In-Vehicle Networks: Attacks, Vulnerabilities, and Proposed Solutions

Paul Carsten
University of South Alabama
Mobile, Alabama USA
pwc1221@jagmail.southalabama.edu

Todd R. Andel
University of South Alabama
Mobile, Alabama USA
tandel@ southalabama.edu

Mark Yampolskiy
University of South Alabama
Mobile, Alabama USA
yampolskiy@ southalabama.edu

Jeffrey T. McDonald
University of South Alabama
Mobile, Alabama USA
jtmcdonald@ southalabama.edu

ABSTRACT

Vehicles made within the past years have gradually become more and more complex. As a result, the embedded computer systems that monitor and control these systems have also grown in size and complexity. Unfortunately, the technology that protects them from external attackers has not improved at a similar rate. In this paper we discuss the vulnerabilities of modern in-vehicle networks, focusing on the Controller Area Network (CAN) communications protocol as a primary attack vector. We discuss the vulnerabilities of CAN, the types of attacks that can be used against it, and some of the solutions that have been proposed to overcome these attacks**.**

Categories and Subject Descriptors

A.1 [**General Literature**] INTRODUCTORY AND SURVEY; C.3 [**Computer Systems Organization]** SPECIAL-PURPOSE AND APPLICATION-BASED SYSTEMS – *Real-time and embedded systems*; J.7 [**Computer Applications**] COMPUTERS IN OTHER SYSTEMS - *Consumer products*

General Terms

Security

Keywords

Automotive Vulnerabilities, CAN bus, In-Vehicle Networks

1. INTRODUCTION

Over the last 20 years, the computer systems embedded in automobiles have become increasingly complex. As demands for sophisticated vehicle systems increased, auto manufacturers introduced more and more computer-controlled systems into vehicles. In addition, systems like electric power steering, adaptive cruise control, and anti-lock braking systems required the removal of many of the traditional mechanical connections and their replacement with sensors. The monitoring of these systems and the necessity of communication between computers that monitor various mechanical devices led to the growth of the in-vehicle network.

However, such advances did not come without a cost. By removing the traditional mechanical connections once present in vehicles and replacing them with sensors on communication buses, the manufacturers created a system in which drivers “drive by suggestion”, or convey their intent to the sensors which then perform the actual actions requested. In the isolated environment in which these systems were originally designed, this is an acceptable decision. Unfortunately, the environment can no longer be considered isolated.

Modern vehicles are vulnerable. The environment in which modern vehicles transmit information is not secure, and it is susceptible to attack. If an adversary were to gain access to the network, modern vehicle networks would have little power to stop them, and a variety of attacks could be performed, some of which could be dangerous, even fatal, to both car and driver.

This paper aims to provide an understanding of the way in which modern vehicles behave, to present the vulnerabilities of one of the most widely used of the protocols which govern communication, to describe some of the potential attacks that could be carried out against an in-vehicle network, and to describe some of the proposed solutions to this tenuous situation.

2. BACKGROUND

In order to understand the vulnerabilities of a system, one must first know how that system works. Modern in-vehicle networks could be compared to a SCADA (Supervisory Control and Data Acquisition) system. Such systems use a single device to gather and process information from an array of sensors connected to various devices [1]. The aforementioned devices for in-vehicle networks are simple computer modules alternately called *engine control units* or *electronic control units* (more commonly referred to as ECUs). These ECUs are associated with sensors that monitor

CISR '15, April 07 - 09, 2015, Oak Ridge, TN, USA Copyright 2015 ACM 978-1-4503-3345-0/15/04…$15.00
http://dx.doi.org/10.1145/2746266.2746267

a wide variety of systems in a vehicle, from entertainment systems to those that govern the control of the vehicle. A typical vehicle can have around 50-70 ECUs [2] all of which are in contact over a series of communication buses. These communications are governed by one or more communication protocols, including the following:

Controller Area Network (CAN): One of the most widely used protocols, CAN is responsible for many systems in a modern vehicle, including vehicle control, safety, and electrical systems. It operates like a broadcast network in which packets are conveyed to all nodes on the network and it is the responsibility of the nodes to determine whether or not to process them. CAN packets have CAN IDs to identify packet content, but their packets do not have sender or receiver addresses.[3]

FlexRay: "A deterministic and error-tolerant high-speed bus," [3] FlexRay has been proposed as a possibly more secure successor to CAN. It behaves in a similar way to CAN and deals with many of the same systems, but it features a system of *communication cycles* which can add some measure of identification of the sender and intended receiver (because nodes can identify packets relevant to them based on the communication cycle). It features synchronous and asynchronous transmission and handles errors through redundancy, checksums, and a bus guardian that is designed to handle logical errors [3].

Local Interconnect Network (LIN): A single wire subnetwork for lower bandwidth transmissions. Designed for use when CAN's versatility is not required. As with CAN, the responsibility to identify whether or not the packets are relevant falls to the receiver [3].

Media Oriented System Transport (MOST): The primary system for multimedia transport in modern vehicles, the ISO/OSI standardized MOST protocol stands apart from the other protocols mentioned primarily because its packets include a sender and receiver address. As with FlexRay, MOST features a configurable time access system. Error management is handled through an internal system service that detects errors through parity bits [3].

These are not the only protocols extant in modern vehicles. In addition, most modern vehicles have multiple protocols in use. One of the greatest challenges to modern vehicle network security is the interconnected nature of these protocols.

For the purposes of this paper, the primary protocol considered will be CAN due to widespread use and the ease with which its vulnerabilities can be understood. Some of the vulnerabilities mentioned here can be applied to other protocols as well. In addition, the interconnected nature of the protocols means that as long as CAN is in widespread use, these vulnerabilities will exist, regardless of measures taken by the other protocols.

3. VULNERABILITIES OF CAN

3.1 CAN Behavior

Controller Area Network (CAN) is one of the most widely used of all modern protocols. Originally designed in the mid 1980s by BOSCH, CAN behaves like a broadcast network. Packets created by a node are distributed to all areas of the network and the receiving nodes determine for themselves whether to use or discard the packets. The rationale for this behavior is to keep the information consistent across the network. [4]

CAN features two different packet styles: the standard and extended identifier packets. The structure of a standard CAN packet is shown in Figure 1.

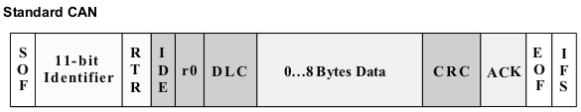

Figure 1. Standard CAN Packet [4]

The packet is divided into the following sections:

SOF: a single bit start-of-frame flag

Identifier: designates the priority of the message and also which ECUs can process the information; a lower value indicates higher priority

RTR: a bit used to indicate when information is required from a specific node (note that the packet is still distributed to all nodes on the network)

IDE: a single bit to identify the packet as a standard length (not extended)

r0: a bit reserved for later use; not currently used

DLC: 4 bits to indicate the number of bytes of data to be transmitted

Data: up to 8 bytes of data can be transmitted in a standard packet

CRC: 16 bit cyclic redundancy check contains the checksum value

ACK: a 2 bit value that, when received, is set to zero; a receiving node will flip it to one if the message contained no errors

EOF: a 7 bit end-of-frame flag; the length is used to help identify bit-stuffing; if 5 bits of the same logic level occur in succession, a bit of the opposite level is stuffed into the data

IFS: 7 bit value that indicates the time for the controller to move a correct frame into the buffer

In addition to the standard length, there is an extended version which features a 29-bit identifier field (the standard 11 bits, plus an alternate version of the RTR, then 18 more bits of identifier) which allows for a wider variety of priorities.

The individual nodes decide whether or not to process the packets by using the CAN ID. If a packet can be processed by a node, it will do so. In addition, there is no restriction against multiple nodes processing the same packet.

Bus access in CAN uses a system that arbitrates the situation that arises if two packets try to use the bus at the same time. Access is random and event driven, meaning that such collisions are likely to occur. In such a situation, the higher priority packet (determined by the CAN ID) wins arbitration and is given preference. However, the other packet is not destroyed and is allowed to arbitrate again afterward.

CAN features robust error-checking mechanisms to ensure that packet errors that occur during transmission are contained and handled efficiently. CAN packets feature five methods of error checking (two at the bit level and three in the message) along with a system that will generate a "resend request" should the frame detection fail one of the five checks. Moreover, CAN features a system which will disable a faulty node if a threshold of repeated

erroneous transmissions is breached [4]. All of this error-checking supports the focus on speed and efficiency (which are always required in real-time networks).

3.2 CAN Vulnerabilities

Several potential vulnerabilities exist in Controller Area Network. As mentioned before, CAN packets do not carry any form of sender or receiver address. Receiving nodes decide to process a packet based on the CAN ID and all of the nodes that can operate on a packet will do so. Therefore individual nodes cannot tell if a packet is designed for them, nor do they know which node that packet came from. This results in another kind of vulnerability, namely CAN's inability to identify whether nodes are legitimate. CAN does not feature any sort of authentication of its nodes. Therefore an attacker could easily spoof messages through a compromised node and the receiving nodes would have no reason to question the spoofed packets' origins. All of these concerns result in a system that is not only insecure but also ill-equipped to even identify threats.

4. ATTACK METHODS

With a system that is so vulnerable, attackers have found a variety of different avenues of attack. In the following section we discuss some ways in which an attacker can interact with a system and some of the different ways in which a CAN network can be attacked, including data stealing, control overrides, data fabrication, vehicle degradation, and external sensor attacks.

4.1 Interaction with CAN

In order for an attacker to attack a CAN network, they must be able to connect to it. There are a few ways this can be accomplished.

The first method is also the most straightforward: direct connection to the maintenance port. "The most significant automotive interface is the OBD-II port, federally mandated in the U.S., which typically provides direct access to the automobile's key CAN buses and can provide sufficient access to compromise the full range of automotive systems." [5] The OBD-II (*On Board Diagnostics*) port, as seen in Figure 2, is a connection located under the steering column of most vehicles.

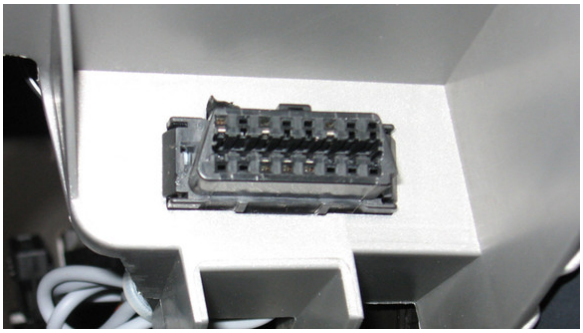

Figure 2. OBD II Port (image courtesy http://www.carcheckup.com/support/knowledgebase.php?article=4)

This port's primary purpose, as indicated by the name, is for vehicle maintenance and engine failure diagnostics. A mechanic can plug a scanning tool into the port and read data by capturing packets being produced by the offending subsystem. However, this port can easily be accessed by anyone with the proper equipment and a window of opportunity. An attacker could easily plug into the OBD-II, gather information or install malware, and then remove the device and exit the vehicle. Alternately, an attacker could implant some sort of remote device on the port and gather information in this way. Because of the necessity of this access, the port has to remain present, ensuring this attack vector will continue to exist.

An alternate method of physically interacting with the vehicle might be through a compromised third-party device. "A similar entry point (to the OBD-II) is presented by counterfeit or malicious components entering the vehicle parts supply chain- either before the vehicle is sent to the dealer, or with a car owner's purchase of an aftermarket third-party component (such as a counterfeit FM radio)."[6] Such methods would be almost completely undetectable.

The recent trend of allowing automobiles to communicate with external sources has raised a new way in which vehicles can be attacked. Many modern vehicles come equipped with systems that allow the vehicles to communicate wirelessly with Internet based services (including systems like GM's OnStar, Toyota's SafetyConnect, and BMW's BMW Assist [5]). These systems provide a new form of access to attackers and allow the attackers to deal remotely with the higher bandwidth systems of the vehicle.In addition, many modern vehicles include Bluetooth capabilities, allowing for short range wireless attacks as well.

Finally, a new system called FOTA (Firmware update Over The Air) is designed to allow manufacturers to provide service updates for automobile systems wirelessly [7][15]. This could potentially give attackers unprecedented access to operations on critical systems.

4.2 Data Stealing

The most basic type of attack that could be performed against a CAN network is a data stealing attack. As mentioned before, a CAN network operates in a broadcast fashion. Each individual node receives each packet and decides for itself whether to discard the packet or perform an operation. As a result, a malicious node on the network could easily snoop each transmission and simply compile the data for use at another time.

There are a number of methods to perform this sort of attack for legitimate uses. The Rev iPhone application allows a user to communicate with a wireless OBD-II reader to gather data and perform a variety of other basic tasks. In addition, the automotive insurance company Progressive offers a device called Snapshot that records driving information with the intended purpose of profiling a driver's habits and providing better insurance rates [8].

However, with these legitimate uses come the dangers associated with gathering information. An attacker could use the information gathered to set up further attacks. The team in [6] used such a method to set up the rest of their attacks. "Through a combination of replay and informed probing, we were able to discover how to control the radio, the Instrument Panel Cluster (IPC), and a number of the Body Control Module (BCM) functions." [6]

4.3 Control Override

"Imagine driving down the highway at 70 miles per hour, when suddenly the wheel turns hard right. You crash. And it was because someone hacked your car."[9] This is the type of attack

that immediately comes to mind when someone thinks about their car being "hacked" while they're driving: the control override. As frightening as it is, it has been shown possible in some situations. The attack takes advantage of the susceptibility of CAN networks to denial-of-service attacks. An attacker could constantly introduce topmost priority messages to the communication channel. Doing so would jam the priority-driven channel as the attacker's messages would always be processed first [3]. In addition to shutting out communication, an attacker could potentially convey messages with deliberately incorrect information with high priority to shut out legitimate messages (as shown in Figure 3).

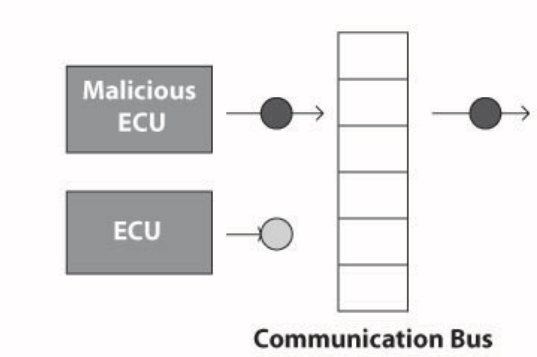

Figure 3. A control override attack. The malicious ECU sends a higher priority packet and prevents the lower priority packet from accessing the bus.

The outcomes of this type of control range from disabling equipment to causing them to react in unusual ways. The team in [6] performed several road tests to accomplish this type of attack. "Even at speeds of up to 40 MPH on the runway, the attack packets had their intended effect, whether it was honking the horn, killing the engine, preventing the car from restarting, or blasting the heat." [6] In addition, the team found that they had extensive control over the EBCM (Electronic Brake Control Module) during the tests and could both disable the brakes and lock up the brakes unevenly, both of which represent a great threat at high speeds.

The good news for this type of attack is that there is some contention over how feasible it is. In their extensive report in [10], Miller and Valasek stated that many of their tests required the vehicle to be in a particular state or required the input of messages of a variety of CAN ID's in order to actually happen. One of the tests involved the use of diagnostic packets which require the vehicle to be in a safe state. "Similar to the previous example that engages the brakes, there is another *DiagnosticCommand* that bleeds the brakes. During the bleeding, the brakes cannot be used. You cannot physically depress the brake pedal. Again, this can only work when the vehicle is moving rather slowely, say less than 5 mph." [10] This would suggest that an attack while a vehicle is in motion would be rather difficult, as an attacker would not only have to be very precise but would also need to send packets with a variety of CAN ID's.

4.4 Vehicle Degradation

In addition to attacks that focus on causing damage through direct control methods, an attacker could also carry out an attack that damages the vehicle by simply causing it to run inefficiently or improperly by conveying false information. Such an attack could degrade or disable a vehicle without the user ever realizing it was happening.

Such an attack would take advantage of the packets sent within the system that provide updates about the current condition of the engine. By spoofing these packets an attacker could misinform the vehicle and lead to the vehicle behaving in odd ways. In [10] Miller and Valasek performed a series of tests using diagnostic packets. One test involved sending a diagnostic packet that was designed to kill the engine in the case of a lack of fuel. As long as this packet is in circulation, the engine will completely shut off and will be unable to be restarted. Moreover, this type of attack can be performed at any speed.

The team in [6] performed similar tests on the engine control systems. "We were able to boost the engine RPM temporarily, disturb the engine timing by resetting the learned crankshaft angle sensor error, disable all cylinders simultaneously (even with the car's wheels spinning at 40 mph when on jack stands), and disable the engine such that it knocks excessively when restarted, or cannot be restarted at all."[6] Tests like these indicate that an attacker can exploit the mechanisms inherent in the vehicle to cause it to destroy itself from within.

In addition to attacks that focus on causing damage through direct control methods, an attacker could also carry out an attack that damages the vehicle by simply causing it to run inefficiently or improperly by conveying false information. Such an attack could degrade or disable a vehicle without the user ever realizing it was happening.

Such an attack would take advantage of the packets sent within the system that provide updates about the current condition of the engine. By spoofing these packets an attacker could misinform the vehicle and lead to the vehicle behaving in odd ways. In [10] Miller and Valasek performed a series of tests using diagnostic packets. One test involved sending a diagnostic packet that was designed to kill the engine in the case of a lack of fuel. As long as this packet is in circulation, the engine will completely shut off and will be unable to be restarted. Moreover, this type of attack can be performed at any speed.

The team in [6] performed similar tests on the engine control systems. "We were able to boost the engine RPM temporarily, disturb the engine timing by resetting the learned crankshaft angle sensor error, disable all cylinders simultaneously (even with the car's wheels spinning at 40 mph when on jack stands), and disable the engine such that it knocks excessively when restarted, or cannot be restarted at all."[6] Tests like these indicate that an attacker can exploit the mechanisms inherent in the vehicle to cause it to destroy itself from within.

4.5 Data Falsification

Just as with the vehicle degradation attack, an attacker could provide incorrect information to the driver, resulting in unsafe behaviors.

One such example occurs in [11] in which the team suppressed the warnings from the system. Vehicles with airbags typically have an airbag control system designed to ensure that the airbag is present and functional in case of a crash. Should the airbag not be present, a warning light should appear on the vehicle's instrument cluster. The team determined the relevant diagnostics packets which would be conveyed by the airbag control system during a diagnostics session. "After recording the reactions to diagnostic

queries during a regular diagnostics session, these replies could successfully be replayed in the absence of the airbag control module. The diagnostics software reports the presence of the device (including its name, part no., etc.) and attests the absence of any error conditions." [11]

According to [10], "the hello world of CAN packet injection is something having to do with display." So an attacker could provide false information to something like the speedometer or odometer to fool the driver. Such an attack was used in [6] where the team caused the speedometer to display 10 mph slower than they were actually moving. Such an attack could easily cause a driver to drive in an unsafe manner by removing their frame of reference to their speed.

Potential other attacks of this nature could target critical systems like temperature control (hiding warnings of excessive temperature) or oil pressure. Either of these could cause the driver to potentially drive their vehicle to death.

4.6 External Sensor Attacks

An emerging type of attack relates to the growing trend to create vehicles that are interconnected in some way with their environment. Systems like rear-view cameras, back-up warning systems, and wireless infrastructure communications introduce a new sort of danger to drivers.

Just as with any data on the instrument cluster, a proximity warning could potentially be spoofed. By simply providing the warning packet that indicates that the vehicle is in close proximity to another, the attacker could cause the driver to react to threats that are not present.

In addition to external sensor spoofing, the trend towards using FOTA to update vehicle firmware introduces a new threat. As mentioned previously, this system allows vehicles to receive firmware updates from the manufacturer which it then applies by flashing the memory on the actual ECUs. In [12], Phung and Nilsson describe various scenarios in which an attacker could potentially exploit the vulnerabilities of this system. "The attacker creates a firmware that can perform arbitrary actions for the corresponding ECU. The firmware is flashed to the target ECU. For example, the ECU could be controlled to read messages that it normally should not have access to or spoof messages that it typically should not send." [12] The authors go on to mention that the ECU firmware that can be updated in this way includes vehicle control ECUs which could allow a new avenue for the vehicle control attacks mentioned previously.

Another external threat that may become a major issue in the near future relates to Vehicular Ad-hoc Networks (VANETs). "VANETs use vehicles as mobile nodes able to self-configure in order to create a communication network via wireless links. The communication takes place both between vehicle-to-vehicle (V2V) and vehicle-to-infrastructure (V2I), aiming to enable *automated* cooperation among different vehicles on the road." [13] VANETs with this capability are the core of a new V2V system currently under consideration by the United States Department of Transportation (USDOT). The system is designed to allow for communication between vehicles and road infrastructure in order to perform calculations and provide driver warnings, advisories, and potentially take pre-emptive actions to avoid and mitigate crashes. [14] While the goals seem reasonable, the implementation of this system as a standard for auto manufacturers would guarantee a new avenue by which attackers could provide false information to the vehicle.

One final type of external sensor attack available relates to Automated Transportation Systems (ATS). The concept emerged from the Intelligent Transportation Systems (ITS) that were envisioned in the late 1990s. These systems would use the same intercommunication proposed by the V2V system to coordinate vehicles and propagate information about road conditions, emergency vehicles, and other driving concerns. According to [16], several European countries including the Netherlands, Germany, and Austria have already implemented such systems in limited capacities.

The ATS takes the system a step further by removing drivers altogether and controlling the vehicles automatically. In [17] the authors present the use of an ATS with a shipping convoy. There are a variety of systems under consideration for this purpose including SARTRE, ARGO, and KON-VOI. These systems arrange the vehicles into a *platoon* which uses sensors located on the exterior of the automated vehicles to judge relative speed and adjust movement accordingly. Aside from the obvious potential of catastrophic damage (in the sense of a standard control override), such a system presents the possibility for a new type of attack: one motivated by diminishing the efficiency of such a platoon. By continually conveying incorrect information between vehicles, the attacker could cause the vehicles to constantly vary their speed, resulting in damage from excessive wear to both braking and acceleration systems. Such an attack would result in financial loss on behalf of the owner of the convoy.

5. POTENTIAL SOLUTIONS

As a result of these and other very real threats posed by the vulnerabilities of modern vehicles, many researchers have been seeking solutions to these vulnerabilities. Several of these solutions focus on the major weakness of in-vehicle network protocols: the inability to detect foreign intrusion.

One of the greatest challenges to implementing any major changes to these protocols is the limited computing power of the ECUs involved. "Usually, a micro controller commonly used in ECUs of current automotive systems (e.g., in off-the-shelf cars) has only a memory of 40-50 KB on average (distributed over RAM and EEPROM) and the computing power is less than 10 MHz." [18] This means that any systems that would be designed to improve this situation would need to be able to function in these limited circumstances. In addition, in-vehicle network protocols need to emphasize speed and efficiency, both of which are necessary in safety critical real-time systems.

5.1 Data Management Systems

In [18], Schulze et al. proposed the idea of a data management system (DMS) to monitor communication and identify malicious behavior. They contend that the decision about whether a packet is malicious or not places too much strain on the already overtaxed ECUs. By sending all communications through a DMS, the system would see many benefits, including uniform data structures, concurrency control, access management, and a potential trip recorder for insurance purposes.

The team listed three paradigms for the DMS: centralized, distributed, and hybrid. A centralized approach would require all communications to pass through a central location which would

monitor and verify all incoming transmissions. Such a system would provide benefits in maintainability and extensibility, since all operations would be at a single location. However, such a system would come with numerous drawbacks. Since all transmissions would pass through a single point, this point could easily become a bottleneck, thus defeating the necessity for speed and efficiency. In addition, if this one system failed in any way, the entire system would likely be compromised. Schulze et al. concluded that this would not be a viable solution.

The distributed solution distributes the DMS hardware throughout the system. Such a solution would overcome the bottleneck concern raised by the centralized approach. It also allows for the use of multiple ECUs in the DMS, allowing for greater computing power. However, this approach also has a major drawback in that the system would increase the traffic on the already overtaxed communication buses. This could, again, lead to a significant danger in the speed of safety critical systems. As such the team recommended the distributed approach over the centralized, but conceded that there could be further improvements.

The hybrid approach is, as expected, a combination of the previous two. The proposed system would use a single DMS for each subbus on the network. Doing so would allow for specialization of the DMS to the particular system involved, and although it would increase the traffic somewhat, it would not be as significant as the distributed approach and it would not see bottleneck to the same degree as the centralized approach. Not surprisingly, the team recommended the hybrid as the preferred DMS approach.

The team does a good job of conveying the benefits of a DMS approach. Moreover, the team makes a point of examining three paradigms of a DMS approach and evaluating the advantages and disadvantages of each. However, the analysis at times feels as though it were presented merely to show why it would not work. This somewhat weakens the argument. In addition, the paper was primarily introducing the concept; no actual implementation was shown, so it is still somewhat difficult to determine how well such a system would function.

5.2 Algorithmic Approach

In [19], Ling et al. provided an algorithmic solution to two of the major vulnerabilities of CAN protocols: denial-of-service attacks and error flag exploitation. They identified the denial-of-service attack as an attack in which a malicious ECU continuously transmits high priority messages. These packets will always win arbitration, thereby sealing off the bus to legitimate lower priority messages. Error flag exploitation uses the error-checking mechanism present in CAN protocols to cut off communication. Whenever the error-checker sees five consecutive bits of identical value, a complementary bit is automatically added. An attacker can exploit this to manufacture error situations, which the error-checker will then respond to without discrimination.

The proposed system uses a monitoring system with a threshold to identify suspicious repetition. At the outset, the system creates a log of all known IDs and clears a counters and flags for both known and unknown IDs. Each time a message appears on the bus, the system identifies if it is a known ID or an unknown ID. If the ID is known, the known-ID counter is incremented and the flag is set, and the unknown ID counter and flag are cleared. If the ID is unknown, the same happens but with the unknown ID flag and counter. This system of monitoring continues until a predetermined threshold is reached. At that point the system triggers an alarm relating whether known or unknown suspicious activity has been detected. A model of this system is shown in Figure 4.

The team was able to show that their system did function properly and was able to identify suspicious behavior. Moreover, they modeled the system using the capacity limitations of a CAN system. Despite these successes, their system seems ambiguous at best. No information is provided to indicate how the initial index of valid IDs would be compiled. Several steps of the outlined algorithm are ambiguous (listed simply as “Do further judgment”) and no indication is given as to what form the alarm should take (whether it should notify the driver, the system, or something else). While these concerns weaken the report,

the idea of an algorithmic solution holds a great deal of promise, and the effort to solve multiple vulnerabilities at once is laudable.

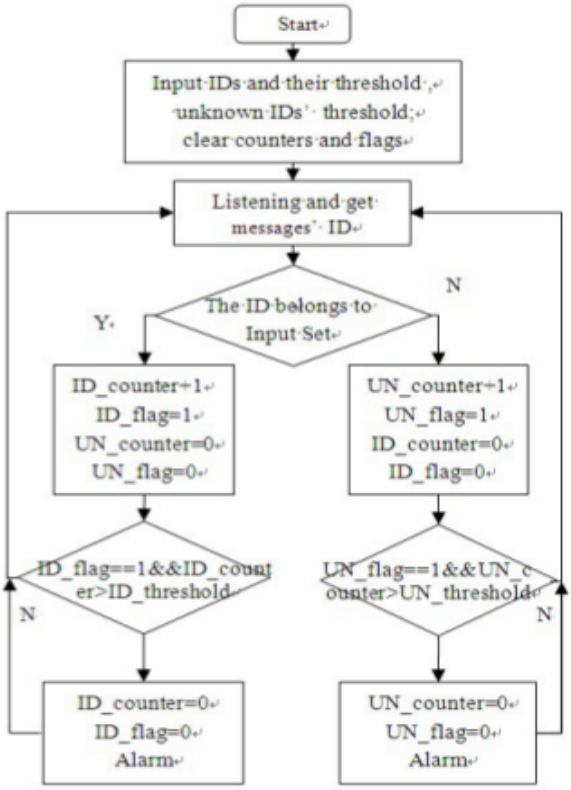

Figure 4 A model of the algorithm proposed by Ling et al. [19]

5.3 Attestation-Based Security

In [20], Oguma et al. propose an algorithmic approach that uses a system of attestation and encryption to identify nodes which were present at the startup of the system. The process starts during the manufacture of the vehicle. A specific ECU designated as the Master ECU is provided a serial number and a key which is stored in a database at the manufacturing center. During assembly of the vehicle, the Master ECU provides the number and key to the center. The center and Master ECU then perform a series of decrypted communications that verify the Master ECU through attestation and identify hash values for all the ECUs in the vehicle. This hash table is then stored in the Master ECU and all individual ECUs. At this point the initialization of the vehicle is complete.

Each time the vehicle is started, a similar process occurs between the Master ECU and other ECUs. The Master ECU generates two random numbers and sends them to all the ECUs. The ECUs perform an operation using one of the two numbers and respond to the Master ECU. The value returned should be one of the values in the hash table. This verifies to the Master ECU that all the ECUs present are valid.

During operation of the vehicle, the other random value is used as a sort of "clearance" for that ECU, as the value is encrypted and included in every message. Also included in the packets is a counter designed to defeat replay attacks.

This solution seems very ambitious, yet the authors do a thorough job in explaining mathematically how it should work. In addition, they present a sample experiment to prove that their solution not only meets their requirements for a solution but also surpasses other similar solutions. However, the project relies heavily on the ECUs being able to process a great deal of encryption, something which may prove difficult with current ECU technology. In addition, the solution requires space in packets for the counter and encrypted value, but the authors give no indication where these elements should be inserted. It is unclear as to whether or not this would meet the requirements for speed and efficiency in real-time systems.

5.4 External Download Security

Another of the ways in which vehicles are vulnerable is through the FOTA system mentioned earlier. In [12] Phung et al. proposed a system to counteract this type of attack. Their proposed solution uses a device called a transformation module. Figure 5 shows the image as provided in the document.

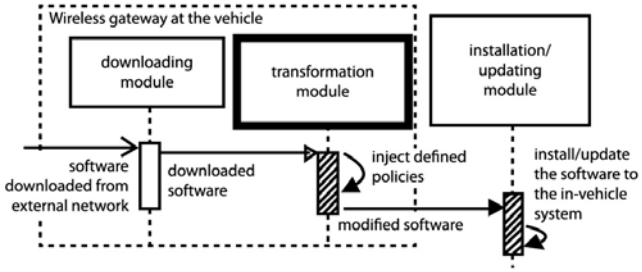

Figure 5 The model transformation module at the wireless gateway. [12]

The team concluded that the primary vulnerability of vehicles to FOTA came from that the ECUs themselves were responsible for determining if the information provided was valid or not. They stated that the ECUs should not be responsible, but rather that a device called a Transformation Module should be placed at the wireless gateway to examine all incoming traffic for validity before disseminating it to the network.

Since all FOTA traffic comes in through a single point, the team proposed including a device at that wireless gateway that could be responsible for examining all traffic. This device, the aforementioned Transformation Module, would take the downloaded software and perform an operation on it to make sure that it adhered to security guidelines provided by the manufacturer.

While this solution seems straightforward enough, the paper seems to be directed less at the security of ECUs and more towards the security of add-on software. The paper mentions that add-on software is developed using Java, therefore the paper outlines a Java-based solution to the problem. In terms of ECUs, however, the paper only gives modest recommendations as to how the transformation module would be constructed, indicating that since ECU firmware is typically designed in C or nesC, a solution could be devised along those lines. This, in addition to a suggestion that the solution might lie in hardware changes, leaves the reader with only a vague impression of how the Transformation Module solution could be implemented for ECUs. In addition to this problem, the paper provides little indication as to how long such a transformation operation might take.

6. CONCLUSIONS AND FUTURE WORK

In this paper we have discussed several aspects relating to the vulnerability of modern in-vehicle networks. The function of standard in-vehicle networks has been discussed, along with an explanation of the weaknesses facing such networks, particularly those using the CAN communications protocol. We have discussed several different ways in which a vehicle can be attacked, and concluded with a discussion of some of the countermeasures that have been recommended to combat these issues.

It is clear that a great deal more work will be required before modern vehicles can be said to be secure. The vulnerabilities of CAN and the interconnected nature of in-vehicle networks combine to create a situation in which an attacker can reach any point in a vehicle's network by compromising a single ECU. In addition, the lack of any form of authentication in modern CAN systems means that any attacker who enters the system will be treated as a legitimate node.

We believe that the first step towards securing vehicles lies in the issue of authentication. A vehicle's network cannot be expected to defend itself against the unknown. Future work in this area could focus on a solution that allows the system to identify foreign nodes (both foreign at startup and during operation). Such a solution would also have to be viable using the ECU technology currently available.

7. ACKNOWLEDGEMENTS

This material is based in part upon work supported by the National Science Foundation under Grants No. CNS-1305369, No. DUE-1241675, and No. DGE-1303384.

8. REFERENCES

[1] Fernandez, John D., and Andres E. Fernandez. "SCADA systems: vulnerabilities and remediation." *Journal of Computing Sciences in Colleges*20.4 (2005): 160-168.

[2] Nilsson, Dennis K., Phu H. Phung, and Ulf E. Larson. "Vehicle ECU classification based on safety-security characteristics." *Road Transport Information and Control-RTIC 2008 and ITS United Kingdom Members' Conference, IET*. IET, 2008.

[3] Wolf, Marko, André Weimerskirch, and Christof Paar. "Security in automotive bus systems." *Workshop on Embedded Security in Cars*. 2004.

[4] Texas Instruments. *Introduction to the Controller Area Network (CAN).* SLOA101A. Dallas, TX. 2008.

[5] Checkoway, Stephen, et al. "Comprehensive Experimental Analyses of Automotive Attack Surfaces." *USENIX Security Symposium*. 2011.

[6] Koscher, Karl, et al. "Experimental security analysis of a modern automobile."*Security and Privacy (SP), 2010 IEEE Symposium on*. IEEE, 2010.

[7] Kleberger, Pierre, Tomas Olovsson, and Erland Jonsson. "Security aspects of the in-vehicle network in the connected car." *Intelligent Vehicles Symposium (IV), 2011 IEEE*. IEEE, 2011.

[8] Vasilev, Pavel. *Driver's Efficiency Analyzer*. Diss. Cornell University, 2012.

[9] Pagliery, Jose. "Your car is a giant computer-and it can be hacked." *CNN Money.* Web. 2 June, 2014.

[10] Miller, Charlie, and Chris Valasek. "Adventures in Automotive Networks and Control Units." *Last Accessed from http://illmatics. com/car_ hacking. pdf on*13 (2013).

[11] Hoppe, Tobias, Stefan Kiltz, and Jana Dittmann. "Security threats to automotive CAN networks–practical examples and selected short-term countermeasures." *Computer Safety, Reliability, and Security*. Springer Berlin Heidelberg, 2008. 235-248.

[12] Phung, Phu H., and Dennis Kengo Nilsson. "A model for safe and secure execution of downloaded vehicle applications." (2010): 06-06.

[13] Groza, Adrian, Bogdan Iancu, and Anca Marginean. "A multi-agent approach towards cooperative overtaking in vehicular networks." *Proceedings of the 4th International Conference on Web Intelligence, Mining and Semantics (WIMS14)*. ACM, 2014.

[14]. "Vehicle-to-vehicle safety application research plan." USDOT. Oct 2011.

[15] Larson, Ulf E., and Dennis K. Nilsson. "Securing vehicles against cyber attacks." *Proceedings of the 4th annual workshop on Cyber security and information intelligence research: developing strategies to meet the cyber security and information intelligence challenges ahead*. ACM, 2008.

[16] Kargl, Frank. "Securing the intelligent vehicles of the future." (2013).

[17] Gerdes, Ryan M., Chris Winstead, and Kevin Heaslip. "CPS: an efficiency-motivated attack against autonomous vehicular transportation." *Proceedings of the 29th Annual Computer Security Applications Conference*. ACM, 2013.

[18] Schulze, Sandro, et al. "On the Need of Data Management in Automotive Systems." *BTW*. Vol. 144. 2009.

[19] Ling, Congli, and Dongqin Feng. "An Algorithm for Detection of Malicious Messages on CAN Buses." *2012 National Conference on Information Technology and Computer Science*. Atlantis Press, 2012.

[20] Oguma, Hisashi, et al. "New attestation based security architecture for in-vehicle communication." *Global Telecommunications Conference, 2008. IEEE GLOBECOM 2008. IEEE*. IEEE, 2008.