagnostics port of the vehicle or inserting a malicious node onto the CAN bus.
- *Lack of Authentication*: The typical CAN bus has no authentication process. Nodes do not have a process in place to ensure the message they receive is from a valid source. This makes it very easy for a masquerade attack, a type of attack where someone pretends to be someone they are not.
- *Lack of Addressing*: Nodes typically have no identification address, which allows all (real or malicious) nodes to send or receive information without any verification that the source of information is valid.
- *Common Point of Entry*: Once an attacker has access to the CAN bus there is no limit to what parameters the attacker can obtain. Usually, one common gateway is used to connect all vehicle CAN bus systems. The diagnostics port in a vehicle for example allows access to all of systems. OBD-II Port.
- *Limited Bandwidth*: The CAN bus has a limited bandwidth. This creates a challenge to allow for any robust authentication process to be implemented. Complex authentication algorithms would not be supported by the low bandwidth of the CAN bus. The 11-bit and 29-bit identifier message structures of the CAN bus only allows for 64 bits of data in the payload, leaving little room for additional data for security purposes. Robust encryption algorithms

need much higher bandwidth and processing power.

- *Lack of Encryption*: CAN bus is designed for ease of use. Data over the network is not encrypted making aftermarket ECU manipulation easy. To implement a robust encryption of data the CAN bus would need a larger bandwidth than what is available currently.
- *Multi-System Integration*: Multiple suppliers integrate onto these CAN bus networks. Given that there is typically no security measures in the CAN bus standards, there is no objective by these system suppliers to create a secure communication protocol.

D. Threats:

Attackers could exploit these vulnerabilities with ease. The diagnostic port of a vehicle usually grants access to the gateway which connects all CAN bus (or other) network systems. This allows an attacker with physical access to a vehicle the ability to develop a library of system data, and allows them to manipulate that data. CAN bus nodes could be manipulated to inject malicious data or listen to the bus and send data to the attacker.

III. CAN BUS SECURITY

A Cyclic Redundancy Check (CRC) is used for detecting changes to data during transmission. It is also known as the safety field. The field consists of a 15 bit check code and a 1 bit delimiter bit. When data is sent, the 15 bit check value is determined based on the remainder of a polynomial division of the data. When the data is received, the division is repeated by the receiving device and compared to the original check value. If it matches, the receiving device sends a dominant state (0) in the acknowledge (ACK) bit slot which will overwrite the recessive state (0) of the sending device. If it doesn't match, the device sends a recessive state in the ACK slot and sends an Error Frame after the ACK delimiter bit [3]. CRCs are usually used to detect transmission errors caused by noise. We note that the sending device can't tell how many or which receivers correctly received the message, it will only know if nobody or at least one received it. CAN FD improves the CRC algorithm and reduces the number of undetected errors. To perform a cyclic redundancy check in the original CAN 2.0, a dividend polynomial was defined which has coefficients given by a bit stream consisting of the following fields: Start of Frame||Arbitration||Control||Data||and CRC which was preloaded with 15 zero bits. The dividend polynomial was divided by a 15 degree divisor polynomial defined as $x^{15} + x^{14} + x^{10} + x^8 + x^7 + x^4 + x^3 + 1$ [4]. The remainder of the division is the 15 bit CRC block sent to the receiving device. The receiving device divides the CRC block by the above divisor and should get a remainder of zero if there are no errors. This CRC produces a Hamming distance of h = 6 and can detect up to h - 1 = 5 bit errors per data block. The 15 degree divisor polynomial can support data block lengths up to $2^7 - 1 = 127$ bits. This is fine for CAN 2.0 which has a maximum length of 102 bits. CAN FD however can have data blocks of up to 690 bits and therefore needs a CRC polynomial divisor of 21 degrees ($2^{10} - 1 = 1023$ bits). This polynomial is defined as $x^{21} + x^{20} + x^{13} + x^{11} + x^7 + x^4 + x^3 + 1$ [5].

IV. EXISTING SECURITY PROPOSALS

During our research we found that one of biggest hurdles to implementing security on the CAN bus architecture is its limited bandwidth, 500kbps [1]. Studies and experiments in [2] [6] [7] show that an authentication process is possible, but the implementation is tightly constrained by the bandwidth and latency of the system architecture. The CAN bus 11 bit ID base frame format is made up of a total of 134 bits, consisting of a 64 bit payload, 46 bits for error checking, and 24 bits for bit stuffing. Any additional bit that are added for security would be added to the 64 bit payload. If the normal payload plus the security bits go over the 64 bit limit, then the message will have to be split into two messages. This results in higher bus utilization, possibly degrading performance. The CAN bus also lacks a global clock and time stamp so randomness can be hard to achieve.

One implementation for security is given in [2]. This implementation consists of setting up trusted communication groups within the CAN bus network, utilizing asymmetrical keys and a common gateway which acts as a Key Distribution Center (KDC). This requires a system change to existing cars due to the addition of a dedicated cryptographic ECU. This ECU would be in a central position of all CAN bus networks and generate temporary keys for each trust group, sending each key to all ECUs defined in the specific trust group. Each group is contained in an access control list which has certificates signed during the manufacturing process of the car. This Access Control List is contained in the KDC. The KDC then compares the responses from each ECU to ensure only the authorized ECUs communicate. This establishes private communication groups within the CAN bus. During the manufacturing process each ECU would be assigned a unique ID, and given a private key. The public key is also stored in memory at each ECU and the KDC. An initialization function of the car is used to start the authentication process. An example would be a vehicle door opening. The KDC would send a challenge to each node along with a key for its associated trusted group. Each ECU would respond to the KDC with an encrypted message with its private key answering the challenge. After all ECUs in a group are authenticated they then encrypt messages between each other for communication with the public key. Each node that belongs to multiple trusted groups would need to repeat the process for each group.

Another security implementation is intrusion detection. There are several methods of intrusion detection currently under development. One intrusion detection system (IDS) presented by [7] observes message frequency to detect attacks. ECUs connected to a CAN bus periodically send data at a specified interval. Under normal conditions, the time between messages with the same CAN ID is the same. For each message that appears on the bus, the IDS checks it's CAN ID and calculates the time since this ID was last seen.

TABLE I. ATTACKER PROFILES

Attacker Profile	Probability of Access to OBD- II	Example of Potential Threat	Seriousness of Threat
Thief	Low	Injecting software to forward data of the vehicle to the thief	High
Mechanic	High	Inject harmful data into the bus affecting safe operation	Medium
Car Owner	High	Accidentally injecting incorrect critical data resulting in unsafe operation	High

If the time interval of the new message is shorter than normal, it assumes the message is part of an injection attack. It can also recognize DoS attacks, when the time interval of the new message is significantly shorter than usual. This IDS proposed is lightweight and fast. A typical time interval of ECU CAN bus messages is 0.5 milliseconds. The response time of the IDS for a typical ECU would therefore be a value less than 0.5 milliseconds, plus the calculation time of the simple interval subtraction. Testing in [7] using this method shows a 100% accuracy rate with no false positives.

Another IDS method is described in [9] which uses specification based detection. Specification based systems have a correct behavior defined by the system policy and the expected system usage. Any observed deviation from this behavior is flagged. This type of detection works well for the vehicle CAN bus because the ECUs have limited set tasks and there is rarely any deviation from standard CAN protocol. The structure of a message is examined to determine unexpected values. Individual fields in a message should contain specific content or fall in a value range. For example the first three bits of a server command specifier should only contain a number between 0-3 and 5-6. Field value ranges of certain messages are also dependent on previous related messages. If the supposed response to the previous message contains unexpected values, it is flagged.

V. PROPOSED CAN BUS SECURITY IMPLEMENTATION

Given the architecture of the CAN bus protocol, we propose a simple authentication process to address some of the vulnerabilities listed in Section 2. It builds on some of the existing ideas found in [2] [6] [7]. Authentication would help prevent malicious attacks but would not stop passive attacks by itself. Our solution specifically protects against malicious masquerade attacks through the ODB-II port. Our definitions and assumptions are below

- *Masquerade Attacks through ODB-II port:* A type of attack where an attacker pretends to be an authorized user of a system in order to gain access or to gain greater privileges than they are authorized for. The attacker may want to disguise themselves as a node on the system or inject data to cause a Denial of Service (DoS) attack.
- *Assumption 1:* All ECUs are connected to one CAN bus network.
- *Assumption 2:* The security node is connected to all nodes.
- *Assumption 3:* All ECUs receive information from the speedometer.
- *Assumption 4:* There is no path for the attacker to gain access to the security key stored in the security node.
- *Attack Model:* Our model of an attacker is someone who has access to the OBD-II port inside the car. Thus the attacker can directly access the CAN bus network of the vehicle. The attacker disguises themselves to be a valid node on the system gaining access to critical data on the CAN bus. The attacker's goal is to communicate with valid nodes or block data to valid nodes. In Table I we show the probability of an attacker gaining access to the OBD-II port.

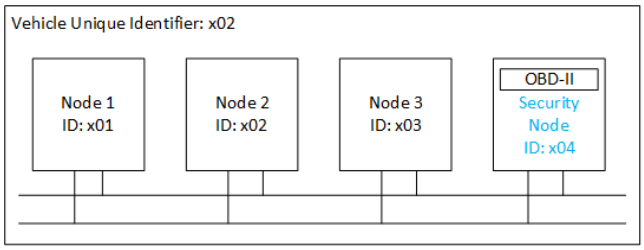

Figure 3. Simple block diagram of attack environment

The security improved would be classified as an authentication process with symmetric keys. Our proposed security method would be to implement a physical security node which is connected to the OBD-II port, shown in Fig. 3. This security node would be able to physically break the connection of the OBD-II port for the CAN bus network upon a failed attempt at authentication. This would reduce the time an attacker would have access to the OBD-II port. The security node would be an improvement to the security methods described in [6] to limit malicious attacks while the OBD-II port is connected to the system. Once an OBD-II port has a connection the security node would enable the authentication process to start. Our view was that the study performed in [6] did not address the OBD-II port, and might even make current standard car maintenance impossible.

The security method in [6] is very beneficial for the amount of bandwidth and processing power it takes. Each node contains an identification table to compute MACs. Each node creates a unique MAC for each receiver of its messages. This will also allow each node to manage how many MACs it needs to create and send in each message, authenticating each message with an efficient use of bandwidth.

This method also introduces a pairwise secret key [6]. This is very similar to the Diffie-Hellman authentication process between two communication parties. Table II gives an example of the Diffie-Hellman key exchange. Nodes *i* and *j* would be pre-loaded during production with a private key.

TABLE II. DIFFIE-HELLMAN EXAMPLE

Shared Key Process Steps	**Node$_i$**	**Node$_j$**
1	Generates random value *i*. $I = g^i \pmod p$	Generates random value *j*. $J = g^j \pmod p$
2	sends Node$_j$ message *I*	
3		sends Node$_i$ message *J*
4	$Kij = J^i \pmod p$	$Kij = I^j \pmod p$
5	Communicating $E_k(M_{ij})$	Communicating $E_k(M_{ij})$

The shared communication key K_{ij} would be the key used to by two nodes to communicate messages. This would be a symmetric key which is more efficient than public keys. This allows the keys to be private, and only known to the manufacturer. We propose that each node which needs the potential to be accessed for maintenance have a unique key to authenticate communication with the security node. The security node now has a unique key that has been generated by the manufacturer and is unique to each vehicle. The security node would need to have an interface that allows user input of the specific key. This way a mechanic would have to get the secret key of the OBD-II port from the manufacturer. The mechanic would only be able to communicate with the necessary nodes for maintenance. We would also propose that each vehicle off the production line has a unique security key. This would only allow mechanics to have the key for the vehicle which they are working on. It would then be easy to keep record of when the CAN bus was accessed. The difficulty of this is that the keys cannot be large in size due to the bandwidth and timing limitations of the CAN bus.

Finally, this method implements a counter function [6]. The counter is used to prevent masquerade attacks in which the attacker tries to replay a message of an authenticated node to get another node to accept a message from them or to get a node to reject a valid message. This is done by keeping a counter at the receiver node and sending nodes. In each message between nodes, M_{ij}, the counter value will be sent in the payload. The receiving node will check that the counter value sent in the message matches the counter value in the node's memory. In our proposed improvement this would help ensure that the mechanic only sends valid messages when on the CAN bus preventing them from pretending to be any other node besides the maintenance node. The sending and receiving process is given in [6]. For the maintenance portion the counter challenge also allows a random challenge that needs to be answered every time entrance is attempted.

Table III is a slight modification of the sender node from [6] to be compatible with the maintenance port. The difference is that the maintenance port would only talk to the security node. The security node authenticates the user by the user providing the correct ID which is the access key. After this the security node manages all commands sent and

TABLE III. MODIFICATION OF SENDER TABLE IN [6]

	Security Node N$_i$
1	User obtains vehicle unique OBD-II port key, *i* from manufacturer, entering the ID to gain access to the security node.
2	(i, nk = 1, rk,s) = ID-Table(k); Security Node verifies the user has entered the correct key.
3	If the key does not match, prompt user for reentry up to three times. Upon failure physically break link of the OBD-II port for a long duration of time.
4	(i, nk, rk,1, rk,2, . . . , rk,nk) = ID-Table(k); Stored ID table
5	Ci,k = Ci,k + 1; set counter
6	$\forall$ `s, 1 ≤ s ≤ nk,Ak,s = f(Mk, Ci,k,Ki,rk,s ); authenticate outgoing messages.`
7	Send Mk, Ci,k,Ak,1,Ak,2, . . . , Ak,nk; normal communication

retrieved from the maintenance port. This limits the access to only the necessary parts of the CAN bus. The security node then would take care of all the authentication process itself and all the user would do is enter the desired normal system functionality into the CAN bus.

This would also work well because the additional security methods only work well up to a three-node system. This is explained in [6]. With three nodes, a probability of attack of .001 and a probability of counters being out of synchronization (due to a communication fault) of .0001 causes the bandwidth to rise to 422.817 kbps. In our case the security node could only communicate with two other nodes at a time while staying under the bandwidth of 500kbps.

There are some drawbacks to this security method. From the manufacturer's standpoint, more money would need to be invested into the production of the car. Production time would increase as the keys would need to be generated and tested before delivery of the vehicle. The manufacturer would also need to keep a database of all the vehicles and the associated keys. A program would be ideal for the Mechanic to access to obtain a vehicles key. This would make the mechanic enter their credentials and upon success give them the key for the specific vehicle. This would create another area where an attacker could gain access to a vehicles security key. Another area of concern is it could cause mechanics repair time to be longer. Since the mechanic only has access to a couple nodes at a single time, they might have to reauthenticate multiple times. This would make them repeat the request process with the security node increasing car turnaround time for the owner.

Overall this simple solution would make the OBD-II port more secure from our attack model. A vehicle owner that wants to tune their car could obtain the key from the manufacturer, but this voids the warranty. Car manufacturers recommend that the owners do not use the maintenance port

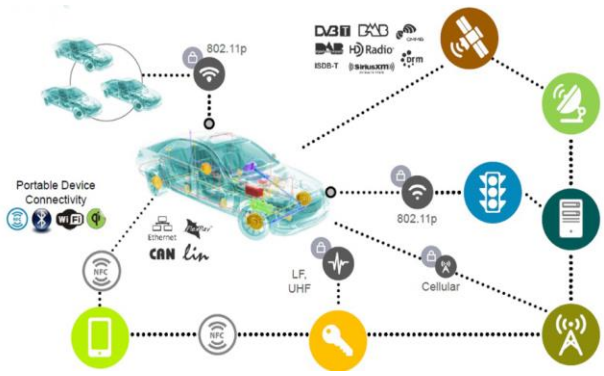

Figure 4. Future vision of connected vehicles [12]

for modifications. The thief has no fast way of obtaining the unique maintenance port key to authenticate themselves. The probability of them access the ODB-II port now is very low. Finally, the mechanic can only access the ports necessary for maintenance and no longer communicate with every ECU on the bus. Their access is also tracked.

A. Future Security of CAN Bus:

This solution does not address all the vulnerabilities in section two. In fact, it only address having a physical common access point the CAN bus. We would not recommend this solution to be implemented for autonomous vehicles in the future. Due to the architecture of the CAN bus and its limitations, system redesign should be a discussion vehicle manufacturers have. Some new protocols have already been introduced such as FlexRay [7]. FlexRay is more expensive than CAN bus but offers data rates up to 10Mbps. This would address some of the bandwidth limitations of the CAN bus, allowing for stronger encryption techniques to be used. FlexRay was first installed on a vehicle in 2006. Newer vehicles implement more safety critical functions on the CAN bus such as back up cameras, blind spot sensors, auto braking, and even autonomous driving. These functions need to be implemented on a secure network because drivers will rely on these functions for safe driving. Fake data injected onto these systems could result in an unintended reaction of the car. Additionally, these vehicles are being connected to cellular networks, Wi-Fi, Bluetooth, etc. A survey of the top 16 automobile manufacturers showed that 100% of their new cars for sale had some form of wireless communication [8]. Manufacturers are quick to add wireless features to appeal to customers, but slow to implement security on them. The last remaining security benefit the CAN bus had in its favor was being isolated from the outside world, but that benefit has been eliminated for new vehicles, and more access points vulnerable to attackers are being created. This is a serious concern which should be addressed with third party testing, legislature, or both. A future industry vision of a connected car, which we share is given in Fig. 4.

VI. CONCLUSION AND OUTLOOK

In our study we found out that the CAN bus is a vulnerable network. There are a lot of ideas on how to improve the security of the CAN bus, but big challenges such as bandwidth and latency need to be overcome. The proposed additional security node managing the connection and authentication of the OBD-II port would be an improvement to the current design. In the existing work examined, the authors were able to compromise the CAN bus after examining and understanding the network of the vehicle and creating a model of it. In the majority of cases they accessed the vehicle through the OBD-II port. Eliminating this point of entry would force attackers to enter the system using other means. It would prevent models from being easily created and published on the internet, reducing the number of potential attackers. If time allows we would continue investigating this security measure and simulating the results. As vehicles become more complex and electrical systems perform more safety critical functions the security of the OBD-II port should be addressed or the maintenance personnel or vehicle owner could cause a malfunction of the car. Manufacturers and their suppliers need to coordinate to design security into the vehicle from the beginning. As vehicles become more connected to outside networks for the purposes of firmware updates or multimedia, the CAN bus becomes more vulnerable. C. Miller and C. Valasek shows just how serious an attack on the CAN bus can be in [8].

REFERENCES

[1] S. Corrigan, “Introduction to the Controller Area Network (CAN),” SLOA101A–August 2002–Revised July 2008.
[2] A. Groll, C. Ruland, “Secure and Authentic Communication on Existing In-Vehicle Networks,” University of Siegen, Germany, 2009.
[3] "CAN in Automation (CiA)." CAN in Automation (CiA): Cyclic redundancy check (CRC) in CAN frames. N.p., n.d. Web. 12 Mar. 2017.
[4] Bosch. CAN Specification Version 2.0. Rep. N.p., Sept. 1991. Web. 12 Mar. 2017.
[5] Bosch. CRC for CAN with flexible data rate (CAN FD). Rep. N.p., n.d. Web. 12 Mar. 2017.
[6] C. Lin, A. Sangiovanni-Vincentelli, “Cyber-Security for the Controller Area Network (CAN) Communication Protocol,” University of California, Berkeley, 2012.
[7] H. Song, H. Kim and H. Kim, “Intrusion Detection System Based on the Analysis of Time Intervals of CAN Messages for In-Vehicle Network,” Korea University, Seoul, Republic of Korea, 2016.
[8] C. Miller and C. Valasek, “Remote exploitation of an unaltered passenger vehicle” in BlackHat USA, 2015.
[9] U. Larson, D. Nilsson, and E. Jonsson, “An approach to specification-based attack detection for in-vehicle networks,” Chalmers University of Technology, Goteborg, Sweden, 2008.
[10] S. Corrigan, “Introduction to the Controller Area Network (CAN),” Texas Instruments, 2008.
[11] Satkunas, Darren. "What is Can Bus?" What is CAN Bus? N.p., n.d. Web. 12 Mar. 2017. <http://canbuskit.com/what.php>.
[12] Bowers, Nat, “What autonomous driving technology is now available?,” Electronic Specifier, 6 Aug. 2014