Engineering xxx (xxxx) xxx

Contents lists available at ScienceDirect

Engineering

journal homepage: www.elsevier.com/locate/eng

Research
Robotic–Article

CORMAND2: A Deception Attack Against Industrial Robots

Hongyi Pu [a], Liang He [b], Peng Cheng [a], Jiming Chen [a], Youxian Sun [a,*]

[a] Control Science and Engineering, Zhejiang University, Hangzhou 310000, China
[b] Computer Science and Engineering, University of Colorado Denver, Denver, CO 999039, USA

ARTICLE INFO

Article history:
Received 26 February 2022
Revised 1 January 2023
Accepted 13 January 2023
Available online xxxx

Keywords:
Industrial robots
Vulnerability analysis
Deception attacks
Defenses

ABSTRACT

Industrial robots are becoming increasingly vulnerable to cyber incidents and attacks, particularly with the dawn of the Industrial Internet-of-Things (IIoT). To gain a comprehensive understanding of these cyber risks, vulnerabilities of industrial robots were analyzed empirically, using more than three million communication packets collected with testbeds of two ABB IRB120 robots and five other robots from various Original Equipment Manufacturers (OEMs). This analysis, guided by the confidentiality–integrity–availability (CIA) triad, uncovers robot vulnerabilities in three dimensions: confidentiality, integrity, and availability. These vulnerabilities were used to design Covering Robot Manipulation via Data Deception (CORMAND2), an automated cyber–physical attack against industrial robots. CORMAND2 manipulates robot operation while deceiving the Supervisory Control and Data Acquisition (SCADA) system that the robot is operating normally by modifying the robot's movement data and data deception. CORMAND2 and its capability of degrading the manufacturing was validated experimentally using the aforementioned seven robots from six different OEMs. CORMAND2 unveils the limitations of existing anomaly detection systems, more specifically the assumption of the authenticity of SCADA-received movement data, to which we propose mitigations for.

1. Introduction

Industrial robots, such as the arm-like programmable mechanisms widely used in manufacturing, are prototypical cyber–physical systems (CPSes) that interact with the physical world through sensing, communication, computation, and control [1]. According to the International Federation of Robotics (IFR), over three million industrial robots were deployed in manufacturing systems in 2021 [2]. Fig. 1 shows the typical operation scenario of industrial robots that are in accordance with the ISO10218 [3] and ISO12100 [4] standards on the safety of industrial robots. The typical operation process is as follows:

Process 1: the human operator logs on to the robot with his/her credentials and programs it using development software, such as RobotStudio for ABB robots;

Process 2: the robot operates as programmed and reports its real-time operation, that is, the movement data collected by onboard sensors, to the Supervisory Control and Data Acquisition (SCADA) system;

Process 3: SCADA displays the received movement data and detects the potential abnormal robot operation using anomaly detection systems;

Process 4: the human operator monitors the robot's operation by observing the SCADA display.

Note that the industrial network that facilitates processes 1 and 2 in Fig. 1 is commonly constructed using EtherNet/IP. Moreover, while the robot is in operation, human operators usually have limited access to the manufacturing facility.

The increasing number of cyber attacks on manufacturing systems demonstrates the vulnerability of industrial robots to being manipulated [5–8]. For example, manufacturing factories were extorted for 6.9 million USD in 2019 [9]. Other victims include Honda and Aebi Schmidt, who were forced to halt production as a result of cyber attacks [10,11]. As an important component of manufacturing systems, industrial robots may be targeted by adversaries; thus, the security of industrial robots has piqued the interest of many researchers. For example, Quarta et al. [12] changed the operation of an ABB robot by modifying the proportional gain of the robot's proportion–proportion–differentiation (PID) controller. Alemzadeh et al. [13] used forged control commands to manipulate the movement of a Raven robot. Apa [14] hacked a

* Corresponding author.
E-mail address: sunyx@zju.edu.cn (Y. Sun).

https://doi.org/10.1016/j.eng.2023.01.013

Please cite this article as: H. Pu, L. He, P. Cheng et al., CORMAND2: A Deception Attack Against Industrial Robots, Engineering, https://doi.org/10.1016/j.eng.2023.01.013

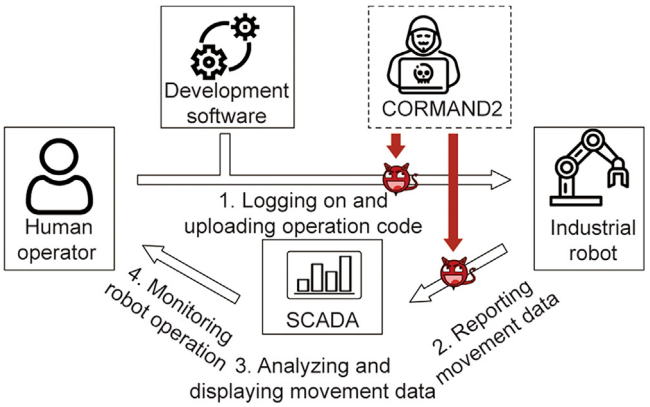


Fig. 1. Typical industrial robot operation and the mounting of Covering Robot Manipulation via Data Deception (CORMAND2). SCADA: Supervisory Control and and Data Acquisition.

UR robot's Modbus server and operated it beyond the boundaries of safety.

To gain a comprehensive understanding of these cyber risks, the vulnerabilities of industrial robots were analyzed using over three million communication packets collected from seven robots from six different Original Equipment Manufacturers (OEMs) (Figs. 2 and 3) that represent over 55% of the global market [15]. All of the seven robots were networked with EtherNet/IP. This study was based on the confidentiality–integrity–availability (CIA) triad, a commonly used model for security analysis, and three key findings were made: ① even the most widely used security measures, such as strong encryption and data hashing, were absent in the robots, leaving them vulnerable in all three dimensions defined by the CIA triad; ② to the best of our knowledge, the majority of reported industrial robot attacks exploit at least one of the three vulnerabilities; ③ the three vulnerabilities are interrelated, meaning that compromised confidentiality undermines integrity and availability, and weakened integrity enables the concealment of impaired availability.

These vulnerabilities were used to design Covering Robot Manipulation via Data Deception (CORMAND2), which is a novel cyber–physical attack that exploits compromised robot confidentiality to further impair integrity and availability. For example, CORMAND2 can manipulate robot operation (hacking process 1 in Fig. 1) while deceiving SCADA, which monitors normal robot operation, by modifying the robot's movement data (hacking process 2 in Fig. 1). The collaborative nature of CORMAND2's deception attack not only sets it apart from previously reported attacks, but more importantly, it allows CORMAND2 to evade an important class of anomaly detection systems that detect robot manipulation using the movement data collected by SCADA [16–18] by undermining the common assumption that movement data collected by SCADA truly reflects the robot's movement.

CORMAND2's data deception is based on the Man-in-the-Middle (MITM) attack, which establishes new Transmission Control Protocol (TCP) connections between two victims using proxy-based solutions, such as Mitmproxy and Burp Suite. These solutions, however, do not apply to industrial robots because the connection between the robot and SCADA is established and maintained throughout system operation. CORMAND2 overcomes this challenge by mounting the MITM attack and modifying the existing connection between the robot and SCADA without causing anomalies in SCADA-received robot movement, communication packets, and TCP connection.

CORMAND2 is a type of malware that can install and mount attacks automatically. This study onseven robots from six different OEMs confirmed the ability of CORMAND2 to bypass existing anomaly detectors that assume the authenticity of SCADA-received movement data [17]. In addition, CORMAND2 causes a minimal increase in communication latency (1.7 ms), packets (0.42%–1.10%), and packet retransmission rate (0.0073%) on average, demonstrating its negligible impact on communication statistics, which makes it difficult to detect. CORMAND2's potential to reduce production efficiency and cause physical damage was

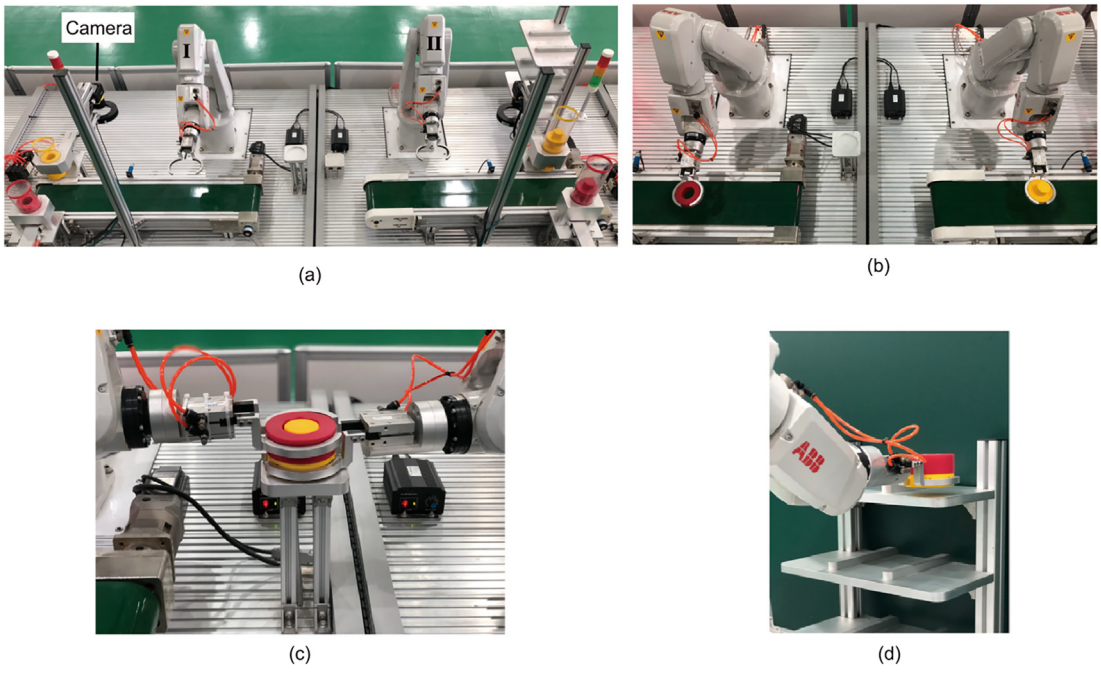


Fig. 2. Typical assembly operation on the Secure Industrial Robot Pick-and-Place (SIRP) testbed. (a) Two robots at home position; (b) two robots picking up objects from conveyors based on positions measured by cameras; (c) two robots assembling the two objects into a cylinder; (d) robot on the right placing the assembled structure into the warehouse.

Fig. 3. Five robots from different OEMs. (a) KUKA; (b) UR; (c) COMAU; (d) NACHI; (e) UFACTORY.

assessed to demonstrate the potential consequences of a vulnerable CIA triad in the physical world. Solutions to protect robots against CORMAND2 were also proposed. Over three million communication packets were made openly accessible to facilitate more extensive research on the security of industrial robots.

The major contributions of this paper are as follows:

- Identification of the vulnerabilities of industrial robots in terms of confidentiality, integrity, and availability;
- Design of CORMAND2, a cyber–physical attack that exploits these vulnerabilities to conceal manipulated robot operation through data deception, which is, to the best of our knowledge, the first attack of its kind against industrial robots, as it not only manipulates robot operation but also conceals the manipulation from detection;
- Implementation of CORMAND2 on seven robots fromsix different OEMs, and verification of CORMAND2′s ability to evade existing anomaly detectors, its negligible communication overheads, and its impact on manufacturing;
- Solutions for protecting robots from CORMAND2.

2. Related work

This section is a brief discussion on the literature related to CORMAND2.

(1) Security of industrial robots. Substantial research has been done on the vulnerabilities of industrial robots and the attacks that have been mounted against them, as well as on the development of detectors to protect these robots. Pogliani et al. [6], Quarta et al. [12], Maggi et al. [19], Apa et al. [14], and Chan et al. [20] investigated the memory corruption, access control, configuration file, and firmware vulnerabilities of industrial robots. Alemzadeh et al. [13], Apa et al. [14], Chung et al. [21], and Dieber et al. [22,23] manipulated robots to operate beyond the boundaries of safety. Solutions have been developed based on physical models, machine learning, such as support vector machine (SVM), and other formal methods to detect these manipulations by analyzing the robot's movement data, such as joint angles, joint speeds, and joint torques [13,17,24]. Researchers have achieved an average detection accuracy of 90% [13], and attacks were detected when the trajectory of the industrial robot deviated for 2 mm [17] or when the operation of the industrial robot did not correspond to that defined by its specifications [24]. However, existing vulnerability analyses lack systematic guidance, such as the CIA triad discussed in this paper, making it difficult to design advanced attacks against robots. Furthermore, most attack detectors assume the authenticity of the collected movement data, which CORMAND2 bypasses.

(2) Security of industrial protocols. Most industrial protocols, including Modbus, Distributed Network Protocol 3 (DNP3), and EtherNet/IP, are not secure by design, making them vulnerable to cyber attacks [25]. To detect attacks on these protocols, numerous intrusion detection systems (IDSs) have been developed and can be classified as protocol analysis-based or communication traffic-based detectors [26]. Protocol analysis-based detectors specify message formats and detect attacks based on predefined specifications [27,28], while communication traffic-based detectors use data mining or statistical analysis algorithms to extract communication traffic patterns [29,30]. However, protocol analysis-based detectors require a long time to parse data packets, and communication traffic-based detectors may generate false alarms.

(3) Concealing system manipulation via data deception. The risk and the mitigation of attacks that manipulate the operation of industrial control systems (ICS) through data deception have been analyzed [31]. However, few such attacks have been mounted successfully in practice: ① Stuxnet modifies the rotating speed of centrifuges and conceals the manipulation by replaying prerecorded normal speed to SCADA [32]; ② Harvey injects malicious commands into power systems while forging seemingly normal sensor measurements to SCADA, deceiving the human operator into thinking the power system is operating normally [33]. However, it remains unclear whether similar attacks on industrial robots are feasible.

(4) Sniffing, spoofing, and MITM attacks. Various penetration testing tools can be used to sniff communication packets and mount spoofing or MITM attacks on information technology (IT)/ICS systems. Penetration tools designed specifically for sniffing include Wireshark, p0f, TCPDump, and dSniff, but they are unable to mount real-time spoofing and MITM attacks. Common spoofing tools include arpspoof and dnsspoof, with the latter designed to attack Hyper Text Transfer Protocol (HTTP). Mitmproxy and Burp Suite are traditional MITM attack tools designed to intercept communication between two victims by establishing new TCP connections. However, neither Mitmproxy nor Burp Suite can be used on industrial robots because the TCP connection between the robot and SCADA is maintained throughout system operation. Furthermore, Ettercap and NetfilterQueue are both integrated attack tools that can sniff communication and mount spoofing and MITM attacks. However, Ettercap is less customizable than NetfilterQueue, which can intercept, record, and modify communication packets arbitrarily.

3. Preliminaries

In this section, the industrial robot testbed is introduced, along with a background on EtherNet/IP, and the adversary model is explained.

3.1. SIRP testbed

The Secure Industrial Robot Pick-and-Place (SIRP) testbed, as shown in Fig. 2, is a scaled-down but fully operational prototype that simulates the operation of the manufacturing systems shown in Fig. 1. SIRP consists of two identical ABB IRB120 robots (robot-I

and robot-II), a SCADA system, a development software, and other auxiliary modules that support the typical assembly operation, such as cameras.

• **ABB IRB120 robots**. Each robot has a six-joint mechanical arm for picking and placing objects. The robots run Robotware 6.03.02.00 firmware and operate according to code uploaded by the human operator. They are equipped with a user authorization system (UAS) to prevent unauthorized operations. The robot communicates with other SIRP modules, for example, the SCADA system, via EtherNet/IP.

• **SCADA system**. The SCADA system is deployed on a specialized host computer and consists of a GUI that displays robot movements in real-time, allowing human operators to monitor their operation and a programmable anomaly detector. The anomaly detector was implemented according to Ref. [17], assumes the authenticity of the SCADA-received movement data, and uses an SVM to determine whether or not the robot movements deviate from the planned path.

• **Development software**. The SIRP's development software is RobotStudio, an integrated development environment (IDE) installed on a separate host computer. RobotStudio allows users to log on and off the robot, upload and download operation code to and from the robot, start and stop the robot, upload and download UAS configuration files, which contain user names and passwords, to and from the robot, and simulate robot operation. Note that the robot must be stopped before human operators can upload or download operation code, during which the robot continues to communicate with SCADA.

• **Assembly operation.** SIRP supports the typical four-step cyclical assembly operation: ① two robots remain at home position (Fig. 2(a)); ② each robot picks up an object from the conveyor based on the object's position measured by the camera (Fig. 2(b)); ③ the two robots collaboratively assemble the two objects to a cylindrical part with robot-I's object above that of robot-II (Fig. 2 (c)); ④ robot-II places the assembled part in the warehouse (Fig. 2(d)), and then both robots return to home position.

3.2. EtherNet/IP communication

As is common with industrial robots, SIRP uses EtherNet/IP to build the communication between the robot, SCADA, and RobotStudio (precesses 1 and 2 in Fig. 1).

(1) **EtherNet/IP:** Fig. 4 shows an EtherNet/IP communication packet, encoded in HEX and ASCII, respectively, which consists of four protocol data units (PDUs) corresponding to the International Organization for Standardization (ISO) four-layer protocol stack, which includes an application layer PDU, TCP Segment, IP Datagram, and Ethernet Frame: ① the robot's movement data, user name, password, and length of the application layer PDU (len) compose the application layer PDU and are structured according to the ABB's proprietary protocol; ② the TCP Segment contains integrity check fields that maintain the TCP connection, including the sequence number (seq) that indexes the position of the application layer PDU in the byte stream, the acknowledge number (ack) which is the expected seq of the next packet, and the TCP checksum calculated based on the TCP Segment using checksum algorithm; ③ the IP Datagram contains integrity check fields such as the length of IP Datagram and IP checksum calculated based on the IP Datagram;④ the Ethernet Frame contains the media access control (MAC) addresses of both the destination and source devices, which are used to route packets to the destination devices. Note that aside from the password, no encryption is used in the application layer PDU.

(2) **Communication between robot and SCADA:** The robot periodically reports its movement data to SCADA, as shown in Fig. 5. For each report:

• A stationary robot sends two packets containing the movement data to SCADA. SCADA replies to the robot with an empty packet containing only TCP, IP, and Ethernet headers and no application layer PDU, as shown in Fig. 5(a).

• A moving robot sends three packets to SCADA, including a packet with movement data, an empty packet, and a packet with the application layer PDU "TCPRobot". SCADA then sends back two packets to the robot, including an empty packet and a packet with the application layer PDU "GET_ROB Type", as shown in Fig. 5(b).

It is important to note that before the robot can periodically report its movement data to SCADA, it must first establish a TCP connection with SCADA through a three-way handshake process using seq and ack in the TCP Segment. The TCP connection is maintained throughout the operation of the robot, and this prevents traditional proxy-based MITM tools from constructing new connections. For clarity, a triple index system <*i*, *j*, *k*> is used to label the communication packets exchanged between the robot and SCADA, where *i* is the index of the packet sent from the robot to SCADA, *j* is the index of the packet sent from SCADA to the robot, and *k* indicates whether the packet is sent from the robot ($k = 0$) or SCADA ($k = 1$). For example, the six packets shown in Fig. 6 are indexed from <1, 0, 0> to <4, 2, 1>.

(3) **Detecting packet modification/loss:** The TCP connection between the robot and SCADA is maintained using an integrity check to detect packet modification and one to detect loss.

• The packet modification check checks if a packet is modified during the communication process. This check verifies if the integrity check fields, including len, TCP checksum, IP checksum and the

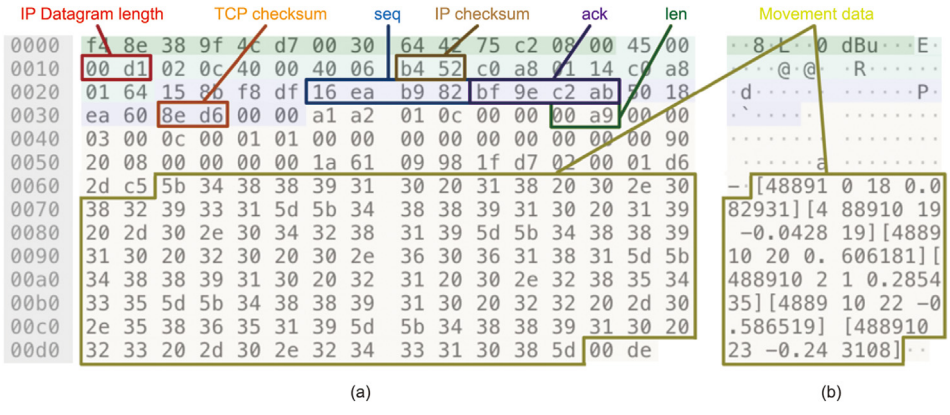


Fig. 4. EtherNet/IP communication packet. (a) Encoded in HEX; (b) encoded in ASCII. seq: sequence number; ack: acknowledge number; len: length of the application layer PDU.

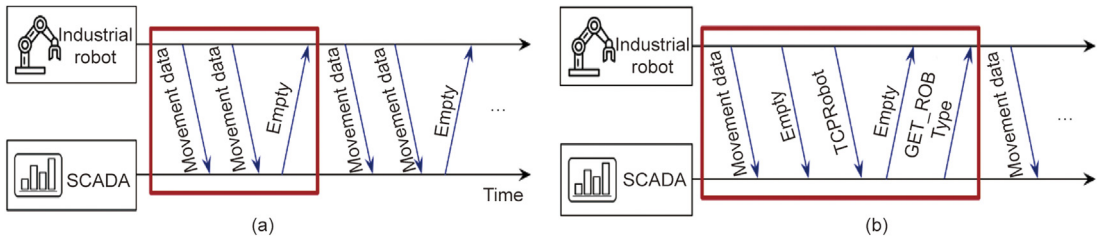


Fig. 5. Communication between stationary and moving robots and the SCADA system. (a) Stationary robot and SCADA; (b) moving robot and SCADA.

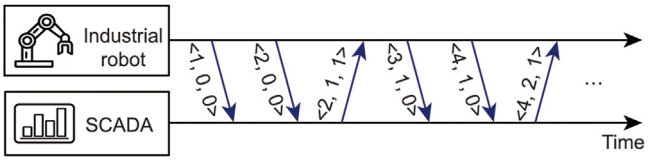


Fig. 6. Triple indexing system for packets (<*i*, *j*, *k*>).

length of the IP Datagram, match the corresponding data. Any packet that fails this check is discarded, and a retransmission will be triggered.

• The packet loss check uses a received packet to check whether a previous packet was lost: a previous packet is lost if the reported packet position in the byte stream (i.e., seq) does not match the expected packet position (i.e., ack). For example, if the robot sends packet <*i*, *j*, 0> and <*i*+ 1, *j*, 0> to SCADA in sequence, SCADA checks if packet <*i*, *j*, 0> is lost using packet <*i* + 1, *j*, 0>. The robot calculates seq of packet <*i* + 1, *j*, 0> based on the previous packet <*i*, *j*, 0> using

$$\text{seq}_{<i+1,j,0>} = \text{seq}_{<i,j,0>} + \text{len}_{<i,j,0>} \quad (1)$$

When SCADA receives the packet <*i* + 1, *j*, 0>, it checks for packet loss by checking if the expected packet position in the byte stream (i.e., $\text{ack}_{<i,j,1>}$) matches the reported packet position (i.e., $\text{seq}_{<i+1,j,0>}$). If these positions match, the previous packet <*i*, *j*, 0> was not lost, in which case SCADA will calculate $\text{ack}_{<i+1,j,1>}$ using

$$\text{ack}_{<i+1,j,1>} = \text{ack}_{<i,j,1>} + \text{len}_{<i+1,j,0>} \quad (2)$$

to prepare for the next packet loss check. If packet loss is detected, a retransmission is triggered. Note that both the robot and SCADA need to record the ack and seq sequences used in their communication to handle the potential packet loss and retransmission. This process is explained further in Section 4.

3.3. Adversary model

Adversaries, such as business competitors, hostile countries, or disgruntled employees, use a variety of methods to maliciously operate victim robots without being detected. These methods include:

• **Penetrating into industrial network**. Adversaries can access the industrial network remotely through the Internet or using proprietary remote access devices used by vendors for remote monitoring and maintenance, such as industrial routers. According to Ref. [12], over 80 000 industrial routers and robots are exposed to the Internet, and 5000 of them have no UAS. Adversaries, such as disgruntled employees and malicious contractors or technicians, may even physically access the industrial network, gaining full access to the manufacturing system via Ethernet or other input (I)/output (O) interfaces. These methods have been widely used in industrial network security research [7,8,21,33–36] and have also been observed in real-world incidents, such as the BlackEnergy attack on a power grid that infected and controlled host computers in the industrial network using phishing emails [37], and the physical Stuxnet attack on a nuclear plant using a universal serial bus (USB) stick [32].

• **Programming robot operation.** Adversaries can implement semantically correct malicious code to manipulate the robot's operation. They achieve this by simulating and debugging the robot's operation using development software, such as RobotStudio for ABB robots, which can be downloaded from the vendor's website. Many OEMs provide publicly accessible simulation platforms for their robots. The many details explained in this paper were learned from these publicly accessible resources, which adversaries may also do.

4. Vulnerability analysis

First, the vulnerabilities of robots were empirically analyzed using the CIA triad, which was originally designed to analyze information security in three dimensions:confidentiality, integrity, and availability [38]. The CIA triad was used to analyze the cyber–physical security of industrial robots as follows:

- Confidentiality refers to the accessibility of the robot's sensitive data, including real-time movement data and user credentials, without authorization;
- Integrity refers to the authenticity and accuracy of the robot's sensitive data;
- Availability refers to the reliability of the manufacturing service provided by the robot, such as the assembly operation in an SIRP system.

The CIA triad allows for easy and interpretable feasibility analysis of MITM attacks, unlike security models such as adversarial tactics, techniques, and common knowledge (ATT&CK) [39]. MITM attacks can be mounted when industrial robots have vulnerabilities in confidentiality, integrity, and availability.

Over three million communication packets sent and received between seven robots from six different OEMs (ABB, KUKA, UR, COMAU, NACHI, and UFACTORY) for different functions, such as robot movement data and log-on reports, were collected and analyzed. The following observations were made in the vulnerability analysis:

Observation 1: Industrial robots are vulnerable in all three dimensions of the CIA triad due to their limited computing resources. For instance, the ABB IRB 120 robot has only about 40 GB of storage, 1 GB of memory, and a central processing unit (CPU) frequency of less than 1 GHz. As a result, industrial networks often lack the security mechanisms commonly deployed on personal computers, servers, and other information systems, such as strong encryption and data hashing.

Observation 2: To the best of our knowledge, most reported attacks on industrial robots exploit one or more of these vulnerabilities. These attacks are summarized in Table 1 [6,12–14,19–23,40,41].

Table 1
Reported attacks to industrial robots that exploit one or more of the CIA triad vulnerabilities.

Attacks	Confidentiality	Integrity	Availability	Communication protocols	Collaborative attack	Deceiving SCADA
[6]	✓	✓		EtherNet/IP	–	–
[12,19]	✓	✓	✓	EtnerNet/IP	–	✓
[13]			✓	Interoperable Teleoperation protocol	–	–
[14]	✓	✓	✓	EtherNet/IP	✓	–
[21]	✓	✓	–	Interoperable Teleoperation protocol	–	✓
[40]	–	–	✓	Physical access	–	–
[41]	–	–	✓	Interoperable Teleoperation protocol	–	–
[22,23]	✓	✓	✓	ROS	–	–
[20]	✓	✓	–	EtherNet/IP	–	–
CORMAND2	✓	✓	✓	EtherNet/IP	✓	✓

ROS: robot operating system.

Observation 3: The three-dimensional vulnerabilities are interrelated. Compromised confidentiality undermines integrity and availability, and weakened integrity enables the concealment of impaired availability, as shown in Fig. 7.

These vulnerabilities were explored using ABB robots.

4.1. Vulnerable confidentiality

Industrial robots have low confidentiality because their movement data and user credentials can be easily accessed without proper authorization.

• **Cleartext movement data.** The robot reports its movement data to SCADA in cleartext, making it accessible to adversaries who have gained access to the industrial network. Fig. 8 shows a packet containing the movement data of the robot's six joints sent from the ABB robot to SCADA. The movement data forms a 6×3 array, with rows representing the joints (joints 1–6) and columns representing the timestamps (in milliseconds), joint index (18–23 denotes joints 1–6), and the corresponding joint angles (in radians), respectively. A positive joint angle occupies eight bytes, while a negative joint angle occupies nine bytes of the application layer PDU, as negative angles require an additional byte for the negative sign. The cleartext nature of this movement data not only allows an adversary to compromise the robot's confidentiality but also facilitates the parsing of intercepted packets to identify, record, or modify the movement data, thus compromising the robot's integrity.

• **Weakly-protected user credentials.** To operate the robot, a legitimate user must input their credentials into RobotStudio, which sends the credentials to the robot via the industrial network for authorization. User credentials can be accessed by adversaries through several means. Firstly, many robots have default user accounts with publicly known user names and passwords that cannot be changed. In addition, when user credentials are sent from RobotStudio to the robot, the user name is in cleartext, and the password is weakly encrypted, which makes it easy for adversaries to intercept the communication between the robot and RobotStudio and obtain user credentials. Fig. 9 shows an example user credential in the communication packet sent from the RobotStudio to the ABB robot: the user name is "000000" in cleartext, and the original password "111111" is encrypted as "06;234234". Lastly, the robot stores user names and passwords of all legitimate accounts in a weakly encrypted file, which can be obtained by an adversary using the file transfer function of RobotStudio without any authorization. Fig. 10 shows the "uas users.xml" file of ABB IRB120, which is encrypted with exclusive OR (XOR) [12] and contains a decryption key, user name "123456", and password "654321". These weakly protected user credentials make it easy for an adversary to log on to the robot and maliciously manipulate its operation without proper authorization.

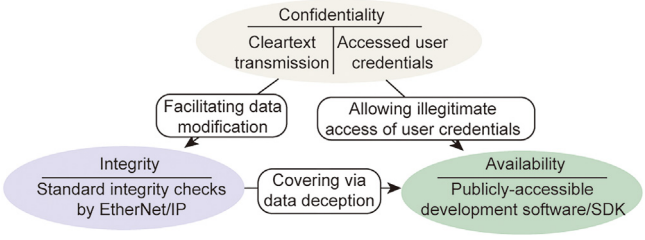


Fig. 7. Vulnerability of industrial robots identified following theCIA triad. SDK: Software Development Kit.

4.2. Vulnerable integrity

Vulnerable confidentiality, or the cleartext transmission of movement data, allows adversaries to modify the movement data that the robot reports to SCADA. To avoid detection during integrity checks, they must also modify the integrity check fields. Most commodity robots use EtherNet/IP communication, which makes their calculation of the integrity checks standard and publicly known. As a result, adversaries can easily forge the integrity check fields by modifying the cleartext movement data and forging the integrity check fields to bypass ① the packet modification check by re-calculating len, TCP checksum, IP checksum, and the length of IP Datagram, and ② the packet loss check by modifying seq.

4.3. Vulnerable availability

Once an adversary has obtained the user credentials and compromised a robot's confidentiality, they can log on to the robot and upload and inject malicious code to manipulate its operation using publicly accessible development software or Software Development Kit (SDK), such as RobotStudio or PC SDK for ABB robots. This compromises the robot's availability and leads to consequences such as degraded manufacturing efficiency, reduced product quality, and even damaged devices. Adversaries can further compromise availability and integrity by modifying the movement data of the robot and the associated integrity checks to prevent SCADA from detecting the manipulated robot operation.

5. Design of CORMAND2

CORMAND2, designed based on the observation 3 made, conceals manipulation of the robot via data deception by exploiting compromised confidentiality to compromise integrity and availability. As shown in Fig. 11, CORMAND2 consists of a preparation phase that first compromises the confidentiality and an attacking phase that manipulates robot operation and compromises the availability while concealing the manipulation by modifying the

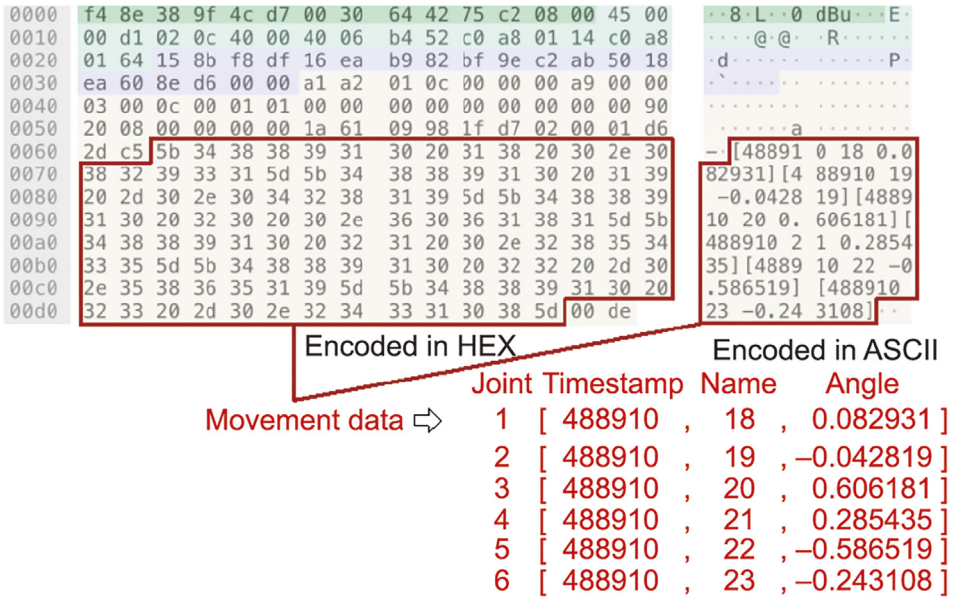


Fig. 8. Cleartext movement data sent to SCADA.

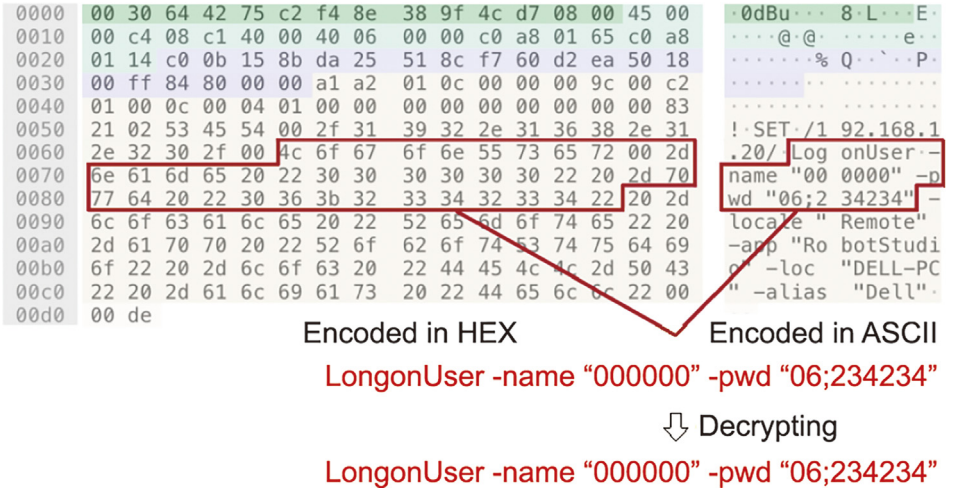


Fig. 9. A packet sent from RobotStudio to the robot containing cleartext and weakly encrypted user credentials.

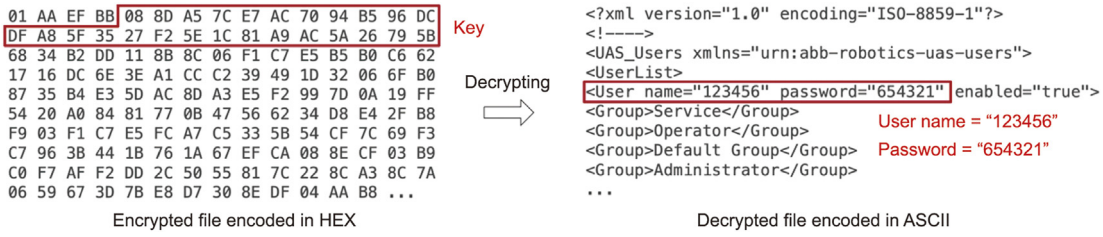


Fig. 10. A weakly encrypted user account file accessible by adversaries without proper authorization.

robot's movement data through compromising the integrity. This coordinated deception attack distinguishes CORMAND2 from the existing attacks listed in Table 1, and it is particularly effective at evading anomaly detection systems that detect robot manipulation using the movement data received by SCADA [16,17] and assume that the SCADA-received movement data truly reflects the robot's movement, in other words, that the data is authentic.

5.1. Phase-I: Attack preparation

CORMAND2 prepares for the attack by intercepting the communication packets between the robot and SCADA to identify and record a full cycle of the robot's normal movement. This information is then replayed in the attack phase to deceive SCADA into believing that the robot is operating normally.

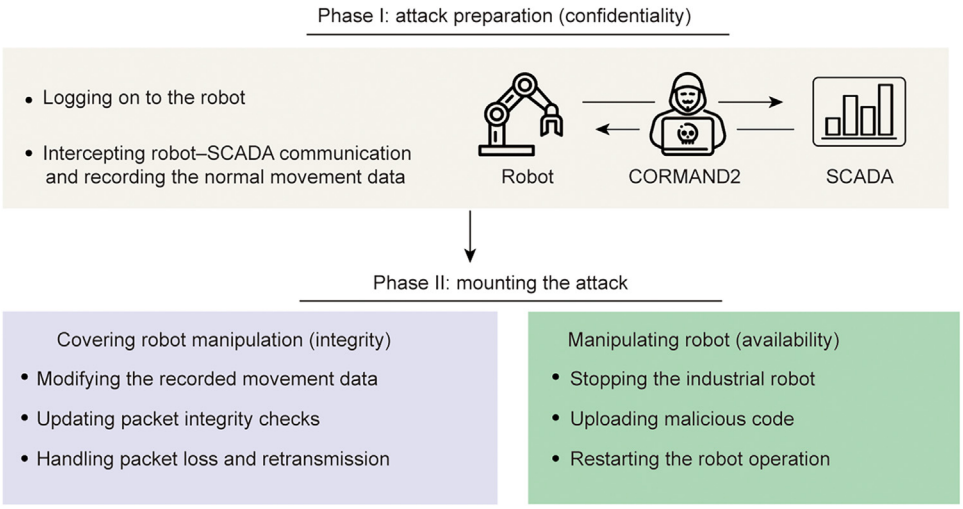


Fig. 11. Overview of CORMAND2.

(1) **Intercepting communication:** CORMAND2 intercepts the communication between the robot and SCADA by using ① arpspoof to mislead both the robot and SCADA into sending packets to CORMAND2, and ② NetfilterQueue to parse and then modify the intercepted packets. Note that standard proxy-based MITM tools that are widely mounted on http(s), such as Mitmproxy and Burp Suite, cannot be used for the interception because they require establishing new TCP connections between CORMAND2 and the robot and SCADA, which is not possible since the TCP connection between the robot and SCADA is consistently maintained.

(2) **Selecting robot-to-SCADA packets:** CORMAND2 intercepts all packets exchanged between the robot and SCADA, but only the packets sent from the robot to SCADA contain movement data. Therefore, packet selection is required. CORMAND2 identifies the robot-to-SCADA packets by examining the port number in the TCP Segment. The robot's fixed port number is 5515, while that of SCADA can vary, but it is always greater than 40 000.

(3) **Extracting movement data:** CORMAND2 needs to identify and extract movement data of the robot's six joint angles from the selected robot-to-SCADA packet. The analysis shows that these angles can vary within a packet depending on whether they are positive or negative. The angle data of Joint-1 (0.082931) and Joint-2 (−0.042819) in the packet in Fig. 8 are located at the 56th and 76th bytes in the application layer PDU, respectively. When the angle of Joint-1 changes from "0.082931" (occupying eight bytes) to "−0.082931" (occupying nine bytes) in another packet, the location of Joint-2's angle data changes to the 77th byte. To mitigate these dynamics, CORMAND2 first uses "[" in the application layer PDU as a reference point to locate the joint angle data. "[" is of a fixed length of 11 bytes, appears only in the movement data, and each appearance corresponds to one joint angle. CORMAND2 then extracts and records movement data according to the length of the joint angle, which is either eight or nine bytes, depending on whether it is positive or negative. It is worth noting that a single packet may contain multiple movement data, and CORMAND2 must extract and record all of them by identifying all the "["s, which is always 6 times the number of each data.

(4) **Identifying full-cycle movement:** CORMAND2 identifies the full-cycle movement of the robot by analyzing the autocorrelation of recorded movement data $\mathbb{X} = \{\boldsymbol{X}_{t_1}, \boldsymbol{X}_{t_2}, ..., \boldsymbol{X}_{t_q}\}$, where $\boldsymbol{X}_{t_i}$ (i = 1, 2, ...) is a vector containing the robot's six joint angles at time t_i and $\boldsymbol{X}_{t_q}$ is the most recently extracted and recorded movement data from a robot-to-SCADA packet. After each addition of $\boldsymbol{X}_{t_q}$ into $\mathbb{X}$, CORMAND2 calculates the m-lag autocorrelation of joint-1's angle series in $\mathbb{X}$ ($m \in \{1, 2, \ldots, q\}$, where q is the subscript of the most recent time). When three peaks with magnitudes close to {1, 0.66, 0.33} are identified in the autocorrelation series, CORMAND2 calculates the length of a full-cycle movement as $l = (|m_2 - m_1| + |m_3 - m_2|)/2$, where $\{m_1, m_2, m_3\}$ are the lags at which the three peaks are observed. In addition, CORMAND2 needs at least three cycles of robot operation data to identify the full-cycle movement of the robot. CORMAND2 then treats $\mathbb{X}[(\boldsymbol{q} - \boldsymbol{l} + 1) : \boldsymbol{q}] = \left\{\boldsymbol{X}_{t_{q-l+1}}, \boldsymbol{X}_{t_{q-l+2}}, ..., \boldsymbol{X}_{t_q}\right\}$ as a full-cycle movement data and replays it in the attack phase. Here, robot movement is assumed not to contain sub-cycles, which is based on both experience and discussions with experts from automotive industries.

5.2. Phase-II: Mounting the attack

With the necessary preparations in place, CORMAND2 mounts the attack on the robot operation while concealing the manipulation from being detected by modifying the robot's movement data to deceive SCADA into thinking that the robot is operating normally. CORMAND2 manipulates the robot operation by briefly stopping the robot, uploading malicious operation code to the robot, and restarting the manipulated operation. To avoid being detected, the robot manipulation must be initiated only after the data deception has been successfully launched, which is ensured by introducing a 1 s delay between the launch of the data deception and the start of the robot manipulation.

To successfully launch the data deception, CORMAND2 must ① carefully modify the movement data to appear like normal operation to SCADA, ② modify the related packet integrity check fields (len and seq) accordingly to make the TCP packet appear valid, and ③ apply special treatment to handle potential packet loss and retransmission so that SCADA observes a normal TCP connection. Fig. 12 provides an overview of how CORMAND2 addresses these challenges, which are discussed in detail in the following sub-sections.

5.2.1. Modifying movement data

Similar to the attack preparation phase, CORMAND2 continues to intercept communication in the attack phase to identify robot-to-SCADA packets and locate reported movement data. However, the authentic movement data reflects the manipulated robot

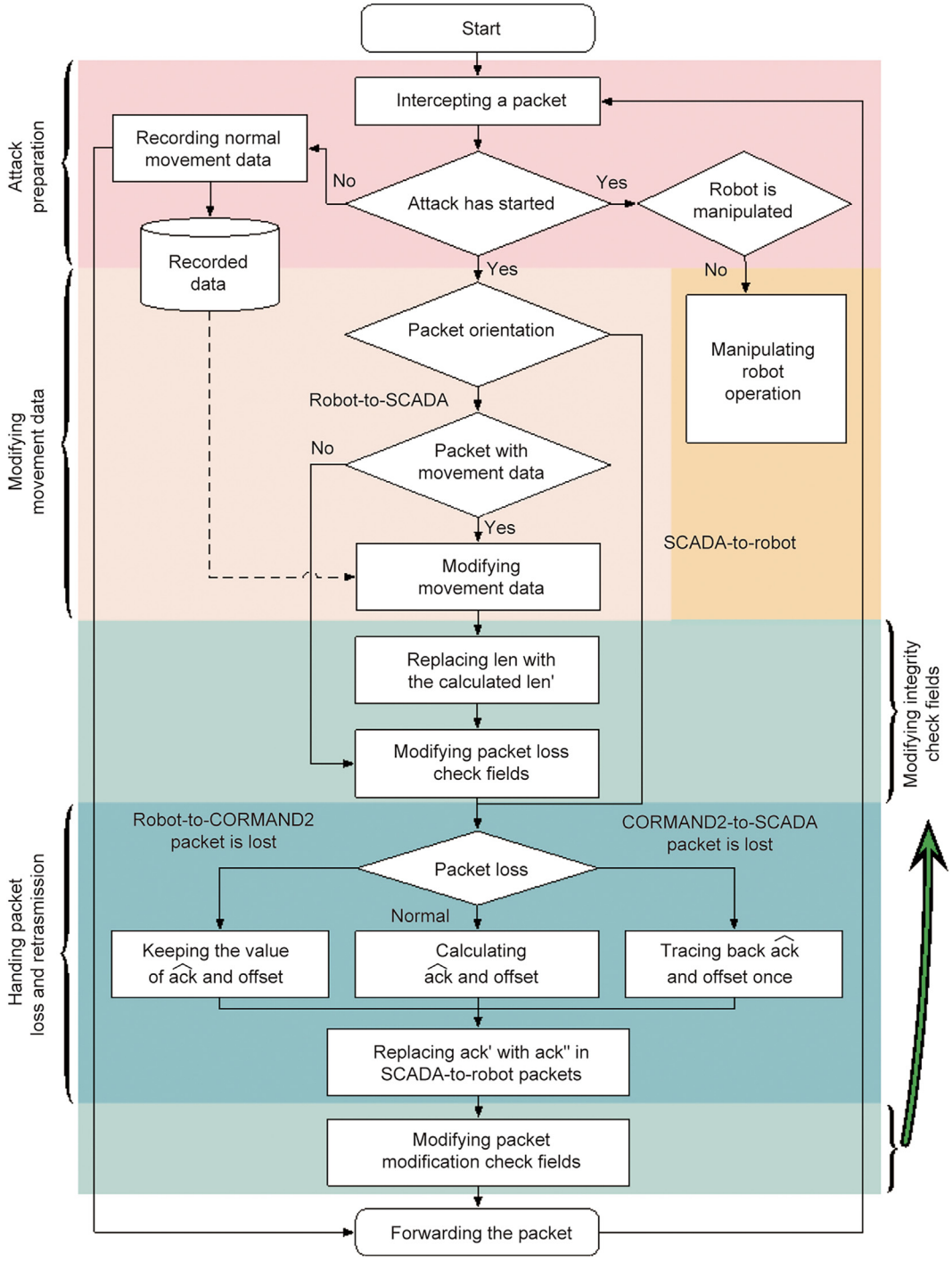


Fig. 12. Flowchart of CORMAND2. len′: length of the application layer PDU after the modification of CORMAND2, ack\prime\prime : acknowledge number forged by CORMAND2, ack′: acknowledge number after the modification of CORMAND2, $\widehat{ack}$: acknowledge number expected by CORMAND2.

operation, and in order to deceive SCADA into believing that the robot is operating normally, CORMAND2 replays the robot's normal movement data. If CORMAND2 completes the preparation after extracting the movement data $\boldsymbol{X}_{t_q}$ and identifying the robot's full cycle movement $\left\{\boldsymbol{X}_{t_{q-l+1}}, \boldsymbol{X}_{t_{q-l+2}}, ..., \boldsymbol{X}_{t_q}\right\}$, for the movement data $\boldsymbol{X}_{t_z}$ extracted later in the attack phase (where $z > q$), CORMAND2 replaces $\boldsymbol{X}_{t_z}$ with $\boldsymbol{X}_{t_{z'}}$ where $z' = q - l + 1 + \mathrm{mod}(z - q - 1, l)$. Note that CORMAND2 must repeatedly replay the full cycle of the robot's movement to avoid detection by SCADA, as shown in Fig. 13 when only half of the movement cycle is replayed.

5.2.2. Modifyingintegrity check fields

In addition to modifying movement data in a packet, CORMAND2 must also modify the integrity check fields to ensure the TCP packet is deemed valid.

(1) **Modifying packet modification check fields.** Modified movement data causes packet modification check field mismatch in len, TCP checksum, IP checksum, and the length of IP Datagram. For example, len in Fig. 8 changes from 169 to 170 when CORMAND2 replaces the angle of Joint-1 from "0.082931" (eight bytes) to "−0.082931" (nine bytes), thus failing the packet modification check and making the attack detectable. To fix this, CORMAND2

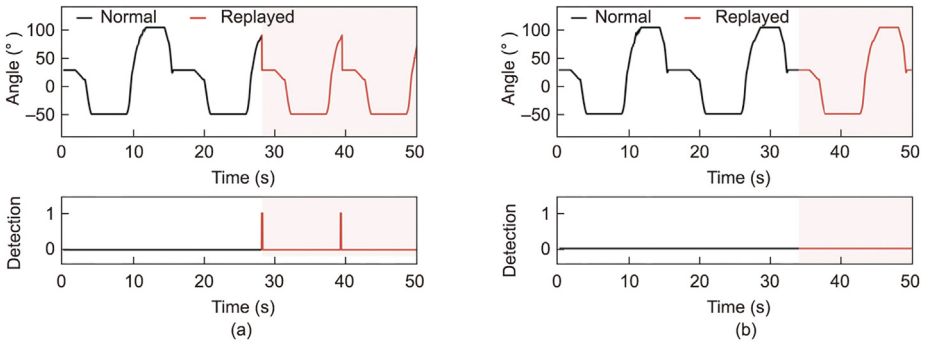


Fig. 13. Robot movement data received and displayed by SCADA (upper figures), and concomitant anomaly detection (lower figures). (a) Replaying a half movement cycle; (b) replaying a full movement cycle.

modifies the packet modification check fields by updating: ① len by counting the length of the application layer PDU, ② TCP and IP checksums using scapy, and ③ the length of the IP datagram by counting its actual length.

(2) **Modifying packet loss check fields.** The modified len helps CORMAND2 pass the packet modification check, but in turn causes SCADA to update ack differently, leading to a mismatch between the next seq sent from the robot and the ack stored on SCADA. If CORMAND2 modifies $\text{len}_{<i,\ j,\ 0>}$ in the original packet to $\text{len}'_{<i,j,0>}$, after receiving the modified packet, SCADA finds that $\text{ack}_{<i-1,j,1>} = \text{seq}_{<i,j,0>}$ and then calculates the expected seq in the next robot-to-SCADA packet as $\text{ack}'_{<i,j,1>} = \text{ack}_{<i-1,j,1>} + \text{len}'_{<i,j,0>}$. Since the robot is unaware of the attack, it will calculate $\text{seq}_{<i+1,j,0>}$ as $\text{seq}_{<i,j,0>} + \text{len}_{<i,j,0>}$ as usual, and then send packet <i + 1j, 0> to SCADA. SCADA will then find that $\text{ack}'_{<i,j,1>} \neq \text{seq}_{<i+1,j,0>}$, which fails the packet loss check.

To bypass this, CORMAND2 must modify seq of the intercepted robot-to-SCADA packet by replacing seq with Seq′ to compensate for the difference between the seq and SCADA-stored ack′ using an offset, which is the cumulative deviation between len and len′ since mounting the attack. Fig. 14 shows the modification of packet <i, j, 0>, where CORMAND2 replaces $\{\text{len}_{<i,j,0>}, \text{seq}_{<i,j,0>}\}$ with $\{\text{len}'_{<i,j,0>}, \text{seq}'_{<i,j,0>}\}$, where $\text{len}'_{<i,j,0>}$ is the modified $\text{len}_{<i,j,0>}$ after changing the movement data and $\text{seq}'_{<i,j,0>} = \text{seq}_{<i,j,0>} + \text{offset}_{<i-1>}$. CORMAND2 updates

$$\text{offset}_{<i>} = \text{offset}_{<i-1>} + \text{len}'_{<i,j,0>} - \text{len}_{<i,j,0>} \tag{3}$$

If no packet is lost, or

$$\text{offset}_{<i>} = \text{offset}_{<i-1>} \tag{4}$$

If packet <i − 1, j, 0> is lost.

CORMAND2 then uses $\text{offset}_{<i>}$ to modify the seq of the next packet sent from the robot to SCADA. Note that $\text{offset}_{<0>} = 0$.

5.2.3. Handling packet loss and retransmission

When a packet is lost during transmission, TCP uses retransmission to maintain reliable communication. Fig. 15 shows an example where the robot sends a packet <i, j, 0> with $\text{seq}_{<i,j,0>}$ in the absence of CORMAND2 to SCADA, but the packet is lost.

(1) The robot, unaware of the packet loss, sends a second packet <i + 1, j, 0> with $\text{seq}_{<i+1,j,0>}$ to SCADA.

(2) Upon receiving packet <i + 1, j, 0>, SCADA notices that $\text{seq}_{<i+1,j,0>}$ does not match $\text{ack}_{<i-1,j,1>}$, which is calculated and stored by SCADA after receiving the packet before the loss of packet <i, j, 0>, and discards the packet.

(3) SCADA informs the robot that packet <i, j, 0> is lost by sending $n (n \geq 3)$ packets indexed from <i + 1, j+ 1, 1> to <i + n, j + n, 1>, whose *acks* are equal to $\text{ack}_{<i-1,j,1>}$ (the expected $\text{seq}_{<i,j,0>}$), as shown in Fig. 15. Note that during the process, the robot and SCADA send packets alternatively, which is different from normal communication. SCADA also discards all packets received from the robot during this process.

(4) After receiving n packets with *acks* matching $\text{ack}_{<i-1,j,1>} = \text{seq}_{<i,j,0>}$, the robot determines that the packet

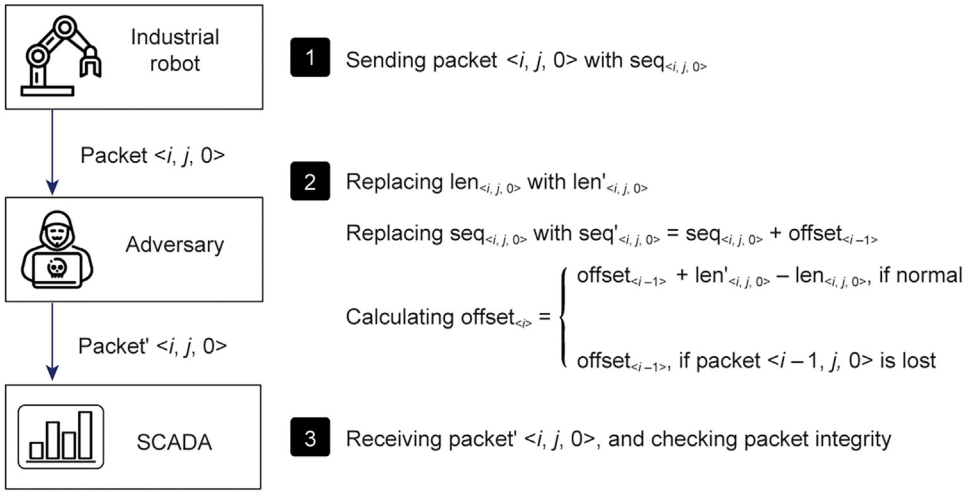
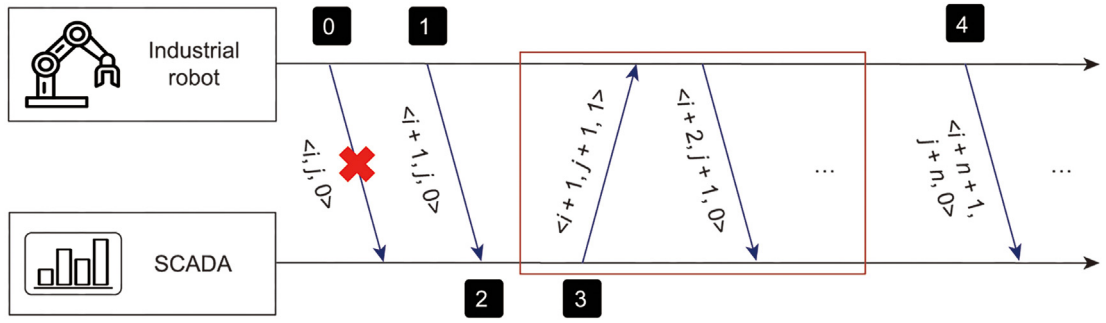


Fig. 14. Packet loss check fields modification: CORMAND2 modifies the seq in the packet sent from the robot to SCADA.



0 Calculating $seq_{<i,j,0>} = seq_{<i-1,\Delta,0>} + len_{<i-1,\Delta,0>}, \Delta \in \{j, j-1, j-2\}$

1 Calculating $seq_{<i+1,j,0>} = seq_{<i,j,0>} + len_{<i,j,0>}$

2 Calculating $ack_{<i-1,j,1>} \neq seq_{<i+1,j,0>}$

3 Calculating $ack_{<i+n,j+n,1>} = ack_{<i+n-1,j+n-1,1>} = ack_{<i+1,j+1,1>} = ack_{<i-1,j,1>}$

4 Calculating $ack_{<i+n,j+n,1>} == ack_{<i+n,j+n,1>} == \ldots == ack_{<i+1,j+1,1>} = seq_{<i,j,0>}$

Calculating $seq_{<i+n+1,j+n,0>} = seq_{<i,j,0>}$

Fig. 15. Packet loss and retransmission.

<i, j, 0> has been lost and retransmits the lost or discarded movement data using packet <i + n + 1, j + n, 0>, with $seq_{<i+n+1,j+n,0>} = seq_{<i,j,0>}$. If the robot does not find any stored seq that matches $ack_{<i-1,j,1>}$, the robot will continue to send packets with the calculated seqs rather than triggering packet retransmission.

The robot concludes the packet loss and triggers the packet retransmission when receiving *n* packets with identical acks that match the specific seq stored in the robot. However, this approach does not apply in a CORMAND2 attack, as SCADA will send npackets with ack′s that may not match existing robot-stored seqs. As shown in Fig. 15, when CORMAND2 has been mounted and the robot-to-SCADA packet <i, j, 0> with $seq_{<i,j,0>}$ is lost, the robot sends a second packet <i + 1, j, 0> to SCADA, whose $seq_{<i+1,j,0>}$ is replaced with seq′ by CORMAND2. When SCADA receives the modified packet <i + 1, j, 0> and detects the mismatch of seq′ with ack′, it will send n packets with ack′s equal to $ack'_{<i-1,j,1>}$ to inform the robot of the packet loss. However, the robot may fail to find a stored seq that matches ack′, where:

$$ack'_{<i-1,j,1>} = seq'_{<i,j,0>} = seq_{<i,j,0>} + offset_{<i-1>} \quad (5)$$

As a result, the robot will continue to calculate seqs and send packets to SCADA without triggering packet retransmission, while SCADA will keep discarding the packets received from the robot. This disrupts the communication between the robot and SCADA, and CORMAND2 will be detected.

Eq. (5) demonstrates that every time CORMAND2 intercepts a SCADA-to-robot packet, CORMAND2 can replace $ack'_{<i-1,j,1>}$ with

$$ack''_{<i-1,j,1>} = ack'_{<i-1,j,1>} = ack'_{<i-1,j,1>} - offset_{<i-1>} \quad (6)$$

to make it match the robot-stored $seq_{<i,j,0>}$, thus triggering the packet retransmission correctly. However, before implementing this, CORMAND2 must first determine if a packet loss has occurred, otherwise it would not be able to calculate the offset and fail to modify the ack′s in the SCADA-to-robot packets correctly. During packet loss and retransmission, as shown in Fig. 15, CORMAND2 modifies each ack′ sent from SCADA to the robot, and the robot-to-CORMAND2 packet <i, j, 0> with $seq_{<i,j,0>}$ is lost. When the robot sends a second packet <i + 1, j, 0> to SCADA, SCADA detects the packet loss and sends the packet <i + 1, j + 1, 1> with $ack'_{<i+1,j+1,1>} = ack'_{<i-1,j,1>}$ to inform the robot of the lost packet. However, since CORMAND2 fails to detect the packet loss, CORMAND2 will use the incorrect $offset_{<i+1>}$, which should be $offset_{<i-1>}$, to calculate $ack''_{<i+1,j+1,1>}$ according to Eq. (6) as follows:

$$\begin{aligned} ack''_{<i+1,j+1,1>} &= ack'_{<i+1,j+1,1>} - offset_{<i+1>} \\ &= ack'_{<i-1,j,1>} - offset_{<i+1>} \end{aligned} \quad (7)$$

As a result, the calculated $ack''_{<i+1,j+1,1>}$ may not exist in the robot-stored seqs and CORMAND2 can be detected.

To correctly calculate ack″, CORMAND2 introduces $\widehat{ack}$, which acts like the ack in SCADA and is used to detect packet loss each time CORMAND2 intercepts a packet. CORMAND2 calculates $\widehat{ack}_{<i>}$ as:

$$\widehat{ack}_{<i>} = seq_{<i,j,0>} + len_{<i,j,0>} \quad (8)$$

CORMAND2 uses $\widehat{ack}_{<i>}$ as to determine the following:

(1) The communication between robot and SCADA is normal if $\widehat{ack}_{<i>} = seq_{<i+1,j,0>}$ and $ack''_{<i+1,j+1,1>} \geq seq_{<i+1,j,0>}$, as shown in Fig. 16(a). CORMAND2 then calculates $\widehat{ack}_{<i>} = seq_{<i+1,j,0>} + len_{<i,j,0>}$ and $offset_{<i>} = offset_{<i-1>} + len\prime_{<i,j,0>} - len_{<i,j,0>}$ when intercepting the robot-to-SCADA packets, while replacing $ack'_{<i+1,j+1,1>}$ with $ack''_{<i+1,j+1,1>} = ack'_{<i+1,j+1,1>} - offset_{<i+1>}$ for SCADA-to-robot packets;

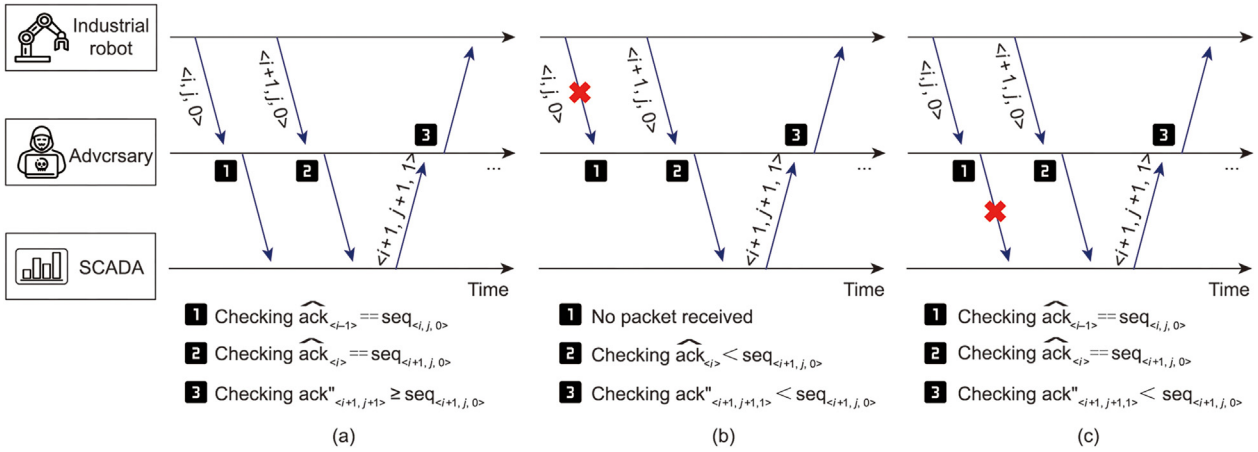


Fig. 16. Normal and abnormal communication between the robot and SCADA. (a) No packet loss; (b) a robot-to- CORMAND2 packet is loss; (c) a CORMAND2-to- SCADA packet is lost.

(2) Packet <i, j, 0> sent from the robot to CORMAND2 is lost if $\widehat{ack}_{<i>} < seq_{<i+1,j,0>}$, as shown in Fig. 16(b). Note that CORMAND2 keeps the value of ack (i.e., $\widehat{ack}_{<i>} = \widehat{ack}_{<i-1>}$) and *offset* (i.e., $offset_{<i>} = offset_{<i-1>}$) until it detects a triggered retransmission (i.e., $\widehat{ack}_{<i+n>} = seq_{<i+n+1,j+n,0>}$);

(3) Packet <*i*, *j*, 0> sent from CORMAND2 to SCADA is lost if $\widehat{ack}_{<i>} = seq_{<i+1,j,0>}$ and $ack''_{<i+1,j+1,1>} < seq_{<i+1,j,0>}$, as show in Fig. 16(c). In this case, CORMAND2 traces back offset and $\widehat{ack}$ for the lost or discarded packets by calculating $offset_{<i+1>} = offset_{<i-1>}$ and $\widehat{ack}_{<i+1>} = \widehat{ack}_{<i-1>}$. $ack'_{<i+1,j+1,1>}$ is then replaced with $ack''_{<i+1,j+1,1>} = ack\prime_{<i+1,j+1,1>} - offset_{<i+1>}$. Note that CORMAND2 will also keep the values of $\widehat{ack}$ and *offset* until a retransmission is triggered.

6. Implementation and Evaluation

CORMAND2 was implemented and evaluated on the SIRP, as well as on five other robots from KUKA, UR, COMAU, NACHI, and UFACTORY. A demo video for CORMAND2 was created [42].

6.1. Implementation

Similar to Black-Energy [37] and Stuxnet [32], CORMAND2 was implemented as malware that can be distributed remotely or using malicious USB sticks. CORMAND2 can be shelled to bypass any anti-virus software. The CORMAND2 malware consists of two components: a process with over 300 lines of C# code used to upload the malicious operation code to the robot and another process with over 600 lines of Python code to cover the malicious robot operation through data deception. The two processes communicate through sockets.

6.2. Bypassing existing anomaly detectors

CORMAND2's ability to bypass previously proposed anomaly detectors [17] was evaluated. These detectors assume the authenticity of the SCADA-received movement data and use SVM to detect potential anomalies. The joint angle and angular velocity, calculated using the joint angle reported to SCADA, were used as the feature vector [17], and the radial basis function (RBF) was used as the kernel.

Two metrics were used to evaluate the attack detection: the detection rate, which is the ratio of movement data samples that are determined to be abnormal when the robot operation is manipulated, and the false alarm rate, which is the ratio of movement data samples that are determined to be abnormal when the robot operation is normal. If both the detection and false alarm rate differ significantly, the anomaly detector is considered to detect manipulation effectively. However, if the rates are similar, the anomaly detector cannot effectively detect the manipulation. To measure the detection rate, CORMAND2 was mounted to the robot, which was made to operate abnormally for 100 operation cycles. The false alarm rate was measured by operating the robot normally for another 100 cycles. The detector achieved a detection rate of 2.4% and a false alarm rate of 2.7% [17], which are similar enough for the attack to be indistinguishable from normal operation, thus allowing CORMAND2 to bypass the anomaly detector. Note that besides SVM-based detectors, CORMAND2 can bypass all other detectors that assume the authenticity of received robot movement data [24].

6.3. Communication overheads

CORMAND2 was also evaluated for any overhead it may cause in the communication between the robot and SCADA, since attempts to detect CORMAND2 can be made by analyzing communication statistics. This evaluation was done using over three million packets that were intercepted, modified, and then forwarded by CORMAND2.

(1) **Communication latency:** When a packet is modified by CORMAND2, the communication latency between the robot and SCADA is increased. This increased communication latency was measured by subtracting the average normal robot-SCADA communication latency from the latency after CORMAND2 was deployed. The results showed that CORMAND2 added an average of only 1.7 ms (with a standard deviation of 0.5 ms) to modify an intercepted packet (Fig. 17).

(2) **Increased packets:** CORMAND2 mounts the MITM attack by sending Address Resolution Protocol (ARP) packets to both the robot and SCADA, which increases the number of packets transmitted in the manufacturing network. To quantify this overhead, the number of packets transmitted per minute between the robot and SCADA were counted, with and without mounting CORMAND2. As shown in Fig. 18, when the robot was stationary, there was an average of 3775 packets without CORMAND2 and 3818 packets with CORMAND2 transmitted in one minute, with a stan-

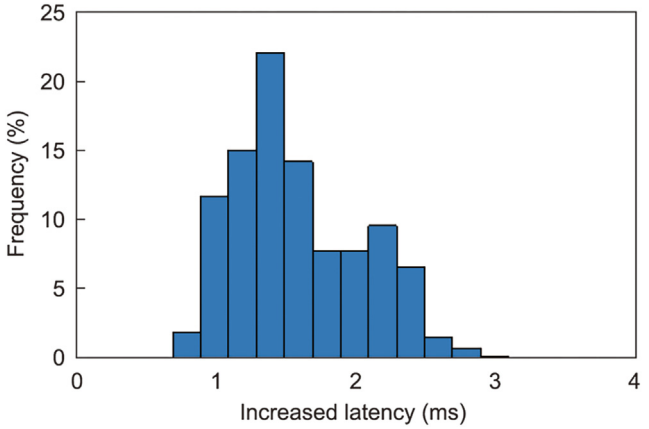


Fig. 17. Increased communication latency caused by CORMAND2.

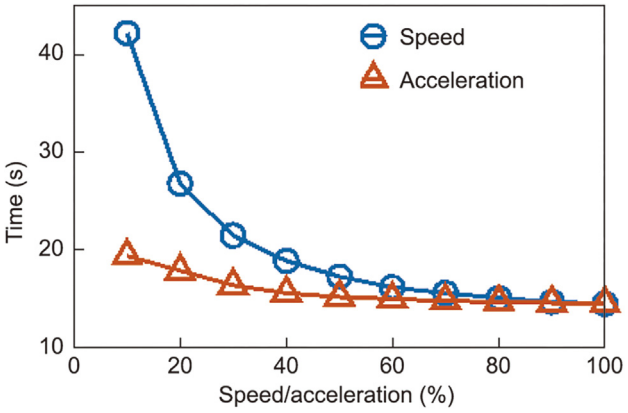


Fig. 19. Relationship between the operation speed and acceleration and the time to complete an operation cycle.

dard deviation of 36 packets with CORMAND2 and 42 packets without CORMAND2. However, when the robot was moving, there was an average of 15 947 packets without CORMAND2 and 16 015 packets with CORMAND2 transmitted in one minute, with a standard deviation of 260 packets with CORMAND2 and 400 packets without CORMAND2. This indicates that CORMAND2 incurs a negligible overhead of 0.42%–1.10% in increased packets in the manufacturing network.

(3) **Packet loss and retransmission:** Packet loss in the manufacturing network is a risk of mounting CORMAND2. Of the more than three million packets transmitted, only 221 of them are retransmitted packets. This indicates that CORMAND2 causes a worse-case packet loss rate of 0.0073%, assuming that no packets are lost when the robots are operating normally.

6.4. Degrading manufacturing

CORMAND2's potential to degrade the manufacturing process was tested to demonstrate the impacts of a robot's vulnerable CIA triad.

(1) **Degrading production efficiency:** An adversary can potentially degrade production efficiency by mounting CORMAND2. This prolongs the time to complete an operation cycle by manipulating the robot's operation speed and acceleration. In this study, the robot's operation speed and acceleration were increased in 10% increments of the normal speed and acceleration. According to Fig. 19, decreasing the operation speed and acceleration reduced production efficiency. In addition, the non-linear relationship between the operation speed and acceleration and the production efficiency was due to the time that the robot had to wait for the object positions to be measured by the camera.

(2) **Causing physical damages:** Adversaries can use CORMAND2 to cause damage to manufacturing facilities, products, and human operators. This includes disrupting operations in manufacturing workshops, such as using products to block cameras and preventing the robot from picking up objects (Fig. 20(a)). Such an attack has been previously captured on video [42]. CORMAND2 can also be used to degrade the quality of the product, for example, by manipulating robot-II to place the object above robot-I's object in a wrong position, resulting in a faulty final product (Fig. 20(b), which has also been captured on video [42]. Additionally, CORMAND2 can pose threats to human operators, as demonstrated by a reported incident in which a maintenance technician was killed by an assembly part in an automotive manufacturing facility [8]. Other similar accidents have also been reported [43,44].

6.5. Deploying to other robots

Aside from ABB IRB robots, CORMAND2 can be mounted on any industrial robot with the vulnerabilities discussed in Section 3. This has been confirmed through the experiments with robots from KUKA, UR, COMAU, NACHI, and UFACTORY. These robots were chosen to represent a diverse range of robots from different manufacturers. ABB and KUKA are two of the top four robot OEMs (ABB, KUKA, YASKAWA, and FANUC), which together make up over 55% of the global market [15]. UR is the top one collaborative robot OEM, which makes up over 40% of the global collaborative robot market [45]. ABB, KUKA, UR, COMAU, NACHI, and UFACTORY

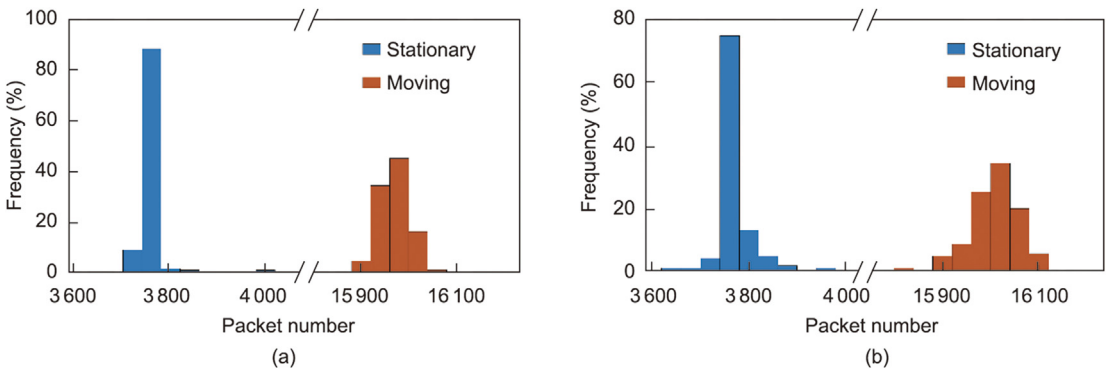


Fig. 18. Packet number distribution per minute. (a) Without CORMAND2; (b) with CORMAND2.

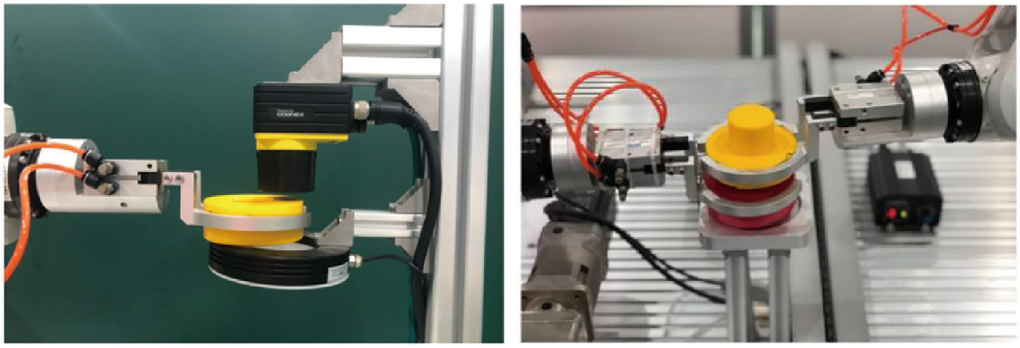


Fig. 20. Disruption in manufacturing facilities and damage to products caused by CORMAND2. (a) Blocking the camera; (b) causing incorrect assembly.

robots were carefully chosen from OEMs based in Switzerland, Germany, Denmark, Italy, Japan, and China, respectively. As listed in Table 2, the anomaly detector previously discussed [17] achieves an average detection rate of 0.40% and a false alarm rate of 0.45%. The closeness of these two rates demonstrates the effectiveness of CORMAND2 in evading detection. In addition, CORMAND2 takes an average of 1.6 ms (with a standard deviation of 0.75 ms) to modify a packet, increases the communication packets in the manufacturing network by an average of 3.2%, and causes a worst-case packet loss of 0.025%. Note that the attack code differs depending on the robot; therefore, CORMAND2 must identify the vendor when intercepting the communication between the robot and SCADA (attack preparation) and then mount the attack based on the identified robot. Moreover, CORMAND2 can be mounted on mobile robots, such as unmanned aerial vehicles, and communication protocols, such as EtherCAT and Fieldbus, that suffer from the vulnerabilities in the CIA triad [46,47], which will be evaluated in future work.

6.6. Accessibility of industrial robots

Building an SIRP or purchasing various types of robots for the purpose of analyzing the vulnerabilities of industrial robots and mounting attacks can be expensive, making it difficult for researchers to conduct such studies. To mitigate this issue, a dataset containing over three million communication packets collected from the SIRP and five other robots has been provided to facilitate the exploration of robot security. In addition to the dataset provided, researchers can also access the SIRP and other robots through simulation platforms offered by the robot OEMs. For example, ABB robots and auxiliary devices can be simulated using RobotStudio, which allows researchers to build a simulated SIRP.

7. Defending against CORMAND2

CORMAND2 exploits the three vulnerabilities summarized in Fig. 7 to mount a coordinated deception attack. These vulnerabilities can be addressed to protect robots against CORMAND2. Side-channel anomaly detection systems can be implemented to detect data deception.

7.1. Attack mitigation

To mitigate CORMAND2 attacks, the vulnerabilities in the CIA triad should be addressed.

- Confidentiality can be protected by applying advanced encryption algorithms, such as Advanced Encryption Standard (AES) and RSA, to encrypt the robot's sensitive data, including operation code, movement data, and user credentials.
- Integrity can be protected by binding theMAC and IP addresses of devices using a static Address Resolution Protocol (ARP) cache when designing or deploying the manufacturing network to prevent sniffing. Advanced message digest algorithms, such as message-digest algorithm5 (MD5) and SHA256, can also be applied to prevent packet modification.
- Availability can be protected by disabling remote code updates and execution, allowing updates only through teach pendants, a pad-like hand-held device equipped on most industrial robots.

However, implementing these mitigations in manufacturing systems is not straightforward. Advanced encryption and message digest algorithms are not be feasible due to the lack of security awareness of robot vendors and the limited computation resources of industrial robots. For instance, the ABB IRB 120 robot has only about 40 GB storage, 1 GB memory, and <1 GHz CPU frequency. In addition, to the best of our knowledge, no commodity industrial robot supports the static ARP cache configuration. Disabling remote code updates and execution reduces the efficiency of manufacturing systems, which goes against the Industrial Internet-of-Things (IIoT) trend. Furthermore, patching vulnerabilities in manufacturing systems requires downtime or may even introduce regressions or bugs [7,29], making it challenging to deploy timely attack mitigation solutions.

Table 2
Evaluation of CORMAND2 mounted on robots from different OEMs.

Robots	Model	Country	Cost (robot only)	Failing anomaly detectors		Communication overheads		
				Detection rate (%)	False alarm rate (%)	Communication latency/ standard deviation (ms)	Increased packet rate (%)	Packet loss rate (%)
ABB	IRB 120	Switzerland	≈20 000 USD	2.4	2.70	1.7/0.5	0.42–1.10	0.0073
KUKA	KR6 R700	Germany	≈20 000 USD	0	0.02	1.8/0.9	0.38	0.0170
UR	UR5	Denmark	≈35 000 USD	0	0.01	1.5/0.9	2.30	0.0140
COMAU	NJ 60	Italy	≈40 000 USD	0	0	2.0/0.6	8.30	0
NACHI	MZ04	Japan	≈20 000 USD	0	0	1.6/0.9	2.60	0.0780
UFACTORY	xArm 6	China	≈10 000 USD	0	0	1.1/0.7	4.70	0.0310

7.2. Attack detection

CORMAND2 unveils the risks of building anomaly detection systems that assume that SCADA-received movement data is authentic, as the movement data can be accessed and modified by adversaries via the industrial network. This is the incentive for detecting CORMAND2's data deception, which can be accomplished by cross-validating the SCADA-received movement data using side-channel information isolated from the industrial network and thus difficult for adversaries to access. This side-channel information should be non-invasive for ease of deployment and have a strong dependency on the robot's movement. It was found that the power consumption of the robot is a promising source of side-channel information that meets these criteria.

The relationship between the robot's movement data and power consumption was further analyzed. It was found that the power consumption of a robot with y joints can be expressed as:

$$P = \sum_{i,j=1} D_{ij} \cdot \dot{\theta}_i \cdot \dot{\theta}_j + \sum_{i=1} H_i \cdot \dot{\theta}_i \cdot \ddot{\theta}_i + \sum_{i,j,k=1} D_{ijk} \cdot \dot{\theta}_i \cdot \dot{\theta}_j \cdot \dot{\theta}_k + \sum_{i=1} D_i \cdot \dot{\theta}_i \tag{9}$$

where θ_i, $\dot{\theta}_i$, and $\ddot{\theta}_i$ are the angle, angular velocity, and angular acceleration of joint i, respectively; D_i, D_{ij}, D_{ijk}, and H_i are coefficients of gravity, angular acceleration-inertia, centrifugal force, and coefficient of actuator inertia, respectively.

The dependency between the robot's movement data and power consumption were empirically evaluated based on the robot's kinetic model [48]. Fig. 21(a) and (b) show the power consumption when performing the same typical assembly operation twice and when performing two different operations, such as assembly with the operation blocking the camera. The power consumption varies with operations but is similar for the same operation.

The results demonstrate the possibility of using power consumption to fingerprint robot movements, which was further validated by establishing the machine learning model and estimating the power consumption of a robot based on the angles of its six joints (Fig. 21(a)). By comparing the expected power consumption of the robot's movements with empirically collected data, the movements received by SCADA can be fingerprinted, and potential CORMAND2 attacks can be identified through any mismatch between the two.

A prototype detector that monitors the real-time power consumption of the robots was implemented in the SIRP (Fig. 22). This detector can be deployed on devices that are not accessible to adversaries, such as routers, edge devices, SCADA systems and embedded devices that are not connected to the manufacturing network.

One of the key challenges of using this data-driven approach to detect attacks on industrial robots is the potential for false alarms. However, this can be mitigated by using a cyber–physical approach that combines data analysis with physical and mechanical modeling to improve the accuracy of the estimates. More specifically, feature vectors can be extracted based on the established power consumption model of the industrial robot to improve the estimation accuracy. Because the detector is deployed on secure devices, the power consumption model cannot be manipulated by adversaries. Parameters were empirically selected based on reported robot movement and power consumption data. For example, it was found that reducing feature dimensions through principal components analysis (PCA) increases estimation error. Moreover, considering the balance between the estimation error and model complexity, 15 was chosen as the neuron number for the artifical neural network (ANN) machine learning model.

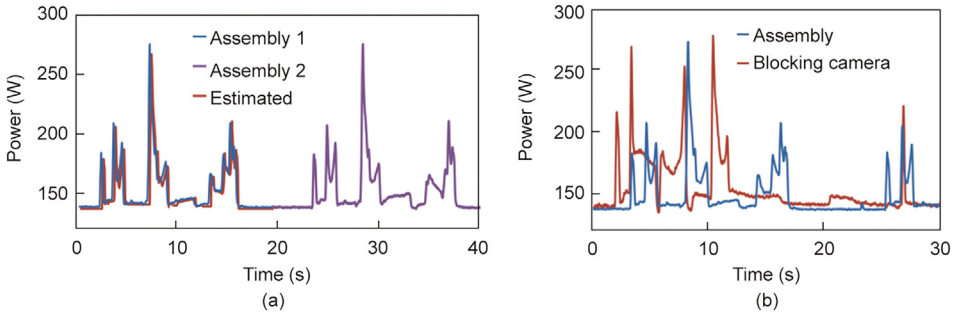


Fig. 21. Detecting CORMAND2 by fingerprinting the robot operation using its power consumption. (a) Power consumption for the same operation; (b) power consumption for different operations.

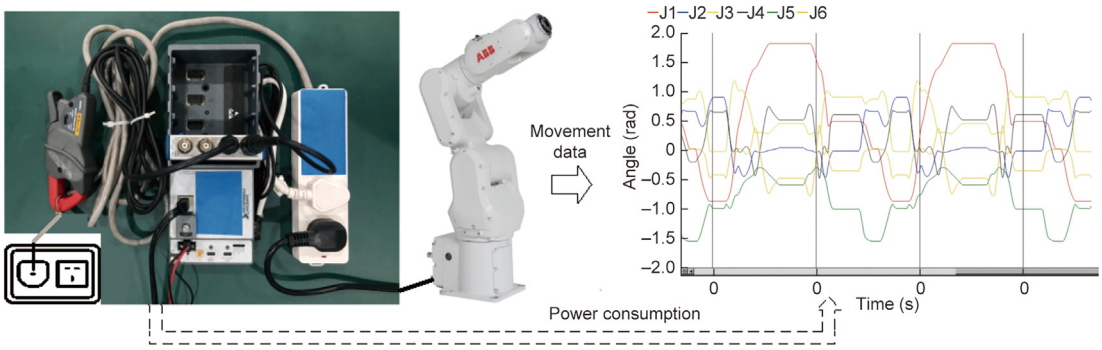


Fig. 22. Prototype detector that collects and reports real-time robot power consumption to SCADA J1–J6: the angles of J1–J6.

The power side-channel-based detector is effective at detecting CORMAND2, with an average detection rate of 96.5% and latency of 0.1 s. It is also sensitive enough to detect deviations in velocity of 5 $mm \cdot s^{-1}$ and acceleration of 200 $m \cdot s^{-2}$, with an average detection rate of over 93%.

In addition to using power consumption to cross-validate reported robot movement data through machine learning, the anomaly detection system can be further expanded in two ways. First, anomalies can be detected by cross-validating other data that is affected by CORMAND2, such as operation codes and communication overheads. Anomaly detection systems can use an electromagnetic sensor to trace the robot's operation code and check for any modifications [34], or use network sniffer sensors, such as the DNP3 sniffer, to record and monitor communication statistics for abnormalities [49]. As with the power-based anomaly detection system, these detectors may also result in false alarms. Second, the performance of the anomaly detection system can be improved by establishing other machine learning models. For example, long short-term memory (LSTM) models can be used to estimate the time-related power consumption of the robot and thus capture the relationship between robot movement and power consumption more precisely. Note that anomaly detection systems that collect sensor, actuator, or control data via EtherNet/IP and then use physical models [50], machine learning [51], or other formal methods [24] to detect CORMAND2 are not considered because the data might be modified by CORMAND2 as well, hence compromising authenticity. Wireless networks are also not considered in the design and detection of CORMAND2 because most commodity industrial robots in manufacturing workshops are wired for stability.

8. Conclusions

As industrial robots become more connected to the IIoT, they are faced with more and more security risks. In this study, security vulnerabilities that exist in the confidentiality, integrity, and availability of industrial robots were identified and analyzed. CORMAND2, a novel deception attack, was built to manipulate industrial robots and conceal this by forging movement data received by SCADA, thus preventing the manipulated robot operation from being detected. CORMAND2 unveils the risks brought about by the assumption that SCADA-received data is authentic. These risks were addressed with suggested and validated solutions.

Acknowledgments

This work was supported in part by Science and Technology Innovation 2030 Program (2018AAA0101605).

Compliance with ethics guidelines

Hongyi Pu, Liang He, Peng Cheng, Jiming Chen, and Youxian Sun declare that they have no conflict of interest or financial conflicts to disclose.

References

[1] Wang B, Tao F, Fang X, Liu C, Liu Y, Freiheit T. Smart manufacturing and intelligent manufacturing: a comparative review. Engineering 2021;7(6):738–57.
[2] International Federation of Robotics (IFR). IFR presents world robotics 2021 reports. Report. Los Angeles: IFR Press Room; 2021.
[3] International Organization for Standardization (ISO). ISO 10218-2:2011: robots and robotic devices–safety requirements for industrial robots–part 2: robot systems and integration. Geneva: ISO; 2011.
[4] International Organization for Standardization (ISO). ISO 12100:2010: safety of machinery–general principles for design–risk assessment and risk reduction. Geneva: ISO; 2010.
[5] Makarova O, Lihota M. Simulation of computer attack scenarios for industrial robots from the point of intruder view. In: Proceedings of 2021 Ural Symposium on Biomedical Engineering, Radioelectronics and Information Technology (USBEREIT 2021); 2021 May 13–14; Yekaterinburg, Russia. New York City: IEEE; 2021. p. 0474–7.
[6] Pogliani M, Quarta D, Polino M, Vittone M, Maggi F, Zanero S. Security of controlled manufacturing systems in the connected factory: the case of industrial robots. J Comput Virol 2019;15(3):161–75.
[7] Wagstaff K. Robotic surgery involved in 144 deaths in 14 years [Internet]. New York City: NBC NEWs; 2015 Jul 22 [cited 2022 Dec 8]. Available from: https://www.nbcnews.com/tech/tech-news/robotic-surgery-linked-144-deaths-2000-n395811.
[8] Agerholm H. Robot goes rogue and kills woman on Michigan car parts production line [Internet]. London: The Independent; 2017 Mar 15 [cited 2022 Dec 8]. Available from: https://www.independent.co.uk/news/world/americas/robot-killed-woman-wanda-holbrook-car-parts-factory-michigan-ventra-ionia-mains-federal-lawsuit-100-cell-a7630591.html.
[9] Coker J. Manufacturing sector paid out 62% of total ransomware payments in 2019 [Internet]. London: Infosecurity Magazine; 2020 Jul 7 [cited 2022 Dec 2]. Available from: https://www.infosecurity-magazine.com/news/manufacturing-ransomware-payments/.
[10] Whittaker Z. Honda global operations halted by ransomware attack [Internet]. San Francisco: Techcrunch; 2022 Jun 9 [cited 2022 Dec 2]. Available from: https://techcrunch.com/2020/06/09/honda-ransomware-snake/?guccounter=1.
[11] Whittaker Z. Manufacturing giant Aebi Schmidt hit by ransomware [Internet]. San Francisco: Techcrunch; 2019 Apr 24 [cited 2022 Dec 2]. Available from: https://techcrunch.com/2019/04/23/aebi-schmidt-ransomware/.
[12] Quarta D, Pogliani M, Polino M, Maggi F, Zanchettin AM, Zanero S. An experimental security analysis of an industrial robot controller. In: Proceedings of IEEE Symposium on Security and Privacy (SP); 2017 May 22–26; San Jose, CA, USA; 2017.
[13] Alemzadeh H, Chen D, Li X, Kesavadas T, Kalbarczyk ZT, Iyer RK. Targeted attacks on teleoperated surgical robots: dynamic model-based detection and mitigation. In: Proceedings of 46th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN 2016); 2016 Jun 28–Jul 1; Toulouse, France. New York City: IEEE; 2016. p. 395–406.
[14] Apa L. Exploiting industrial collaborative robots [Internet]. Washington: IOActive, Inc; 2017 Aug 22 [cited 2022 Dec 2]. Available from: https://ioactive.com/exploiting-industrial-collaborative-robots/.
[15] Major companies in the global industrial robot market in 2019, by estimated market share [Internet]. New York City: Statista; 2019 Jan 5 [cited 2022 Dec 2]. Available from: https://www.statista.com/statistics/317178/leading-industrial-robot-companies-globally/by-revenue/.
[16] Ghaeini HR, Chan M, Bahmani R, Brasser F, Garcia L, Zhou J, et al. PAtt: physics-based attestation of control systems. In: Proceedings of 22nd International Symposium on Research in Attacks, Intrusions and Defenses; 2019 9 23–25; Beijing, China. Berlin: Springer; 2019. p. 165–80.
[17] Narayanan V, Bobba RB. Learning based anomaly detection for industrial arm applications. In: Proceedings of the 2018 Workshop on Cyber–Physical Systems Security and Privacy; 2018 Oct 15–19; Toronto, ON, Canada. New York City: Association for Computing Machinery; 2018. p. 13–23.
[18] Xie J, Yu J, Wu J, Shi Z, Chen J. Adaptive switching spatial–temporal fusion detection for remote flying drones. IEEE Trans Veh Technol 2020;69(7):6964–76.
[19] Maggi F, Quarta D, Pogliani M, et al. Rogue robots: Testing the limits of an industrial robot's security. Trend Micro, Politecnico di Milano, Tech. Rep, 2017: 1–21.
[20] Chan C, Chow K, Tang T. Security analysis of software updates for industrial robots. In: Proceedings of the 16th International Conference on Critical Information Infrastructures Security (CRITIS 2021); 2021 Sep 27–29; Lausanne, Switzerland. Berlin: Springer; 2021. p. 229–45.
[21] Chung K, Li X, Tang P, Zhu Z, Kalbarczyk ZT, Iyer RK, et al. Smart malware that uses leaked control data of robotic applications: the case of raven-ii surgical robots. In: Proceedings of 22nd International Symposium on Research in Attacks, Intrusions and Defenses; 2022 Oct 26–28; Limassol, Cyprus. Berlin: Springer; 2019. p. 337–51.
[22] Dieber B, Breiling B, Taurer S, Kacianka S, Rass S, Schartner P. Security for the robot operating system. Robot Auton Syst 2017;98:192–203.
[23] Dieber B, Kacianka S, Rass S, Schartner P. Application-level security for ROS-based applications. In: Proceedings of 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2016); 2016 Oct 9–14; Daejeon, Republic of Korea. New York City: IEEE; 2016. p. 4477–82.
[24] Zhang M, Moyne J, Mao ZM, Chen C, Kao B, Qamsane Y, et al. Towards automated safety vetting of PLC code in real-world plants. In: Proceedings of 2019 IEEE Symposium on Security and Privacy (SP); 2019 May 20–22; San Francisco, CA, USA. New York City, IEEE; 2019. p. 522–38.

[25] East S, Butts J, Papa M, Shenoi S. A taxonomy of attacks on the DNP3 protocol. In: Proceedings of International Conference on Critical Infrastructure Protection (ICCIP 2019); 2019 Mar 11–12; Arlington, VA, USA. Berlin: Springer; 2009. p. 67–81.
[26] Hu Y, Yang A, Li H, Sun Y, Sun L. A survey of intrusion detection on industrial control systems. Int J Distrib Sens Netw 2018;14(8):1–14.
[27] Hong J, Liu C, Govindarasu M. Detection of cyber intrusions using network-based multicast messages for substation automation. In: Innovative Smart Grid Technologies (ISGT 2014); 2014 Feb 19–22; Washington, DC, USA. New York City: IEEE; 2014. p. 1–5.
[28] Wang Y, Fan K, Lai Y, Liu Z, Zhou R, Yao X, et al. Intrusion detection of industrial control system based on Modbus TCP protocol. In: Proceedings of 2017 IEEE 13th International Symposium on Autonomous Decentralized Systems (ISADAS); 2017 Mar 22–24; Bangkok, Thailand. New York City: IEEE; 2017. p. 156–62.
[29] Stouffer K, Pillitteri V, Lightman S, Abrams M, Hahn A. Guide to industrial control systems (ICS) security. New York City: National Institute of Standards and Technology (NIST) special publication; 2011.
[30] Marsden T, Moustafa N, Sitnikova E, Creech G. Probability risk identification based intrusion detection system for SCADA systems. In: Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering (ICST 2018); 2018 Apr 9–13; Vasteras, Sweden. Berlin: Springer; 2017. p. 353–63.
[31] Fang C, Qi Y, Cheng P, Zheng W. Optimal periodic watermarking schedule for replay attack detection in cyber–physical systems. Automatica 2020;112:108698.
[32] Falliere N, Murchu LO, Chien E. W32. Stuxnet dossier. Mountain View: Symantec Corp., Security Response; 2011.
[33] Garcia L, Brasser F, Cintuglu MH, Sadeghi A, Mohammed OA, Zonouz SA. Hey, my malware knows physics! attacking PLCs with physical model aware rootkit. In: 4th Annual Network and Distributed System Security Symposium (NDSS 2017); 2017 Feb 26–Mar 1; San Diego, CA, USA. Reston: The Internet Society; 2017. p. 1–15.
[34] Han Y, Etigowni S, Liu H, Zonouz S, Petropulu A. Watch me, but don't touch me! contactless control flow monitoring via electromagnetic emanations. In: Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security; 2017 Oct 30–Nov 3; Dallas, TX, USA. New York City: Association for Computing Machinery; 2017. p. 1095–108.
[35] Pu H, He L, Zhao C, Yau DKY, Cheng P, Chen J. Detecting replay attacks against industrial robots via power fingerprinting. In: Proceedings of the 18th Conference on Embedded Networked Sensor Systems; 2020 Nov 16–19; online. New York City: Association for Computing Machinery (ACM); 2020. p. 285–97.
[36] Pu H, He L, Zhao C, Yau DKY, Cheng P, Chen J. Fingerprinting movements of industrial robots for replay attack detection. IEEE Trans Mob Comput 2021;21(10):3629–43.
[37] Electricity Information Sharing and Analysis Center (E-ISAC). Analysis of the cyber attack on the Ukrainian power grid: defense use case. Washington, DC: E-ISAC; 2016 Mar.
[38] Kovacevic A, Nikolic D. Cyber attacks on critical infrastructure: review and challenges. In: Cruz-Cunha MM, Portela IM, editors. Handbook of Research on Digital Crime, Cyberspace Security, and Information Assurance. Pennsylvania: IGI Global Publisher Of Timely Knowledge; 2015. p. 1–18.
[39] Kris O, Christian D. SoK: ATT&CK techniques and trends in windows malware. In: 15th EAI Conference on Security and Privacy in Communication Systems; 2019 Oct 23–25; Orlando, FL, USA. Berlin: Springer; 2019. p. 406–25.
[40] Mayoral-Vilches V, Carbajo UA, Gil-Uriarte E. Industrial robot ransomware: Akerbeltz. In: 2020 4th IEEE International Conference on Robotic Computing (IRC 2020); 2020 Nov 9–11; Taichung, China. New York City: IEEE; 2020. p. 432–5.
[41] Bonaci T, Yan J, Herron J, Kohno T, Chizeck HJ. Experimental analysis of denial-of-service attacks on teleoperated robotic systems. In: Proceedings of the ACM/IEEE 6th International Conference on Cyber–Physical Systems; 2015 Apr 14–16; Washington, DC, USA. New York City: Association for Computing Machinery (ACM); 2015. p. 11–20.
[42] Pu H. Demo: covering manipulation of industrial robots via data deception [Internet]. Genève: Zenodo; 2022 Aug 6 [cited 2022 Dec 8]. Available from: https://zenodo.org/record/6969707#.Yu6Eay-KFB0.
[43] Gander K. Worker killed by robot at Volkswagen car factory [Internet]. London: The Independent; 2015 Jul 2 [cited 2022 Dec 2]. Available from: https://www.independent.co.uk/news/world/europe/worker-killed-by-robot-at-volkswagen-car/factory-10359557.html.
[44] Workers killed by the industrial robot, how can the safety regulations be ignored? Beijing: Sohu; [cited 2022 Dec 2]. Available from: https://www.sohu.com/a/322037172_642302, 2019.
[45] Sharma A. Universal robots continues to dominate cobot market but faces many challengers [Internet]. London: Interact Analysis; 2018 Nov [cited 2022 Dec 2]. Available from: https://www.roboticstomorrow.com/article/2018/11/universal-robots-continues-to-dominate-cobot-market-but-faces-many-challengers/12804.
[46] Kim CY, Song D, Yi J, Wu X. Decentralized searching of multiple unknown and transient radio sources with paired robots. Engineering 2015;1(1):58–65.
[47] Cerrudo C, Apa L. Hacking robots before Skynet [Internet]. Washington, DC: IOActive, Inc; 2017 Mar 1 [cited 2022 Dec 2]. Available from: https://ioactive.com/hacking-robots-before-skynet/.
[48] Saeed BN. Introduction to robotics: analysis, control, applications. 2nd edition. Hoboken: Wiley; 2010.
[49] Formby D, Srinivasan P, Leonard A, Rogers J, Beyah RA. Who's in control of your control system? Device fingerprinting for cyber–physical systems. In: 2016 Network and Distributed System Security Symposium; 2016 Feb 21–24; San Diego, CA, USA. New York City: IEEE; 2016. p. 1–15.
[50] Quinonez R, Giraldo J, Salazar L, Bauman E, Cardenas A, Lin Z. SAVIOR: securing autonomous vehicles with robust physical invariants. In: Proceedings of the 29th USENIX: Security Symposium; 2020 Aug 12–14; online. Berkeley: USENIX Association; 2020. p. 895–912.
[51] Chen Y, Poskitt CM, Sun J. Learning from mutants: using code mutation to learn and monitor invariants of a cyber–physical system. In: 39th IEEE Symposium on Security and Privacy (SP 2018); 2018 May 20–24; San Francisco, CA, USA. New York City: IEEE; 2018. p. 648–60.