



Smart Car Hacking: A Threat to The Rising Automotive Industry

DECLARATION

I, Vinay Unnikrishnan Nair, hereby declare that the research presented in this work is entirely my original creation and has not been previously submitted to any institution or university for a Degree or Diploma. Moreover, I have diligently referenced all the literature and sources used in accordance with the academic honesty policy of Dublin Business School.

Student Name: **Vinay Unnikrishnan Nair**

Student Number: **10609822**

Dublin Business School

Dated: **29/08/2023**

ACKNOWLEDGEMENT

I wish to express my deepest gratitude to the following individuals without whom the completion of this study would not have been possible. First and foremost, I am immensely thankful to my friends and colleagues, who stood by my side throughout my journey at Dublin Business School. Your unwavering love and encouragement have been invaluable, and I dedicate this study to all of you.

A special thanks to my supervisor, **Tejas Bhat**, whose exceptional expertise and guidance were instrumental in achieving the desired level of research quality.

Furthermore, I would also like to extend my appreciation to all the teaching staff at DBS and my fellow classmates with whom I collaborated in groups, fostering an environment of teamwork and shared ideas to attain fruitful outcomes.

Abstract

The purpose of this research paper is to highlight the importance of cybersecurity in modern automobiles, by focusing on the various methods and techniques used by malicious actors to hack and steal automobiles. With the increasing use of technology in automobiles, new avenues for potential security breaches have opened up, necessitating the examination of most impactful automotive vulnerabilities. The focus of this study is on the controller area network (CAN) bus architecture, a key component of modern vehicles that allows communication between different electronic control units (ECUs). By comprehensively studying the intricacies of CAN bus, we aim to identify potential weak points and propose effective countermeasures. Using an analysis of existing hacking incidents and cybersecurity challenges in the automotive industry, this paper emphasizes the urgency for auto manufacturers to develop and maintain secure vehicular information systems.

Keywords: Controller Area Network (CAN), Electronic Control Units (ECU), CAN Bus, Car Hacking, Car Pentest, CAN Injection.

Table of Contents

1. Introduction.....	7
2. Literature Review.....	9
3. Role of ECU's and CAN Bus	11
4. Attacks on Modern Vehicles	14
4.1 Remote Keyless Entry (Relay Attacks).....	14
4.2 Replay Attacks.....	15
4.3 Roll Jam Attacks.....	16
4.4 Roll Back Attacks.....	17
4.5 CAN Injection Attacks	19
5.1. Decoding Replay Attacks.....	22
5.1 Further Analysis	27
6. Conclusion	30
7. References.....	32

Table of Figures

FIGURE 1: REASON OF CAN INVENTION	11
FIGURE 2: CAN DATA FRAME	12
FIGURE 3: CAN MESSAGE FORMAT	12
FIGURE 4: CAN BUS ARCHITECTURE IN MODERN CAR	13
FIGURE 5: HACKER ONE RF	14
FIGURE 14: FLIPPER ZERO USED FOR CAPTURING RFID SIGNALS	15
FIGURE 6: SAMMY KAMKAR'S REPRESENTATION OF ROLL JAM ATTACK	16
FIGURE 7: DEVICE USED FOR ROLLJAM ATTACK	17
FIGURE 8: WORKING OF ROLL BACK ATTACK	18
FIGURE 9: OBD PORT	19
FIGURE 10: BREAKING INTO TOYOTA RAV4	20
FIGURE 11: WIRING OF ECU IN RAV4	21
FIGURE 12: A CAN INJECTION DEVICE MASKED AS A JBL SPEAKER	21
FIGURE 13: INSIDE THE SPEAKER	22
FIGURE 15: VIRTUAL SIMULATOR	24
FIGURE 16: CAN LOGS	25
FIGURE 17: SAMPLE FLOW CHART OF REVERSE ENGINEERING	26
FIGURE 18: REPLAYING THE FRONT UNLOCK DOOR MESSAGE	29

1. Introduction

The automobile industry has experienced rapid growth since its evolution. The first automobiles ran on steam and electricity. On January 29, 1886, Carl Benz applied for a patent for his “vehicle powered by a gas engine.” Patent – number 37435 which might can be marked as the birth certificate of the automobile. In July 1886 Benz Patent Motor Car, model No.1 became the first three-wheeled car available for public. Since then, the world has seen a rapid acceleration of modern vehicles starting from the age of innovations between 1920s to 1950s. The then car manufacturers came up with innovations like automatic transmissions, power steering, and air conditioning, making cars more comfortable and easier to drive. As cars became faster and more powerful, concerns about safety escalated which resulted in the introduction of seat belts as standard equipment, and in the 1970s, safety regulations led to the inclusion of features like airbags, anti-lock brakes, and crumple zones. These advancements aimed to mitigate the risks associated with accidents. Post which marked the entry of digital technology into cars. Onboard computers were introduced to control fuel injection and engine management, improving efficiency and emissions. Electronic components also became common in dashboards, bringing digital displays, climate control, and infotainment systems fuelled by the integration of technologies aimed at providing consumers with unparalleled driving experiences. In this last two decades of innovation, vehicles are now equipped with advanced features, including artificial intelligence (AI) capabilities and voice command assistants, all designed to collect and analyze vast amounts of data from consumers. The ultimate goal is to deliver personalized recommendations and results, enhancing the convenience and luxury of driving. This evolution has led to significant investments by car manufacturers, resulting in the production of millions of technologically advanced vehicles annually. However, amongst this progress, a crucial aspect tends to be overlooked – the inherent risks that accompany technological advancements. In the race to develop smarter and more connected vehicles, the concept of security is often overshadowed. The increasing reliance on AI-driven systems, coupled with the data exchange, introduces a wide variety of vulnerabilities that, if left unaddressed, have the potential to jeopardize not only customer data but also the reputation of car manufacturers. While billions of dollars are allocated to the development of these technological beasts, the importance of these innovations against cyber threats is often underestimated. The very technologies that promise convenience and customization are the same ones that can be exploited by malicious actors if left unseen. Hackers, operating in the shadows, are actively seeking vulnerabilities within these high-tech vehicles. Their intent is not limited to data breaches alone; they have the ability to execute remote vehicle thefts, potentially leading to catastrophic consequences. What's alarming is that these cyber attackers do not necessarily require an advanced technical background; a plethora of tools for such exploits can be easily be bought from the Dark Web or crafted with minimal knowledge of electronics and coding. Recent years have witnessed a surge of incidents involving the exploitation and theft of vehicles through electronic means. These alarming instances have captured the attention of both mainstream news outlets and social media platforms, implicating the urgent need for car manufacturers to address the critical issue of automotive cybersecurity. The relentless advancement of automobile technology, coupled with the imminent rise of autonomous and self-driving vehicles, paints a much bigger picture of security in automotive landscape. As vehicles evolve from modes of transportation to interconnected hubs of technology, ensuring the safety of the electronic and information systems within them becomes paramount. To delve into the vulnerabilities inherent in modern vehicles, it is imperative to intricate web of components that constitute these automobiles. Among the various technological growth in modern vehicle one of the crucial aspect of this technological ecosystem is the Control Area Network (CAN) Bus, a pivotal communication framework that facilitates seamless interaction between various electronic control units (ECUs) within the vehicle. The interplay of these

components forms the foundation of modern vehicular functionality, highlighting the significance of safeguarding them against potential cyber threats. This paper intends to highlight the multifaceted landscape of car pen testing and car hacking. By examining the nuances of vulnerabilities, methodologies for testing, emerging trends, and potential countermeasures, this paper aims to provide a comprehensive understanding of the challenges that the automotive industry faces in this era of technological innovation. Through this critical analysis, car manufacturers, stakeholders and even consumers can pave the way for a safer, more secure automotive future, where the promise of cutting-edge technology harmonizes seamlessly with the imperatives of cybersecurity.

2. Literature Review

This literature review draws from a comprehensive range of peer-reviewed articles, conference papers, and reports from reputable sources. The focus is on recent publications within the last five years to provide a current perspective on the field of car pen testing and car hacking. Researchers have emphasized the importance of employing robust methodologies in car pen testing to uncover vulnerabilities effectively. A study by Wang et al. (2019) advocates for a three-stage approach, including reconnaissance, vulnerability analysis, and exploitation. This approach mirrors traditional penetration testing methodologies while accommodating the unique challenges posed by automotive systems. Wang further stated that by following such methodologies, testers gain insight into potential entry points that attackers might exploit. Numerous studies have identified recurring vulnerabilities in modern vehicles. Ralf et al. (2020) highlight insecure communication protocols as a prime concern. They reveal that inadequate encryption and authentication mechanisms in vehicular communication can expose vehicles to remote manipulation. Samy Kamkar (2015) was the first ever to identify and demonstrate how rolling code attacks can be used to hack your car. Samy demonstrated the vulnerability during DefCon conference which is a DEF CON is a hacker convention held annually in Las Vegas, Nevada. Additionally, key fob relay attacks, as demonstrated by Checkoway et al. (2015), underscore the vulnerability of remote keyless entry systems. Such findings underline the importance of addressing weaknesses in these critical interfaces. Hoon Wei Lim and Levente Csikor (2022) discovered the roll back attack flaw indicated that the way the attack works is to replay a series of previously valid codes which resets the car's internal PRNG counter to an older state, which allows an attacker to reuse the known prior keys and unlock your vehicle. Furthermore, Dr. Ken Tindell and Ian Tabor (2023) identified CAN Injection attack that is done by manipulating the CAN bus inside the car by physically breaking the side of the car to get into cars internal communication and then use a simple electronic device to send fake CAN frames on to the CAN bus which will deactivate the immobilizer function and unlock the car door. As automotive technology advances, new attack vectors emerge. Gopinath et al. (2022) discusses the increasing risk associated with over-the-air (OTA) updates. While OTA updates offer a convenient means of patching vulnerabilities, they also present an avenue for attackers to compromise vehicle software remotely. Additionally, as vehicles evolve towards autonomy, researchers like Li et al. (2021) explore potential threats related to the integration of autonomous driving systems. These emerging trends reflect the evolving landscape of car hacking and emphasize the need for adaptable pen testing methodologies. Recent research has seen the development of specialized tools and techniques for car pen testing. A study by Miroshnikov et al. (2018) introduces CANSPLY, a tool designed to analyze in-vehicle communication on the CAN bus. This tool assists testers in identifying potential vulnerabilities related to communication between ECUs. Moreover, Srdic et al. (2020) propose the use of dynamic analysis techniques to identify potential buffer overflow vulnerabilities in ECUs' firmware. These advancements in tools and techniques streamline the pentesting process and enable more efficient vulnerability discovery. Academic research has also focused on proposing countermeasures and solutions to mitigate car hacking risks. Shen et al. (2021) introduce a lightweight encryption protocol for in-vehicle communication, enhancing the security of data exchange between ECUs. Similarly, Fischer et al. (2019) present an approach that leverages blockchain technology to secure OTA updates, preventing unauthorized modifications. These innovations in security measures contribute to the ongoing efforts to fortify vehicles against potential attacks. The academic discourse on car hacking also acknowledges the role of regulatory bodies in ensuring automotive cybersecurity. Research by United Nations Economic Commission for Europe (UNECE) WP.29 emphasizes the necessity of implementing cybersecurity regulations and standards within the automotive industry. Such regulations serve as a crucial driving force for manufacturers to invest in secure design

practices and rigorous testing. By exploring methodologies, vulnerabilities, emerging trends, and potential countermeasures, this review contributes to the holistic understanding of car pen testing's significance. The insights garnered from the latest academic research not only emphasize the vulnerabilities and risks associated with modern vehicles but also highlight the ongoing efforts to safeguard them through innovative tools, techniques, and regulatory frameworks. As the automotive landscape continues to evolve, continued research and collaboration in the field of car pen testing are imperative to ensure a secure and resilient future for the automotive industry.

3. Role of ECU's and CAN Bus

The Primary role of ECUs is to keep the engine of a car running smoothly. A modern car can have as much as 70 ECUs. ECUs control most of the car's functions including safety critical engine control, airbag deployment, and anti-lock braking system. To have a safe driving, ECUs should have a reliable communication network. In order for all these units to communicate with each other efficiently a CAN Bus is used. Before the invention of the CAN Bus automotive manufacturers used point-to-point wiring systems. As we started having more and more electronic components in the cars, this only became bulky and too expensive to maintain. Imagine connecting every wire of every single component to another wire. The challenge was diagnosing and troubleshooting. This problem was then fixed by replacing it with a CAN. CAN was originally developed by BOSCH in the year 1985. Thanks to BOSCH CAN soon began to be introduced in cars to become a single wire which is connected with each component for communication between each unit.

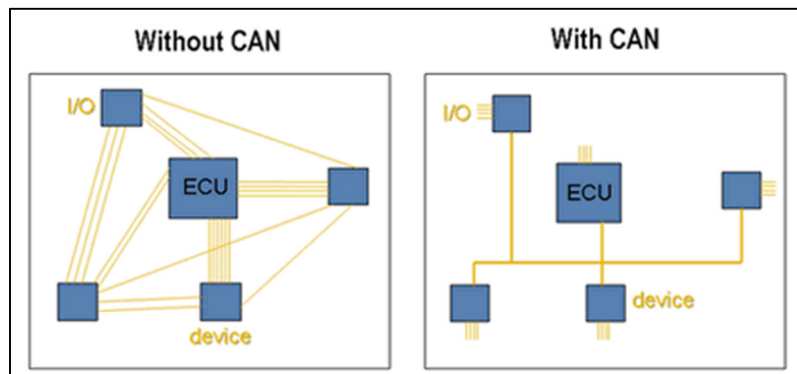


Figure 1: Reason of CAN Invention

One feature of CAN is that it uses a message-based protocol to transfer information. A car CAN have multiple nodes that are able to send and/or receive messages. The message usually consists of an ID, which is a priority of the message and also it can contain CAN message that can be of eight bytes or less at a time. Messages transmitted with the dominant ID will be replaced with those sent with a less dominant ID if two or more nodes start transmitting messages at the same time. Priority-based bus arbitration is what this is known as. The priority is higher for the transmission of messages with numerically smaller value IDs comes first. This is how a node recognizes that messages with a higher priority are being sent to a buffer. For example, messages coming from the brakes has a higher priority than a message from the audio player. If two or more nodes begins sending messages at the same time, the messages sent with the dominant ID will overwrite with that of less dominant.

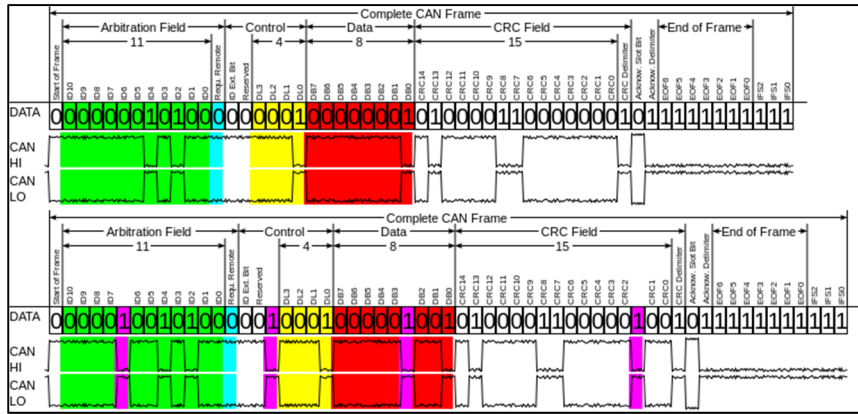


Figure 2: CAN Data Frame

CAN bus consists of two different wires. As it is a bus, multiple devices can be connected to these wires. A CAN frame has 3 major parts:

- i) Arbitration Identifier
- ii) Data Length Code
- iii) Data field

In CAN Bus any CAN node can start transmitting when the bus is free, two or more nodes can start transmitting at the same time. Arbitration is the method through which various nodes compete for bus control. Proper arbitration is crucial to CAN performance because it ensures that message collisions do not diminish bandwidth or result in message loss. Each data or remote frame begins with an identifier, which specifies the message's priority and content.

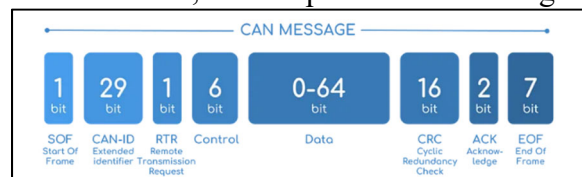


Figure 3: CAN Message format

There are two message frame formats supported by the CAN protocol, with the only essential difference being the length of the identifier (ID). Identifiers are 11 bits long in the standard format and 29 bits long in the extended format. The above figure shows how a message in the standard format begins with the start bit “start of frame”, this is followed by the “arbitration field”, which contains the identifier and the “RTR” (remote transmission request) bit, which indicates whether it is a data frame or a request frame without any data bytes (remote frame). The “control field” contains the IDE (identifier extension) bit, which indicates either if it is a standard format or extended format, a bit reserved for future extensions and – in the last 4 bits – a count of the number of data bytes in the data field. The “data field” ranges from 0 to 8 bytes in length and is followed by the “CRC field”, which is being used as a frame security check for detecting bit errors. The “ACK field” has a ACK slot (1 bit) and a ACK delimiter (1 recessive bit). The bit in the ACK slot is sent as a recessive bit and is overwritten as a dominant bit by those receivers which have at this time received the data correctly. Correct messages are acknowledged by the receivers regardless of the result of the acceptance test. The end of the message is indicated by “end of frame”. “Intermission” is the minimum number of bit periods separating consecutive messages. If there is no following bus access by any station, the bus remains idle (“bus idle”).

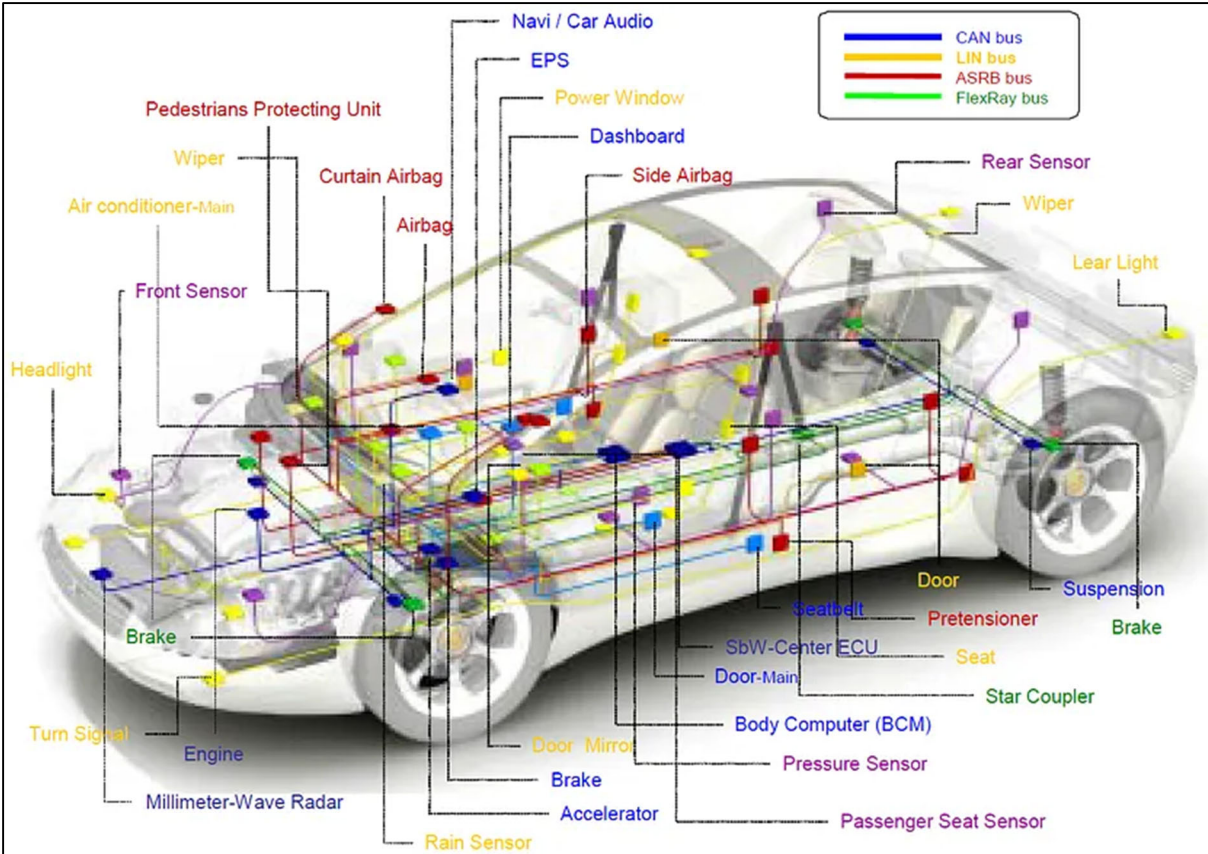


Figure 4: Can Bus architecture in modern Car

4. Attacks on Modern Vehicles

CAN is vulnerable to attacks despite its resistance to electrical noise and several security measures. As system security becomes a major concern, intensive study on CAN vulnerabilities and potential remedies is being conducted. Some of this research were effective in their experimental strikes against commercial vehicles. Although most attacks are carried out by physical access to the vehicle, wireless attacks are on the rise. The wireless attack surface will grow with new wireless interfaces such as vehicle-to-vehicle and vehicle-to-infrastructure. There have been several popular attacks which is been carrying on for a last decade or so. Here, we will see and understand few of the most impacting attacks carried on different car models. We will see what those attacks are, what tools can be used to carry such attacks and, what car models have been impacted.

- Remote Keyless Entry (Relay Attacks)
- Replay Attacks
- Roll Jam Attacks
- Roll Back Attacks
- CAN Injection Attacks

4.1 Remote Keyless Entry (Relay Attacks)

There are various kinds of keyless attacks but for our understanding we will look into how relay attack works. This attack works by picking up radio signals coming from your key fob and forwarding/relaying it over. Consider a scenario where

1) **Two Thieves with Electronic Cloning Devices**

- i) Thief 1 pings the car entry system which is always 'listening' for the key fob 24/7.
- ii) Car responds by requesting an authentication signal from the key fob.
- iii) Thief 1 forwards this request to thief 2, who is near the real key fob (inside the house) which is always on, listening for the car.
- iv) The Key fob then responds with a valid authentication which thief 2 then forwards back to thief 1, who then has everything needed to enter and drive away your car.



Figure 5: Hacker One RF

Now the attack vector is quite simple and devices like 'Hacker One RF' or any other similar device can be purchased from the market anywhere ranging from €20 to €200. Many new cars have keyless entry either as standard or as an optional extra, and there are thousands of used

vehicles already on the road with this technology. Researchers claim that it is 90% faster to steal a keyless entry car than a car with old-fashioned locks and ignition – in as little as 10 seconds (WHATCAR, n.d.) ADAC, known as the Allgemeiner Deutscher Automobil-Club, is Europe's largest automobile association. Their research claims that more than 30 brands have made insecure cars, including Audi, BMW, Honda, Hyundai, Kia, Peugeot, Renault, Skoda and Volvo (The Guardian, 2019). As per their research only Jaguar i-Pace (2018), Land Rover Discovery (2018), Land Rover Range Rover (2018) were not vulnerable to unlock or engine start.

4.2 Replay Attacks

The recent attack on many modern Honda and Acura vehicles makes them vulnerable to Replay attacks. This attack can be performed using cheap hardware. This vulnerability allows an attacker to capture and use the signal to unlock the cars and start their engines wirelessly. According to researchers, the vehicles impacted by this bug primarily include the 2016-2020 Honda Civic (LX, EX, EX-L, Touring, Si, Type R) cars. Furthermore, there is also a known CVE published (CVE-2019-20626) (Berry, 2020) for Honda Car vulnerable to this attack. Berry was able to record a lock signal “653-656, 667-668, 677-680, 683-684, 823-826, 837-838, 847-850, 853-854” and then by flipping these signals he was able to unlock the Car. Interestingly, attack like these can be prevented with rolling codes but Honda has never implemented these security measures in their cars and atleast they wouldn't likely seem to implement any fix in their existing vulnerable cars.



Figure 6: Flipper Zero used for capturing RFID signals

Attacks like this can be done using tools like Flipper Zero, HackerRF One etc. Flipper Zero can be purchased at around \$200 from Amazon. Currently, looking at the range of attacks that can be performed using this mini device government and security forces are actively trying to ban its buying/selling.

4.3 Roll Jam Attacks

Discovered by Spencer White which was later refined by Samy Kamkar and many others. The roll jam attack works by blocking and capturing the key fob signals and then using it later to unlock the car. Since, the fashion of keyless entry has taken the car market by storm most automobile manufacturers employ a keyless entry technology known as rolling codes. This was implemented to avoid replay/relay attacks. The rolling code system is based on an algorithm that generates a new code each time the keyfob is pressed, and the next code in the sequence can only be predicted by the car and the keyfob (this means that even if an attacker captures one of the codes, it is immediately classified as expired because both the car and the keyfob have moved on to the next code in the sequence). The data is sent from the keyfob to the automobile through radio (typically inside the ISM band, which is allocated worldwide for industrial, scientific, and medical uses rather than telecommunications). This data is often modulated using on-off keying (OOK), in which binary bits are transferred one by one (a high power level for a one and a low power level for a zero). Because key fobs do not need to send vast volumes of data, the data rate (or baud rate) is relatively modest (meaning that each binary bit lasts longer). This implies that with just a few lines of Python code, these signals may be readily received and processed. In simple terms, the way the attack works is as one presses the key fob to unlock his/her car that code is jammed and captured before it reaches the automobile. This means that the code the attacker received hasn't "expired" yet since the car still hasn't moved onto the next code in the sequence. The automobile, however, does not open since it does not receive this signal. But what would someone do if their vehicle key failed to open? Press the button one more time. On that second press, the RollJam is programmed in a way to jam the signal again and record that second code, but also to simultaneously broadcast its first code. That replayed first code unlocks the door, and the user immediately forgets about the failed key press. But the RollJam has secretly stored away a second, still-usable code. (Wired, 2015).

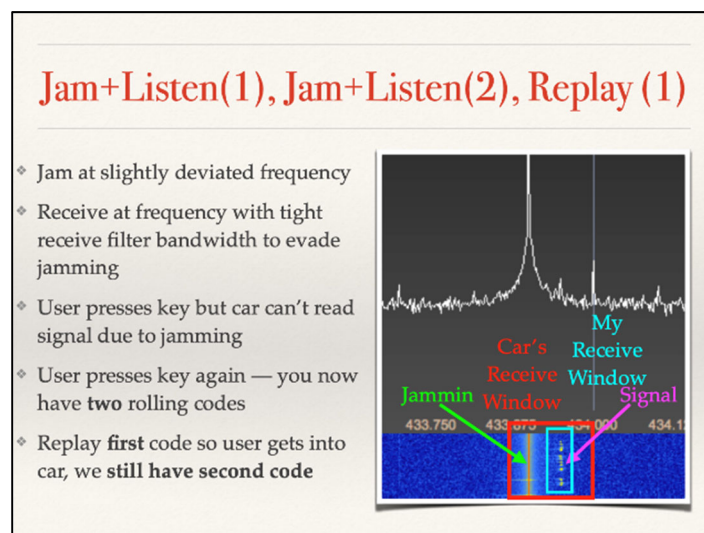


Figure 7: Samy Kamkar's representation of Roll Jam Attack

This attack demonstrated by Samy Kamkar (Hackaday, 2017) can be done using a \$32 radio device which in size is smaller than a cell phone and can not only be used to unlock keyless car but also can open modern garage doors, alarm system etc. In short, any electronic device which operates on rolling code algorithm.



Figure 8: Device used for RollJam attack

The successful attacks were tested on car manufacturers like Nissan, Cadillac, Ford, Toyota, Lotus, Volkswagen, and Chrysler vehicles, as well as Cobra and Viper alarm systems and Genie and Liftmaster garage door openers. Kamkar estimates that millions of other vehicles and garage doors may be vulnerable. But he says he believes the problem is rooted in the chips used by many of those companies: the Keeloq system sold by the firm Microchip and the Hisec chips sold by Texas Instruments.

4.4 Roll Back Attacks

This attack was initially discovered by a group of researchers in Singapore however, further researched by Hoon Wei Lim and Levente Csikor published their research (Linn & Csikor, 2022) wherein they demonstrated that they were able to find vulnerability inside car's rolling codes. To avoid replay attacks car manufacturers came up with disposable rolling codes this makes every key fob button press unique preventing relay and replay attacks. Meaning, every code that are used once or unused (codes which vehicle never receives) are invalidated by the next code, thus preventing replay attacks. The way safety provision features of rolling code system is built that the key fob can be out-of-sync compared to the vehicle's counter. A vehicle counter is nothing, but an internal counter built inside a vehicle to keep track of all the unlocks. However, out-of-sync does not mean that the key fob can be completely out of sync with the car's counter, but it can be only few steps ahead of what the car presumes it will receive next. The key fob signal receiving unit in the vehicle, does not have just one future code it waits for but a set of future codes which is generated by PNG (Pseudorandom Number Generator). This code is generated using the same algorithm in both the key fob and the receiving unit which will unlock the car as per the instruction from the code. The main drawback of rolling code attack is its generally a timing attack i.e., it is particularly sensitive to timing, and it must be aware of the next valid unused code. If the attacker misses the future codes, the attacker has to restart from scratch. The way Roll Back attack works is if an attacker is able to capture and replay two consecutive unlock signals that would unlock the car.

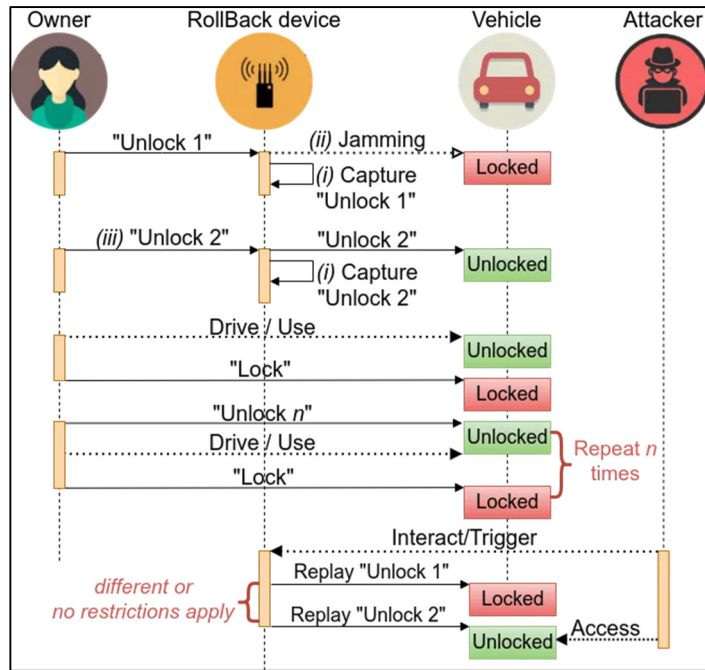


Figure 9: Working of Roll Back attack

Remember, that the car invalidates the present code as soon as the next code is generated but that is also the reason this attack works. Because, by replaying those already invalid signals, the rolling code counter in the vehicle re-synchronizes back to the counters present in the replayed signals. Now, since the counter is re-synchronized the previous invalid unlock signals (if captured) can now be a valid code and can be used to unlock the car. The reason this attack becomes dangerous is if an attacker is able to two signals once and have indefinite access to that vehicle. Currently, Rollback attack is only successful for 70% of the vehicles. Currently, these car's are vulnerable to Rollback attacks as per the researchers: Honda Fit (hybrid) 2016-2018, Fit 2018, City 2017, Vezel 2016-2022, Hyundai Elantra 2013-2015, Elantra 2012, Avante 2018-2020, Kia Cerato/Forte K3 2016-2018, Cerato/Forte K3 2012-2018, Mazda 3 2018, 2 Sedan 2018, 2 HB (facelift) 2020, Cx-3 2019, Cx-5 2018, Nissan Teana 2014, Latio 2007, Sylphy 2012-2019, Toyota Wish 2009-2017, Corolla Axio 2015-2017, Altis 2005, Prius (hybrid) 2020.

4.5 CAN Injection Attacks

This is something very latest attack happening around. Unlike other attacks where you wait for the victim to press key fobs or try sneaking near to place where a key fob is kept to capture the signals. This attack does not require you to be around the key fob or the victim for that matter. Again, the goal here is to steal the car. Dr. Ken Tindell in his blogpost (Tindell, 2023) shares about a similar incident that happened with one of his friend Ian Tabor whose car was stolen. Ironically, Ian is a Cyber Security expert and quite known to find vulnerabilities in automobiles and how together they tried to understand how this attack works. It is important to note modern vehicles have built-in diagnostic systems to detect any car faults. It's called 'on-board diagnostics' (or OBD for short) and when an Electronic Control Unit (or ECU) detects a fault, it records a code. In the automotive industry, it's called 'dropping a DTC' (or Diagnostic Trouble Code). The car manufactures usually have their own app which can be installed on your phone and these DCT's will be shown on your app whenever your car detects any failures.



Figure 10: OBD Port

Whenever your car has any trouble, and you take it to a mechanic. They connect a wire to the OBD port to the laptop for further diagnostics. These are codes that indicate what the detected fault is and when it occurred. Modern cars connects ECU with CAN bus and one of the ways an ECU will identify a problem is if it doesn't hear from another ECU it needs to communicate with, and this is frequently done with a timeout: if a CAN message isn't regularly received, then after some time without hearing anything, the listener assumes there's a problem with the CAN bus or the other ECU. Additionally, there are situations when it is clear that the CAN bus has failed, such as when an ECU's own messages are not transmitted or the CAN bus interface hardware reports that communication has been lost. There's a known CVE-2023-29389 for Toyota RAV4. An ECU that manages the lights (the high and low beam headlights as well as the turn indicators) is located in the front of the RAV4.



Figure 11: Breaking into Toyota RAV4

Because lights are now intelligent and include features like motors to level the headlights (so that when the car is loaded with heavy luggage, the lights are turned to compensate), steering headlights to illuminate the corners, the ability to automatically detect when the lights have failed, the ability to turn on pumps to spray water on the lights, and more, most cars now have such an ECU. In order to avoid blinding approaching drivers, it is also possible to select which LEDs in a grid on the RAV4 are lighted up to avoid dazzle the oncoming drivers.

The way the CAN injection attack works is by manipulating the CAN bus inside the car. Toyota RAV4 has its CAN bus near the headlights. The attacker would usually break the side of the car to get into cars internal communication and then use a simple electronic device to send CAN frames on to the CAN bus to send fake “Key is validated” messages as if from the smart key receiver. The gateway ECU (a simple device that just copies certain CAN messages back and forth) will copy that fake message over to the CAN bus, and the engine control system will accept the message and deactivate the immobilizer function. The theft device is designed to be connected to the control CAN bus to impersonate the smart key ECU. The wires for this CAN bus can be accessed in a variety of ways; the only condition needed is that they come to the edge of the car so that they can be reached (wires buried deep inside the car are difficult for criminals to reach while attempting to steal a parked automobile on the street). The RAV4's may bus may be accessed from the headlight connector by lifting the bumper aside and going through the headlights, which is by far the simplest method. When the device is first powered on, the CAN Injector does nothing: it's just listening for a particular CAN message to know that the car is ready. As soon as it receives this CAN message it does two things: it starts with sending a burst of CAN messages (at about 15-20 times per second) and it activates that extra circuit connected to its CAN transceiver. This burst of CAN messages contains a signal ‘smart key is valid’, and the gateway will forward this to the engine management ECU on the other bus. Now, this would obviously cause confusion on the control CAN bus since it will be difficult to identify which CAN is sending these messages because CAN messages from the real smart key controller would clash with the imposter messages from the CAN Injector, and this could stop the gateway from forwarding the injected message. This is where that extra circuit comes in. It changes the way a CAN bus operates to ensure that other ECUs on that bus cannot talk. Of course, the gateway can still listen to messages, and can still send messages on to the CAN bus. The burst continuously repeats 20 times a second and sometimes the gateway is not listening because its CAN hardware is resetting itself because it thinks that being unable

to talk is an indication of a fault. When the fake CAN device button is played the burst of CAN messages changes slightly and they instruct the door ECU to unlock the doors (as if the ‘unlock’ button on the wireless key had been pressed). The thieves can then unhook the CAN Injector, get into the car, and drive it away.

The below image shows how ECUs in a RAV4 are wired together with CAN Bus. We assume this can be very similar to other modern vehicle cars.

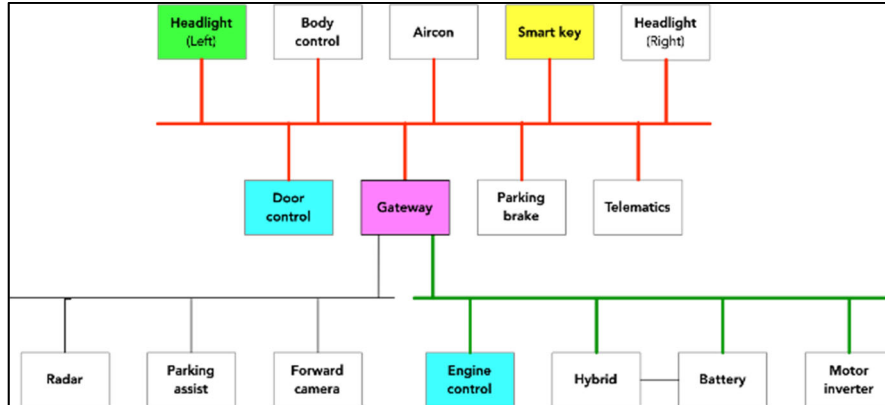


Figure 12: Wiring of ECU in RAV4

The above image consists of three CAN buses shown:

- i) A control CAN bus (this contains ECUs for headlights, telematics, door control, aircon, etc.)
- ii) A powertrain CAN bus (which has ECUs for engine control, the hybrid battery and motor control, etc.)
- iii) An autonomy CAN bus (which has ECUs for radar, forward looking camera, and self-parking)

These internal communication inside the CAN is not protected so anyone who breaks into the wiring for the red CAN bus and connects a rogue CAN device to send fake CAN frames will be copied and forwarded to the green CAN bus which will deactivate the immobilizer and can unlock the car.

Here's an image of a CAN Injection device



Figure 13: A CAN Injection device masked as a JBL Speaker

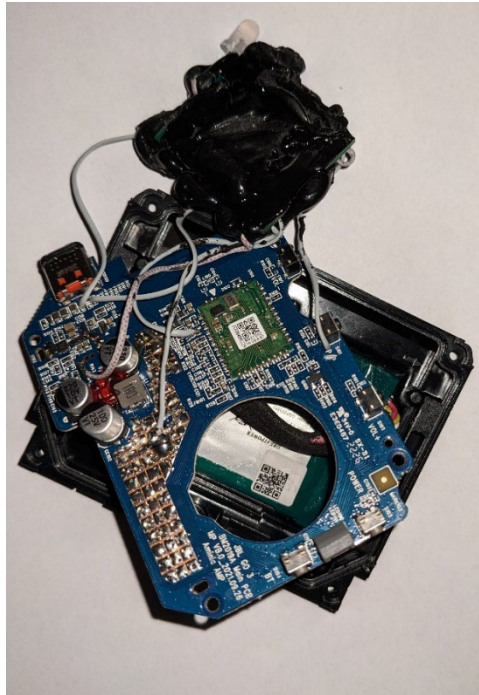


Figure 14: Inside the Speaker

This is a custom-made chip board consisting of about \$10 of components: a PIC18F chip which has a CAN hardware, plus a software pre-programmed into the chip known as firmware, a CAN transceiver which is a standard CAN chip that turns digital signals from the CAN hardware on the PIC18F into the analog voltages sent on CAN wire, and an extra circuit connected to the CAN transceiver. The device is built to take its power from the speaker battery, which then connects to a CAN bus. Surprisingly there are products sold targeting many car models, including from Jeep, Maserati, Honda, Renault, Jaguar, Fiat, Peugeot, Nissan, Ford, BMW, Volkswagen, Chrysler, Cadillac, GMC and Toyota.

5.1. Decoding Replay Attacks

The goal of replay attack is to intercept, steal, and store the information contained in a message delivered from the automobile remote or key fob, saving it for a future attack. Once the hacker has the pertinent message, they may utilize it whenever they want to launch an attack. Now, since this is one of the most oldest type of attack which still exist the second reason of choosing this demonstration is to paint an understanding how can message works and what tools can we use to capture it and how easily such tools can be downloaded from internet to practice and understand the working of CAN message and due to its lack of encryption and cryptographic absence how an attacker once access to your vehicles CAN message can intercept, read and modify the message to steal your car.

Disclaimer: The actions described herein are conducted solely within a controlled and secure network environment. The purpose of these actions is to gain a comprehensive understanding of potential vulnerabilities within CAN messaging systems. It is imperative to emphasize that these activities are not intended to serve as a reference or guide for carrying out malicious attacks on any vehicle or system.

So, lets understand how the attack works? When you acquire your brand-new vehicle, a key fob is given to you in place of a set of keys. The fob functions as a short-range radio transmitter that always stays in contact with your automobile. A receiver unit in the automobile and the fob recognize one another with a coded signal when they are near the specified vehicle. A vehicle can be given instructions via sent signals to carry out a variety of tasks, including as unlocking or locking doors, opening or closing windows, turning on the headlights, or even turning on the air conditioning. These

messages are passed over through a CAN Bus. As we studied earlier CAN bus acts very much like a Hub (networking device) whose role is to just forward whatever messages it has received over the network. Hub in general is a very noisy and a dumb device since it cannot filter a data, i.e., it cannot identify the destination of the packet, so it broadcasts or send the message to each port. Also, its is considered as an insecure device due to the ability to eavesdrop on that network. Every time you press the lock button on our car, an electrical signal is sent through the CAN bus to every device on the network, and the doors interpret the data packet. The data packet will contain a command: 'lock' or 'unlock'. Upon receiving the data packet, the doors will execute the directives in the packet. A number of devices in the vehicle operate in this manner, with an action being triggered, data being sent over the network, and the appropriate device taking the appropriate action. The gas pedal would cause data packets in the network to be sent telling the gas flow to increase, so the car would go faster. As a result, the speedometer may rise to reflect the appropriate speed of the vehicle. Depending on the amount pressed on the gas pedal, the speedometer will rise by the amount instructed on the network. As stated earlier, the packets sent over the network are made of two parts: an 'identifier' and the 'data'. The identifier is a representation for the device in the vehicle meaning door, window etc and the data field represents what instruction need to be completed with the mentioned device. Consider the below example.

120#F289632003200321

The beginning message of the packet is the identifier. Here the identifier is 120. The portion after # is the data field. It is important to note that the identifier will be different for different vehicles, and it is subjective to the car model, year and the make of the car. Meaning, the identifier 120 may be a gas pedal for a Honda Civic 2019 but it may represent car window for Toyota RAV4 2022. We may be able to identify the devices on one vehicle with its identifier tag, but only other vehicles with the same year/make/model will have the same device controlled with that identifier tag value. For decoding the attack, we will need some tools since the attack is a virtual simulation and not performed on a real vehicle however, the same tools can be used if this needs to be mimicked for a real-world replay attack. For the purpose of demonstration, we will use the below software's:

a) Kali Linux: It is a Debian-derived Linux distribution designed for digital forensics and penetration testing. This can be downloaded free from the internet by visiting the link:
<https://www.kali.org/docs/introduction/download-official-kali-linux-images/>

b) ICSim (Instrument Cluster Simulator): ICSim is a software utility designed to produce CAN signals in order to provide a lot of seemingly "normal" background CAN noise—essentially, it's designed to let you practice CAN bus working and reverse engineering without actually needing to have a real car. It's a free tool and can be downloaded via the link.
<https://github.com/zombieCraig/ICSim>

c) Socketcand: It is a set of open source can drives that provides access to CAN interfaces on a machine via a network interface. It can be downloaded from the below link.
<https://github.com/linux-can/socketcand>

d) Kayak: Kayak is an application for CAN bus diagnosis and monitoring. It is opensource and can be downloaded using the link below.
<https://github.com/dschanoeh/Kayak>

All the below tools can be installed using below commands inside the Kali Machine

```
kali@kali:~$
```

```
sudo apt install libsdl2-dev libsdl2-image-dev can-utils maven autoconf
```

```
git clone https://github.com/zombieCraig/ICSim.git
```

```
git clone https://github.com/linux-can/socketcand.git
```

```
git clone https://github.com/dschanoeh/Kayak.git
```

```
cd ICSim
```

```
sudo make
```

```
cd socketcand
```

```
wget https://raw.githubusercontent.com/dschanoeh/socketcand/master/config.h.in
```

```
autoconf
```

```
./configure
```

```
make clean
```

```
make
```

```
sudo make install
```

```
cd Kayak
```

```
mvn clean package
```

Once installed the command: `cd ../ICSim && ./icsim vcan0` can be run to start the virtual simulator



Figure 15: Virtual Simulator

This is how the simulator looks when its running. The simulator is mapped to our keyboard arrow keys. Pressing the Up-arrow key will increase the speedometer, indicating the vehicle accelerating. The left arrow key and right arrow key will signal the vehicle indicators respectively. If we press the Right Shift+A|B|X|Y, one door will be unlocked at a time. Similarly, to lock the doors again press the Left Shift key with any of the door buttons. If we press the Left SHIFT then Right Shift, all the doors will be unlocked. Now, to record our can message we can use a tool called **candump** which we installed previously, and we can run it

using command: **candump vcan0 -l** (Note: Messages coming from CAN Bus are extremely loud for the purpose of demonstration we can quickly close it to avoid capturing large numbers of messages). **Ctrl+C** to close the tool. Checking the logs we can see a huge number of CAN messages

```

File Actions Edit View Help
(1687353227.160035) vcan0 17C#000000001000021
(1687353227.160098) vcan0 18E#00006B
(1687353227.160100) vcan0 1CF#80050000003C
(1687353227.160103) vcan0 1DC#02000039
(1687353227.160104) vcan0 183#0000000700001022
(1687353227.161711) vcan0 143#6B6B00E0
(1687353227.162469) vcan0 244#0000000148
(1687353227.164240) vcan0 095#800007F400000017
(1687353227.164302) vcan0 1A4#0000000800000010
(1687353227.164304) vcan0 1AA#7FFF000000006810
(1687353227.164306) vcan0 1B0#000F00000000157
(1687353227.165518) vcan0 1D0#000000000000000A
(1687353227.167371) vcan0 166#D0320027
(1687353227.167464) vcan0 039#00039
(1687353227.167472) vcan0 158#0000000000000028
(1687353227.167478) vcan0 161#000005500108002B
(1687353227.167483) vcan0 191#010090A1410012
(1687353227.169077) vcan0 305#8035
(1687353227.169165) vcan0 164#0000C01AA8000013
(1687353227.169171) vcan0 133#0000000086
(1687353227.169176) vcan0 136#0002000000000039
(1687353227.169180) vcan0 13A#0000000000000037
(1687353227.169185) vcan0 13F#000000050000003D
(1687353227.169189) vcan0 17C#0000000010000030
(1687353227.169194) vcan0 18E#00007A
(1687353227.171270) vcan0 294#040B0002CF5A000E
(1687353227.171368) vcan0 21E#03E83745220601
(1687353227.171375) vcan0 309#00000000000000B1
(1687353227.171382) vcan0 428#01040000521C3E
(1687353227.171387) vcan0 405#0000040000000038
(1687353227.171394) vcan0 183#000000D0000103B
(1687353227.171400) vcan0 143#6B6B00FF
(1687353227.173314) vcan0 095#800007F400000026
(1687353227.173426) vcan0 244#000000010F
(1687353227.177183) vcan0 166#D0320036
(1687353227.177499) vcan0 158#0000000000000037
(1687353227.177519) vcan0 161#000005500108003A
(1687353227.177530) vcan0 191#010090A1410021
(1687353227.179802) vcan0 164#0000C01AA8000022
(1687353227.179887) vcan0 133#0000000089
(1687353227.179894) vcan0 136#000200000000000C
(1687353227.179899) vcan0 13A#000000000000000A
(1687353227.179903) vcan0 13F#0000000500000000
(1687353227.179907) vcan0 17C#0000000010000003
(1687353227.179912) vcan0 18E#00004D
(1687353227.179917) vcan0 1CF#80050000000F
(1687353227.179922) vcan0 166#D0320036

```

Figure 16: Can Logs

The CAN messages are broken down wherein 1) arbitration ID 2) the interface of the CAN packet and 3) the CAN message. (Note: This can also be done using Wireshark which is a network capturing tool). If you prefer more of a GUI interface, Kayak, which we installed previously can also be used. Kayak offers a few unique capabilities that you can't easily get on the command line, such as documenting the identified packets in XML (.kcd files), which and displaying virtual instrument clusters and map data. Our goal here is to identify the exact code which replayed again can unlock the car doors. To do that we will follow the below steps:

- a) Run candump to log on the vcan0 interface.
- b) Keep the CAN bus Control Panel window active.
- c) Unlock all the doors.
- d) Lock all the doors.
- e) Click in the candump terminal window.
- f) Press CTRL-C to stop the network capture.

For the purpose of the demo these steps need to be completed as quickly as possible to keep the captured file (log) being as small as possible. This would help us to identify the CAN code easily. Once we complete the above steps now, we can use another tool **canplayer** to re-run the captured code command: **canplayer -l candump-2023-06-06_XXXX.log**. Running the tool, we

can see the car door opening and closing this ensures that the captured CAN message contains the door locking and unlocking code. We have the captured log file which confirms our desired action, now to in order to find the exact code we can make a best guess as to where in the file the action occurs. With that guess, we can decide to play the first half or the last half of the file until we get the correct code. In order to do that we first need to determine the size of our log and then split it in half and then replay the split file using canplayer tool. To check the size of our log we can use command: `wc -l candump-xxx-xxxx.log` [X represents the complete name of the log file]. We can use `head` or `tail` with the `-n` option and take approximately half the file to replay.

Command: `head -n 4500 candump-2023-06-06_xxxx.log > split1` [we named the file split1]
`canplayer -I split1` [Replaying the file split1 using canplayer]
`wc -l split1` [this would now give the current size of the log file name 'split 1']

Now, we keep halving the size of the packet capture until we are left with a single packet, at which point we can identify what bits are used to unlock and lock the door. The below image is from the 77th page of “The Car Hacker’s Handbook” by Craig Smith (Smith, 2016) which perfectly puts in a flow chart on how the above process needs to be carried out.

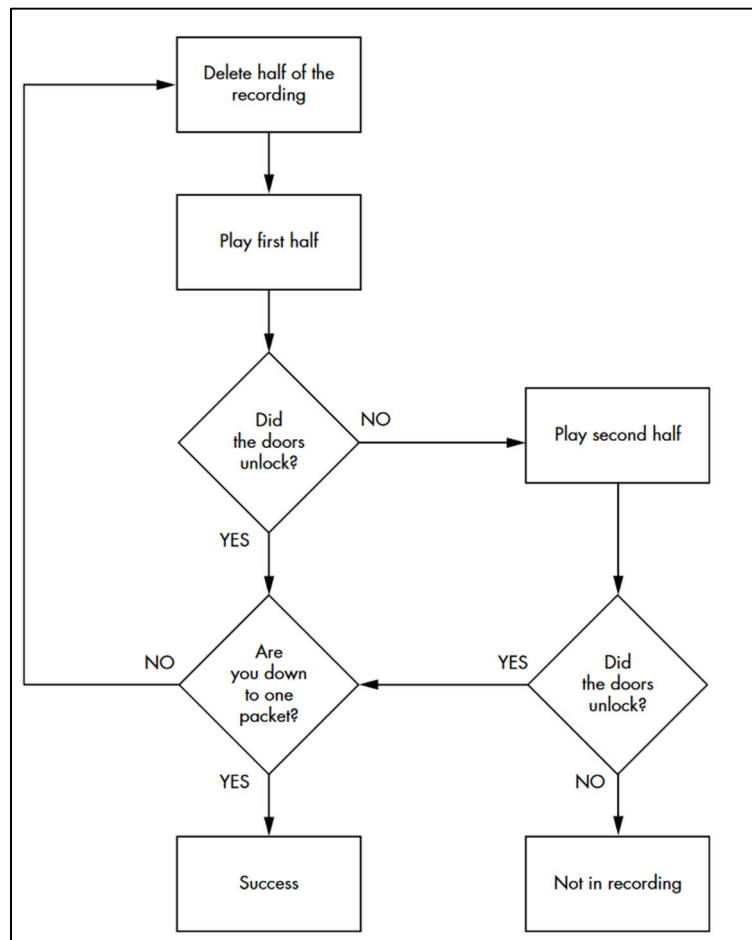


Figure 17: Sample flow chart of reverse engineering

With enough repetition of this process, we will be able to identify the CAN packet responsible for unlocking the doors. However, each time we play a CAN file we will have to manually lock the door again using right shift and left shift because the IC Simulator will not show a change

if the doors are already in an unlocked state. During the demonstration we identified the CAN packet which played can open the vehicle door. Let us decode the CAN message to understand it better.

(1654550641.304018) VCAN0 19B#000000000000

The highlighted part of the packet is the identifier. This means that the doors are identified with the hexadecimal code of 19B. The second highlighted portion of the packet is the data. Here, the data is all set to 0's, With the doors identifier now known, let us search deeper for the original packet capture for that identifier to see what other variations to the data portion of the packet exist. We can use the below command for that:

grep 19B candump-2022-06-06_142359.log

This will identify and only display message containing 19B inside the log file. Looking at the message we find another message *(1654550642.112533) vcan0 19B#00000F000000*. Playing back the same packet we identified that this particular message locks the Car door.

5.1 Further Analysis

We have successfully identified two CAN messages. One which unlocks all car door and one which locks back the door. We can further analyze the data more to get more granularity and identify if we can control each door.

(1654550641.304018) VCAN0 19B#000000000000

(1654550642.112533) VCAN0 19B#00000F000000

As per our analysis the data is changed on the second nibble of the third byte. If we break this down to its binary form, we can show it like this:

8	4	2	1	Hex
0	0	0	0	0
1	1	1	1	F

The packet which unlocks all the doors is represented with nibble 0 and the packet that locks all doors represents the hexadecimal F. If we break this down further, we will get 16 possible door combinations.

8	4	2	1	Hex	Door
1	0	0	0	8	?
0	1	0	0	4	?
0	0	1	0	2	?
0	0	0	1	1	?

Trying all the possible combinations we came to conclusion that below packets are responsible which locks each door when replayed.

(1654550741.306018) VCAN0 19B#000008000000

(1654550342.123853) VCAN0 19B#000004000000

(1654550741.306018) VCAN0 19B#000002000000

(1654550342.123853) VCAN0 19B#000001000000

We saved each door message with its name file like doorUnlock1, doorUnlock2 and so on and replaying each message we find the below findings:

8	4	2	1	Hex	Door
1	0	0	0	8	Back Door to the left
0	1	0	0	4	Back Door to the right
0	0	1	0	2	Fron Door to the left
0	0	0	1	1	Front Driver Window

If we account for the action of each door, 1 is the action to lock a specific door and 0 is the action to unlock it. Now, considering we understand how the CAN packet works let us try to modify the CAN packet to lock both the back doors of the vehicle and unlock only the front doors. To identify the correct Hex code let's consider the below table.

Back Door (L)	Back Door (R)	Front Passenger	Driver Seat	
8	4	2	1	Hex
?	?	?	?	?

As per our analysis, if an unlock bit is 0, then we can fill the front doors with the value of 0. A value of 1 will result in locking the doors, so the table would look like

Back Door (L)	Back Door (R)	Front Passenger	Driver Seat	
8	4	2	1	Hex
1	1	0	0	C

The binary result of this is $1(8) + 1(4) + 0(2) + 0(1) = 8+4+0+0 = 12$. Hexadecimal format of 12 is represented as C. This changes our CAN packet to:

(1654550342.123853) VCAN0 19B#00000C000000

Saving this file as doorUnlock and replaying back it again confirms that we can unlock the front door as seen in the image below.

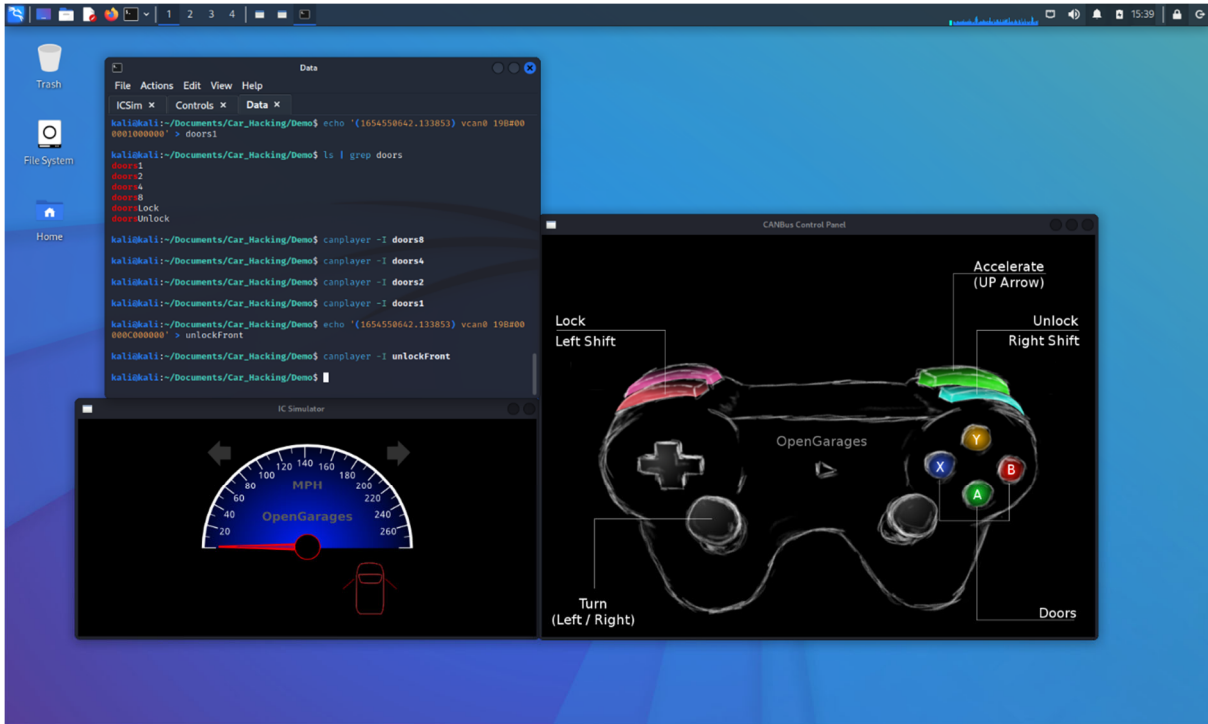


Figure 18: Replaying the front unlock door message

6. Conclusion

In an era defined by technological progress and digital connectivity in modern vehicles, security still stands as one of the most prominent factors yet to be considered seriously. There is no doubt that the automotive industry is growing rapidly, and security would be one of the stand points the ever growing customer market will consider. Customers would not only like to feel physical security in their luxury cars but would believe the digital security would also be taken care by the car manufacturers. The paramount importance of cybersecurity in the realm of smart cars cannot be overstated; it is the cornerstone that safeguards not only the driving experience but also human lives. As we navigate towards a phase of innovation and risk, a comprehensive approach to cybersecurity emerges as an imperative, requiring the collaboration of manufacturers, researchers, policymakers, and consumers. It is important to not only protect the Car communication network but also web applications and mobile applications which car companies build. Having a mobile application of the Car you own becomes handy because the application can send you alerts and notifications about the ongoing activities of your car which needs your attention for e.g. to get your car service done after a certain time period or getting your engine oil changed but, car application can significantly become a threat if the customer data stored in the application is not protected which is generally the case with most car companies. Sam Curry demonstrated (Curry, 2023) vulnerabilities in car applications of most of the car companies which not only allowed him to access customer sensitive information, vehicle information but also to remotely unlock the car without the need of any radio device or tools.

The statement, "Cybersecurity is now the need of the hour," resonates with undeniable urgency. Smart cars, equipped with AI, connectivity, and autonomy, are positioned at the forefront of technological advancement. Yet, this very advancement renders them vulnerable, a susceptibility that hackers are quick to exploit. The prospect of a car being hacked while in motion, with vital systems such as airbags, brakes, and accelerators manipulated remotely, unveils a dangerous imagination if not taken seriously can easily become a scary reality. Through this research paper it is prominent that the central axis of most vulnerability lies within the CAN bus system, the nerve centre of automobile that provides communication among various electronic control units within a vehicle. Through these research paper we can conclude the need of the below steps which car manufacturers and the government should actively ensure

I) **Secure Communication Channels:** Implementing strong encryption protocols for communication between different parts of the car, such as the Control Area Network (CAN) Bus and applying time stamps on those communication messages most of the attack vectors can be reduced. Apply secure coding practices to prevent vulnerabilities in the software that controls the car's functions.

II) **Regular Software Updates:** Ensuring that all car software, including infotainment systems and control units, are up to date with the latest security patches. Regularly updating the car's firmware to fix any vulnerabilities that may have been discovered.

III) **Multi-Layered Authentication:** Introducing multi-layered authentication for accessing the car's systems and controls. Requiring strong, unique passwords for remote access and device pairing, making it harder for hackers to gain unauthorized access.

IV) **Built-In Intrusion Detection Systems:** Manufacturers should try finding a way to install intrusion detection systems that can monitor the car's network for any unusual or unauthorized activities. These systems should alert the car owner or manufacturer if any suspicious activity is detected.

V) Ethical Hacking and Car Pentesting: Car manufacturers should collaborate with independent security researchers/companies who can ethically hack and do penetration tests to identify vulnerabilities before hackers do and using these insights to fix weak points in the car's software and systems.

VI) Secure Boot Process: Implement a secure boot process to ensure that only trusted and authenticated software is loaded when the car starts. This prevents malicious software from infiltrating the car's systems during startup.

VII) Over-the-Air Updates: Most manufactures now use OTA approach to implement security fixes. Using secure over-the-air (OTA) update mechanisms to deliver software updates to the car and ensuring that OTA updates are signed and verified to prevent unauthorized updates.

VIII) Hardware Security Measures: Using robust hardware security modules to store sensitive data and cryptographic keys securely. Employing secure hardware components that resist tampering and physical attacks.

IX) User Education: Educating car owners about the importance of cybersecurity and safe practices, such as not sharing sensitive information or connecting to unsecured Wi-Fi/Bluetooth networks. Encourage users to report any unusual behaviour or anomalies in their car's systems and car's software application.

X) Collaboration and Regulation: Collaborating with cybersecurity experts, researchers, and industry groups to stay informed about the latest threats and best practices. Investing in cyber security and adhering to regulatory standards and guidelines, such as the UNECE WP.29 regulations, to ensure a minimum level of cybersecurity in vehicles.

XI) Continual Monitoring: Car manufactures should ensure continuous monitoring of the car's systems for any signs of unusual or unauthorized activity. Implementing a strong mechanism to detect anomalies and respond promptly to potential threats.

By adopting these preventive measures, car manufacturers can fortify the cybersecurity of smart vehicles and reduce the risk of car hacking incidents. The United Nations Economic Commission for Europe (UNECE) WP.29 regulations are indicative of a collective acknowledgment of the gravity of the situation. These regulations mandate stringent cybersecurity measures within the automotive industry, incentivizing manufacturers to invest in secure design practices, resilient software architecture, and continuous monitoring mechanisms. In conclusion, the fusion of smart technology and vehicles has introduced an era of unparalleled convenience and innovation. Yet, this transformation reflects a parallel reality, one where vulnerabilities loom large and cyber threats are ever-present. The critical juncture at which the automotive industry finds itself demands proactive measures, collaboration, and a relentless pursuit of cybersecurity excellence. By fortifying the CAN bus system, conducting rigorous pentesting, adhering to regulatory standards, and fostering an informed consumer base, we can pave the way toward an automotive future that is not only smart but also secure. With these concerted efforts, we can rise to the challenge of mitigating cyber risks, ensuring the integrity of smart cars, and ultimately, safeguarding the lives of those who trust in the promise of modern transportation.

7. References

1. Berry, B., 2020. *CVE-2019-20626*. s.l. Patent No. CVE-2019-20626.
2. Curry, S., 2023. *Blog*. [Online]
Available at: <https://samcurry.net/web-hackers-vs-the-auto-industry/>
[Accessed 20 08 2023].
3. Hackaday, 2017. *Hackaday*. [Online]
Available at: <https://hackaday.com/tag/samy-kamkar/>
[Accessed 17 08 2023].
4. HackingIntoYourHeart, 2022. *Github-Unoriginal-Rice-Patty*. [Online]
Available at: <https://github.com/HackingIntoYourHeart/Unoriginal-Rice-Patty>
[Accessed 20 08 2023].
5. Linn, H. W. & Csikor, L., 2022. A New Time-Agnostic Replay Attack Against the Automotive Remote Keyless Entry Systems. *A New Time-Agnostic Replay Attack Against the Automotive Remote Keyless Entry Systems*, p. 24.
6. Max Eddy, 2022. *Security*. [Online]
Available at: <https://uk.pcmag.com/security/142046/is-your-car-key-fob-vulnerable-to-this-simple-replay-attack>
[Accessed 18 08 2023].
7. Smith, C., 2016. The Car Hacker's Handbook. In: *The Car Hacker's Handbook*. s.l.:s.n., p. 77.
8. The Guardian, 2019. *Cars*. [Online]
Available at: <https://www.theguardian.com/money/2019/jan/28/uk-keyless-cars-theft-which-ford-fiesta-vw-golf-nissan-qashqai-focus>
[Accessed 15 08 2023].
9. Tindell, K., 2023. *CTO Blog*. [Online]
Available at: <https://kentindell.github.io/2023/04/03/can-injection/>
[Accessed 19 08 2023].
10. WHATCAR, n.d. *WHATCAR*. [Online]
Available at: <https://www.whatcar.com/news/car-theft-group-test-britains-most-secure-cars/n19875>
[Accessed 15 08 2023].
11. Wired, 2015. *Security*. [Online]
Available at: <https://www.wired.com/2015/08/hackers-tiny-device-unlocks-cars-opens-garages/>
[Accessed 17 08 2023].