SECURITY AND COMMUNICATION NETWORKS
Security Comm. Networks 2016; **9**:4330–4340
Published online 7 September 2016 in Wiley Online Library (wileyonlinelibrary.com). DOI: 10.1002/sec.1610

RESEARCH ARTICLE

Security in automotive telematics: a survey of threats and risk mitigation strategies to counter the existing and emerging attack vectors

Alex Oyler[1] and Hossein Saiedian[2]*

[1] SBD North America and University of Kansas, Lawrence, KS, U.S.A.
[2] EECS, Universityof Kansas, Lawrence, KS, U.S.A.

ABSTRACT

This manuscript surveys a variety of topics related to the security of connected vehicles and their associated services, often referred to as *telematics*. Current challenges as well as emerging and future security risks are discussed. A number of specific technologies and implementations are discussed, such as Next Generation Telematics Patterns, KeeLoq, vehicular ad hoc networks, and controller area network. Principles and practices are outlined to counter or minimize risks associated with the existing and potential attack vectors as well as reducing risks both on the internal vehicle network and the telematics service delivery platforms. Copyright © 2016 John Wiley & Sons, Ltd.

KEYWORDS
telematics; automotive security; controller area network (CAN); vehicular ad hoc networks (VANETs); Next Generation Telematics Pattern (NGTP)

***Correspondence**
Saiedian, Hossein, EECS, Universityof Kansas, Lawrence, KS, U.S.A.
E-mail: saiedian@eecs.ku.edu

1. INTRODUCTION

The emergence of what is colloquially known as the internet of things has had many implications on the way we use and interact with electronic devices from gas pumps and phones to stoplights and infrastructure. Vehicles are no exception. Most premium and many entry-level vehicles now contain some sort of a combined hardware and software module that supports connectivity via a cellular network or Bluetooth connection to a data-enabled cell phone. Such a platform is known primarily as *telematics*. Many companies provide telematics services for consumer and commercial vehicles, such as Verizon, OnStar, and SiriusXM. While most telematics service delivery platforms have been designed with security in mind, there remain a number of challenges to completely securing a solution from vehicle to user. For example, security mechanisms on the vehicle network itself are scarce and rarely well implemented. The shortcomings of these systems manifested themselves publically the Summer of 2015 when two security researchers from IOActive remotely affected a vehicle in motion. The resulting study published in *Wired Magazine* [1] shifted the attention of original equipment manufacturer (OEM) and government stakeholders to the real-world implications of poorly secured connected vehicles.

It is important that researchers, manufacturers, and suppliers alike give serious thought to the overall security of any telematics offering as the stakes are human safety. A security breach could mean unfettered access to vehicle systems, endangering its occupants. There are no universal security standards for telematics or vehicle platforms, and thus no holistically designed offering that withstands a comprehensive security assessment. This paper surveys the particularly dangerous existing and emerging attack vectors on local vehicle networks and telematics service delivery platforms.

While electronics have been integrated with automotive components since the first commercially available microchip, the idea of connecting these individual components first actualized in 1987 using the recently introduced controller area network (CAN) standard. Development of the protocol began at Bosch in 1983 and publicly introduced in 1986, preceding the first commercially available CAN-enabled components by only 1 year. Today, dozens of networked electronic control units (ECUs) communicate on the CAN bus on all types of vehicles, each serving a unique and important function. Some examples include the blind spot module, powertrain control module, and, perhaps the most relevant to this discussion, the telematics gateway. Newer vehicles, such as the Tesla Model S, use automotive Ethernet as an alternative to CAN.

The telematics gateway manages all consumer and cellular radio frequency (RF) communication, which consists primarily of Bluetooth and standard cellular technologies such as CDMA, EVDO, GSM, and, most recently, Long Term Evolution. The first and most well-known service offered through this component was General Motors' OnStar, which launched in 1996. Customer features included in-vehicle conveniences such as emergency dialing and roadside assistance, while additional services such as stolen vehicle location assistance were available in a situation that warranted location tracking. Going forward, the telematics gateway will play an increasingly important and visible role in the user experience as higher throughput connectivity enables richer features.

One of the primary tools used by mechanics and automotive technicians in diagnosing and repairing issues with the vehicle is called on-board diagnostics. The standard implementation, called on-board diagnostics port 2 or OBD-II has been available on all vehicles since the mid-1990s. OBD-II specifies a number of parameters that may be queried by a device, such as ECU statuses and configuration as well as certain standard diagnostic trouble codes. Diagnostic trouble codes indicate to a technician if there is an issue with certain vehicle systems such as the engine or transmission.

In order to access these codes and parameters, an OBD-II port is installed on every vehicle, generally under the steering column. The OBD-II specification calls for a standard connector implementation. The two most important pins on this connector are 6 and 14, as those connect to the high-speed and low-speed CAN busses on the vehicle. In general, all powertrain ECUs reside on the high-speed bus with baud rates up to 500 Kbps. All other ECUs, such as interior units like the amplifier, reside on the low speed bus. Some ECUs such as the body controller and the telematics gateway may send and transmit messages on both busses. It is important to note that every OEM has a different CAN bus architectures, and even varying architectures within their own product lines and vehicle platforms.

One of the identified uses of the telematics gateway going forward is the principal manager of a vehicle's involvement in what is known as a VANET, or vehicle ad hoc network. A VANET is just as the name implies: an ad hoc collection of automotive peers that exchange information to enable a service. There could be other peers on the VANET as well, called RSUs, or roadside units. These connect to infrastructure to enable services depending on the function of the service being enabled. Various emerging telecommunication protocols are currently being evaluated as the basis for VANET or vehicle-to-vehicle/vehicle-to-infrastructure (V2X) implementations. The two major candidates are dedicated short-range communications also known as DSRC [2] and 5G [3].

The VANET is perhaps the primary area of automotive research as it has important implications in public safety infrastructure. Enabling this technology in the public safety sector would pave the way for consumer-facing services. Within public safety, a VANET could enable smarter stoplights, better communication between officers, and real-time fleet monitoring. Consumer services could include traffic routing, collision assistance, and road condition and traffic reporting. As a use case, a consumer vehicle equipped with VANET-enabled technology could automatically report to a roadside unit in the event of an accident, which would then notify the closest emergency personnel.

These types of networks are also critical in the world of the autonomous vehicle. The key efficiencies gained by automating mobility can only be realized if vehicles have the ability to network directly with other vehicles for the purpose of optimizing intersections, avoiding collisions, and detecting anomalous environment conditions.

While there are no practical implementations of VANETs as of yet, universities and manufacturers are investing heavily in the concept. Many cities are implementing "smart corridors" for the purpose of evaluating next-generation V2X networking technologies. The University of Michigan's MCity, in partnership with the city of Ann Arbor, has made major strides in testing emerging technologies in closed facilities while building the public infrastructure required to bring these technologies to mass market [4,5]. Furthermore, the US Department of Transportation is evaluating potential pilot cities for deploying DSRC-based "smart city" infrastructure. The partnership evident here between government, academia, and private industry provides the key ingredients for realizing the steps required to bring next-generation technologies such as DSRC and 5G to market.

2. THE CURRENT STATE OF AUTOMOTIVE SECURITY

The most important principle in designing a secure system is to exercise the idea known as defense in depth, defined within the context of computing as a strategy that employs multiple layers of security. The idea extends to automotive systems. The following is an analysis of a number of these layers within the automotive and telematics model and the robustness of their respective authentication and authorization mechanisms. Researchers from the group known as the Center for Automotive System Security as well as the automotive cybersecurity company Secured By Design (SBD) have published research on this topic over the past 5 years and provide the groundwork for the surfaces discussed later. The full scope of security threats to the connected car spans a much greater surface than what can be discussed here; in fact, SBD identified nearly 20 separate attack points in the connected car ecosystem.

2.1. Controller area network and electronic control unit security

Understanding defense in depth within the context of telematics is important because most vehicle-centric services do not and thus cannot assume a layered security paradigm. The biggest problem is the security of the internal vehicle network itself. In most cases, only the most basic security measures are in place for on-bus communications. While one issue is that all bus communications are carrid out without any form of encoding or encryption, the most relevant security concern is with trust. Nearly all communications are implicitly trusted. The only real validation is carried out through the use of a cyclic redundancy check as shown on Figure 1 [6], but this is only applicable for error detection, not necessarily security and trust.

The CAN protocol itself does not have any provision for data integrity protection or authentication (Nilsson [7]). Furthermore, CAN is peer-to-peer (as opposed to something like point-to-point), making the implementation of a trust mechanism that much more difficult. The bottom line is that the CAN protocol itself does not support any security mechanism. If any authentication and/or authorization is to be implemented, it must be at the electronic control unit or physical firewall level, which could be equated to the application layer of the Open Systems Interconnection (OSI) model.

For any remote action on a vehicle such as door unlock or remote start, a nonlinear feedback shift register block cipher may be used by the receiver ECU, which manages the given function to authenticate incoming requests. A relatively common authentication mechanism for RF hub manufacturers is the KeeLoq NLFSR block-based cipher, which then requires a rolling code to be transmitted from a requesting device—in this case, the telematics gateway. Many high-volume auto manufacturers use this technology manufactured by Microchip Corporation for all of their RF hub implementations [8].

KeeLoq, however, has been both mathematically and practically broken for the purposes of establishing sufficient trust with a requester [9]. The keyspace itself is 64 bits with a 32-bit hopping code, which for this type of security is sufficient; however, a number of factors mitigate its effectiveness. First, Microchip suggests that each manufacturer implement an error tolerance of 16 codes. That is, the RF hub will allow not just the next calculated code, but any of the next 16 calculated codes. Effectively, the keyspace is reduced by 16 times to a 28-bit block (which still allows for nearly 270 million combinations). The biggest security concern, however, is that the secret manufacturer key can be derived using a differential power analysis attack—a type of side-channel attack—which measures the power consumption of the chip during encryption. Once performed, an attacker could know the manufacturer code for a particular OEM, and successfully apply his or her findings to any make and model that implements the cipher on that OEM's platforms. Knowing the manufacturer key is particularly devastating because any transmitter, whether it may be a key fob (a small security token) or the telematics gateway, could be cloned and valid hopping codes transmitted using this information.

While this is particularly concerning because of the threat of cloned transmitters, the real issue with the CAN and the KeeLoq trust model for remote access is that each device on the CAN is inherently trusted. Because the telematics gateway is a trusted device, all requests coming from it are not independently authenticated. There is no current method for the receiver ECU to validate that the request from the telematics gateway is from a legitimate sender. It simply trusts that any invalid requests have already been blocked at a higher level. For current applications, this may be secure enough. A true break of this mechanism requires physical access to the vehicle. However, as vehicles become more technologically advanced, and each control unit has more influence on the vehicle's actions, a more secure protocol must be introduced to prevent injury and loss of life because of nefarious access to the bus messaging. The telematics gateway is merely the beginning as the introduction of VANETs, and self-driving cars will leave more control of the vehicle to non-human components.

2.2. On-board diagnostics

In many vehicle platform electrical architectures, a device connected to the OBD-II port of a vehicle may transmit and receive any message on any CAN bus in the vehicle. This presents a number of security risks when physical access to the vehicle is available. With limited knowledge of a particular automaker's CAN messaging definitions, one could send any number of control messages to the engine, transmission, brakes, and safety systems without the driver's awareness or consent.

This security risk has been demonstrated numerous times over by white hat hacking organizations for high-profile media outlets such as *Forbes* and *Vice* [10]. In these demonstrations, the hackers install a device with a cellular modem on the vehicle's OBD-II port. This device has been configured to accept commands remotely and retransmit them on the powertrain CAN bus. The hackers themselves were able to reverse engineer certain control commands that cause the vehicle to cut the engine. Many consumer telematics devices for common applications such as usage-based insurance and fleet telematics are composed of the same basic architecture, and any security vulnerability in those devices enable an extremely fertile attack

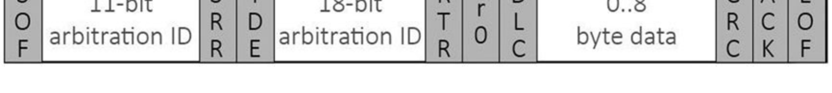

Figure 1. A standard controller area network frame with 8-byte payload. CRC, cyclic redundancy check.

surface. In two excellent and comprehensive reports, Checkoway and his colleagues at the University of California San Diego (2011) and Miller and Valasek [11] have shown that it is possible to disable a vehicle's breaks, turn off head-lights, or potentially take-over steering via breaking into the OBD-II port. A number of other researchers at UCSD report similar attacks and concerns [12].

The thought of this is particularly frightening: If hackers were to be able to find out the networking addresses of the device (such as Mobile Device Number (MDN), IP address, or International Mobile Subscriber Identity (IMSI)), they could theoretically perform a remote intrusion on the vehicle's CAN bus. However, the same principle that enables this attack also mitigates its risk to the public at large: physical access. It is not feasible to break in to thousands of vehicles at once and install devices or figure out, which vehicles already have a target device installed; this is a more specialized attack that requires not only access to the vehicle, but also reverse engineering technically challenging problems in order to understand what CAN frames are required to affect a vehicle's behavior. That said, if certain mitigations are not in place on the cellular network, an attacker could theoretically flood a range of possible IP addresses with requests to determine which devices respond to valid requests. In fact, this is exactly what enabled the scale of the Jeep Cherokee attack.

Car automation has become a reality, and new research and technology promise and offer hands-free steering of vehicles [13]. In such technological progress, driving a vehicle becomes a task shared between a human being and technology. In addition to the common risks, there are additional concerns with security, especially if the technology parts of such vehicles are hacked.

2.3. Service architecture security

The lack of security at the CAN bus level requires that the security of a telematics solution be bulletproof. While most automotive companies claim proprietary stakes to these solutions, some companies have proposed open standards for telematics to help bolster the security of their own solutions. One such company is BMW. In 2010, they, alongside WirelessCar and Connexis, proposed NGTP 2.0, which stands for NGTP [14]. Figure 2 illustrates the architectural pattern described by NGTP. Implementations like this include standard security measures such as the use of Transport Layer Security (TLS), defense in depth, PIN authentication, and separation of concerns.

Some problems, however, are universal. Externally, social engineering is as much of a threat in telematics as it is in most information technology settings. If someone clandestinely learned a driver's authentication information to a service, that person could theoretically unlock, start, and stop his or her vehicle without any real evidence outside of IP address, which is easily spoofed. Shared accounts pose a similar threat. There are also internal threats, both intentional and unintentional. Those with understanding of the technology behind these solutions can exploit its known weaknesses and their elevated access to cause harm.

In 2010, researchers uncovered a practical attack against a widespread implementation of telematics communications on GSM networks. Airbiquity, a telematics integrator based out of Seattle, developed and sold a technology named aqLink used in the Ford Sync telematics system. Researchers from Center for Automotive System Security reverse engineered the aqLink protocol and discovered a weakness in its usage of the voice channel for data transfer. Using a suite of signal processing tools, the researchers reverse engineered the initiation message and the protocol used to send data across the voice channel. Once packet size, ECC, and cyclic redundancy check were understood, they were able to take binary logs of certain commands to and from the vehicle. Using all of these elements, they implemented a

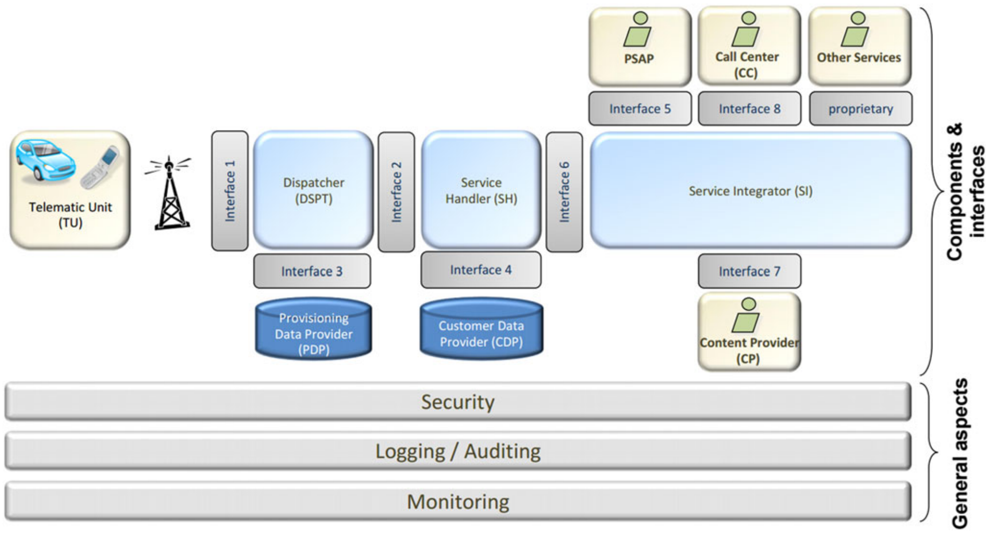

Figure 2. Next Generation Telematics Pattern high-level architecture pattern [14].

C program to send custom commands to the modem using the known elements of the aqLink protocol [15]).

This is another prime example of where defense in depth should have been utilized to protect against intrusions. Once the researchers were able to mimic a valid transceiver, the implementation had very few additional security measures to protect against malicious or invalid commands. A number of buffer overflow vulnerabilities were identified as the telematics gateway was largely written in C, but the real problem manifested itself with the only other security measure implemented—the response-challenge mechanism for incoming commands. aqLink used a nonce to protect itself from replay attacks. Counter-intuitively, aqLink always seeded that nonce with the same constant when the gateway powered on. Thus, it did not actually protect against replay attacks at all.

Using the previous two vulnerabilities, the researchers authenticated to the vehicle using their rogue C transceiver and placed an Internet Relay Chat (IRC) client on the telematics gateway using one of the buffer overflow exploits. When the vehicle came online, this client would join a channel through which commands would be sent. To concretely demonstrate the severity of this break, they used this channel to turn on the audio systems as well as periodically post the vehicle's GPS coordinates to Twitter. Those, however, represent more innocent uses of this break. A more devious hacker could use this channel to remotely engage or disengage the brakes, which could lead to disastrous results.

Another cellular communication protocol, which is used commonly in automotive telematics service delivery applications, is short message service, or SMS. An SMS is often used to trigger some behavior by the vehicle from the network because of a lack of bi-directional data connectivity. Despite inherent problems with SMS such as its size limitation, its role within the context of telematics service delivery in the status quo is broad. These limitations mean that data cannot be efficiently encrypted or signed. With access to certain areas of the modem, an attacker could theoretically read the contents of an SMS to find potentially valuable information such as vehicle identification number (VIN), server uniform resource identifier (URIs), or IP addresses.

As newer technologies emerge, additional security challenges will be faced. As an example, Sprint Velocity, which provides telematics services to Chrysler, announced that it would be using IBM's MessageSight technology (Armonk, NY, USA), which implements a lightweight application protocol called MQTT (MQ Telemetry Transport). MQTT is built for lightweight applications, such as sensor networks. Generally, security is of minimal concern for these networks as most of the data have no value to a potential hacker, and no known implementation of MQTT provides substantial motivation for subversion or disruption. Newer versions of MQTT support TLS encryption as well as other high-level security mechanisms that make securing this type of solution much easier, but emerging solutions will have to take security considerations to the forefront before any new technology would be considered for integration.

3. NEXT-GENERATION FEATURES AND THEIR SECURITY RISKS

For a design to be secure, the specific weaknesses and security challenges associated with that particular feature need to be called out and designed for at the onset of research and development. It is not good enough to simply react to a security threat; suppliers and OEMs must expect the threats. Certain development frameworks such as SBD's Automotive Secure Development Lifecycle promote awareness of and adherence to security requirements throughout a product's lifecycle. In the succeeding texts are some features that will likely come to market in the next 5–10 years, each introducing a specific security challenge.

3.1. Rich features and controller area network access

In order for richer features to be delivered to customers, increased communication must take place between the components on the vehicle CAN. In the case of the telematics gateway, this poses a particularly interesting challenge as exposing those features on the network to the gateway introduces the link between the Internet and the required electronic control units on the CAN. As an example, a driver could configure his or her own profile, calling out specific environmental preferences such as radio presets, seat adjustment, heating, venting, and air conditioning (HVAC) settings, and transmission behavior. All of these settings are controlled by individual ECUs: radio presets by the telematics gateway, seat adjustment by the seat controller, air conditioning by the HVAC ECU, and transmission by the transmission control module. Allowing the radio to configure each of these settings means that each individual ECU must now accept CAN messages from the telematics gateway, exposing an attack vector to each ECU. A visualization of this potential attack vector is show in Figure 3.

The attack surface is not just limited to those functions, which are exposed specifically to the telematics gateway.

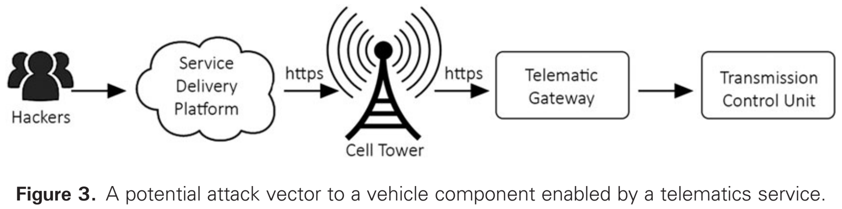

Figure 3. A potential attack vector to a vehicle component enabled by a telematics service.

Many ECU behaviors, including powertrain ECUs such as the engine control module and transmission control module, are affected by CAN messages sent by other ECUs. If an attacker is able to reverse engineer the encoding of these messages, they could then manipulate the vehicle's behavior remotely, resulting in a severe compromise of the vehicle's integrity and, more importantly, the passengers' safety. This is exactly what happened when the Jeep Cherokee's telematics gateway was compromised: The researchers were able to manipulate the vehicle's driving behavior by sending malicious messages specifically designed to invoke certain functions in steering, throttle control, and braking [16].

3.2. Publish/subscribe pattern

While today's common telematics application implementations such as NGTP utilize standard security measures such as TLS, other service providers are looking at more lightweight application protocols for delivery of these features. HTTPS, while certainly secure for this application, also creates a large amount of processing and data overhead. Message Queuing Telemetry Transport (MQTT) and its cousin advanced message queuing protocol (AMQP) are a lightweight connectivity protocol that utilizes publish/subscribe semantics in order to transfer data through less reliable connections. This makes sense for a telematics application: The service provider cannot guarantee the level of service provided through the cellular network. At least one telematics solutions provider, Sprint Velocity, has implemented this application protocol for its solution. This poses interesting security challenges, such as secure authentication to the message broker and application-level encryption.

One potential strategy for implementing a secure platform is to follow semantics proposed previously for publish/subscribe architectures. Abdullahi and Wang [17] identify five discrete security challenges within this type of architecture: publication authentication, subscription authentication, publication and subscription integrity, publication confidentiality, and subscription confidentiality. Addressing each of these individual concerns ought to yield a holistically secure solution. Utilizing standard encryption measures such as TLS and authentication mechanisms that include services such as lightweight directory access protocol (LDAP) and a CA addresses these specific concerns. However, a complete analysis each of those five concerns for threats from both outside and inside should be completed during design of a publish/subscribe architecture.

3.3. Vehicle ad hoc networks

Another emerging technology to consider is the VANET. VANETs are peer-to-peer collections of vehicles and nodes such as traffic lights within a certain area. An example would be a network of vehicles and a collection of traffic lights so that the lights can actively monitor the flow of traffic and adjust its signaling appropriately for the configured corridor. The challenges around implementing VANETs, in particular the security aspect, are perhaps the most widely researched topics in vehicle technology today.

While there are a number of practical challenges to successfully implementing VANETs (primarily manufacturer and consumer adoption), a precursor to such an implementation is a standardized security model. Very real attacks have already been identified against DSRC applications. The following were identified by SBD in its evaluation of DSRC applications in 2014:

- forging, eavesdropping, or blocking of SOS messages;
- theft of RSU control unit data;
- forging, eavesdropping, or blocking RSU warning messages;
- forging, eavesdropping, or blocking vehicle warning messages; and
- impersonation of other vehicles.

Note, of course, that these attacks are purely theoretical until the technology is fully implemented, deployed, and tested in a real-world environment.

The impact of compromising these messages is self-evident. Prohibiting a vehicle access to critical environmental data may cause critical failures within the system when it is designed to rely on these messages. Any vehicle system must be thoroughly designed and assessed to account for these types of failures, but it is nearly impossible to predict every single failure mode. Understanding the implications of compromises to the integrity of messages sent between vehicles and other vehicles or related infrastructure will be a key enabler of the various V2X applications.

For additional coverage of the above concerns, please see two excellent reports, one covering the state of the art in embedding security in vehicles [18] and one providing a report on the V2X security and privacy, discussing the current state and its future [19] both co-authored by Andre Weimerskirch who is known for his research on vehicle security. Eiza and his colleagues [20,21] and Laymin *et al*. [22] present other security and privacy threats and vulnerabilities. When developing a new application, it is best to investigate and identify the security requirements early in the process and incorporate them into the application development from the very start of the development. Tetmeyer *et al*. [23] introduce a new approach for capturing such security requirements early in the development.

4. COUNTERMEASURES IN CURRENT AND FUTURE TECHNOLOGY

In order to better secure existing and future telematics platforms, a number of simple security measures can be implemented and accounted for at each component. As each solution is different, the specific artifact of each suggestion may be different; however, these are all strategies that ought to make it much more difficult to compromise the integrity and availability of telematics services. Some principles are ubiquitous: User education, strong passwords, and two-form authentication should be considered for all user-facing services.

Risk mitigations or countermeasures are generally framed within the context of a classification scheme wherein threats are assigned classifications and each classification has a specific type of countermeasure. Within the context of Automotive Secure Development Lifecycle, the STRIDE scheme is used. Each letter in STRIDE stands for the threat classification: spoofing user identity, tampering with data, repudiation, information disclosure, denial of service, and elevation of privilege. Each of the countermeasures later addresses these specific threats as countermeasures and is summarized in Table I.

4.1. The principle of least privilege

Perhaps the most important principle to follow when designing any system is using least privilege for all application programming interface (APIs) and access. Least privilege means that for any external access to a component, whether it may be through an API or administrative purposes, the access should be limited to only what is needed to complete the work or action. This can be realized at a number of components and layers within known and potential telematics platforms.

At the vehicle level, any signaling between ECUs on the CAN should be limited to only what signals each ECU requires to do its job. For example, the occupant restraint controller should not need to be a receiver on the bus for the signal indicating whether or not cruise control is enabled. Exposing unnecessary signaling only increases the chance for defect or worse exploit.

This is particularly important at the telematics gateway level as unfettered access to the CAN bus would allow a remote agent to inject any number of signals onto the bus that may affect vehicle behavior. In order to protect this at the vehicle level, each individual ECU on the bus should explicitly reject any message from the telematics gateway that it does not expect. The transmission control module should not accept messages from the telematics gateway telling it to change gear; it should simply manage gear changes internally. This functionality could also be provided by an external firewall gateway, which resides between the telematics gateway and the CAN bus.

Table I. Countermeasures and STRIDE threats.

Component	Countermeasure	S	T	R	I	D	E
Telematics gateway	Detection of fake cellular networks	√	√	√	√	√	
	Secure boot process		√				
	Debug port authentication	√	√				
	Over the air software updates (reprogramming)		√		√	√	
	Buffer overflow protection via address space layout randomization		√				√
	Intrusion detection and prevention system	√	√	√	√	√	√
	Secure client-server communications				√		
	V2X jamming detection and prevention			√			
	Trust anchor for external communications	√	√	√	√		
	SMS authentication	√					
	Hardware security module	√	√	√	√	√	√
Mobile network operator	SMS firewall		√		√	√	√
	Secure SIM data	√	√	√	√		
Telematics service provider	Encrypted communications		√		√		
	Adherence to security standards (ISO27001)	√	√	√	√	√	√
	Mutual authentication for all client communications	√					
CAN bus/OBD	OBD hardware coverings	√					√
	CAN bus firewall	√	√			√	√
	Message authentication codes	√	√				
	ECU key management	√	√	√	√	√	√
	CAN bus anomaly detection network monitor	√	√	√	√	√	√
	Centralized authentication	√	√				
In-vehicle infotainment	Digital signatures for applications	√	√	√			
	Embedded virtualization	√	√		√	√	√
	Wi-Fi password policy	√	√		√		
	Wi-Fi NIST guidelines	√	√		√	√	√
	Bluetooth NIST guidelines	√	√		√	√	√
	USB best practices		√				√
	Recovery by design						√
	Bug bounties	√	√	√	√	√	√

OBD, on-board diagnostics; CAN, controller area network; ECU, electronic control unit; SMS, short message service; V2X, vehicle-to-vehicle/vehicle-to-infrastructure; SIM, subscriber identity module; NIST, National Institute of Standards and Technology; USB, universal serial bus.

Additionally, the software layer of the telematics gateway should include a layer of abstraction between the CAN bus service and whatever platform the remote services process is running on. This reinforces defense in depth by obfuscating CAN access through automotive APIs, meaning even a rogue application would have limited functionality. A visualization of this idea is represented in Figure 4.

The service provider for the cellular network may also take a number of steps to reduce the number of potential attack vectors. If the remote operation service depends on SMSes for some of its activities, the service provider ought to wholly secure the SMS delivery procedure using network management tools such as whitelisting. Leaving this open is dangerous as a common use case for SMSes is to tell the radio to wake up and consume some sort of remote operation. A misfeasor could flood the radio with SMSes, which would keep the radio awake, eventually depleting the battery of the vehicle. At the data level, the service provider could whitelist IP addresses for only the backend services that directly communicate with the vehicle radio. This would prevent any sort of exploit in the radio's port configuration from being exploited. For example, if a radio had left open port 23 for telnet debugging, one could theoretically exploit this by determining the IP address for the vehicle's radio and attempting to telnet into it from a remote host if that port is not blocked at a bastion firewall. In the event that the radio is Unix based and does not have a root password, this is particularly devastating as the hacker would have unfettered access to the radio's operating system.

A number of software and operating system models for telematics security have been proposed in the academic community. One such model, proposed by a group of computer science researchers from Belgium, dichotomizes the runtime environment within a telematics module to a service runtime environment (SRE) and core runtime environment (CRE) [24]. The CRE hosts applications that demand real-time computing requirements and must be highly available. Thus, the CRE operates at a lower level and will block any "illegal" attempts by the SRE to modify its behavior or initiate invalid requests. The SRE may request data or action from the CRE through a cross-runtime interface, but all data are obfuscated to prevent any man-in-the-middle attacks. The SRE hosts all non-critical applications, such as infotainment services. Applications within the SRE may be updateable over the air and may be created by different entities (specifically companies), but all must be verified by an external entity using a mechanism such as digital signing. Furthermore, the architecture supports two types of entity access control: local and remote. These engines validate the authorization and authentication of requests within the local system and from a remote service provider to ensure all data are valid and come from a trusted source. A holistic approach such as this reinforces the principle of least privilege by isolating functions within certain runtime environments and managing their behavior through explicit access control components.

From the service delivery perspective, simple practices could be employed to secure the publish-subscribe architecture described earlier. For any device that attempts to connect to the message broker, employ a simple whitelist so that the user must be explicitly authorized to subscribe and publish on topics. This can be implemented simply by using an LDAP server, which also supports TLS [RFC2830]. This is a secure method for authorizing users, although it would add some overhead to connection setup times.

4.2. Digital signatures and the application runtime environment

A robust data validation mechanism must be employed within the telematics module, and eventually for all messages on the CAN bus. Within the context of the service runtime environment, all applications that are delivered and installed over the air ought to be signed and verified using public key infrastructure. A public–private key pair ought to be generated by the certificate authority, and the public key distributed to the telematics module manufacturer to be included in the SRE application management service. Any application intent for the module ought to be digitally signed by the telematics service provider using the private key. Using this mechanism, no application will successfully be installed unless signed using the service provider's private key. This is important in ensuring the validity of the applications being installed on the head unit and should prevent the installation of rogue applications given a strong enough hashing algorithm (such as SHA-256).

Furthermore, any access to the native operating system of the telematics module ought to be strictly regulated at the service runtime environment. Any application that requires read or write access to the filesystem ought to use explicitly programmed interfaces provided by the SRE so that the ability of applications to modify the filesystem is regulated. The SRE should provide obfuscated interfaces for persistent storage so that data are available between

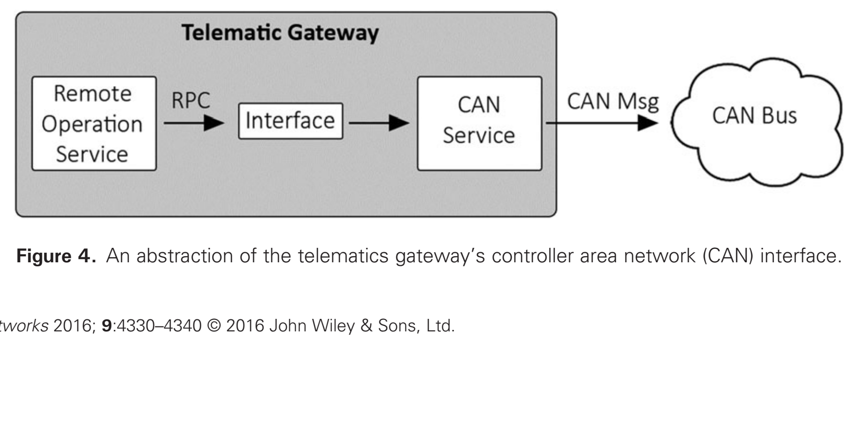

Figure 4. An abstraction of the telematics gateway's controller area network (CAN) interface.

power cycles. There should be no other requirement for any application to read or write to the filesystem.

At the layer below the SRE, we can also define a security architecture as it relates to a node on a network, whether it be a telematics network or VANET. One proposal attempts to define both the types of security requirements for a vehicle network node as well as the specific components where security measures may be implemented [25]. This includes service, system, and communication security requirements within a telematics architecture, as well as ensuring overall privacy of data. The architecture itself requires an independent security module within the system stack of the telematics module. That security module must support security within the communication stack as well as APIs to applications that must communicate on the network. The module is broken into two parts: the hardware module and the security APIs. The hardware ought to be tamper proof (such as a Trusted Platform Module or SmartCard), and the API component must use public key infrastructure. The API itself is broken into four parts: PKI, secure communication, privacy, and platform security.

This architecture is just an example of some of the proposals that provide a logical blueprint for designing a telematics module. Whether the software or hardware component, there are basic security requirements that need to be taken into account now that vehicles are connected to networks that may be open to compromise.

4.3. In-vehicle networking

Following the suggestions previously outlined will significantly bolster the robustness of any telematics offering, but the CAN protocol still does not contain any sort of meaningful security mechanisms. This violates the principle of defense in depth and ought to be addressed in the next iteration of the controller area network standard, or the automotive community should deprecate the use of CAN for embedded automotive networks.

One proposal from a group of researchers in Sweden calls for the use of a delayed data authentication mechanism using compound message authentication codes (Nilsson, 2008). This would utilize what is known as cipher-block chaining (in this case, cipher block chaining (CBC)-media access control (MAC)) where two ECUs share MAC keys. The MAC itself is calculated on the compound of four CAN frames, which are generally 64 bytes, meaning the MAC would cover 256 bytes worth of data. The biggest challenge in implementing a data authentication mechanism like this is ensuring that the CPU and memory overhead of this calculation is not obtrusive to ECU operation as all ECUs are built on embedded platforms with limited resources. The encryption algorithm proposed in this paper, KASUMI, requires about 10 kB in memory for code and 124 bytes for data. Each byte of plaintext requires 550 CPU cycles to encrypt on a 16-bit reduced instruction set computing (RISC)-based microcontroller, which would be common (if not underpowered) for many ECU implementations.

Many OEMs and suppliers, however, exhibit significant reservation in adopting this type of enhancement. Various reasons are cited such as cost of adding CPU cycles, implementation of protocol enhancements, and alignment of all ECUs to the standard. Unfortunately, it seems that time may not be on the side of the automotive community. Disruptors such as Tesla have changed the conversation with widespread use of Ethernet as an alternative to CAN, and many traditional OEMs have publically announced that they are investigating replacements for CAN. The greater scale provided by these newer standards gives manufacturers additional overhead for designing cryptographically secure messaging protocols and encryption standards for their vehicle platforms.

4.4. Additional countermeasures

Weimerskirch and Gaynier [26] of the Transportation Research Institute at the University of Michigan describe a separated architecture, firewall, and intrusion detection system (IDS) system that separates safety-critical network segments from external interfaces. The separated architecture

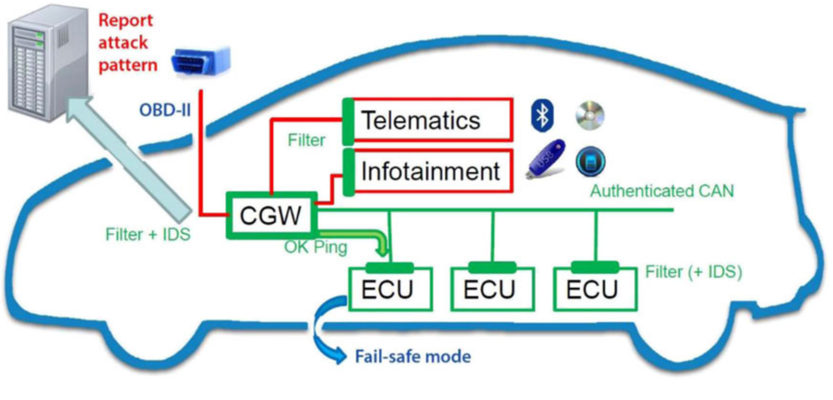

Figure 5. Separated Architecture, Firewall, and IDS [26].

will protect the safety-critical systems when a subsystem, for example, the infotainment system, has been compromised. A separated architecture will also provide protection for vehicle electronics from attacks to the OBD-II or from a compromised OBD-II. The separated architecture is illustrated in Figure 5.

As can be expected, the attack surface present for a connected car is far greater than what can be discussed within this context. Using the various types of threats identified within the STRIDE model, a number of countermeasures are cataloged in Table I with each mapped to the different types of threats it mitigates. This countermeasure catalog was built in partnership with the SBD automotive cybersecurity analyst team, which evaluates and tests these components on a daily basis. Note that this list is not meant to be exhaustive, but rather a survey of the most significant countermeasures for various threats to each component or attack point. As a reminder, the six component of the STRIDE model are as follows:

- spoofing user identity,
- tampering with data,
- repudiation,
- information disclosure,
- denial of service, and
- elevation of privilege.

5. CONCLUSIONS

Innovation and new technologies necessitate new security requirements. This is not a unique or new challenge, and it once again is required as new technology is introduced to the telematics space. The innovators of this technology ought to pay close attention to security requirements while designing these solutions and ensure that every level of the platform is secured in some way.

Overarching theme automakers may heed understanding what users want. If there is no demand for a particular service enabled through telematics, do not offer it. Providing services that have no demand only opens security risks and does not generate revenue. Drivers may not want fine control over their vehicle's behavior remotely. In fact, often times it is quicker and easier to configure these things within the vehicle itself.

Technology within vehicles will continue to evolve. From the controller area network, to the telematics gateway, to the backend services, and to the users themselves, security measures ought to be implemented at every layer of the technology stack so that the risk to driver safety and data is minimized. The current infrastructure at both the cloud level and the internal vehicle network has been demonstrably broken by researchers, and if active effort is not taken by automakers and telematics services providers alike to ensure a holistically secure ecosystem, more devious breaks of these systems could create massive problems for consumers in the future.

REFERENCES

1. Wired Magazine. 2015. https://www.wired.com/2015/07/hackers-remotely-kill-jeep-highway/ [Accessed on 9 May 2016].
2. US DoT. 2016. http://www.its.dot.gov/factsheets/dsrc_factsheet.htm [Accessed on 9 May 2016].
3. 5GPPP. 2014. https://5g-ppp.eu/wp-content/uploads/2014/02/5G-PPP-White-Paper-on-Automotive-Vertical-Sectors.pdf [Accessed on 9 May 2016].
4. Transportation. 2016. https://www.transportation.gov/smartcity/infosessions/dsrc [Accessed on 9 May 2016].
5. UMichigan. 2016. http://www.mtc.umich.edu/test-facility [Accessed on 9 May 2016].
6. National Instruments. Controller Area Network (CAN) Overview, 2011. Available at http://www.ni.com/white-paper/2732/en/ [Accessed on 9 May 2016].
7. Nilsson D, Phung P, Larson U. Vehicle ECU Classification Based on Safety-security Characteristics. *Proceedings of the Road Transport Information and Control (RTIC 2008)*, 2008; pp. 1–7 IET.
8. Microchip Corporation. Embedded Security, 2013. Available at http://www.microchip.com/pagehandler/en-us/technology/embeddedsecurity/technology/home.html [Accessed on 9 May 2016].
9. Eisenbarth T, Kasper T, Moradi A, Paar C, Salmasizadeh M, Manzuri Shalmani M. On the Power of Power Analysis in the Real World: A Complete Break of the KeeLoq Code Hopping Scheme. In *Proceedings of Crypto 2008, Volume 5157 of LCNS*, 2008; pp. 203–220. Also available from: http://www.iacr.org/archive/crypto2008/51570204/51570204.pdf [Accessed on 9 May 2016].
10. Aaronson X. We Drove a Car While It Was Being Hacked, Vice Motherboard, 2014. Available from: http://motherboard.vice.com/read/we-drove-a-car-while-it-was-being-hacked [Accessed on 9 May 2016].
11. Miller C, Valasek C. Adventures in Automotive Networks and Control Units, 2011. http://illmatics.com/car_hacking.pdf [Accessed on 9 May 2016].
12. Foster I, Prudhomme A, Koscher, K Savage, S. Fast and Vulnerable: A Story of Telematics Failures, Proc. of 9th USENIX Workshop on Offensive Technologies (WOOT 15), Washington, D.C., 2015, USENIX Association, 2015. (The report is also available from the UCSD website: http://www.autosec.org/pubs/woot-foster.pdf [Accessed on 9 May 2016].
13. Casner S, Hutchins E, Norman D. The challenges of partially automated driving. *Communications of the ACM* 2016; **59**(5):70–77.
14. NGTP. NGTP 2.0 Introduction, 2013. Available at http://www.ngtp.org [Accessed on 9 May 2016].
15. Checkoway S, McCoy D, Kantor B, Anderson D, Shacham H, Savage S, Koscher K, Czeskis A, Roesner F, Kohno T. Comprehensive Experimental Analysis of

Automotive Attack Surfaces. Center for Automotive Embedded Systems Security, 2011. Available from: http://www.autosec.org/publications.html [Accessed on 5 January 2016].

16. IOActive. 2015. http://www.ioactive.com/pdfs/IO-Active_Remote_Car_Hacking.pdf [Accessed on 9 May 2016].
17. Abdullahi M, Wang G. Secure Publish-Subscribe-Based In-Network Data Storage Service in Wireless Sensor Networks. *Proceedings of the 2012 IEEE 8th International Conference on Distributed Computing in Sensor Systems*, 2012; pp. 297–299 IEEE.
18. Wolf M, Weimerskirch A, Wollinger T. State of the art: embedding security in vehicles. *Journal of Embedded Systems* 2007; **2007** Article ID. 74706:1–16.
19. Weimerskirch A. V2X Security & Privacy: The Current State and Its Future, 2013. http://weimerskirch.org/papers/Weimerskirch_V2XSecurity.pdf [Accessed on 9 May 2016].
20. Eiza M, Owens T, Ni Q. Secure and robust multi-constrained QoS aware routing algorithm for VANETs. *IEEE Transactions on Dependable and Secure Computing* 2016a; **13**(1):32–45 IEEE.
21. Eiza M, Ni Q, Shi Q. Secure and privacy-aware cloud-assisted video reporting service in 5G enabled vehicular networks. *IEEE Transactions on Vehicular Technology* 2016b. doi:10.1109/TVT.2016.2541862 IEEE.
22. Laymin N, Vinel A, Jonsson M, Loo J. Real-time detection of denial-of-service attacks in IEEE 802.11p vehicular networks. *IEEE Communications Letters* 2014; **18**(1):110–113 IEEE.
23. Tetmeyer A, Saiedian H, Hein D. A tagging approach to extract security requirements in non-traditional software development processes. *International Journal of Secure Software Engineering* 2014; **5**(4):31–47.
24. Maerien J, Michiels S, Van Baelen S, Huygens C, Joosen W. A Secure Multi-application Platform for Vehicle Telematics, IEEE 72nd Vehicular Technology Conference Fall 2010, pp. 1–5, IEEE Digital Library, 2010.
25. Eichler S. A Security Architecture Concept for Vehicular Network Nodes, 6th International Conference on Communications & Signal Processing, pp. 1–5, IEEE Digital Library, 2007.
26. Weimerskirch A and Gaynier R. An Overview of Automotive Cybersecurity: Challenges and Solution Approaches, TrustED '15: *Proceedings of the 5th International Workshop on Trustworthy Embedded Devices*, pages 53–53, October 16, 2015, Denver, CO. The slides are available here: http://www.umtri.umich.edu/sites/default/files/Ron.Gaynier.UMTRI_.IT_.2015.pdf [Accessed on 6 May 2016], 2015.