CHAPTER

Security and data privacy of modern automobiles

6

Richard Brooks[1], **Iwinosa Aideyan**[1] **and Mert Pese**[2]

[1]*Department of Electrical and Computer Engineering, Clemson University, Clemson, SC, United States* [2]*School of Computing, Clemson University, Clemson, SC, United States*

6.1 Introduction

Surface transportation systems have improved safety, mobility, and environmental footprints since the introduction of advanced technologies, and a new paradigm, known as intelligent transportation systems (ITS), has been inducted to promote technology-focused research and development in the government and private sectors. In addition, traditionally automobiles have been mechanical devices, but advances in electronics and information and communication technology (ICT) have radically changed the industry. Modern automobiles are heavily computerized and increasingly networked. Electronic Control Units (ECUs) are embedded computer systems that control transmission, engines, brakes, air conditioning, entertainment systems, etc. These ECUs coordinate internal functions over multiple in-vehicle bus networks. Increasingly, automobiles are connected over external wireless networks to other vehicles, roadside units (i.e., ITS infrastructure), mobile devices, and original equipment manufacturer (OEM) service centers.

The evolution from traditional automobiles to connected vehicles has been accepted, in large part because it reduces manufacturing costs and increases automobile efficiency. However, these emerging technologies also bring real and reasonable concerns about the security and privacy implications of these changes.

The landscape of connected vehicles has witnessed monumental shifts. The integration of AI and machine learning, coupled with the advent of 5G technology, has not only enhanced vehicle connectivity but also introduced a new set of challenges. As vehicles become more intertwined with our digital personas, the boundaries between personal devices and vehicular systems blur, elevating the importance of robust security measures. The recent years have seen a surge in high-profile cyberattacks targeting connected vehicles, underscoring the evolving threat landscape. Moreover, collaborations between tech behemoths, innovative startups, and traditional automobile manufacturers have emerged, aiming to fortify vehicular systems against potential threats. As consumer awareness and expectations around vehicular data security grow, the industry finds itself at a crossroads. This chapter delves deep into the current challenges, vulnerabilities, and solutions in the realm of connected vehicle security and privacy.

Data Analytics for Intelligent Transportation Systems. DOI: https://doi.org/10.1016/B978-0-443-13878-2.00005-9

Typically, computer and network security requires securing all components in the infrastructure stack at multiple layers. This requires:

- Physical security,
- Communications security, and
- Application security.

Unlike traditional computing systems, where computers and routers are kept in isolated secure facilities, automobiles are left unattended in public places such as parking facilities for most of the day. This means that physical security may not always be possible. At the other end of the stack, automotive applications are usually implemented by a number of small companies for the OEMs. The small companies have small profit margins, and the OEMs have multiple vendors to choose from. While this is economically advantageous, the OEMs have limited control over the development process and limited liability for data security mistakes. These issues are representative of the security challenges facing the automobile industry. Other challenges include:

- Each automobile design has to integrate the needs of different stakeholders (including the OEM, component vendors, repair shops, people with leasing agreements, car dealerships, car owners, car fleet operators, the police and environmental regulators, transportation safety regulators, and transportation infrastructure operators) with multiple conflicting interests.
- The automotive supply chain is relatively complex, mainly consisting of OEMs and Tier 1 and Tier 2 suppliers. Tier 1s are the primary suppliers to OEMs. They provide the OEMs with major vehicle components and systems, such as engines, transmissions, chassis, and other critical parts. Tier 1s are typically large and well-established companies with the capacity to design, engineer, and manufacture complex systems. Examples for Tier 1s are Bosch, Denso, Harman, Continental, and Aptiv. Tier 2 is one step below Tier 1 in the supply chain. They provide Tier 1 suppliers with specialized components, subassemblies, or materials that are integrated into the final vehicle. For instance, Tier 1s buy the physical hardware for their Electronic Control Units (ECUs) from Tier 2, which tend to be semiconductor companies such as NXP, Infineon, and Renesas.
- Automobiles integrate a large number of communications networks, such as in-vehicle bus networks (e.g., Control Area Networks (CANs), Local Interconnect Networks (LINs), Media Oriented Systems Transport (MOST), FlexRay, and Automotive Ethernet), dedicated short-range communication (DSRC), Wi-Fi, vehicular ad hoc networks (VANETs), cellular networks, mandated tire pressure monitoring systems (TPMS), Wireless Personal Area Networks (WPANs), entertainment systems, and keyless entry systems.

In the realm of autonomous driving, the Robot Operating System (ROS) has been recognized as a pivotal tool. The second generation of ROS, ROS2, has been particularly influential in the development of self-driving car architectures. It offers the necessary reliability and real-time performance required for automated driving applications. ROS2 provides a framework that ensures safe and reliable real-time behavior while retaining the advantages of a distributed architecture and standardized message types. This integration has been instrumental in enhancing the capabilities of vehicles, making them smarter and more autonomous, and ensuring safer, more efficient operations on the road [1].

- As an automobile travels, it moves between multiple networks and network vendors. Handoffs between networks need to maintain both communication connections and security levels.

- Many vehicular applications and services are not designed with security in mind. A single implementation error in one component can be exploited to gain access to the bus networks. From the buses, it is easy to access and control the ECUs. This gives the attacker control over the entire vehicle, including the brakes and steering.
- ECUs can be accessed, apart from directly through the Onboard Diagnostics-II (OBD-II) port, remotely through wireless communication. This provides a much larger attack surface than most traditional systems.
- Multiple ECUs from multiple vendors are integrated into one single platform. The OEMs may, or may not, have access to the source code of vendor software. If two vendors make different assumptions, for example, they use different network packet sizes, this can create an exploitable vulnerability. To be certain that the system is secure, the OEM would have to test all possible combinations of components, which is not commercially attractive.

This chapter surveys connected vehicle security and privacy issues. In Section 6.2, we give an overview of communications networks and the innovative applications in connected vehicles. Section 6.3 identifies stakeholders within the automotive ecosystem and the assets they need to protect. An attack taxonomy that describes attacks on connected vehicles, originally in Ref. [2], is given in Section 6.4. We analyze existing attacks on connected vehicles and map them to the attack taxonomy in Section 6.5. Discussions of security and privacy solutions are presented in Section 6.6. Conclusions and future research directions (i.e., open issues) are presented in Section 6.7.

6.2 Connected vehicle networks and vehicular applications

6.2.1 In-vehicle networks

Modern automobiles are controlled by embedded computer systems, called ECUs. The number of ECUs is increasing; high-end vehicles have up to 120 ECUs. ECUs collect sensor data and control a broad range of automobile functions, including the powertrain, entertainment systems, brakes, power steering, and lighting. ECUs communicate over a number of in-vehicle bus systems, CANs, LINs, FlexRay, Automotive Ethernet, and MOST networks. CAN and FlexRay buses are for critical ECUs that require fast networks, such as the powertrain. LINs are for ECUs that require less transmission speed, such as lights, air conditioning, seats, and doors. MOST networks are mainly for infotainment systems such as audio, video, and voice. Automotive Ethernet is an emerging communication protocol in recent years that is a response to increasing bandwidth needs. It can be used standalone between two ECUs (point-to-point) or as a high-speed backbone between different functional domains (see Fig. 6.1). Because different bus networks use different physical media and protocol stacks, gateway ECUs are needed to read and write between the different buses and manage protocol conversions. A gateway ECU sends, receives, and translates messages between connected buses.

Wireless technologies are also used for ECU communications. Tire Pressure Monitoring System (TPMS) is mandated on modern automobiles in the United States and European Union. TPMS uses battery-powered TPM sensors mounted on the tires of a vehicle to continuously monitor the air pressure of all the tires. The sensors periodically broadcast the pressure and temperature measurements together with their unique identifiers to the in-vehicle TPM ECU. The transmission uses

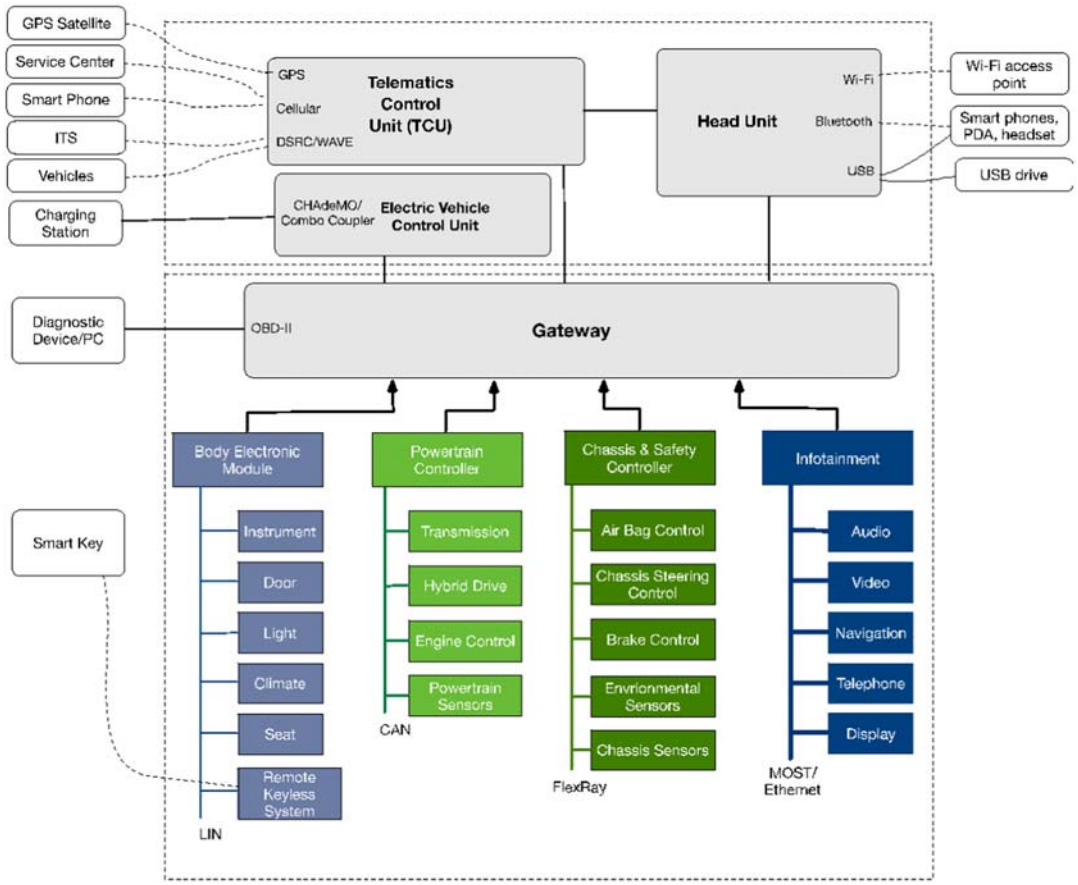

FIGURE 6.1

Architecture of connected vehicle networks.

Data from https://www.linkedin.com/pulse/information-security-connected-vehicle-shashank-dhaneshwar, last accessed on May 24, 2016 and H. Kitayama, S. Munetoh, K. Ohnishi, N. Uramoto, and Y. Watanabe, Advanced security and privacy in connected vehicles, IBM Res. Dev., 58, 1, 2014.

radio frequency technology. The TPM ECU, in turn, analyzes the data and triggers a TPM warning light and message on the vehicle board if the data suggest underinflated tires. TMPS increases overall road safety by detecting underinflated tires. TPMS also improves fuel economy because proper tire inflation improves traction and tire rolling resistance, and reduces braking distance. Similarly, antitheft systems (e.g., remote keyless entry, engine immobilizer, passive entry) are also common. Radio Frequency Identification (RFID)-based antitheft systems have an RFID embedded in a key or keyfob. The RFID communicates directly with a reader device in the car.

In-vehicle WPAN connects personal devices (e.g., cell phone, PDA, headset) via short-range wireless technology, most popularly Bluetooth. WPAN can be interconnected to internal buses via

a WPAN gateway ECU. This allows consumers to control lights, windshield wipers, airflow, heat, entertainment units, and many other features using a Bluetooth-enabled PDA or a Bluetooth-enabled headset with voice-activated control [3].

6.2.2 External networks

Connected vehicles can communicate with each other, or roadside units, by transmitting basic safety messages (BSM) using VANETs. The BSM data transmitted in VANETs include vehicle location, heading, speed, distance measured by millimeter radar, and traffic conditions. VANETs can use multiple wireless technologies: Wireless Access in Vehicular Environments (WAVE), Dedicated Short Range Communications (DSRC), Worldwide Interoperability for Microwave Access (WiMAX), Universal Mobile Telecommunications System (UMTS), Long Term Evolution (LTE), etc.

Connected vehicles also communicate with OEM service centers using cellular networks and traffic management centers through roadside units. Many manufacturers have business-to-vehicle offerings, such as Ford's Sync [4], GM's OnStar [5], Toyota's Safety Connect [6], Lexus' Enform [7], BMW Connected Drive [8], and Mercedes-Benz's Mbrace [9]. These services provide safety (crash reporting), roadside assistance (remote diagnostics), vehicle monitoring (location tracking, battery monitoring), and antitheft (remote engine stopping and locking) services.

Fig. 6.1 [10,11] shows an example architecture for connected vehicle networks. Internal bus networks communicate with external networks via gateway ECUs. Innovative connected vehicle applications, as envisioned in USDOT-developed connected vehicle reference implementation architecture (CVRIA) [12], leverage connected vehicle networks.

6.2.3 Innovative vehicular applications

Over-the-air (OTA) ECU update services connect a vehicle with cellular equipment to the OEM service center to remotely reprogram ECU firmware. Currently, ECU updates at a dealership can be time-consuming and inconvenient for vehicle owners and expensive for OEMs. Some OEMs now provide OTA updates for noncore ECUs. BMW, Audi, and Tesla have recently announced procedures for remotely updating navigation maps [13]. GM OnStar can remotely update its Telematics Control Units [14]. However, OTA updates for core ECUs are not common. Only Tesla publicly claims to remotely update its core ECUs. A remote update has security advantages and disadvantages. They let OEMs quickly correct security flaws for their entire fleet. However, if not implemented carefully, attackers can use this access to modify critical systems. Code signing is essential for remote updates to be certain that only authorized code is used. Poorly implemented auto-update systems have been used to infect machines in the past [15].

Over-the-air (OTA) ECU update services have seen significant advancements, especially with the introduction of the Uptane framework [16]. Uptane is specifically designed to enhance the security and reliability of OTA updates in automotive systems.

Smartphone apps are developed to control connected vehicles. They let customers remotely start/stop the engine, locate vehicles in parking lots, and lock/unlock doors. Apps can also monitor vehicle speed, mileage, and location. An iPhone app, iDriver, allows customers to steer cars remotely via their iPhones [17]. In Section 6.5.2.1, we discuss problems that have been found in current security systems. Given our inability to secure simpler systems, it may be reasonable to

worry about the potential security challenges of connected vehicles and future automated vehicles. *Vehicle Platooning* applications envision a number of vehicles cooperating to maintain a relatively short distance from each other and improve their fuel efficiency and road capacity. Vehicles use each other's position and velocity data for collaborative control to reduce their fuel consumption by reducing air drag. An *Automated Vehicle* is a self-driving vehicle. It senses its environment and navigates without human operations and will eliminate crashes due to human driver errors and problems such as falling asleep. Vehicle drivers are free from driving tasks and continuous monitoring of the environment. Automated vehicles are anticipated to improve fuel efficiency, increase road capacity/utilization, and reduce pollution. Automated vehicles are attracting the attention of IT companies, academia, and OEMs. Google announced that its prototype automated vehicle has driven hundreds of thousands of miles in self-drive mode under restricted conditions [18]. Recently, Google, Uber, and Tesla have made headlines by developing self-driving taxis for big cities [19]. While Google, Uber, and Tesla made initial headlines with their forays into self-driving technologies, their advancements have been substantial in the past eight years.

Waymo, a subsidiary of Alphabet (Google's parent company), has been at the forefront of these developments. In San Francisco, Waymo's autonomous ride-hailing service, known as Waymo One, has become a reliable mode of transportation for thousands of residents across various neighborhoods since late 2022. The service operates with an all-electric fleet, providing a sustainable and futuristic transportation solution. Waymo's introduction of robot taxis in San Francisco was a significant milestone, although it was met with both enthusiasm and concerns from city officials and activists [20].

Meanwhile, companies such as Uber and Tesla have also made significant strides in the autonomous vehicle sector, with Tesla's Autopilot feature and Uber's self-driving car trials marking notable achievements. As the landscape of transportation evolves, the integration of autonomous vehicles into our daily lives becomes more imminent, promising safer roads, efficient commutes, and a revolutionized automotive industry.

SAE defines 5 levels of autonomy in autonomous driving systems. Generally speaking, the higher the level of autonomy (and thus complexity), the larger is the attack surface due to additional perception, planning, and/or control modules:

- Level 1 (L1): The vehicle may have some driver assistance systems, such as adaptive cruise control or lane-keeping assistance, but human intervention is essential. The driver is required to be ready to take control of the vehicle at any time, as the vehicle cannot fully control any critical dynamic driving tasks on its own.
- Level 2 (L2): These are already widely available and target consumer vehicles. Examples include Tesla Autopilot, Cadillac Super Cruise, and OpenPilot. These systems require a human driver to be present and ready to take control if necessary.
- Level 3 (L3): The vehicle can handle some driving tasks but will require the driver to take over when the system requests; this level features substantial autonomous capabilities but still necessitates human intervention in more complex driving environments or scenarios. The driver can disengage from active control during specific conditions, but must be ready to take over if needed.
- Level 4 (L4): These systems are rapidly being deployed, targeting transportation services such as self-driving taxis, buses, trucks, and delivery robots. Some of these have already entered the commercial market. However, they can only operate in limited Operational Design Domains (ODDs).

- Level 5 (L5): These represent the ultimate goal of autonomous driving technology, capable of handling all possible driving scenarios without any human intervention.

VANET applications have been proposed to enhance safety (e.g., collision avoidance, lane-changing assistance), increase comfort (e.g., automatic toll/parking payment and fuel payment), and improve road utilization (e.g., congestion notification, route selection). These applications rely on other vehicles to provide reliable information, even though no reliable approach for mutual authentication has been proposed. If such a method were to exist, there would be worrisome privacy concerns.

6.3 Stakeholders and assets

The automobile ecosystem has witnessed significant advancements, with major IT companies playing pivotal roles:

- Google/Waymo: Initially known for producing automated vehicles as well as Android Automotive, which is poised to become the market leader for in-vehicle infotainment systems. Google's subsidiary, Waymo, has since made substantial progress. Waymo's autonomous ride-hailing service, Waymo One, has become a staple in cities like San Francisco, offering residents a futuristic mode of transportation [20].
- Microsoft: Microsoft continues to enhance its offerings for automobiles, with a focus on cloud computing and edge technologies that are revolutionizing the automotive industry.
- Apple: While Apple had announced plans for an electric vehicle, its focus has shifted towards developing advanced vehicular technologies and software solutions that integrate seamlessly with its ecosystem.
- Amazon: Amazon first revealed its electric delivery van in 2020, and now there are more than 1000 of these vans on the road. These vehicles, produced in collaboration with electric vehicle maker Rivian, are part of Amazon's Climate Pledge, aiming to reach zero net carbon by 2040. The vans currently make deliveries in cities such as Baltimore, Chicago, Dallas, and Seattle, among others. Amazon's CEO, Andy Jassy, emphasized that these vehicles were designed with driver input and are among the safest and most comfortable delivery vehicles today. To support this new fleet, Amazon has installed thousands of charging stations at delivery stations [21].
- GPS and Mapping Services: Google Maps and Apple Maps have evolved to provide more than just driving directions. They now offer real-time traffic updates, route optimization, and integration with other smart technologies.
- Communications Providers: The Continuous Air-interface for Long and Medium Range (CALM) [22] is an ISO initiative that defines a set of standardized wireless communication protocols and interfaces for ITS services. CALM supports uninterrupted transparent networking and handover between different communications networks, such as WiMAX, DSRC, Millimeter Wave, Wi-Fi, etc. This initiative decouples ITS services from the underlying communications technologies, allowing services to select networks through a standard interface [23].

The integration of these stakeholders and their assets has transformed the automotive landscape, making vehicles smarter, more connected, and more efficient.

Perfect security does not exist and would probably be prohibitively expensive if it did exist. A reasonable engineering approach considers the value of assets that need to be protected and uses limited resources to secure assets when it makes sense economically. Therefore, we identify the stakeholders involved in connected vehicle ecosystems and their assets to be protected. Fig. 6.2 shows the stakeholders and their assets. In the past, private vehicle owners only needed to protect their vehicles from being stolen. Modern vehicles generate, store, and transmit personal, private information. For example, many vehicular applications record vehicle locations and send the data to service centers. Internet access in vehicles is used by vehicle owners to access online banking, calendars, email, etc. Sensitive information (e.g., login, itineraries, and billing records) is stored in vehicles. Private vehicle owners should want to protect their data privacy. In contrast, fleet owners view vehicles as commodities and do not store sensitive information [24]. Thus, conflicts are common. For example, car rental companies track the driving history of their clients. Clients who speed or leave predefined regions must pay hefty fines. Needless to say, the clients would prefer for this information to be private.

Vehicle manufacturers have large inventories of automobiles to protect prior to sale. After sale, unauthorized modifications of ECU software can have warranty and liability implications for manufacturers and software suppliers. As automobiles become increasingly computerized, manufacturers make large investments in software development. This intellectual property must be protected as well.

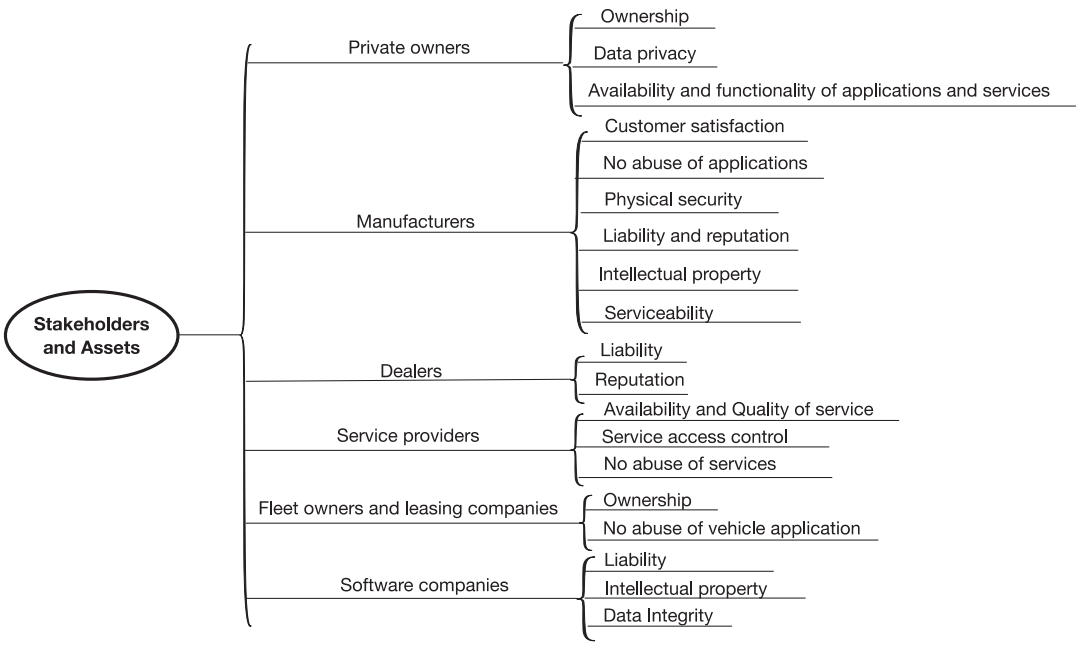

FIGURE 6.2

Stakeholders and their assets to be protected.

Extended from R.R. Brooks, S. Sander, J. Deng, and J. Taiber, Automobile security concerns, IEEE Veh. Technol., 4(2) (2009) 52–64.

Dealers need to protect automobile inventories, but intellectual property issues and unauthorized software modifications do not concern them.

Vehicle manufacturers offer services to their customers through cellular networks, making them service providers. Service providers generate revenues from service provision, so they restrict service access only to paying customers. To keep paying customers satisfied, they must guarantee both availability and quality of service, including security and privacy. Software companies that develop software for vehicle manufacturers also need to protect intellectual property and prevent unauthorized software modifications.

6.4 Attack taxonomy

Fig. 6.3 shows the attack taxonomy that describes potential attacks on connected vehicles. It is a modified version of the attack taxonomy developed by the Computer Emergency Response Team (CERT) to describe attacks to computer networks (i.e., computers connected through a network such as the internet or local area network) [25]. We analyze attacks on automotive systems using

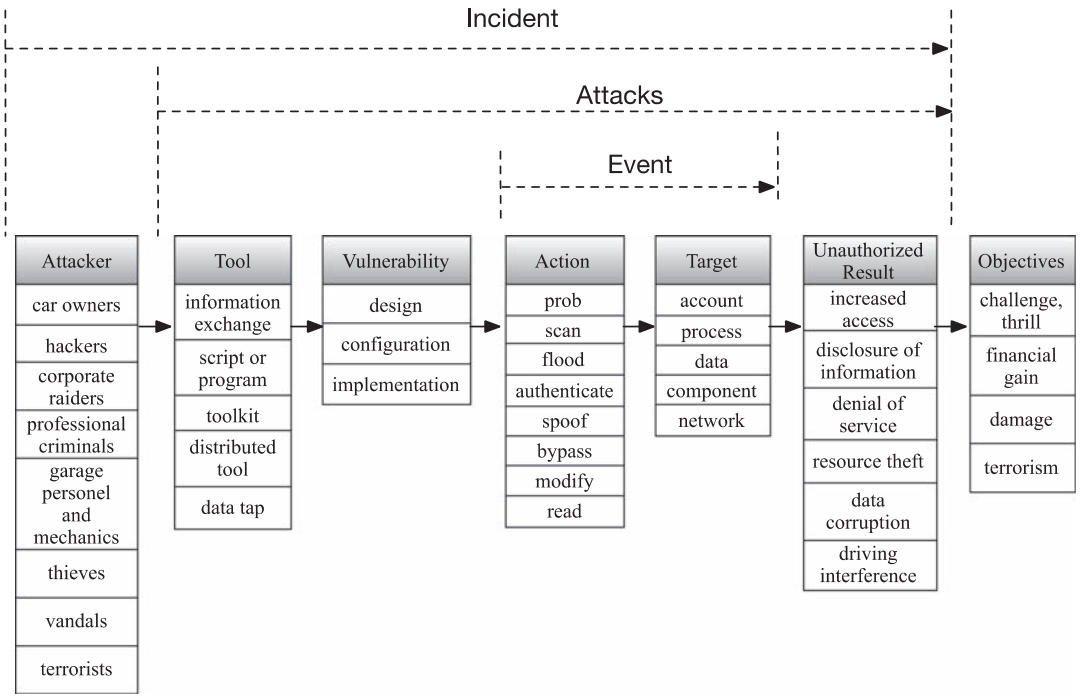

FIGURE 6.3

Attack taxonomy for connected vehicles, extended from the Computer Emergency Response Team attack taxonomy.

From R.R. Brooks, S. Sander, J. Deng, and J. Taiber, Automobile security concerns, IEEE Veh. Technol., 4(2) (2009) 52–64.

the taxonomy. Mapping attacks on automobiles to the taxonomy helps find the gaps between attacks and existing security solutions. It also finds common problems shared by known vulnerabilities, like the use of inadequately long cryptographic keys [26,27]. In Fig. 6.3, an incident occurs when an attacker launches a set of attacks to achieve an objective. In a specific attack, a tool exploits the vulnerability of a target to gain an unauthorized result. Each attack includes multiple actions to compromise specific target functions.

6.5 Security analysis

The internal bus networks of automobiles lack necessary security mechanisms. Wireless internal networks (e.g., TPMS and antitheft systems) are also vulnerable. In this section, we analyze the vulnerabilities of various connected vehicle networks and their protocol stack. Attacks leveraging these vulnerabilities will be enumerated. These attacks are mapped to the CERT attack taxonomy. Inconsistency was found during the mapping. Hence, the CERT taxonomy was modified to fit in the connected vehicle context.

6.5.1 Network and protocol vulnerability analysis

In-vehicle bus networks are broadcast and use shared access. No encryption is implemented. Messages on these bus networks are in clear text. No authentication is in place to verify the source of a message. And most of all, the buses are interconnected via gateway ECUs. Even though some gateway ECUs include firewalls, most of them allow diagnostic functions and interfaces that access all the internal bus networks without any restriction. Leveraging these vulnerabilities, a malicious ECU can eavesdrop (i.e., "read" in the language of CERT attack taxonomy), spoof, replay (i.e., "copy" in the language of CERT attack taxonomy) messages, and flood the network. A single compromised ECU on a bus could endanger all ECUs on the connected buses.

Controller area network vulnerabilities: The Controller Area Network (CAN) is the most prominent communication protocol in vehicles. It is a broadcast-based protocol, and every ECU is constantly listening to the traffic. An ECU decides to process the CAN message if its CAN ID matches its acceptance filter. Since CAN has not been designed with security in mind, recent research deals with enhancing it by the key cyber-security properties of confidentiality, authenticity, and availability. The main threat and key part of every cyberattack against vehicles to date are CAN injection attacks, which can lead to serious malfunctioning of the vehicle. Examples of malfunctioning include, but are not limited to, arbitrarily accelerating, braking, or steering the vehicle, but also tampering with body operations, such as breaking the HVAC or disabling the airbags. The most common form of CAN injection attack is message spoofing, which entails suspending the authentic broadcast from a legitimate ECU and transmitting a message with identical CAN ID but modified payload.

In particular, Pesé et al. [28] defined three categories of CAN injection attacks. (1) Fabrication attacks allow the adversary to fabricate and inject messages with a forged CAN header and payload at a higher frequency to override cyclic CAN messages sent by legitimate ECUs. (2) Suspension attacks on the compromised ECU prevent its broadcast of legitimate, potentially safety-critical

CAN messages to the intended recipient(s). Suspension attacks comprise Denial-of-Service (DoS) attacks that exploit the CAN protocol design. CAN messages with smaller IDs are higher priority and will always win arbitration on the CAN bus compared to messages with higher CAN IDs. An attacker can inject messages with low CAN IDs to effectively silence certain ECUs that transmit lower-priority messages. Another example is the bus-off attack, which takes advantage of the CAN error handling mechanism. In the absence of an adversary, the latter can prevent faulty ECUs from negatively affecting CAN communications by putting them in a bus-off state after a repeated number of communication violations. An attacker can easily exploit this mechanism to deliberately suspend legitimate, nonfaulty ECUs. (3) Masquerade attacks combine both of the above attacks by suspending the CAN broadcast of one ECU and deploying another ECU to fabricate malicious CAN messages. Last, but not least, eavesdropping on CAN traffic is possible due to the lack of encryption, which results in compromise of personal information or makes CAN bus reverse engineering possible. The latter is a necessary precursor to CAN injection attacks since the attacker needs to form a well-crafted CAN message to achieve a visible outcome of the vehicle malfunctioning. Attacks on the CAN bus (or interchangeably any in-vehicle network) can be grouped into three generations. Fig. 6.4 shows the three generations of in-vehicle network attacks.

First-generation attacks that started with the rise of automotive security literature in the early 2010s. Chekoway and coworkers [29,30] were mostly targeting physical interfaces, that is, the attacker needed to have physical access to their victim vehicle. Once the attacker was inside the vehicle, they could simply access the in-vehicle network (IVN) through a physical connector called the Onboard Diagnostics (OBD-II) port. This interface is mandated in all US gasoline vehicles manufactured after 1996. With physical access to this port, it is possible to launch the aforementioned CAN injection attacks but also update ECU software on a CAV using the Unified Diagnostic Protocol (UDS). Key mitigations against CAN-based first-generation attacks include secure software design, digitally-signed messages, and ECU fingerprinting to assure messages were sent from the legitimate ECU. However, implementing the aforementioned mitigations on the CAN bus is nontrivial. Providing message authentication is difficult due to the limited amount of space in CANs data fields along with the necessity of exchanging data in real time.

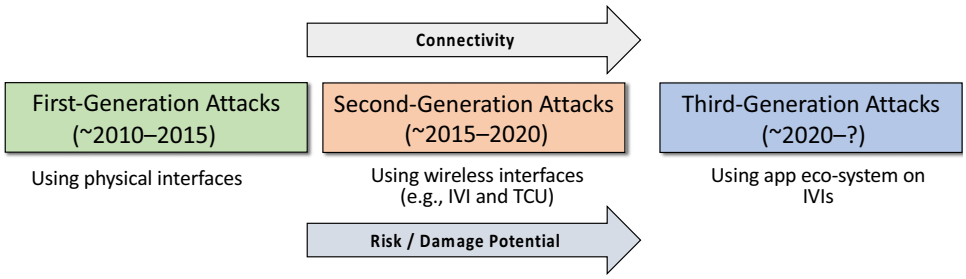

FIGURE 6.4

Three attack generations.

Adapted from M.D. Pesé, Bringing Practical Security to Vehicles (Doctoral dissertation). Miller, C., & Valasek, C. (2015). Remote exploitation of an unaltered passenger vehicle. Black Hat USA, 2015(S 91), 2022, pp. 1–91.

Furthermore, car makers deploy computationally weak ECUs due to cost reasons that limit the implementation of cryptographic algorithms. The added latency at the sender and receiver for encrypting/decrypting, as well as signing/verifying messages, has a direct impact on hard real-time deadlines on the CAN bus, which cannot be missed to conform with functional safety. Alternative mitigation strategies to add a layer of security while addressing these constraints have been proposed in recent literature [28]. In recent years, AUTOSAR SecOC [31] has emerged as an accepted solution among OEMs to add basic security principles to the CAN bus.

Second-generation attacks went further and tried to gain IVN access without being physically inside a vehicle. For this purpose, attackers would exploit vulnerabilities in the wireless interfaces of ECUs, for example, the Wi-Fi or cellular connectivity of Telematic Control Units (TCUs). TCUs are devices embedded into automobiles that integrate various services and features into the vehicle. They provide connectivity using Wi-Fi, Bluetooth, GPS, and mobile data interfaces. Many TCUs are part of In-Vehicle Infotainment (IVI) systems that comprise car radios and navigation systems. Since TCUs are connected to the IVN, CAN injection attacks that have been discussed in the previous subsection can be launched through a compromised TCU analog to first-generation attacks. The most comprehensive and impressive attack of this generation was the famous Jeep hack that happened in 2015 [32]. These hackers were able to obtain CAN bus access through several vulnerabilities in the TCU's software and hardware and could kill a running Jeep Cherokee on the highway (or steer it into a ditch). As a result of this hack, 1.4 million vehicles had to be recalled, and a lawsuit that had been filed against the OEM and Tier-1 was just dismissed in 2020. Compared to the first generation of attacks, this generation is more feasible to conduct since no physical access is required. As a result, the risk and damage potential increase. Furthermore, these attacks are also more scalable, as the high number of recalls proved. A large influence in executing TCU attacks derives from the simplicity of publicly discoverable devices. Targeted devices were discovered by scanning known ports over specified IP address ranges. Using Network Address Translation (NAT) would substantially aid in obscuring the identity of target devices. Removing the transceiver interface from TCUs to prevent CAN communication is another suggested course of action; however, this may cripple many of the TCUs functional purposes.

Finally, *third-generation* attacks take the scalability and damage potential even further. As of the time of this writing, there are no known attacks yet, although the technology required for it is slowly maturing. A new IVI operating system named Android Automotive OS (AAOS) was announced by Google in 2017. A custom flavor of the popular Android mobile operating system, its most distinct feature is its ability to connect to the IVN and read, as well as write data to it. Third-party apps will be gradually supported in a custom Play Store but are limited to media, messaging, navigation, parking, and charging apps at the moment. With an increasing number of third-party apps, as well as OEMs heavily customizing AAOS, there is serious security risk coming from this platform [33]. Now, malicious entities will be able to access the vehicle and its IVN from anywhere, opening the doors for significant damage potential.

LIN vulnerabilities: LIN uses a single-master multislave architecture where only the master ECU of a LIN network can initiate a message, and it polls a slave ECU to respond to a message. Attacking this single point of failure is promising [34]. In LIN, the master can force a slave to sleep by sending a special message. A malicious LIN ECU can spoof the message to deactivate the entire LIN network. A LIN message contains a SYNC field in the message header. The master ECU sets this field to a predefined value to instruct slave ECUs to synchronize. A malicious LIN ECU can spoof the message and modify the SYNC field to disrupt the synchronization.

MOST vulnerabilities: In a MOST network, one MOST device acts as a timing master that periodically sends timing frames that cause MOST slaves to synchronize. A malicious MOST device can fabricate malicious timing frames to interrupt the synchronization. The MOST protocol allows for bandwidth contention. A fixed-length communication segment, called the dynamic segment, is available periodically so that MOST devices can contend for it. Contention depends on message priority. The device with the highest message priority wins. The winning MOST device can transmit until it either finishes its transmission or another device with higher message priority joins the contention. A malicious MOST device can jam the segment by spoofing high-priority messages.

WPAN vulnerabilities: WPAN using Bluetooth is very popular. Security measures including authentication and encryption are used in Bluetooth; however, the security is weak [35]. The keys for encryption and authentication in Bluetooth are based on the Bluetooth device address and a PIN. However, the address of each Bluetooth device, which is a 48-bit unique address assigned by the device manufacturer, is public information. Any Bluetooth device can know the address of any other neighboring Bluetooth device by simply inquiring for it [35]. The PIN is also comprisable. Generally, the PIN is a 4-digit, user-entered code like in standard mobile phones. In some worse cases, the PIN is a nonvolatile built-in at the factory. Therefore, Bluetooth security is crackable. WPAN is interconnected to internal buses; compromised WPAN can be an attack vector (i.e., entry point) to ECUs.

TPMS vulnerabilities: TPMS messages are broadcast over radio. Each TPMS message contains sensitive data: temperature, tire pressure, and a unique identifier. TPMS lacks basic security measures. The message transmissions are not encrypted, and the TPM ECU trusts all received TPMS messages without input validations. Eavesdropping, reverse engineering, and spoofing attacks are possible. TPMS wireless broadcasting also presents a privacy risk, as the unique identifier identifies and tracks a vehicle. The risk is greater since TPMS use is mandatory and it is hard to deactivate.

Antitheft system vulnerabilities: The common wireless antitheft systems for modern automobiles include remote keyless entry, passive keyless entry and start, and engine immobilizers. A key communicates with the car over a wireless channel. For security, challenge and response protocols or secret codes usually authenticate the key to the car, and the communications are encrypted. However, inadequate encryption key length and imperfect cipher structure make it possible to crack these antitheft systems [26,36]. This is often excused by explaining that physical attacks on the car are less expensive than cryptanalysis. It is worth noting that engine immobilizer bypass kits are very inexpensive and sold online.

VANET/V2X vulnerabilities: Connected vehicles talk to each other or roadside units through VANETs. Vehicle-to-vehicle communications are ad hoc. Self-organized networks, formed voluntarily, are dynamic since vehicles frequently join and leave the network. Message authenticity must be verified. Though some mechanism exists to detect fake messages, an attacker can always forge a large number of messages to drown out authentic messages. This may include a Sybil attack, where one car pretends to be many cars. VANETs data include vehicle location, velocity, distance between vehicles, and traffic conditions. A malicious user can misuse the data to track another vehicle. Without reliable authentication, arbitrary packet spoofing is trivial, which can subvert any VANET application. However, reliable authentication would present a privacy risk. In recent years, V2X is a more common and comprehensive term that replaced the use of VANETs.

Vulnerabilities also arise when system implementations either deviate from protocol and standard specifications or when specifications are vague. Results in Ref. [29] reveal the bus implementation vulnerabilities in a car. One potential vulnerability is bus interconnections through ECU gateways. To mitigate the risk, the standard implicitly defines that high-speed bus networks are more trusted than low-speed buses [29]. High-speed buses connect real-time safety-critical ECUs (e.g., engines, brakes), while low-speed buses connect ECUs that are less critical to safety, such as seats and air conditioners. An ECU connected to a low-speed bus should be unable to send packets to a high-speed bus. However, Koscher et al. [29] found this not to be the case in some car models.

6.5.2 Attacks

Existing attacks on connected vehicle systems target VANETs, antitheft systems, internal buses and ECUs, TPMS, and WPANs. Here, an analysis of these attacks is presented and mapped them to the CERT attack taxonomy in Tables 6.1 and 6.2.

Table 6.1 Attackers of antitheft systems and their objectives [2].

Attacker	Objectives
Thieves	Steal cars and costly car components
Vandals	Crack automotive antitheft protection
Hackers	Crack automotive entry system for fun
Professional criminals	Gain access to automobiles

Table 6.2 Attacks on antitheft systems.

Attack	Tool	Vulnerability	Action	Target	Unauthorized result
Keeloq attack [36]	Info. exchange	Design (short key; short block size; and similar key schedule)	Read, authenticate	Data (encryption key)	Disclosure of information (encryption key)
DST attack [26]	Info. exchange	Design (short key; cipher function and structure)	Read, authenticate	Data (cipher function, encryption key)	Disclosure of information (encryption key and cipher function)
Relay attack [37]	Toolkit	Design	Spoof		Resource theft
Bypass kit	Toolkit	Design	bypass		Resource theft
Jamming	Toolkit	Design	Flood	Network	Denial of service
RollJam	Toolkit	Design	Scan, copy	Data	Resource theft

6.5.2.1 Antitheft system attacks

Automobile antitheft systems prevent unauthorized physical access into automobiles. Potential attackers to break antitheft systems and their objectives [2] are listed in Table 6.1. Successful breaking of antitheft systems not only gives attackers physical access to the automobiles but also various onboard vehicular applications. Thieves usually adopt low-tech attacks to steal cars, such as jimmying the lock, looking for keys left in automobiles, cutting alarm wires, and hot wiring the ignition. High-tech cyberattacks on wireless antitheft systems are increasing, especially targeting high-end cars [38]. Table 6.2 maps existing attacks on antitheft systems to the CERT taxonomy.

Keeloq is a 32-bit block cipher used in many remote keyless entries to encrypt the communication between a key fob and a car. Keeloq uses a 64-bit cryptographic key and comprises 528 identical rounds of encryption/decryption cycles. Each cycle of encryption/decryption is equivalent to a non-linear feedback shift register (NLFSR). A key recovery attack [36], revealed the NLFSR and uncovered the encryption key of a Keeloq, exploiting three weaknesses of Keeloq: short key length, short block size, and the existence of an efficient linear approximation of the NLFSR.

The digital signature transponder (DST) is an RFID device that has been used in over 150 million engine immobilizer keys. A DST uses a 40-bit cryptographic key. In its communication with a car, a DST emits a factory-set 24-bit identifier and then authenticates it with a challenge and response protocol. The car initiates the protocol by transmitting a 40-bit challenge. The DST encrypts this challenge using its 40-bit key, truncates the encrypted challenge to a 24-bit response, and returns it back to the reader. An attack on a DST used in a Ford car uncovered its encryption key after just harvesting two challenge-response pairs [26]. Then the attacker was able to clone the DST and used the cloned DST to unlock the car. The attack first used reverse engineering to uncover the complete functions of the encryption cipher, followed by a brute force attack to obtain the key.

The relay attack [37] on passive keyless entry and start is simple and requires no cryptanalysis expertise. A passive keyless entry and start, unlike remote keyless entry, does not need human action to unlock cars. A car periodically scans for a passive key. Once a passive key enters the proximity, the car authenticates the passive key. The car sends a low-powered signal. Passive keyless entry and start only work when the passive key is close to the car (<2 m). The relay attack intercepts the radio signals between a car and a passive key via antennas acting as repeaters, resulting in the passive key activating the car even when it's nowhere near it. The attack has been successfully tested on ten models from eight manufacturers.

Bypass kits can be an attack vector for antitheft systems. Bypass kits are interface kits used to momentarily bypass the antitheft system. They are produced by OEMs and sold to manufacturers. Sometimes manufacturers need "an additional interface kit to allow an after-market (not factory installed) system to work properly" [39]. With the easy availability of bypass kits [40] and a growing market for stolen luxury cars, this particular crime is likely to be cost-effective in the near future.

Radio jamming is another attack on wireless antitheft systems. A thief can use key fob jammers purchased online to block the radio frequencies emitted from the remote keyless entries to lock the door. More and more radio jamming crimes have been witnessed and reported to police [41,42]. This type of attack can be detected. Normally, when a car is locked, it flashes its lights and sounds a beep. When the attack occurs, there is no flashing light or beep.

RollJam is a very small device, available on eBay for less than $50, that attacks remote keyless entries and unlocks cars almost at will [43]. An attacker just needs to put RollJam near a target vehicle and wait for the victim to use the keyfob within the radio range of RollJam. The victim will

notice that the keyfob does not work on the first try but works on the second. Later, the attacker can retrieve RollJam; press a button on RollJam to unlock the car. RollJam exploits the design that remote keyless entry uses a set of rolling secret codes. Remote keyless entry changes the secret code every time it is used. A code is rejected by the car if it is used a second time. When a victim uses the keyfob to lock the car for the first time, RollJam jams the signal and records the first code, so the keyfob button press does not work. Naturally, the victim tries the keyfob again. RollJamm jams for the second time, records the second code, and plays the first code at the same time. This time the car will be locked. In this way, RollJam has stored a secret code that can be used next time.

6.5.2.2 Electronic control unit attacks

Table 6.3 lists major ECU attackers and their objectives. Automotive hobbyists often change their ECUs for enhanced power or sportive shock calibration. Some dishonest car owners modify mileage when selling their cars. European truck drivers tamper with tachographs to avoid punishment for driving extended hours. Garage personnel steal sensitive customer data that is stored in cars.

An attacker needs to access ECUs to deliver malicious inputs. Checkoway et al. [30] analyzed a full range of I/O channels available in a modern automobile and identified the challenges required to access each channel. Table 6.4 lists the channels that can be used to deliver malicious inputs to ECUs. ECUs can be accessed directly through the OBD-II port, which is federally mandated in the United States and can be found under the hood in virtually all automobiles. The OBD-II port is

Table 6.3 Attackers of electronic control units and their objectives [2].

Attacker	Objectives
Private owners	Change ECU software to gain more shock calibration, enhanced power, improved brake behavior, or change digital tachometers
Garage personnel	Sensitive information
Corporate raiders	Obtain proprietary information
Hackers	Hacking for fun

Table 6.4 Channels for attacking electronic control units [30].

Channel	Exploitation
OBD-II port	Plug a malicious hardware directly into the OBD-port
CD player	Automotive media systems are connected to internal buses. A compromised CD can attack other ECUs
PassThru	An attacker can either gain control of the PC running the diagnostic software or the connection between the PC and the PassThru device
Bluetooth	Buffer overflow with paired Android phone and Trajan app, or brute force the PIN of Bluetooth network
Cellular	Reverse engineer the software that a car's telematics unit uses to establish connections between the car and a remote center and authenticate the car

intended to be used by car owners and garage personnel to access ECUs for diagnostic purposes. A manufacturer-specific tool is plugged directly into the OBD-II port to access the internal buses and retrieve diagnostic information from ECUs. An attacker, by directly plugging in malicious hardware to the OBD-II port, can gain control of the vehicle [29]. However, direct access to the OBD-II port is a strong requirement for attackers. The external networks of connected vehicles present a much wider attack surface. Nowadays, vehicle diagnostics can be carried out using PCs instead of dedicated tools. A PassThru device (typically a USB or Wi-Fi device) is plugged into the OBD-II port, and a PC running manufacturer diagnostic software connects to internal buses via the PassThru device. An attacker can either gain control of the PC running the diagnostic software or the connection between the PC and the PassThru device. The Bluetooth WPAN can also be leveraged to deliver malicious inputs to ECUs through a compromised personal device that is connected to the WPAN. The long-range cellular networks offer many advantages for attackers. Cellular networks are generally used to connect a vehicle to the OEM service center. Chekoway et al. [30] reverse engineered the software that a car's telematics unit uses to establish connections between the car and a remote center. They were able to interpose the connections and send arbitrary packets on the connections.

Table 6.5 lists existing attacks in the literature on internal buses and ECUs. Koscher et al. [29] conducted intensive exploits of CAN buses. To begin, they developed CarShark, a CAN sniffer to observe the traffic on the CAN buses. Together with a combination of replay and informed probing, they unveiled how ECUs communicate with each other and the valid packets to control various ECUs, including radio, body control modules, etc. Koscher et al. also conducted fuzzing attacks and reverse engineering to understand how certain hardware features were controlled. They demonstrated that through CarShark eavesdropping and probing, fuzzing attacks, and reverse engineering, they were able to take full control of a wide range of individual ECUs, including radio, body controller, engine, brakes, and HVAC. An automated reverse engineering tool for the CAN bus has been proposed by Pesé et al. [44] and can identify both powertrain and body information. Koscher et al. also implemented composite attacks that exploit multiple ECUs. For example, they manipulated the speedometer to display an arbitrary offset of the current speed, such as 10 mph less than the actual speed. This attack intercepts actual speed update packets on the low-speed CAN bus and transmits maliciously-crafted speed packets with the falsified speed to the display.

Table 6.5 Attacks on internal buses and electronic control units.

Attack	Tool	Vulnerability	Action	Target	Unauthorized result
CAN injection attack [29]	Script or program	Design	Read, spoof, probe	Data	Disclosure of information (e.g., valid CAN messages), Data corruption, DoS
Fuzzing attack [29]	Script or program	Design	Spoof	Data	Disclosure of information (CAN functionality)
Reverse engineering [29,44]	Toolkit (CAN ReadMemory, IDA pro)	Design	Read	Data	Disclosure of information (CAN functionality)
Bridging internal CAN Bus networks [29]	Script or program	Design	Modify, spoof	Network	Increased access

Another significant attack performed in Ref. [29] targets the interconnections of internal bus networks. Each vehicle includes multiple buses, each of which hosts a subset of the ECUs. For functionality reasons, some buses must be interconnected; thus, a small number of ECUs are physically connected to more than one bus and act as logical bridges. Perfect and safe network segmentation will not allow ECUs on the low-speed network to access the high-speed network. However, it is not the case in real implementation. It was found that the telematics unit on a car connected to both low-speed and high-speed networks being exploited. By reprogramming the telematics unit, the ECUs on the low-speed bus were able to disrupt the function of critical ECUs on the high-speed bus. This increased access to critical ECUs can lead to disastrous outcomes.

Two security researchers wirelessly hacked a running Jeep, taking over dashboard functions, steering, transmission, and brake control [32,45]. This attack exploits the software vulnerability in Chrysler's Uconnect infotainment system. An open port in the Uconnect system allowed the researchers to remotely compromise the ECU. Further vulnerabilities in the proper separation of the CAN controller from this ECU allowed the aforementioned CAN injection attack. Shortly after the hack, Chrysler announced a formal recall for 1.4 million vehicles to patch the software vulnerability.

A light-weight side-channel analysis over CAN bus is presented in Ref. [46]. Other attacks on ECUs are presented in Ref. [47,48]. However, these attacks, compared to the attacks in Ref. [29], are weaker and can be achieved using the techniques presented in Ref. [29] as well.

6.5.2.2.1 Fuzzing techniques for controller area network versus robot operating systems

The integration of the Controller Area Network (CAN) bus with the Robot Operating System (ROS) is pivotal in the realm of autonomous military fleets and unmanned systems. This amalgamation bridges the gap between the hardware-level communication facilitated by the CAN bus and the high-level functionalities offered by ROS. Such integration not only streamlines data flow across sensors, actuators, and control units but also enhances vehicle efficiency and responsiveness. By unifying the communication landscape, ROS provides a modular architecture for software development, fostering seamless communication with CAN-based hardware. This accelerates the development of new functionalities, crucial in dynamic military operations, and enhances the vehicle's overall autonomy and safety.

In a recent study [49], the effectiveness of fuzzing techniques on this integrated system was evaluated. Fuzzing, a software testing technique, involves injecting randomized data inputs into a system to uncover vulnerabilities. The research aimed to determine whether fuzzing the unified CAN-ROS system would be more effective than fuzzing its individual components separately. Three distinct fuzzers were employed in the study: CAN fuzzer, ROS fuzzer, and CANROS fuzzer (combination of both, fuzzing the integrated CAN-ROS system simultaneously). The study tested three hypotheses:

- Fuzzing the unified system is more effective in uncovering vulnerabilities and inducing crashes than fuzzing its individual components in isolation.
- There exists a statistically significant difference between the performance of the ROS fuzzer on the CAN bus compared to the CAN fuzzer in terms of crash induction and vulnerability detection.
- There exists a statistically significant interaction between the fuzzing scenarios and the types of bugs injected (dereferencing, buffer overflow, and dividing by zero) concerning the ECU functions.

The findings demonstrated that the fuzzer, which employed both fuzzers simultaneously, had the better performance in terms of crash timings, hence suggesting its effectiveness in promptly identifying software bugs. Although both the CAN Fuzzer and CANROS Fuzzer demonstrated competitive performance in specific cases, the efficiency of the ROS Fuzzer was comparatively lower.

Given these emerging threats, it's imperative for automotive manufacturers and cybersecurity professionals to adopt robust security measures to safeguard ECUs and the broader vehicle network. Continuous monitoring, vulnerability scanning, and timely patching are essential to ensuring the security and safety of modern vehicles.

6.5.2.3 Tire pressure monitoring system attacks

In a TPMS, battery-powered TPM sensors mounted on the wheels of an automobile transmit packets periodically (every 60–90 seconds required by the National Highway Traffic Safety Administration (NHTSA)) to the in-vehicle TPM ECU. The transmission power of TPM sensors is relatively small in order to prolong sensor battery life. Despite the low data rate, low transmission power, and high travel speed of automobiles, eavesdropping TPMS communications is feasible. Rouf et al. [50] demonstrated that with inexpensive hardware, the communications can be easily overheard from a distance of over 40 m away from a passing automobile. Reverse engineering TPMS communications to obtain the unique identifier contained in every TPMS packet is also feasible, given that the communications are unencrypted. The identifier can be used to track an automobile. Rouf et al. in Ref. [50], demonstrated the feasibility of tracking an automobile traveling at 60 km/hour. Message spoofing is another attack on TPMS due to the lack of authentication, input validation, and proper filtering. It is shown in Ref. [50] that the in-vehicle TPM ECU of an automobile accepted forged packets at an increased rate of 40 packets per second, while the expected packet rate is much smaller. It is also shown in [50] that an attacker vehicle was able to fool a victim car traveling at the speed of 110 km/hour to turn on the low-pressure warning light using spoofed packets, which potentially can drain the vehicle battery. Table 6.6 summarizes these attacks.

6.5.2.4 Vehicular ad hoc networks/vehicle-to-everything attacks

The attackers that are likely to compromise VANET communications and their objectives are listed in Table 6.7. Terrorists could misuse VANETs in the hope of creating traffic havoc. Greedy drivers that want to clear traffic along their path could fool other vehicles into choosing other paths [51].

Table 6.6 Attacks on tire pressure monitoring systems.

Attack	Tool	Vulnerability	Action	Target	Unauthorized result
Eavesdropping	Toolkit (frequency mixer, USRP)	Design	Read	Data	Disclosure of information
Identity exposure	Toolkit (TPMS trigger tool, low noise amplifier, USRP)	Design	Read	Data (identity)	Disclosure of information
Packet spoofing	Toolkit (Frequency mixer, TPMS trigger tool)	Design	Spoof	Data	Denial of service

Table 6.7 Attackers of vehicular ad hoc networks and their objectives [2].

Attacker	Objectives
Terrorists	Can cause harm or hysteria
Greedy drivers	Can clear their route/path by redirecting other traffic
Vandals or hackers	Can access anything

VANET attacks can be classified into five categories [52]:

- Network monitoring: an attacker monitors the whole network and listens to the communications.
- Social attack: an attacker disturbs the victim vehicles with insulting messages and indirectly causes problems in the victims' driving behaviors.
- Timing attack: an attacker maliciously delays the message and messes up the time-critical safety applications.
- Application attack: an attacker tampers the contents of the messages and causes accidents by sending the fake safety messages.
- DoS attacks: an attacker consumes network bandwidth by message flooding and ID spoofing.

The future of intelligent transportation systems (ITS) will be spearheaded by vehicle-to-everything (V2X) communication. Whereas VANETs focus mostly on vehicle-to-vehicle (V2V) and vehicle-to-infrastructure (V2I) communication, V2X is a broader term that encompasses all types of communication between vehicles and other entities, including vehicles (V2V), infrastructure (V2I), pedestrians (V2P), and more. V2X is one of the complementary technologies to enhance and support Advanced Driver-Assistance Systems (ADAS) and CA versus Primarily, the V2X communication range is greater than the sensing ranges of current ADAS and CAV sensors, such as radar, LiDAR, and cameras. V2X can be performed over either a cellular network or a short-range network such as DSRC (Dedicated Short Range Communications) and LTE Sidelink. V2X can also be used with cars that have a lower level of automation, such as traditional cars, and help them avoid traffic congestion and collisions. For these purposes, vehicles exchange Basic Safety Messages (BSMs) in the United States, which are defined in the SAE J2735 standard. BSMs contain state information about a vehicle, such as its location, speed, acceleration, heading, and yaw rate. Vehicles listen to BSM broadcasts and can plan their future actions accordingly, for example, by slowing down or speeding up. This can enhance road safety as long as there is no malicious interference. The field of V2X security has received increasing scrutiny over the last decade, with various standardization bodies in the United States and Europe working to add security to the respective V2X protocols. Depending on the attackers' capabilities and attack types, a holistic, multilayered security concept is required. For instance, BSMs from external attackers (e.g., roadside attackers with V2X radio) will be discarded immediately due to lack of valid credentials to join the BSM broadcast.

In contrast, internal attackers (e.g., compromised ECUs) are "real" vehicles that are authenticated to exchange BSMs with their surrounding vehicles and other entities. They can launch a variety of attacks, such as denial-of-service (DoS), sybil, replay, or false data injection. Compared to the first three attack types, which are all of adversarial nature, false data broadcast can also be

caused by faulty sensors in nonmalicious vehicles. This can be achieved through a compromised in-vehicle network or a malicious on-board unit (OBU)—the vehicle's external interface responsible for V2X communication.

A complete list of attacks on VANET/V2X scans can be found in Refs. [53,54] and are mapped to the attack taxonomy in Table 6.8. In a Sybil attack, one vehicle may spoof hundreds of other vehicles to send false road congestion information to fool the nodes and vehicles in the VANETs. DoS attacks are possible, where a large number of spoofed packets absorb all available bandwidth to make the system unavailable for legitimate users. If the attacker launches attacks in a distributed manner from different locations, it becomes distributed denial-of-service (DDoS), which will amplify the power of DoS. Attackers can also send spoofed information to fool the victim vehicle, causing it to halt abruptly to create traffic accidents that may lead to a chain reaction of multivehicle rear-end collisions. In location tracking, an attacker may use data mining technology to track the locations of a vehicle and send spoofed location information for their own benefit. Malicious code/malware/spam attacks can hamper normal network operations and cause serious disruptions in VANETs. A replay attack uses previously generated frames in new connections, which are used by malicious or unauthorized users to impersonate legitimate users. In an illusion attack, the attacker deceives the sensors on his car to produce wrong sensor readings and incorrect traffic information and send them to neighboring vehicles in order to change their driving behaviors. It will lead to a traffic accident or traffic jam [55]. In a black hole attack, the attacker node claims to have the shortest path to the destination node, hoping to fool other nodes to route messages for the destination node to the attacker. Then the attacker can choose either to drop or forward packets. Gray hole and wormhole attacks are variations of black hole attacks.

Table 6.8 Attacks on vehicular ad hoc networks.

Attacks	Tool	Vulnerability	Action	Target	Unauthorized results
Sybil attacks	Script or program	Design (crypto or protocol)	Spoof, authenticate	Identity	Denial of service
Bogus information	Script or program	Design (crypto or protocol)	Spoof, modify	Data	Data corruption
Denial-of-service	Distributed tools	Design (crypto or protocol)	Flood	Resource	Denial of service
Man-in-the-middle attack	–	Design (crypto or protocol)	Modify	Data, identity	Data corruption
Location tracking	Data mining	Design (crypto or protocol)	Read	Location	Disclosure of information
Malicious code	Script or program	–	Modify	–	Denial of service
Replay attack	–	Design (crypto or protocol)	Authenticate	Identity	Disclosure of information, increased access
Illusion attack	Script or program	Design (sensor)	Deceive	Data	Data corruption
Black hole attack	–	Design (protocol)	Spoof, modify	Data	Denial of service

6.6 Security and privacy solutions

Vehicles are becoming increasingly connected. According to Gibbs [56], automotive applications are expected to constitute 53% of Internet of Things (IoT) data transmitted over 5G by 2023. In the future, in-vehicle infotainment (IVI) platforms will likely allow third-party services/apps to collect data from the vehicle using a built-in data connection. Together with the sheer amount of data generated in vehicles (25 GB/hour [57]), data sharing capabilities with third parties could turn into a lucrative after-market business for OEMs. However, this potential data monetization may introduce security and privacy risks to the driver. According to Frost and Sullivan [58], data security and privacy are among the most critical drivers or inhibitors in next-generation mobility services.

The connected vehicle ecosystem includes computers and networks. IT security plays an important role in securing connected vehicles; however, connected vehicle security goes far beyond.

- Vehicles are more physically exposed than computers, as vehicles are operated or parked in an open environment most of the time.
- A connected vehicle has multiple I/O interfaces (e.g., OBD-II access, Wi-Fi, radio, GPS, LTE, Telematics, Bluetooth), whereas computers generally use Wi-Fi and Ethernet.
- A connected vehicle implements multiple protocols in addition to TCP/IP (e.g., CAN protocol, LIN protocol), while a computer connected to the Internet exclusively implements TCP/IP protocols.
- A connected vehicle on the road may go through various networks; handoffs should maintain not only uninterrupted communication connections but also security levels.
- Multiple external wireless networks have access to internal critical buses through gateway ECUs.
- ECUs are constrained by limited resources (processing power and memory). Security means must respect these constraints.

Data consumed or generated by connected vehicles are valuable assets and need to be protected. Increasing demand for connectivity collides with data security and privacy on some levels. Potential data breaches pose considerable challenges to the development of connected vehicles. Various connectivity equipped in modern cars may enable unauthorized access to private data collected by the vehicle, such as location data, driver identity, and payment card information used for dashboard shopping. The automobile industry has a strong commitment to consumer privacy protection. The Alliance of Automobile Manufacturers, the Association of Global Automakers, and their members (23 global major automakers) developed a voluntary set of data privacy principles in 2014 [59]. The principles include:

- Transparency: participating members commit to providing owners and registered users with ready access to clear, meaningful notices about the participating member's collection, use, and sharing of covered information.
- Choice: participating members commit to offering Owners and Registered Users certain choices regarding the collection, use, and sharing of covered information.
- Respect for context: participating members commit to using and sharing covered information in ways that are consistent with the context in which the covered information was collected, taking account of the likely impact on owners and registered users.

- Data Security: participating members commit to implementing reasonable measures to protect covered information against loss and unauthorized access or use.
- Integrity and access: participating members commit to implementing reasonable measures to maintain the accuracy of covered information and commit to giving owners and registered users reasonable means to review and correct personal subscription information.
- Data minimization, de-identification, and retention: participating members commit to collecting covered information only as needed for legitimate business purposes. Participating members commit to retaining covered information no longer than they determine necessary for legitimate business purposes.
- Accountability: participating members commit to taking reasonable steps to ensure that they and other entities that receive covered information adhere to the principles.

Furthermore, the European Union (EU) has established a privacy standard called General Data Protection Regulation (GDPR) in May 2018 [60] that gives more consent opportunities over an individual's data. Although GDPR is only binding for EU residents and entities, OEMs are global companies selling cars worldwide. Hence, GDPR adherence is of great importance to North American and Asian OEMs. Even in the United States, there are state-specific privacy laws, such as the California Consumer Privacy Act (CCPA) [61] and its more stringent 2023 update, the California Privacy Rights Act (CPRA) [62].

Since GDPR is the most comprehensive privacy framework that is applicable to automotive data collection, it is outlined in the following. GDPR distinguishes between data subjects, data controllers and data processors. GDPR ensures adequate protection of the privacy rights of data subjects, that is, drivers in our context. Data controllers dictate how and why data is going to be used by the organization and thus have the most responsibility when it comes to protecting the privacy and rights of the data subject. Data processors process any data on behalf of the data controller. Since OEMs control the data shared with third-party app providers, they fall under the category of data controllers and are subject to increased compliance obligations [63]. These are summarized as seven core principles in the following [64]:

1. Lawfulness, fairness, and transparency: relates to the legality of data collection and transparency of collected data.
2. Purpose limitation: use collected data only for the specific purposes for which it was collected.
3. Data minimization: only request data that is required for a purpose.
4. Accuracy: relates to upkeeping the accuracy and completeness of such data and what a consumer's rights are for correcting inaccuracies.
5. Storage limitation: constrain the amount of time that personal data can be stored for.
6. Integrity and confidentiality: relate to the security of data transmission and storage, for example, by encrypting and pseudonymizing data.
7. Accountability: have appropriate measures and records in place to take responsibility for what you do with personal data and how you comply with the other principles.

Finally, Pesé et al. propose the first automotive GDPR-compliant, privacy-preserving data collection and sharing framework named PRICAR [65].

The objective is to address increasing concerns about privacy issues raised by new connected vehicle technologies. In what follows, a set of security and privacy solutions in the literature to secure connected vehicles are described.

6.6.1 Cryptography basics

Security and privacy rely heavily on cryptography. Asymmetric cryptography, symmetric cryptography, and hash are widely used to provide authentication, confidentiality, and integrity.

Asymmetric cryptography is primarily used to authenticate the communication nodes. A public key and secret key pair is used for encryption/decryption in asymmetric cryptography. Encryption using a public key (or secret key) can only be decrypted using the paired secret key (or public key). Each host holds a pair of public keys and a secret key. The public key is published to other hosts, while the secret key is kept secret. If host A wants to authenticate itself to host B, host A encrypts some predefined value using its secret key and sends it to B. B decrypts it using A's public key and retrieves the value. Then A authenticates itself to B by showing that it owns the secret key. Asymmetric cryptography can also be used for confidentiality. Host A encrypts its messages using the recipient's public key. Then only the recipient can decrypt the messages using its private key.

Symmetric cryptography is primarily used to provide confidentiality. The encryption and decryption use the same key, which is shared only between the communication ends. Compared to asymmetric cryptography, symmetric cryptography is less computationally intensive; therefore, it is general practice to use asymmetric cryptography for authentication and symmetric cryptography for confidentiality.

A hash function maps data of arbitrary size to a bit string with a fixed size, which is called a digest. A hash function is a one-way function, that is, infeasible to invert. Hash is used for detecting communication tampering. When host A sends a message to host B, it first applies the hash function that is previously agreed between the two hosts to produce a message digest, and it sends to B both the message and the digest. Upon receiving them, B applies the same hash function to the received message to produce a digest and compares it with the received digest. If the two digests are the same, then the received message is not tampered with.

6.6.2 Security solutions for automotive communications

A significant vulnerability of bus communications is that messages are broadcast in clear text with no authentication. Many attacks (e.g., Carshark sniffer, fuzzing, reverse engineering [29]) exploit these vulnerabilities. Security techniques to protect internal bus networks include code obfuscation, rootkit traps, cryptography, intrusion detection systems (IDSes), and honeypots [34,66–69].

Despite the importance of automotive security and several proposed solutions from academia, their adoption in real-world commercial vehicles by OEMs is rare. Pesé et al. attribute the lack of adoption to the following three challenges [70]:

1. **Cost**: This is usually the dominant driver behind the lack of adoption. OEMs operate within extremely tight cost constraints and aim to avoid any increase in their production cost. One dimension of how added security can contribute to overall cost is resource constraints. ECUs in current vehicles only require simple operations, and thus most ECUs are not built with high-performance hardware. For instance, current Engine Control Modules can have 80 MHz clock frequency, 1.5 MB flash memory and 72 kB of RAM (Bosch [71]). Adding cryptographic security operations for encryption and authentication on the CAN bus or machine-learning-based techniques for IDSes would require significantly more performant hardware, which

would, in turn, increase the costs to acquire more powerful ECUs. Furthermore, OEMs rely heavily on the economy-of-scale of their supply chain. They source several ECUs directly from different suppliers and Tier-1s. By reusing legacy ECUs for many generations of their vehicles, OEMs can avoid development costs and keep their purchasing costs low. Satisfying new security requirements would lead to new ECU developments, which would increase the OEMs' cost of manufacturing a vehicle [72].

2. **Latency**: This functional parameter is extremely important in vehicles, which come with stringent, hard real-time requirements to satisfy automotive safety certifications such as ISO 26262 [73]. For instance, in the case of CAN security, ensuring confidentiality and authenticity requires encryption and adding MACs. The maximum permissible end-to-end (E2E) latency on CAN is in the subsecond range [74]. Both encryption and MAC calculation add a negligible delay, especially considering the low power and cost hardware. This can lead to deadline misses, which can compromise driver safety and is thus unacceptable. Numerous existing works ignore this requirement while proposing performant IDS solutions and also do not consider how slowly their heavy solutions would run on resource-constrained automotive hardware.
3. **Mindset**: This challenge specifically applies to CAN security. OEMs never disclose DBC files to the general public. They keep them as proprietary secrets within their organization. DBCs are even only partially shared with Tier-1s. By doing so, OEMs believe that they can both prevent eavesdroppers from logging interpretable data on the CAN bus as well as deter attackers from launching CAN injection attacks. In fact, keeping DBCs secret acts as a barrier to CAN injection since attackers need to tediously reverse engineer the signal information they want to target first. Since DBCs differ between vehicle models, they need to repeat this step for each model they want to attack. As a result, OEMs believe that their security-by-obscurity mindset will help them improve the security of the CAN bus by deterring attackers. However, existing work on automated CAN bus reverse engineering [44] has shown that this mindset needs to be overcome.

6.6.2.1 Code obfuscation

An ECU obfuscates messages before sending them to the connected buses [66]. This makes it difficult for attackers to reverse-engineer the ECU firmware and harvest valid messages to control vehicle hardware. Obfuscation is cost-effective. There are quite a few obfuscators out there [75–78].

6.6.2.2 Authentication, confidentiality and integrity

6.6.2.2.1 Authentication

Weimerskirch et al. [34] used asymmetric cryptography to authenticate ECUs. Each accredited ECU OEM is assigned a pair of public keys PK_{OEM} and a secret key SK_{OEM}. PK_{OEM} is publicly published. Each ECU stores a copy of PK_{OEM}, while SK_{OEM} is a secret and only accessible to the OEM. Each ECU is assigned a unique identification ID, a pair of public key PK_{ID} and secret key SK_{ID}, and a digital certificate

$$\{ID, PK_{ID}, Auth_{ID}\}\{SK_{OEM}\}$$

which consists of the ID of the ECU, its public key, PK_{ID} and authorizations $Auth_{ID}$, signed by the secret key SK_{OEM} of the OEM that manufactures the ECU. The SK_{ID} is kept secret. In what follows, we use $\{content\}\{key\}$ to denote encrypted *content* using the *key*.

When joining a local bus, a new ECU sends out its digital certificate to the gateway ECU of the local bus to authenticate itself. The gateway ECU verifies the new ECU by decrypting the certificate using the OEM's PK_{OEM}. There is a trust chain. The gateway ECU trusts accredited OEMs, so it trusts the ECUs that are signed by the OEMs. After authentication, the gateway ECU stores a copy of the new ECU's public key and its authorizations, which define the access rights of the new ECUs to other buses connected via the gateway. The idea of ECU authorizations can potentially solve the security problems introduced by bus interconnections. If an ECU fails to authenticate itself to the gateway ECU, an error code may be generated by the gateway ECU and displayed on the dashboard screen to warn the car owner.

The gateway ECU also authenticates itself by sending its digital certificate. ECUs on the connected bus store a copy of the gateway ECU's public key. However, this form of authentication does not prevent the problem of ECU compromise, that is, when an existing ECU is compromised remotely, like in the Jeep hack [32].

6.6.2.2.2 Confidentiality

To prevent bus sniffers from eavesdropping ECU communications, Weimerskirch et al. [34] used symmetric encryption. The gateway ECU periodically generates a random group key for a local bus and shares the group key with all authenticated ECUs on the local bus. To distribute the group key to an authenticated ECU, the gateway concatenates the group key (denoted as *GK*) and the time stamp (denoted as TS), encrypts the concatenation using the ECU's public key (denoted as PK_{ID}), and sends out the encrypted concatenation, $\{GK, TS\}\{PK_{ID}\}$. Then only the target ECU possessing the paired secret key SK_{ID} can decrypt and obtain the group key. This protects the secrecy of the group key. Adding the time stamp prevents replay attacks. Each authenticated ECU will apply symmetric encryption to encrypt its outgoing messages using the group key. Only authenticated ECUs with the group key can decrypt, thus the communication confidentiality is guaranteed.

Given that symmetric encryption is lighter and faster than asymmetric encryption, asymmetric encryption is used only for authentication and distribution of group keys, which involve only a few message exchanges.

When selecting symmetric encryption algorithms, we must note that canonical symmetric encryption algorithms, like AES, are unsuited for encrypting CAN messages. AES handles 128-byte data blocks, while the maximum allowable data field size in CAN protocol is 8 bytes. Traditional automakers use a proprietary message format unknown to the public as a means of security [79]. Several celebrated hacking incidents in the past few years show that this form of "security through obscurity" has not been effective by demonstrating that this format can be manually reverse engineered. Furthermore, advances in automated reverse engineering can significantly speed up this process [44].

Trillium, a Japanese startup, developed a CAN bus encryption technology called SecureCAN [80] in 2015 that supports payloads of 8 bytes or less. Three operations (substitution, transposition, and time-multiplexing) are adopted in the algorithm. Trillium claims that SecureCAN can encrypt, transmit, and decrypt within 1 millisecond, which meets the requirement for real-time CAN bus applications. The key management system is another innovation of SecureCAN. A new shared master key is generated every time the car's ignition is turned on, which can be changed at random intervals using the frequency-hopping feature.

Another way is to replace the CAN bus with the CAN FD [81] to increase the bandwidth of the in-vehicle network. The data field of CAN FD is now up to 64 byte, and data transmission speed can be over 1 Mbps. To automakers, the replacement means an increase in manufacturing cost, which impedes the promotion and application of CAN FD. Ethernet is still the most promising solution for the long run.

The former two solutions lack backward compatibility. One approach to adding security to CAN on the software layer while maintaining backward compatibility with existing ECUs is S2-CAN [28]. Pesé et al. abandon the use of cryptography for confidentiality by using alternative obfuscation methods. This also leads to an increase in performance, that is, a significant reduction in E2E latency, memory, and bandwidth consumption, while maintaining a sufficient level of security.

6.6.2.2.3 Integrity

Even with authentication and encryption, an attacker could still intercept messages, change some bits, and retransmit. Weimerskirch et al. [34] use Message Authentication Codes (MACs) for data integrity. The gateway ECU periodically generates an authentication key for a local bus and distributes it to all authenticated ECUs on the bus the same way as it distributes the group key. When sending a message, an ECU first hashes the message to produce a digest. The digest is encrypted using the authentication key, producing a MAC. Then the ECU concatenates the message and the MAC encrypts them using the group key, as shown in Fig. 6.5. At the receiver side, it first decrypts the encrypted message using the group key and obtains the message in clear text and the MAC.

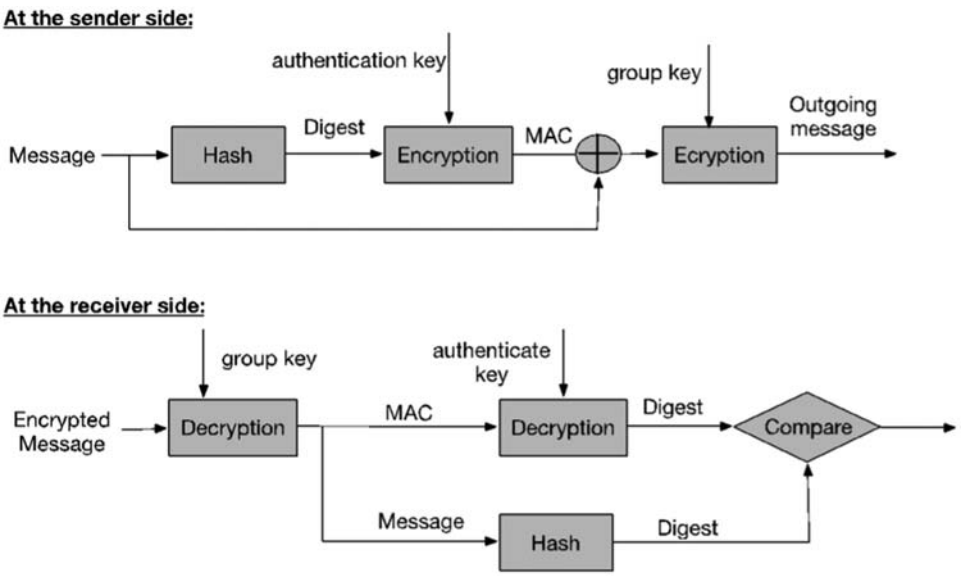

FIGURE 6.5

The use of message authentication codes to ensure data integrity.

Then it decrypts the MAC using the authenticate key and obtains a digest. Next, it applies the hash function to the received message to produce a digest and compares it with the received digest. If they are the same, then the message is not tampered; otherwise, the receiver drops the tampered message and reports the tampering.

vatiCAN [82] offers backward-compatible sender and message authentication, as well as protection against replay attacks for safety-critical CAN messages via HMACs. The HMAC is sent in a separate message with a different CAN ID since it cannot fit into the 8-byte CAN payload. This is a similar issue in the work of Weimerskirch et al. [34]. S2-CAN [28] tries to solve this problem as well by leveraging alternative methods to cryptographic MACs.

The above security means rely heavily on gateway ECUs to perform authentication, generate and distribute group keys and authentication keys, and store the public keys of authenticated local ECUs. This requires that gateway ECUs be granted more computation resource, and a secure memory to store the keys. However, this also means that the gateway must always be available. A Denial-of-Service (DoS) attack against the gateway would render several of the aforementioned approaches useless, making the gateway a single point of failure. The use of digital certificates relies on a Public Key Infrastructure (PKI) to distribute public keys. See Ref. [83] for PKI details.

6.6.2.3 Rootkit traps

Obfuscation and cryptographic means form the very first security barrier preventing attackers from attacking an ECU. However, neither is obfuscation unrecoverable nor is encryption uncrackable. To further strengthen the ECU security, Yu et al. [66] used a second layer of defense, which is to deploy known rootkit vulnerabilities in the ECU to trap attackers.

A rootkit is usually used by hackers to conceal their traces on a compromised system and leaves a backdoor to allow later returns without being detected [84]. For example, a load-able kernel module (LKM) rootkit can be utilized to monitor and report activities on the ECU after the attacker gets inside of an ECU [85]. A LKM is a compiled kernel code that is not built into the kernel but can be loaded when required. It is illustrated in Ref. [85] that a kernel rootkit can deceive most rootkit detectors, such as Tripwire and AIDE. This shows that a deliberately modified LKM rootkit can be used to monitor and track all activities of an intruder while being virtually invisible to the intruder.

Metasploit and the rootkit reference work [84,86] are used to find exploits for most vulnerabilities, such as privilege escalation. Rootkits using known exploits easily attract attackers' attention and thus are more likely to be "taken advantage of." When an embedded rootkit vulnerability is exploited, we can

- Determine if it is a malicious attack or just a system fault;
- Isolate the attack from the rest of the system if it is identified as an attack;
- Identify the type of attack;
- Study the attacker's exploits, especially when it is an emerging attack;
- Trace back to the attacker if possible.

This design adds one more layer of security by siphoning off attackers using rootkit vulnerability traps. It also enables the system to switch from defense to proactive aggression. Moreover, it helps improve the accuracy of intrusion detection by extracting signatures of emerging attacks.

6.6.2.4 Intrusion detection system

Larson et al. [67] proposed and evaluated specification-based IDS for the CAN 2.0 and CANOpen 3.01 protocols. The system places one detector at each ECU. The detector investigates all incoming and outgoing traffic against the specifications of the protocols. An intrusion is deemed to occur when the traffic does not agree with the specifications. Hoppe et al. [68] demonstrated an anomaly-based IDS for CAN protocol. The system is attached to a CAN bus and listens to traffic on the bus. It records the rates of specific messages on the bus and compares them to what is considered to be normal. Most novel IDSes for CAN buses are entropy [87] or machine learning-based [88].

6.6.2.5 Gateway firewall

Another significant vulnerability of internal buses is that they are interconnected via gateway ECUs, riskily enabling less critical ECUs (e.g., light, seat) that are connected to low-speed buses to access high-speed buses where critical ECUs (e.g., brake, transmission) are attached. An attack exploiting this vulnerability is presented in Ref. [29]. Therefore, Weimerskirch et al. [34] proposed to implement a firewall at a gateway ECU. Recall in Section 6.1, a gateway ECU also stores a copy of each authenticated local ECU's authorizations Auth_{ID}, which define the ECU's access rights to interconnected buses. When a gateway ECU receives a message from a local authenticated ECU, the firewall on the gateway ECU checks the authorizations of the local ECU. If it is authorized to access an interconnected bus, then the gateway ECU relays the message. Otherwise, it drops the message and generates an error code. In this way, a logical segmentation of internal buses is achieved. We must note that these firewalls must offer special interfaces that allow diagnostic data or ECU firmware updates to pass. These interfaces must be prevented from misuse. A two-layer firewall system for the next-generation domain architecture (depicted in Fig. 6.1) based on an Ethernet backbone is proposed by Pesé et al. [89]. It consists of a gateway-level hardware firewall for fast access control between domains and a stateful firewall implemented in software at the domain controller to further reduce false positives.

6.6.3 Wireless personal area networks security and privacy

The in-vehicle WPAN must be secure; otherwise, attackers may be able to access private information on personal devices connected to the WPAN and gain increased access to the internal CAN bus to interfere with the functionality of various ECUs [30].

6.6.3.1 Bluetooth security checklist

Bluetooth is the most widely used wireless technology for WPAN. The National Institute of Standards and Technology (NIST) and the US Department of Commerce published "Wireless Network Security 802.11, Bluetooth, and Handheld Devices" [90]. Various recommendations are suggested in the publication for securing wireless communications and handheld wireless devices [3]. A Bluetooth security checklist containing 37 items is recommended. Out of the 37 items, 17 items are most relevant to Bluetooth WPAN, including [3]:

- Ensure that handheld or small Bluetooth devices are protected from theft.
- Ensure that Bluetooth devices are turned off when not used.
- Bookkeep all Bluetooth-enabled devices.

- Change the default settings of Bluetooth devices to reflect the security policies.
- Set Bluetooth devices to the lowest necessary and sufficient power level so that transmissions remain within the secure perimeter of the agency.
- Ensure the environment in which Bluetooth devices are bonding is secure from eavesdroppers.
- Choose PIN codes that are sufficiently random and avoid all weak PINs.
- Choose PIN codes that are sufficiently long.
- Ensure no Bluetooth device is defaulting to the zero PIN.
- At the application layer, use an alternative secure protocol for exchanging PIN codes, for example, the Diffie-Hellman Key Exchange or certificated-based key exchange methods.
- Ensure that combinations keys are used instead of unit keys.
- Use link encryption for all Bluetooth connections.
- Ensure device mutual authentication for all accesses.
- Ensure encryption for all broadcasts for all accesses.
- Configure encryption key sizes to the maximum allowable.
- Establish a "minimum key size" for any key negotiation process.
- Ensure that portable devices with Bluetooth interfaces are configured with a password to prevent unauthorized access if lost or stolen.

6.6.3.2 Secure wireless personal area networks

WPAN is interconnected to internal buses via a WPAN gateway ECU. Maintaining secure communications between a Bluetooth device connected to a WPAN and the gateway is very crucial for normal and safe operation of the vehicle. To address this issue, Mahmud et al. [3] propose to build secure wireless personal area networks (SWPAN), in which:

- Each Bluetooth device to be used is registered to the gateway, so that the gateway knows which devices are allowed to communicate with it.
- The gateway allows removal of Bluetooth devices from the list of registered devices, as devices may be lost or stolen.
- Each device has its own link keys to encrypt the communications with the gateway and to prevent other devices from eavesdropping.
- The link keys are changed frequently so that uncovering these keys are infeasible.
- The gateway generates the link keys for devices and distributes these link keys to each device in a secure way that these keys are not compromised due to man-in-the-middle attacks.

6.6.3.3 Enabling data privacy in wireless personal area network

Bluetooth-enabled devices can be used to track people [91]. Each Bluetooth device is assigned a unique 48-bit address at the factory. The address of a Bluetooth device can be easily obtained by simply initiating the inquiry process.

Manufacturers employ a security feature called Bluetooth low energy (BLE) or Bluetooth smart, which was introduced as part of the Bluetooth 4.0 core specification [92]. BLE disguises the MAC address while advertising packets with a random number that periodically changes. A trusted device uses an Identity Resolution Key (IRK) created during pairing to decrypt these random addresses to a real MAC address. The problem with BLE is that it is poorly implemented or sometimes completely ignored. According to Lester [93], these random addresses of many devices are found to be fixed.

For those that do change their addresses, many of them are easy to identify as they have a counter that increments the last few bytes of the address and often send out constant identifying information.

The most up-to-date Bluetooth version 4.2 was released in 2014, with new security features added. The new specification provides link layer security with controller-based address resolution. The MAC address of Bluetooth devices can be masked unless connecting to a trusted device.

6.6.4 Secure vehicular ad hoc networks

The IEEE 1609.2 standard [94] specifies four security requirements for VANET communications: confidentiality, authentication, integrity, and anonymity. For confidentiality, the standard recommends using the Elliptic Curve Integrated Encryption Scheme algorithm to encrypt messages. for authentication and integrity, the standard recommends using elliptic curve digital signature algorithm to sign messages. The standard also encourages the use of long cryptographic keys. The minimum recommended size of encryption keys is 256 bits, of signature keys is 224, and of public/secret keys is 256 bits. The standard also defines the format and processing of messages and digital certificate format. The European Telecommunications Standards Institute is currently working on standards for protecting data exchanges in VANETs [95].

Privacy protection is one of the main security requirements for VANETs security [96,97] since data in VANETs is transmitted in an open access environment. Sensitive data transmitted over VANETs includes, but is not limited to, vehicle location, driver identity, driving behavior, location of the vehicle, and internal car sensor data.

Many research efforts have considered VANETs privacy. Most efforts focus on communication schemes. The adoption of pseudonyms (PNs) instead of using the identities of vehicles makes the VANET communications anonymous and improves privacy [98,99]. Each vehicle, V generates a set of public/secret key pairs, $\left(PK_V^1, SK_V^1\right), \left(PK_V^2, SK_V^2\right), \cdots \left(PK_V^n, SK_V^n\right)$, and sends over a secure channel the public keys (i.e., $PK_V^1, PK_V^2, \cdots PK_V^n$) to a Certificate Authority (CA). All vehicles and roadside units trust the CA and store a copy of the public key of the CA. The CA generates a set of PNs for the vehicle V (i.e., $PN_V^1, PN_V^2, \cdots PN_V^n$). PN_V^i contains $ID_{CA,}$ the identifier of the CA, T, the lifetime of PN_V^i, and PK_V^i, the public key of vehicle V, signed by the CA:

$$\left\{ID_{CA},\ T,\ PK_V^i\right\}\{SK_{CA}\}$$

CA sends over the same secure channel the set of PNs back to vehicle V. When communicating with other vehicles or roadside units, vehicle V can authenticate itself by using the PNs instead of revealing its true identity.

An alternative to PNs is to use anonymous keys [100]. Each vehicle is preinstalled with a set of one-time anonymous keys certified by a CA. Anonymous keys can be updated in a yearly checkup. A privacy-preserving protocol called Efficient Conditional Privacy Preservation Protocol (ECPP) for anonymous VANETs communication is presented in Ref. [101], where an anonymous key is valid for a short time period after generation. The scheme introduced in Ref. [102] uses a set of short-time PNs for message encryption. Each PN is associated with a key pair and a certificate for message encryption. The receiver of a message turns to a certificate revocation list (CRL) to validate the attached certificate. Messages are signed on behalf of a group of signing keys, so the vehicles' identification will not be divulged. Sampigethaya et al. [103] proposed grouping vehicles traveling at the same speed and in the same direction. Allowing members of the group to anonymously issue and sign messages with a group signature on behalf of the group.

A decentralized group authentication protocol is proposed in Ref. [104] as an alternative to a CA. Vehicles in the range of the same roadside unit or service centers are considered in the same group and managed by each roadside unit. Vehicles in the same group can verify each other's messages using a secret member key obtained from the roadside unit. The viability of the protocol is based on a dense spread of roadside units. The scheme proposed in Ref. [105] uses *k*-anonymity, where *k* vehicles in a region are assigned the same PN for communicating with roadside units or service centers. With this scheme, an attacker can only detect a group of cars receiving the message but cannot determine which one in particular.

Most privacy-preserving technologies rely heavily on time-consuming cryptographic operations and generate a large volume of cryptographic data. For example, in DSRC, a vehicle broadcasts messages to all nearby vehicles every few milliseconds. On the other end of the communication, a vehicle may receive hundreds of messages within a short time period, which need to be verified in real-time. This may cause an unacceptable delay. To tackle this, Zhang et al. propose a privacy-preserving protocol called APPA built on a one-time identity-based aggregate signature [106]. A one-time signature is generated using a secret key obtained from the trusted authority. The secret key is associated with a one-time PN, which guarantees the anonymity of the vehicles. Since an aggregate signature algorithm is employed, the signatures on different messages from different vehicles can be aggregated into one signature, which can be verified as a single signature. This feature greatly curtails the time for verification as well as the storage space for cryptographic data.

Given the above, we can see that most privacy solutions in VANETs involve the use of PNs and require the presence of CAs for key management. Generated cryptographic data adds excessive overhead to practical applications. Privacy-preserving techniques for VANETs still call for effective and efficient solutions.

6.6.5 Secure over-the-air electronic control unit firmware update

Fig. 6.6 below shows an overview of the OTA ECU firmware update. The update process is initiated by the backend server, which remotely sends an update command to the Telematics Control

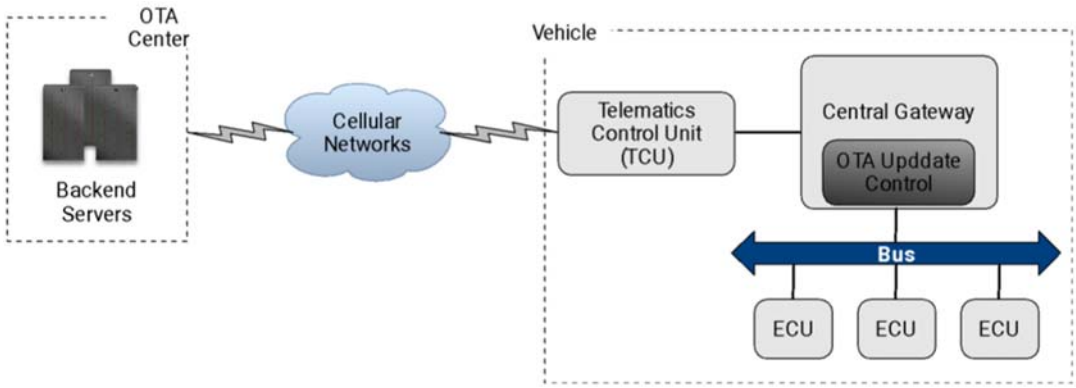

FIGURE 6.6

An overview of over-the-air electronic control unit firmware update.

Unit (TCU) of an on-board network. Before the actual update, the target ECU to be updated needs to send its specifications (e.g., ECU types, firmware version, etc.) to the server to ensure the correct firmware will be used. Though OTA updates are rare now, their security has been studied. It is good practice to embed security at the beginning during the development of an application rather than distribute security add-ons after being attacked.

To secure the OTA update process, all the components and communication channels involved in the process must be secured. End-to-end security must be provided. The security requirements are identified in Ref. [69] as:

- *Code origin authenticity*: An ECU can verify that the firmware to be installed is from the OEM. Any faulty firmware must be denied.
- *Code Integrity*: An ECU can detect any unauthorized modification of the firmware since it leaves the backend servers. Faulty or modified firmware must be denied.
- *Code confidentiality*: The content of the firmware should remain confidential during the transmission and installation.
- *Update metadata confidentiality:* This requirement ensures that an attacker should not gain information out of the update process about the specifications of the target ECU (e.g., ECU types, version of current firmware) and the firmware version to be used.
- *Availability:* This requirement ensures that the TCU, target ECU, and buses are available throughout the update process.
- *Command freshness:* This requirement prevents the replay attack that an attacker sends an old update command to deceit target ECU.
- *Message source authenticity*. This requirement ensures that during the process, the target ECU must verify that the received messages are from the backend server. A man-in-the-middle attack can occur if this requirement is not satisfied.

A secure protocol for OTA firmware updates is proposed in Ref. [107]. The message exchanged defined by the protocol is depicted in Fig. 6.7. The protocol relies on the Hardware Security Module (HSM) to provide security features. HSM is responsible for performing all the cryptographic operations, including symmetric/asymmetric encryption/decryption, integrity checking, digital signature creation/verification, and random number generation. Each ECU includes a HSM. The protocol satisfies all the security requirements above. In Fig. 6.7, each of the three components (backend server, TCU, and target ECU) has a pair of public keys and secret keys, and the public keys of the other two components. The backend server also has SSK, which is the key for encrypting/decrypting firmware. The process is divided into four phases: Remote Diagnosis, ECU Reprogramming Mode, SSK Exchange and Firmware Download.

The remote diagnosis phase authenticates the backend server and the ECU to each other and establishes a session key (MK) for later message encryption/decryption:

- The backend server initiates the update process by requesting the ECU specifications. To do so, it generates MK, concatenates it with a time stamp (T), signs the concatenation with its secret key (SK_{BS}) to produce a MAC, concatenates the MAC with MK and T, encrypts with the public key of TCU, and sends to TCU the encrypted message

$$\{M,\ T, MAC\}\{PK_{TCU}\}$$

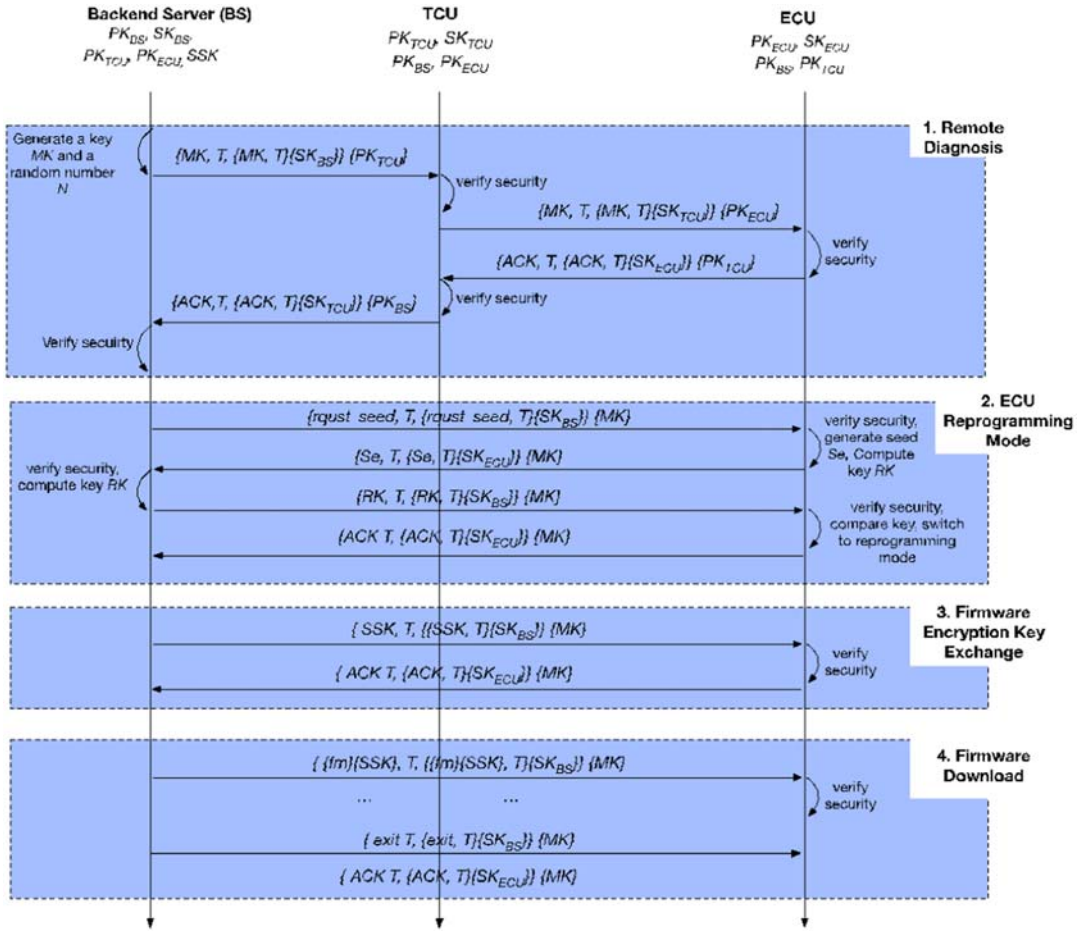

FIGURE 6.7

Secure protocol for over-the-air electronic control unit firmware update.

where $MAC = \{MK, T\}\{SK_{BS}\}$. This message can only be decrypted by TCU. Confidentiality is guaranteed.

- The TCU decrypts with its secret key SK_{TCU} to get MK, T, and the MAC. Then, it decrypts the MAC using the backend server's public key PK_{BS}, and compares the MK and T with the previously obtained values to detect message tampering. The message authenticity is also verified as the MAC can only be signed (encrypted) by the backend server, who owns SK_{BS}. The T is used to check message freshness to prevent replay attacks. After verifications, TCU sends to ECU the similar message

$$\{M,\ T,\ \{M, T\}\{SK_{TCU}\}\}\{PK_{ECU}\}.$$

- The ECU performs the same verification, and sends back an acknowledgment message.

In this way, the backend server can successfully authenticate itself to the ECU and distribute *MK* to the ECU. A secure communication channel between the two can be established. From now on, all the communications will be encrypted using *MK*. The ECU will send its specifications, encrypted using *MK*, to the backend server.

After completing the first phase, the backend server will force the ECU to switch to reprogramming mode. In phase three, the backend server distributes the *SSK* key to the ECU, which will be used in phase 4 to decrypt firmware. In phase 4, the backend server sends the firmware to the ECU. When it finishes, it sends an *exit* message to the ECU. The ECU returns an acknowledgment message.

The protocol has satisfied all the requirements identified. Other security solutions can be found in Refs. [108,109]. Over-the-air (OTA) ECU update services have evolved significantly over the years. One notable advancement in this domain is the introduction of the Uptane framework [16].

Uptane is a comprehensive framework designed to secure OTA software updates for vehicles. Recognizing the unique challenges posed by automotive systems, Uptane addresses potential threats and ensures that even if some parts of the update system are compromised, an attacker cannot install malicious software on vehicles.

Uptane employs multiple repositories to validate software images before they're downloaded to vehicles. The Image Repository serves as a constant source of information about these images, holding every image currently deployed by the OEM and the metadata files that verify their authenticity. The Director Repository, on the other hand, determines which software should be distributed to each ECU based on the current state of the repository.

ECUs communicate with the Director repository by sending their vehicle version manifest, which contains signed information about existing software images. Using this data, the director decides which images should be installed next. Once the updates are determined, the metadata and images are sent to the ECU, which then undergoes a verification process. If this verification is successful, the images are downloaded to the ECU, updating the vehicle's software.

Fig. 6.8 provides a visual representation of Uptane's design, illustrating the interaction between the vehicle components and the repositories. Uptane provides the following features:

- Multiple roles and repositories: Uptane employs a multirepository setup with each repository having a specific role. The four primary roles are director, image repository, timeserver, and vehicles.
- End-to-end security: the director repository instructs the vehicle about which images to install. The image repository holds the metadata about these images. Before any update, the vehicle checks with both repositories, ensuring end-to-end security.
- Compromise resilience: even if one repository is compromised, Uptane's design ensures that the vehicle software remains secure. For instance, even if an attacker takes control of the director repository, they cannot cause a vehicle to install malicious software without also compromising the Image repository.
- Customized updates: the director repository can instruct different vehicles to install different sets of updates. This is particularly useful for staged rollouts, canary deployments, or when specific updates are targeted to a subset of vehicles based on certain criteria.
- Real-time clocks and timestamps: the timeserver ensures that the vehicle has a source of secure time, and the vehicle can use this to check the validity of the timestamps in the metadata it receives.

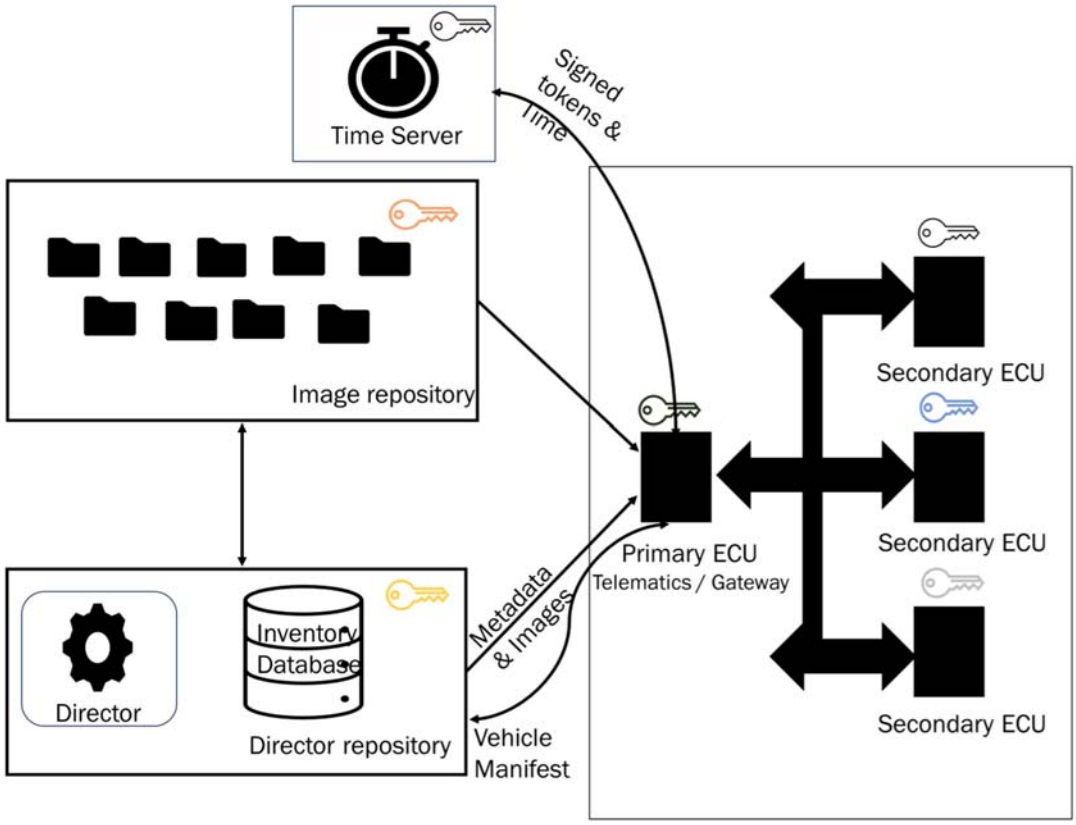

FIGURE 6.8

Uptane system architecture referenced from [16].

- Protection against rollback attacks: Uptane protects against rollback attacks, where attackers might try to make a vehicle install outdated software. The Image repository's metadata contains information about the latest software, preventing such downgrades.
- Delegations: for added flexibility, the image repository can delegate responsibilities to other repositories, allowing third parties to hold and maintain their software images and metadata.

With the advent of frameworks like Uptane, the landscape of OTA updates in the automotive industry has been transformed, offering enhanced security and reliability. It's crucial to implement these updates with care, ensuring that vehicles are safeguarded from potential threats.

6.6.6 Privacy measurement of sensor data

The internals of a connected vehicle are a swarm of sensors. Although sensor data may not contain Personal Identification information (PII), traffic analysis technologies can be adopted to infer

sensitive information from side channels (timing, message size, communication frequency, etc.). Standard privacy-preserving techniques like Privacy Preserving Data Mining (PPDM) [110] use noise addition and information suppression. Research has gone into evaluating privacy in Internet of Things (IoT) systems like connected vehicles.

A recent report from Mozilla's Privacy Not Included project reveals unsettling information about the privacy and security concerns associated with modern internet-connected cars [111]. The study found that all major car brands, including BMW, Ford, Toyota, Tesla, and Subaru, fail to meet basic privacy and security standards in their new models. These vehicles are essentially data-harvesting machines that collect a wide range of personal information, from driving behaviors like seatbelt use to sensitive details like race, health information, and even sexual activity. These cars employ various data harvesting tools such as microphones, cameras, and connections to drivers' phones, while manufacturers also collect data through apps and websites, potentially sharing or selling this data with third parties.

Furthermore, the report uncovered a lack of data encryption in most car brands' practices and identified "privacy washing," where manufacturers provide misleading information that downplays privacy concerns. Many automakers claim to adhere to the nonbinding "Consumer Privacy Protection Principles," but these promises lack transparency and enforcement.

Ukilet et al. [112] propose a risk estimation scheme that allows users of IoT systems like connected vehicles to estimate the risk of sharing private data. In their work, sensitive or private data is defined as *unpredictable anomalous events that may arouse curiosity or undue interest*. Anomalies in sensor data make them distinguishable and thus pose serious threats to data privacy.

The proposed scheme consists of two steps: sensitivity (anomaly) detection and measurement of privacy/sensitivity. The detection algorithm discovers the anomaly points in a given sensor dataset while optimizing the masking and swamping effects. The detection method shows superior detection performance when compared with other outlier detection algorithms [113,114] in terms of larger Kullback–Leibler (KL) distance. According to Sanov's theorem [115], a larger KL distance indicates better detection capability. Given a dataset S, the privacy metric ρ_M is defined as mutual information $I(S, v)$, where v is the sensitive/anomalous part of S. To improve the accuracy of ρ_M as privacy metric, a statistical compensation ρ_S is computed using Kolmogorov-Smirnov (KS) test of S and v. The rule is ρ_S, if the null hypothesis is rejected; otherwise, $\rho_S = W_{S,v}$ where $W_{S,v}$ is the $L1$-Wasserstein metric between S and v and quantifies the distance between distributions S and v. The privacy is defined as $\rho_Q = \rho_S \times \rho_M$, where $\rho_Q \in [0, 1]$ is in proportional to the privacy risk of S. If ρ_Q exceeds a predefined threshold, the privacy-preserving sensor data S' computed using standard privacy preserving data mining is shared with a third party.

6.6.7 Secure handover

As the vehicle roams, it may pass through heterogeneous networks. Current academic and industrial efforts focus on seamless handovers with the least interruption on communication sessions. IEEE 802.21 (Media Independent Handover) mainly defines an architecture to enable low-latency handover across multiple technology access networks [116]. Software-defined networking (SDN) is also proposed to handle handovers to provide session continuity [117].

When a handover occurs, the roaming vehicle needs to authenticate itself to the new network (base station or access point) and other connected vehicles in the new network. This problem is known as AAA (Authenticate and Authenticate Again). AAA can seriously degrade user quality of service (QoS). This is likely to be an especially problematic issue for vehicles traveling at high speed that wish to maintain connectivity with minimal user distraction. A recent survey of AAA optimization techniques (e.g., AAA Context Transfer methodology) can be found in Ref. [118].

A vehicle may travel to a less secure network. To protect ongoing communication sessions, we suggest using Transport Layer Security (TLS) for communication sessions requiring protection. TLS is a cryptographic protocol that provides authentication and communication confidentiality. TLS works at the presentation layer (layer 6) of the OSI stack. Current 802.21 plans work at both layer 2 (data link) and layer 3 (network). TLS should normally be able to coexist with 802.21.

6.7 Future research directions and conclusions

Increasing the security of automotive systems will be difficult. Many of the basic security problems are baked into the current bus standards. No individual manufacturer or supplier would be able to make the necessary changes and remain competitive in a low-margin business. An industry-wide change would be needed, but that would leave a large number of legacy automobiles. Numerous new approaches could be made to monitor and filter information on the automotive bus systems. Technologies also exist to help secure individual ECUs. We present some here. Other ideas could leverage recent advances in trusted computing that add a hardware root of trust to make infecting the devices more difficult. All of these research ideas could be helpful, but in many ways the key problem is constrained by the economics of the automotive industry.

Current update practices, such as whole-vehicle re-flashing, are cumbersome and potentially insecure. A promising avenue of research is the integration of blockchain technology with Unified Diagnostic Services (UDS) to bolster security. Leveraging blockchain's smart contract enforcement can ensure adherence to supply chain security best practices and provide indelible documentation of the software build process. This, combined with modular updates facilitated by UDS, aims to streamline the update process while safeguarding against vulnerabilities. The goal is to create a system where firmware updates are both efficient and secure, mitigating risks like supply chain compromises and vulnerabilities inherent in current vehicular network protocols. Another promising research direction is the study of automotive ethernet security, as it is gaining traction in in-vehicle architectures [119]. Despite similarities with regular Ethernet known from IT systems and networks, the deployment of this communication protocol in novel in-vehicle architectures offers promising avenues for security exploration.

All in all, this chapter presents a security profile for connected vehicle systems. Multiple vulnerabilities are analyzed and existing attacks are surveyed. To analyze the attacks, we map them to the attack taxonomy that describes attacks on connected vehicle systems. This mapping helps us find the problem space and locate common vulnerabilities. Various security solutions to the security and privacy issues are presented as well.

Exercises

Exercise 1. List all the in-vehicle networks and study their security features, if any.
Exercise 2. Explain the vulnerabilities of in-vehicle networks.
Exercise 3. Understand the challenges in securing ECU communications.
Exercise 4. Try designing security policies for an ECU gateway that connects two different in-vehicle networks.
Exercise 4. Understand the attack surface enabled by external vehicle networks.
Exercise 5. Study the privacy issues of VANET and the solutions to address them.

References

[1] M. Reke, D. Peter, J. Schulte-Tigges, S. Schiffer, A. Ferrein, T. Walter, et al., A self-driving car architecture in ROS2, in: 2020 International SAUPEC/RobMech/PRASA Conference, IEEE, 2020, pp. 1–6.
[2] R.R. Brooks, S. Sander, J. Deng, J. Taiber, Automobile security concerns, IEEE Veh. Technol. 4 (2) (2009) 52–64.
[3] S. Mahmud, S. Shanker, In-vehicle secure wireless personal area network (SWPAN), IEEE Trans. Veh. Technol. 55 (3) (2006) 1051–1061.
[4] http://www.ford.com/technology/sync/, last accessed on October 9, 2016.
[5] https://www.onstar.com/us/en/home.html, last accessed on October 9, 2016.
[6] http://www.toyota.com/owners/parts-service/safety-connect, last accessed on October 9, 2016.
[7] http://www.lexus.com/enform, last accessed on October 9, 2016.
[8] http://www.bmw.com/com/en/insights/technology/connecteddrive/2013/, last accessed on October 9, 2016.
[9] https://www.mbusa.com/mercedes/mbrace, last accessed on October 8, 2016.
[10] https://www.linkedin.com/pulse/information-security-connected-vehicle-shashank-dhaneshwar, last accessed on May 24, 2016.
[11] H. Kitayama, S. Munetoh, K. Ohnishi, N. Uramoto, Y. Watanabe, Advanced security and privacy in connected vehicles, IBM Res. Dev. 58 (1) (2014).
[12] http://www.iteris.com/cvria/html/applications/applications.html, last accessed on October 9, 2016.
[13] http://www.oesa.org/Publications/OESA-News/August-2015/ver-the-Air-Updates-to-Become-Commonplace-in-Vehicles.html, last accessed on May 24, 2016.
[14] https://www.onstar.com/us/en/home.html, last accessed on May 24, 2016.
[15] http://resources.infosecinstitute.com/hacking-autoupdate-evilgrade/, last accessed on June 10, 2016.
[16] https://uptane.github.io/, last accessed on November 2, 2023.
[17] https://www.youtube.com/watch?v = oHDwKT564Kk, last accessed on May 24, 2016.
[18] J. Muller, No hands no feet: my unnerving ride in Google's driverless car. Available from: http://www.forbes.com/sites/joannmuller/2013/03/21/no-hands-no-feetmy-unnerving-ride-in-googles-driverless-car, last accessed on May 24, 2016.
[19] http://news.mit.edu/2016/startup-nutonomy-driverless-taxi-service-singapore-0324, last accessed on May 24, 2016.
[20] https://www.nytimes.com/2023/08/21/technology/waymo-driverless-cars-san-francisco.html, last accessed October 29, 2023.
[21] https://www.businessinsider.com/amazon-creating-fleet-of-electric-delivery-vehicles-rivian-2020-2, last accessed October 29th, 2023.

[22] https://www.ietf.org/proceedings/63/slides/nemo-4.pdf, last accessed June 6, 2016.
[23] https://www.jabil.com/blog/automotive-connectivity-trends-fueling-the-future.html, last accessed October 29th, 2023.
[24] R.R. Brooks, S. Sander, J. Deng, J. Taiber, Automotive system security: challenges and state-of-the-art, in: 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, May, p. 26.
[25] J.D. Howard, T.A. Longstaff, A common language for computer security incidents, Sandia Report, SAND98-8867, Sandian National Laboratory, 2007.
[26] S.C. Bono, M. Green, A. Stubblefield, Security analysis of a cryptographically-enabled FRID device, in: Proceeding of 14th conference on Usenix Security Symposium, vol. 14, 2005.
[27] A.I. Alrabady, S.M. Mahmud, Analysis of attacks against the security of keyless-entry systems for vehicles and suggestions for improved designs, IEEE Trans. Veh. Techonol. 54 (1) (2005) 41−50.
[28] M.D. Pesé, J.W. Schauer, J. Li, K.G. Shin, S2-CAN: sufficiently secure controller area network. in: Annual Computer Security Applications Conference, 2021, December, pp. 425−438.
[29] K. Koscher, A. Czeskis, F. Roesner, S. Patel, T. Kohno, S. Checkoway, et al., Experimental security analysis of a modern automobile, in: IEEE Symposium on Security and Privacy, 2010.
[30] S. Checkoway, D. McCoy, B. Kantor, D. Anderson, H. Shacham, S. Savage, et al., Comprehensive experimental analysis of automotive attack surfaces, in: Proceeding of USENIX security, 2016, pp. 1−16.
[31] Autosar specification of module secure onboard communication, https://www.autosar.org/fileadmin/user-upload/standards/classic/4-2/AUTOSARSWSSecureOnboardCommunication.pdf, last accessed on October 4, 2023, 2022.
[32] C. Miller, C. Valasek, Remote exploitation of an unaltered passenger vehicle. Black Hat USA, (S 91), 2015, pp.1−91.
[33] M. Pesé, K. Shin, J. Bruner, A. Chu, Security analysis of android automotive, SAE Int. J. Adv. Curr. Pract. Mobil. 2 (2020-01-1295) (2020) 2337−2346.
[34] M. Wolf, A. Weimerskirch, C. Paar, in: C. Paar (Ed.), Secure in-vehicle communication, ESCAR, 2004.
[35] T.C. Niem, Bluetooth and its inherent security issues, Global Information Assurance Certification Security Essentials Certification, Research Project, Version 1.4b, Nov. 4, https://www.sans.org/reading-room/whitepapers/wireless/bluetooth-inherent-security-issues-945, 2002.
[36] S. Indesteege, N. Keller, O. Dunkelman, E. Biham, B. Preneel, A practical attack on Keeloq, in: N. Smart (Ed.), Eurocrypt'08, volume 4965 of LNCS, Springer-Verlag, 2008, pp. 1−18.
[37] A. Francillon, B. Danev, S. Capkun, in: A. Perrig (Ed.), Relay attacks on passive keyless entry and start systems in modern cars, *NDSS*, 2011.
[38] http://www.bbc.com/news/technology-29786320, last accessed on May 25, 2016.
[39] http://www.autoalarmpro.com/bypass_kits, last accessed on May 25, 2016.
[40] https://www.directechs.com/default.aspx, last accessed on May 25, 2016.
[41] http://www.clickorlando.com/news/local/orlando/thieves-use-device-to-jam-keyless-entry-systems, last accessed on May 25, 2016.
[42] http://articles.sun-sentinel.com/2012-12-28/news/fl-pirate-radio-hollywood-20121229_1_pirate-radio-entry-systems-keyless-entry, last accessed on May 25, 2016.
[43] http://thehackernews.com/2015/08/rolljam-unlock-car-garage.html, last accessed on May 26, 2016.
[44] M.D. Pesé, T. Stacer, C.A. Campos, E. Newberry, D. Chen, K.G. Shin, LibreCAN: automated CAN message translator, in: Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security, 2019, November, pp. 2283−2300.
[45] https://www.wired.com/2015/07/jeep-hack-chrysler-recalls-1-4m-vehicles-bug-fix/, last accessed on June 21, 2016.

[46] H.M. Song, H.R. Kim, H.K. Kim, Intrusion detection system based on the analysis of time intervals of can messages for in-vehicle network, in: Proceeding of 2016 International Conference on Information Networking (ICOIN), IEEE, 2016, pp. 63−68.
[47] T. Hoppe, S. Kiltz, J. Dittmann, Security threats to automotive CAN networks − practical examples and selected short-term countermeasures, in: Proceeding of the 27th International Conference on Computer Safety, Reliability, and Security, Necastle upon Tyne, UK: Springer-Verlag, 2008, pp. 235−248.
[48] T. Hoppe, J. Dittmann, Sniffing/Replay attacks on CAN buses: a simulated attack on the electric window lift classified using an adapted CERT taxonomy, in: Proceeding of the 2nd Workshop on Embedded Systems Security, Salzburg, Australia, 2007.
[49] I. Aideyan, R. Brooks, M.D. Pesé, Fuzzing CAN vs. ROS: an analysis of single-component vs. dual-component fuzzing of automotive systems, SAE WCX, SAE International, 2024.
[50] I. Rouf, R. Miller, H. Mustafa, T. Taylor, S. Oh, W. Xu, et al., in: I. Goldberg (Ed.), Security and Privacy Vulnerabilities of In-Car Wireless Networks: A Tire Pressure Monitoring System Case Study, USENIX Security, 2010, pp. 323−328.
[51] B. Parno, A. Perrig, Challenges in securing vehicular networks, in: Proceeding of the 4th Workshop Hot Topics in Networks (HotNet-IV), 2005.
[52] I.A. Sumra, I. Ahmad, H. Hasbullah, J.L.B.A. Manan, Classes of attacks in VANET, in: Electronics, Communications and Photonics Conference (SIECPC), Saudi International, IEEE, 2011, pp. 1−5.
[53] V.H. La, A. Cavalli, Security attacks and solutions in vehicular ad hoc networks: a survey, Int. J. AdHoc Netw. Syst. (IJANS) 4 (2) (2014) 1−20.
[54] S.K. Das, K. Kant, N. Zhang, Handbook on Securing Cyber-Physical Critical Infrastructure, Elsevier, 2012.
[55] A.S.K. Pathan (Ed.), Security of Self-Organizing Networks: MANET, WSN, WMN, VANET, CRC Press, Boca Raton, FL, 2016.
[56] N. Gibbs, Automakers seek new revenue streams from wireless services, upgrades, May 2020, https://europe.autonews.com/automakers/automakers-seek-new-revenue-streams-wireless-services-upgrades, last accessed on September 1, 2023.
[57] The connected car 'data explosion': the challenges and opportunities, Sep 2020, https://www.information-age.com/tconnected-car-dataexplosion-123473363/, last accessed on August 22, 2023.
[58] Pymnts, "Who controls data in web-connected vehicles?" Jun 2018. https://www.pymnts.com/innovation/2018/data-sharing-smart-carsprivacy/, last accessed on December 2, 2022.
[59] Letter to the FTC. http://www.autoalliance.org/auto-issues/automotive-privacy/letter-to-the-ftc, last accessed on October 9, 2016.
[60] The eu general data protection regulation (gdpr) is the most important change in data privacy regulation in 20 years. https://eugdpr.org/, last accessed on December 2, 2022.
[61] Basics of the california consumer privacy act of 2018. https://www.privacypolicies.com/blog/california-consumer-privacy-act/, last accessed on December 3, 2022.
[62] California consumer privacy act (ccpa), Nov 2020. Available from: https://www.cookiebot.com/en/ccpa-compliance/, last accessed on December 3, 2022.
[63] What does the gdpr have to do with car oems? https://upstream.auto/blog/dgpr/, last accessed on December 3, 2022.
[64] G. Madzudzo, M. Cheah, Data protection and connected vehicles: privacy policy analysis from a consumer perspective, November 2020.
[65] M.D. Pesé, J.W. Schauer, M. Mohan, C. Joseph, K.G. Shin, J. Moore. PRICAR: Privacy Framework for Vehicular Data Sharing with Third Parties, in: 2023 IEEE Secure Development Conference (SecDev), 2023, pp. 184−195. IEEE.
[66] L. Yu, J. Deng, R.R. Brooks, S.B. Yun, Automotive ECU design to avoid software tampering, in: Proceeding of Cyber Information Security Research Conference (CISR'15), Oak Ridge, TN, 2015.

[67] U.E. Larson, D.K. Nillson, E. Jonsson, An approach to specification-based attack detection for in-vehicle networks, in: Proceeding of the IEEE Intelligent Vehicles Symposium, 2008, pp. 220–225.
[68] V. Verendel, D.K. Nilsson, U.E. Larson, E. Jonsson, An approach to use honeypots in in-vehicle networks, in: Proceeding of the 68th IEEE Vehicular Technology Conference, 2008, pp. 163–172.
[69] M.S. Idrees, Y. Roudier, Computer aided design of a firmware flashing protocol for vehicle on-board networks, Res. Rep. RR-09-235 (2009).
[70] M.D. Pesé, Bringing Practical Security to Vehicles (Doctoral dissertation). Miller, C., & Valasek, C. (2015). Remote exploitation of an unaltered passenger vehicle. Black Hat USA, 2015(S 91), 2022, pp. 1–91.
[71] Electronic engine control unit for commercial vehicles, 2020, https://www:bosch-mobility-solutions:com/en/products-and-services/commercial-vehicles/powertrain-systems/natural-gas/electronicengine-control-unit/, last accessed on July 17, 2023.
[72] A.S. Thangarajan, M. Ammar, B. Crispo, D. Hughes, Towards bridging the gap between modern and legacy automotive ecus: a software-based security framework for legacy ecus, in: 2019 IEEE 2nd Connected and Automated Vehicles Symposium (CAVS), 2019, pp. 1–5.
[73] Automotive safety integrity level, https://en.wikipedia.org/wiki/Automotive_Safety_Integrity_Level, Dec 2020, last accessed on July 17, 2023.
[74] R.I. Davis, A. Burns, R.J. Bril, J. J Lukkien, Controller area network (can) schedulability analysis: refuted, revisited and revised, Real- Time Syst. 35 (3) (2007) 239–272.
[75] https://jscrambler.com/en/, last accessed on June 10, 2016.
[76] http://stunnix.com/prod/cxxo/, last accessed on June 20, 2016.
[77] https://javascriptobfuscator.com/, last accessed on June 20, 2016.
[78] http://www.semdesigns.com/Products/Obfuscators/.Last, 2016 (accessed 20.06.16).
[79] R. Currie, Developments in car hacking, Technical report, 2015.
[80] J. Yoshida, Can bus can be encrypted, says trillium. http://www.eetimes.com/document.asp?docid = 1328081, 2015.
[81] F. Hartwich, Can with flexible data-rate, Citeseer.
[82] S. Nürnberger, C. Rossow, Vatican–vetted, authenticated can bus, in: International Conference on Cryptographic Hardware and Embedded Systems, Springer, 2016, pp. 106–124.
[83] https://en.wikipedia.org/wiki/Public_key_infrastructure, last accessed on June 3, 2006.
[84] G. Hoglund, J. Butler, Rootkits: Subverting the Windows Kernel, Addison-Wesley Professional, 2006.
[85] J. Rose, Turningthetables: Loadablekernelmodulerootkits, Deployed in a Honeypot Environment, SANS Institute InfoSec Reading Room, 2003.
[86] Windows Rootkit Overview, Symantec, 2006.
[87] S. Ohira, A.K. Desta, I. Arai, H. Inoue, K. Fujikawa, Normal and malicious sliding windows similarity analysis method for fast and accurate ids against dos attacks on in-vehicle networks, IEEE Access. 8 (2020) 42422–42435.
[88] M.D. Hossain, H. Inoue, H. Ochiai, D. Fall, Y. Kadobayashi, Lstm-based intrusion detection system for in-vehicle can buscommunications, IEEE Access. 8 (2020) 185489–185502.
[89] M.D. Pesé, K. Schmidt, H. Zweck, Hardware/software codesign of an automotive embedded firewall, Technical report, SAE Technical Paper, 2017.
[90] T. Karygiannis, L. Owens, Wireless network security: 802.11, bluetooth and handheld devices, Special Publication (NIST SP), National Institute of Standards and Technology, Gaithersburg, MD, 2002 (accessed 08.08.24).
[91] http://electronics.howstuffworks.com/bluetooth-surveillance2.htm, last accessed on June 16, 2016.
[92] Bluetooth, Adopted specifications, https://www.bluetooth.com/specifications/adopted-specifications.
[93] S. Lester, The emergence of bluetooth low energy, http://www.contextis.com/resources/blog/emergence-bluetooth-low-energy/.

[94] IEEE trial-use standard for wireless access in vehicular environments – security services for applications and management messages, IEEE Std., July 2006.
[95] D. Benhaddou, A. Al-Fuqaha (Eds.), Wireless Sensor and Mobile Ad-Hoc Networks: Vehicular and Space Applications, Springer, 2015.
[96] S. Misra, I. Woungang, S.C. Misra, Guide to Wireless Ad Hoc Networks, Springer Science & Business Media, 2009.
[97] A. Stampoulis, Z. Chai, A survey of security in vehicular networks, Project CPSC, 534, 2007.
[98] P. Papadimitratos, L. Buttyan, J.P. Hubaux, F. Kargl, A. Kung, M. Raya, Architecture for secure and private vehicular communications, in: Proceeding of 7th International Conference on ITS, IEEE, 2007, pp. 1–6.
[99] F. Dötzer, Privacy issues in vehicular ad hoc networks, Privacy enhancing technologies, Springer, 2005, pp. 197–209.
[100] M. Raya, P. Papadimitratos, J.P. Hubaux, Securing vehicularcommunications, in: IEEE Wireless Communications Magazine, Special Issue on Inter-Vehicular Communications, 13(LCA-ARTICLE-2006-015), 2006, pp. 8–15.
[101] R. Lu, X. Lin, H. Zhu, P.H. Ho, X. Shen, Ecpp: efficient conditional privacy preservation protocol for secure vehicular communications, in: INFOCOM 2008, The 27th Conference on Computer Communications, IEEE, 2008.
[102] G. Calandriello, P. Papadimitratos, J.P. Hubaux, A. Lioy, Efficient and robust pseudonymous authentication in vanet, in: Proceedings of the fourth ACM international workshop on Vehicular ad hoc networks, ACM, 2007, pp. 19–28.
[103] K. Sampigethaya, L. Huang, M. Li, R. Poovendran, K. Matsuura, K. Sezaki, Caravan: providing location privacy for vanet, Technical report, DTIC Document, 2005.
[104] L. Zhang, Q. Wu, A. Solanas, J. Domingo-Ferrer, A scalable robust authentication protocol for secure vehicular communications, IEEE Trans. Veh. Technol. 59 (4) (2010) 1606–1617.
[105] C. Zhang, X. Lin, R. Lu, P.H. Ho, X. Shen, An efficient message authentication scheme for vehicular communications, IEEE Trans. Veh. Technol. 57 (6) (2008) 3357–3368.
[106] L. Zhang, Q. Wu, B. Qin, J. Domingo-Ferrer, Appa: aggregate privacy-preserving authentication in vehicular ad hoc networks, Information Security, Springer, 2011, pp. 293–308.
[107] M.S. Idrees, H. Schweppe, Y. Roudier, M. Wolf, D. Scheuermann, and O. Henniger, Secure automotive on-board protocols: a case of over-the-air firmware updates, in: Proceeding of 3rd International Workshop on Nets4Cars/Nets4Trains, Oberpfaffenhofen, Germany, 2011.
[108] D. Nilsson, U. Larson, Secure firmware updates over the air in intelligent vehicles, in: Proceeding of IEEE International Conference on Communications workshops, 2008, pp. 380–384.
[109] S. Mahmud, S. Shanker, I. Hossain, Secure software upload in an intelligent vehicle via wireless communication links, in: Proceeding of IEEE Intelligent Vehicle Symposium, 2005, pp. 588–593.
[110] Y. Lindell, B. Pinkas, Privacy preserving data mining, Advances in CryptologyCRYPTO 2000, Springer, 2000, pp. 36–54.
[111] https://gizmodo.com/mozilla-new-cars-data-privacy-report-1850805416, last accessed November 7, 2023.
[112] A. Ukil, S. Bandyopadhyay, A. Pal, IoT-privacy: to be private or not to be private, in: Computer Communications Workshops (IN- FOCOM WKSHPS), 2014 IEEE Conference, IEEE, 2014, pp. 123–124.
[113] R. Rao, S. Akella, G. Guley, Power line carrier (plc) signal analysis of smart meters for outlier detection, in: 2011 IEEE International Conference on Smart Grid Communications (SmartGridComm), IEEE, 2011, pp. 291–296.
[114] R.Md Nascimento, A.P. Oening, D.C. Marcilio, A.R. Aoki, E. de Paula Rocha, J.M. Schiochet, Outliers' detection and filling algorithms for smart metering centers, in: Transmission and Distribution Conference and Exposition (T&D), 2012 IEEE PES, IEEE, 2012, pp. 1–6.

[115] I. Sanov, On the probability of large deviations of random variables, United States Air Force, Office of Scientific Research, 1958.
[116] http://www.ieee802.org/21/, last accessed on June 1, 2016.
[117] groups.geni.net/geni/raw-attachment/wiki/GEC21Agenda/. . ./GEC21poster-V3.pdf, last accessed on June 1, 2016.
[118] G. Karopoulos, G. Kambourakis, S. Gritzalis, Survey of secure handoff optimization schemes for multimedia services over All-IP wireless heterogeneous networks, IEEE Commun. Surv. 9 (3) (2007) 18–28.
[119] M. de Vincenzi, G. Costantino, I. Matteucci, F. Fenzl, C. Plappert, et al. A systematic review on security attacks and countermeasures in automotive ethernet, ACM Computing Surveys, 2022.