



International Journal of Marketing Management

ISSN 2454 - 5007



www.ijmm.net

Email ID: editor@ijmm.net , ijmm.editor9@gmail.com

A Machining Process Invariant Approach to Cyber-Attack Detection on Cyber-Physical Manufacturing Systems

¹ S. Srinivas, ² Ch. Vanajasri,

¹Assistant Professor, Megha Institute of Engineering & Technology for Women, Ghatkesar.

² MCA Student, Megha Institute of Engineering & Technology for Women, Ghatkesar.

Abstract

With more and more Internet of Things (IoT) devices being used in manufacturing, cyber-physical manufacturing systems (CPMSs) have become simpler targets for hackers. There have been major developments in the area of CPMS intrusion detection recently. Nevertheless, the present approaches entail substantial expenses when gathering past data to train detection models, which are customized for certain machining situations. These procedures are put to the test by the ever-changing nature of machining in practice. In this research, we lay the framework for detection by detailing our finding that the CPMS machining code contains a whole machining process. Using the Machining Process Invariant of the machining code, we provide MPI-CNC, an intrusion detection technique. Automating the analysis of machining codes to build machining process rules and important parameter sets is possible with MPI-CNC. As rules for detection, they are very useful. After then, MPI-CNC seeks out the CPMS's runtime state and checks it against the detection criteria to determine how the cyberattack behaved. We ran 10 actual machining operations using MPI-CNC on two FANUC CNC machines to see how well it performed. The results of the experiments proved that MPI-CNC is quite versatile. While keeping regular machining processes going, MPI-CNC demonstrated it could identify cyber-attacks on CPMS with more accuracy than state-of-the-art detection systems.

Index Terms

Industrial Internet of Things (IoT), cyber assault, cyber-physical manufacturing systems (CPMSs), and computer numerical control (CNC).

I. INTRODUCTION

Without manufacturing, modern industrialization would not have been conceivable. The industrial sector throughout the world is rapidly moving toward networked and intelligent growth, thanks to the rise of intelligent manufacturing and the Internet of Things [1]. The core of every CNC system is the computer numerical control system. On January 22, 2024, January 11, 2024, and October 23, 2023, the wording was modified. Publication date: January 25, 2024; date of revision: May 9, 2024. The Chinese government's 110400ZG21 National Key Research and Development Program helped fund part of this project. (Limin Sun is the author in question.) Zhengdong Li, Xin Chen, Shijie Li, Limin Sun, Shichao Lv, and Hangyu Wang are all listed as contributors. They are affiliated with many organizations, including the Institute of Information Engineering, Beijing 100085, China; the School of Cyber Security, Beijing 101408, China; and the Beijing Key Laboratory of IOT Information Security Technology. Feel free to reach out to them via any of the following email addresses: sunlimin, lvshichao, lizedong, chenxin1990, lishijie, wanghangyu, or liie.ac.cn. To contact Yuqi Chen, please send an email to chenyyq@shanghaitech.edu.cn. Currently residing in Shanghai, China, Chen is linked with ShanghaiTech University's School of Information Science and Technology. A unique number has been assigned to the digital item 10.1109/IIOT.2024.3358798.

Control of the machining activities of production machines via the use of cyber-physical manufacturing control systems (CPMSs). Countless sectors depend on computer numerical control (CNC) machines, including aerospace, automotive, and military. Automated control of production processes made feasible by CNC equipment linked to the industrial Internet allows the Giga plant at Tesla to achieve much improved production efficiency. There are serious threats to the most important CPMS security measures as the number of companies connecting their CNC systems to the Industrial

Internet continues to rise. An exciting new area of study is intrusion detection algorithms for content management systems (CPMS). Side channel data may be anything from current [2], video [3] or audio [4,5], [6] produced by machining; hence, most CPMS security researchers are now training ML classification models to identify abnormalities in this data. Researchers in this area have developed data-driven digital twin models of CNC systems that can detect attacks by comparing the system's runtime state [7]. You may also find altered machining codes offline by training machine learning anomaly detection models using digital attributes extracted from the codes [8].

For some machining situations, these solutions can identify unusual processing characteristics. Unfortunately, cyber-attacks can be easily launched by attackers due to the lack of authentication mechanisms, plain-text transmission, and unfixed vulnerabilities in CNC systems, which are caused by manufacturers not taking adequate security into account. In order to cause processing errors in the product, attackers mostly aim at the machining code. One example is the introduction of a Trojan horse into the CNC system's firmware, which allows the attacker to covertly alter the machining code that is inputted into the system. This allows them to carry out evil hole attacks [9]. The mechanical strength of the final product can be compromised since attackers have shown they can use steganography to alter machining code files in network traffic [10]. The second kind of attack involves the hacker accessing the CNC system's memory and changing important settings. For example, they manipulate settings to replace processing materials, insert smart materials inside gas masks to plant logic bombs physically, and so on. The result is gas masks that burst and let gas out when worn [11]. Additionally, criminals have made use of pre-existing open-source software such as C3PO [12] and Industrial Intrusion Detection System (IDS) [13] to inject CNC systems with harmful instructions that halt operation. These incidents highlight the critical requirement of intrusion detection systems for CNC systems in order to properly combat these types of attacks. Driving force: Various machining codes are used by the CNC system to handle different products in real manufacturing processes, which results in numerous machining scenarios. As a consequence of the varied tool routes, raw materials, and machining equipment needed for each situation, the audio, picture, current,

and voltage side-channel properties also vary. Researchers usually have to spend a lot of time and energy gathering sidechannel data for each new machining situation and starting the training process all over again to build intrusion detection models that use this data across diverse circumstances. However, once an attacker gains access to a CPMS system and successfully executes an attack script, they may easily interrupt various machining situations. As a result, a versatile intrusion detection strategy for the CPMS system is critical for effectively countering current attack tactics in a range of machining scenarios. Observation: One prominent approach to industrial control security for detecting cyber-attack anomalies is invariant rule-based detection. Industrial control devices often use control logic codes, which include invariant control logic, to regulate the typical operation of industrial systems. As a foundation for intrusion detection in industrial control systems, researchers have used data-driven [14] or code-driven [15] techniques to derive control logical invariant principles. They have done a fantastic job with detecting. The invariant rule-based detection served as an inspiration for our discovery that the CNC system's machining process is invariant in the CPMS field, and that the machining code includes extensive information on the invariant machining process. So, by looking at the machining code, we can find all the rules that are invariant to the machining process. Key components of the machining process invariant rules are machining speed and machining trajectory; these rules may characterize the CNC system's machining process in detail and provide a solid foundation for intrusion detection. Approach: This paper suggests MPI-CNC, an intrusion detection approach based on Machining Process Invariant, to solve the problem of the CPMS intrusion detection system's poor adaptive capacity. Machine programming language MPI-CNC quickly and automatically parses machining code for detection criteria. The technique begins by extracting important machining-related information such as tool routes, machining sequences, spindle speeds, and more from the code, which are then used as criteria to detect assaults. Afterwards, MPI-CNC actively gathers critical parameters and realtime machining status from the CNC system while milling. Last but not least, MPI-CNC checks if the data from runtime machining is consistent using the detection rules to find cyberattacks. The result is a prototype built on the FANUC CNC system that demonstrated the viability of the method described in this article. We assessed the deployment time cost, detection

performance, and interference to the CNC system, and we ran trials using genuine CNC machines. We also investigated ten real machining situations and three assault strategies. The results of the experiments proved that MPI-CNC may be easily used for new machining scenarios without preprocessing and can reliably identify cyber-attacks while the CNC is running, without disrupting its normal operation. MPI-CNC outperforms other cutting-edge detection algorithms in terms of detection performance. Machine tool code injection attack has a detection accuracy of 98.81% and parameter injection attack of 100%, whilst other approaches get the best detection results of 93.25% and 98.38%, respectively. In what follows, this piece makes a contribution. 1) A new method for automatically extracting detection criteria from machining codes is proposed. Increased flexibility in CPMS intrusion detection is a result of its ability to quickly develop detection rules for various machining settings. 2) Following a reverse-engineering of the FOCAS protocol utilized by FANUC CNC systems, we created acquisition request packets with little interference that adhere to the protocol's specifications. This method enhances data collecting efficiency while decreasing machining process disturbance. 3) The FANUC CNC system served as the foundation for the development of a prototype CPMS IDS. Based on this concept, this prototype may be expanded and adapted to different CNC systems and protocols, even though it was designed for a specific CNC system. 4) We tested our suggested method in three assault scenarios and ten real-world machining settings to see how well it could adapt and identify threats. Experiments demonstrate that this method can reliably identify cyber-attacks in runtime without impacting the CNC's normal operation and can be rapidly deployed to new machining circumstances without preprocessing. Strategic Plan: What follows is the outline for the rest of the article. While Section III gives a synopsis of the MPI-CNC, Section II lays forth the technical foundation of CPMS. In Section IV, we go over the MPI-CNC and how it was put into practice. In Section V, we discuss the experimental evaluation. The present state of CPMS intrusion detection technologies is introduced in Section VI. In Section VII, we will go over the MPI-CNC's shortcomings. Section VIII serves as the last section.

II. BACKGROUND

The major focus of this piece is to provide a way for CPMS intrusion detection. This section aims to give a synopsis of the relevant literature by highlighting two key points: 1) how CPMS is made and 2) how it is machined, along with the many types of assaults that it faces. Section A. Cyber-Physical Production Networks See Figure 1(a) for an illustration of the typical components of a CPMS, which include an engineering station, a server for machine data collecting (MDC), and distributed numerical control (DNC) [see also: Fig. 1(a)]. The

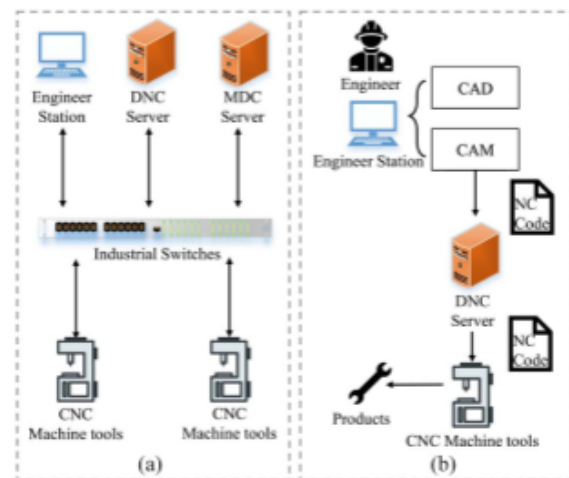


Fig. 1. Network topology and processing process of a CPMS. (a) Network topology of CPMS. (b) Processing process of CPMS.

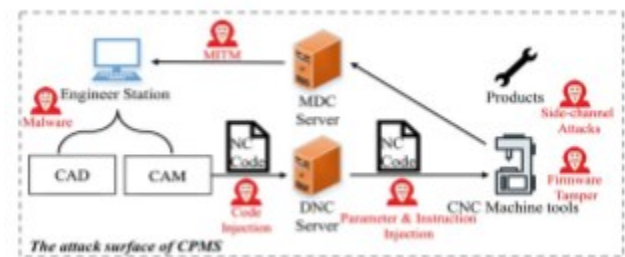


Fig. 2. Attack surface of CPMS.

An engineering station is often a desktop or laptop computer in an office that has CAD and CAM software installed. As seen in Figure 1(b), the Processing is illustrated. Machining code, also known as NC code, is created by engineers in engineering design software and subsequently submitted to the DNC server. The DNC server is responsible for

distributing the NC code to the machinery that needs it during production. By analyzing the NC code, the CNC system takes control of the machining process automatically. The MDC server communicates with the production machinery to gather data on its location, velocity, temperature, and other parameters. In this way, engineers can track the machining process thanks to the data being sent back to the engineering station's monitoring program. On top of that, while machining is underway, engineers may provide control orders to execute runtime operations.

B. Model of Attack Recently, researchers have examined and classified cyber-attacks that target CPMS [16], [17]. Examining the attack surface from the standpoint of the production process (see Fig. 2), this essay delves into the recent cyber-attacks on CPMS. At some point throughout manufacturing, the engineer's workstation will have an Internet or local area network connection. Hackers can acquire CAD models and NC codes, as well as attack system and software vulnerabilities, by targeting the engineer station with malicious vectors like as spear-phishing [18] and BadUSB [19]. Ethernet is the most common method for devices like engineer stations and DNC servers to talk to the CNC system, although the protocols used for this connection aren't always secure enough. To transfer NC code in plain text, DNC servers may utilize the FTP protocol, for instance. Hackers can insert harmful instructions and parameters into network packets by replaying them or by interfering with and stealing NC code during man-in-the-middle attacks [20]. Additionally, the CNC system firmware is vulnerable to manipulation by the attacker, who can disrupt regular operations [9], [21]. Such assaults, however, are more complex and need an in-depth familiarity of the CNC system's code structure. Since CPMS is a typical cyber-physical system, there are a number of ways attackers can disrupt normal processing [23] or gain steganographic access to information about the machine tool's production status and workpiece geometry [24] through side-channel attack methods like electrical measurement interference and acoustic resonance. There were three main types of attacks: 1) Injection of machining code; 2) Injection of parameters; and 3) Injection of instructions. The term "machining code injection" is used to describe an assault when the CNC system's machining code, also known as the NC code, is altered or replaced [9], [10], [25]. It is possible to compromise CNC machines and cause them to malfunction by altering critical sections of code, such as the cutting path,

spindle speed, or auxiliary control code. Parameter Injection: Changing the CNC system's settings is known as parameter injection [11], [26]. Machining is affected by a number of critical CNC system factors, including spindle speed ratio, quick feed rate, and alarm shielding. Consequently, the CNC system, machining precision, and the CNC machine itself are all vulnerable to assaults that compromise these critical metrics. When someone sends harmful control orders to the CNC system, it's called "Instruction Injection." This kind of command injection disturbs the regular machining process. An open-source program known as C3PO was released by McCormack et al. [12]. It examines the network services of 3D printers for any vulnerabilities and then utilizes those flaws to transmit malicious commands that attack CNC systems that are managed remotely. Additionally, attackers can interrupt production by delivering attack scripts and injecting malicious commands through the ISF [13]. Potential security risks to CNC systems include physical assaults that span domains and side-channel information leaking [27]. However, it should be mentioned that these dangers are not addressed in this essay. Their infeasibility, lack of danger potential, or vulnerability to detection by current IDSs are some of the reasons they are not included.

III. OVERVIEW

Here, we present MPI-CNC, a novel approach to intrusion detection for CPMS. For example, as seen in

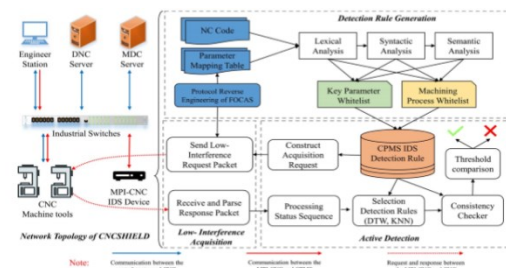


Fig. 3. Systematic approach to build MPI-CNC.

Figure 3 shows that there are three separate steps to this process: First, actively detecting; second, acquiring a low-interference condition; and third, generating detection rules. Here we give a brief outline of the basic structure for evaluating intrusion

detection technologies. **Generating Detection Rules:** Using static analysis, the detection rule generation module parses the NC code and extracts detection rules. The NC code contains all the rules for the machining process, including the rules for the important parameters. By using the key parameter rules, critical parameters in the CNC system may be checked and double-checked to make sure they are accurate, stable, and protected against malicious manipulation that could cause machine faults or reduced cutting precision. In contrast, machining process rules identify malicious attempts such program replacement or tampering with machining trajectories by examining the NC code's machining processes' invariant properties and therefore establishing reference recommendations. The NC code's derived detection rules provide an exhaustive picture of the machining process and allow for the proactive identification of network assaults and unforeseen abnormalities. **Low-Interference Acquisition:** The CNC system's machining states are collected during runtime by the low-interference state acquisition module. In order to keep tabs on how well a CNC system is running, most manufacturers include either development kits or monitoring software. An example of a development component that allows for secondary development is the FANUC Focas 1/2. With active connectivity with the CNC system, it allows for runtime remote monitoring. Important details including NC programs, tool locations, and spindle velocities are supplied by the development component. Nevertheless, regular machining operations and real-time performance might be significantly affected by an increase in the CNC system's network demand caused by directly using the original development kit for high-frequency data gathering. Our research gets around this problem by analyzing proprietary procedures through the process of reverse engineering. In addition, we personalize data gathering requests and do away with unnecessary ones. Consequently, we are able to get runtime machining states within the CNC system at high frequencies with minimum interference. **Awareness Monitoring:** In particular, the active detection module looks for irregularities while the machine is being turned. It receives data on the machining state during runtime from the low-interference state acquisition module, processes it, and then sends out notifications. The CNC system's machining status is checked to see if it follows the regulations for key parameters and machining process. Malicious instruction assaults, code

tampering, and key parameter manipulation are all prevented by this strategy. Two important monitoring parameters—the error threshold and the size of the monitoring window—must be determined during the active detection stage. To find the sweet spot, we first try out various widths of monitoring windows according to the state acquisition frequency. Afterwards, we determine the greatest observed error and use it to define the error threshold. For each monitoring window, we next compute the cumulative normal error under certain window widths.

IV. APPROACH

A. Issue Description The computer numerical control (CNC) system uses data from a number of sensors to manage the relative motion of the tool and the workpiece in a closed-loop system, making CNC production a complicated industrial control process. In this article, we will utilize the CNC's machining state at time t as described by $u(t)$ in (1), where $P(x,y,z)$ is the tool's coordinates in the xyz three axes, $S(t)$ is the spindle speed, $F(t)$ is the feed rate, and $T(t)$ is the current tool number.

$$u(t) = (P(x, y, z), S(t), F(t), T(t)). \quad (1)$$

To characterize the machining process as suggested by the machining code, which stands for the CNC machining process's invariant features, we define (2) with $r(n)$. In particular, the machining path curve equation—typically, straight lines and circular arcs—is represented by $F(x,y,z)$. The beginning and ending positions of the machining route are represented by $Start(x,y,z)$ and $End(x,y,z)$, respectively. Tool number, feed rate, cutting route, and spindle speed all work together to describe machining in detail.

$$r(n) = (F(x, y, z), Start(x, y, z), End(x, y, z), S(n), F(n), T(n)). \quad (2)$$

In equation (3), we define the CNC machining model using $M()$, where $\varepsilon(t)$ stands for the CNC's internal losses and tolerable mistakes caused by external variables.

$$u(t+1) = M(u(t), r(t), \varepsilon(t)). \quad (3)$$

Due to machine wear and tear, current and voltage jitter, and other variables, the CNC machining state experiences tolerable errors $\varepsilon(t)$ during the typical

machining process. While the CNC is normally following the CNC machining model M() and the present machining process $r(n)$, it can get drastically off course when cyberattacks are underway. As a result, we used the constant features of CNC machining to build an intrusion detection system that is detailed in this paper. The three main components of our method are an active detection engine, a low-interference acquisition module, and a module for generating detection rules [see Fig. 3].

B. Rule for Detecting Intrusions For the production and processing of workpieces, the machining process is foundational. Spindle speed, feed rate, and cutting route are just a few examples of the extensive and detailed machining process details included in the NC code. The numerical control code (NC code) is translated into executable instructions by the CNC system, which then controls the different parts of the CNC machine tool to finish the machining processes. It is possible to create key parameter whitelist rules and machining process whitelist rules for intrusion detection by analyzing the NC code. The method of creating detection rules using code analysis has shown to be highly useful in the field of industrial control [15], [28].

I. Rule for the Key Parameter Whitelist: The real manufacturing process in CNC systems is affected by a great number of critical factors. It is necessary to modify the procedure for transmitting orders from the CNC system in order to operate the hardware in accordance with the specified parameters. In real-time machining, for example, the CNC system modifies the tool's landing spot and trajectory of movement according to parameters like tool length compensation and tool radius compensation, or it regulates the feed acceleration according to acceleration and deceleration parameters. These

Parameter Type	FOCAS Address Mapping	Data Type	Number of parameters
Tool Compensation Parameters	0x000800000001 -0x000800000190	Real	400
Macro Variables	0x001500000001 -0x0015000003e7	Real	633
CNC Parameter	0x008d00000001 -0x008d000006b9	Bit(axis), Byte(axis), Word(axis), Real(axis)	27609
PMC Parameter	0x800100000000 00000000 -0x800100000bb7 00000009	Bit(axis), Byte(axis), Word(axis), Real(axis)	6572
All Parameters			35214

factors have an immediate effect on the stability and precision of the machine tool during machining. Decreased machining accuracy or machine tool failure can occur if the critical parameters in the CNC system are intentionally altered by malevolent individuals. Crucial CNC system parameters often have predetermined ranges of values. For example, there are fixed values for the parameters that govern the tool's radius and length of compensation, and the parameters that regulate the feed's acceleration and deceleration typically lie between 140 and 160 ms. To ensure that the FANUC CNC system's critical parameters are always set correctly, we examine the values of these parameters in Table I and create whitelist rules for them. "Whitelist Rule for Machining Processes" For CNC programming languages, the worldwide Organization for Standardization (ISO) has set ISO-6983-1 [29] as the worldwide standard. The CNC codes, which consist of G-codes and M-codes, are defined by this standard, which also specifies their syntax and lexicon. Many companies that make CNC systems have released CNC control devices that are ISO-6983-1 compliant. Some examples of CNC systems that allow NC programming in accordance with this standard include SIEMENS's SINUMERIK 802D and SINUMERIK 840D and FANUC's 0i-md and 0i-mf. Different CNC models and manufacturers may use different NC code representations, but they all share certain features and there aren't many differences when it comes to syntax and programming ideas. Thus, by utilizing our suggested detection technique and taking into account these commonalities, we may adapt the intrusion detection system to work with various CNC system architectures. The NC code is what the CNC system uses to control the machining process. To accomplish automated production machining, the spindle and several servo axes of the CNC system coordinate to control the tool's and workpiece's movement and

TABLE I**FANUC PARAMETERS MAPPING TABLE**

rotation. Consequently, in order to create machining process rules, we need to decipher the NC code and find the constant correlations between tool motion trajectories, feed rates, spindle speeds, and other parameters at each stage of the process flow. Malicious attacks, including parameter injection or tampering with NC code, can be identified using these principles as benchmarks.

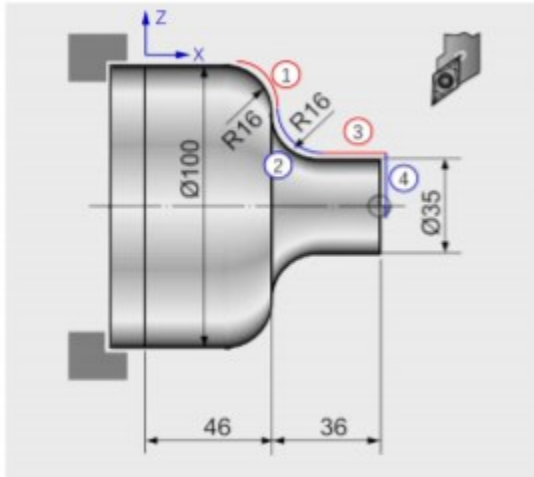


Fig. 4. Case of machining process whitelist rule.

We show how to develop detection rules based on the machining process invariance utilizing NC codes using turning machining as an example [see Fig. 4]. During turning, the NC codes are executed by the CNC system to automatically regulate the machining process. During machining, the NC codes determine the spindle speed, feed rate, and tool number. The tool's movement trajectory is then specified using G codes, for example, G01 for ordinary linear machining and G02 and G03 for circular machining. Hence, in order to create the rules for the machining process, we analyze the NC codes lexically, syntactically, and semantically. For example, rule 1 states that when the spindle speed (S) is 2000 and the feed rate (F) is 1500, tool number 3 follows the curve $(x-20)^2 + (z+16)^2 = 16^2$, which begins at (20,0) and ends at (36,-16). C. Acquisition with Minimal Interference The standard practice for gathering CNC runtime status information involves utilizing the manufacturer-supplied communication interface. Focas is FANUC's communication interface; nevertheless, we discovered that its acquisition frequency is too low, which increases intrusion detection alert latency and impacts detection rule selection accuracy. This is why we took the time to

manually reverse-engineer the Focas protocol format in order to create acquisition packets with minimal interference. Reverse Engineering the FOCAS Protocol: For FANUC CNC systems, we reverse-engineer the proprietary protocol Focas by capturing mirror traffic on an industrial switch. One application-layer protocol that uses TCP/IP is Focas. The Focas protocol needs two rounds of the TCP handshake in order to establish a connection. Initially, the client initiates a connection setup request to port 8139 of the CNC system using port A (or any other accessible port). The customer then uses either port A + 1 or port A + 2 to connect a second time to the CNC system's port 8193. Port A + 1 or A + 2 is where the following request and response procedures are carried out. Turn it backwards

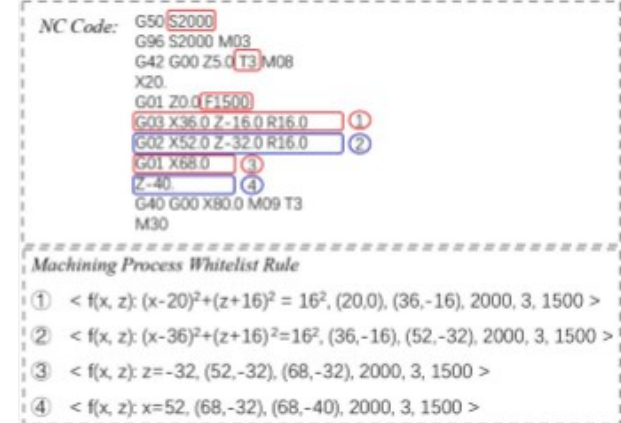


Fig. 5. FOCAS protocol reverse analysis results. This figure shows a binary request packet for collecting the current machining coordinates of a CNC using Focas, which consists of a Focas protocol header and ten subitems. The frame format of the Focas protocol is revealed by protocol analysis [see Fig. 5]. As identification for the Focas protocol, the first four

bytes of the payload part are always a0a0a0a0. The request/response gap is represented by the fifth and sixth bytes. Bytes 7 and 8 contain the code for the Focas function. The length of the payload data is represented by the 9th and 10th bytes. The amount of subitems is indicated by the 11th and 12th bytes. The Focas protocol's header consists of the first twelve bytes. Subitem code, fixed padding, and subitem length are the subitems that make up the Focas protocol. To prevent potential exploitation by nefarious persons, the request parameter address and data format are not explicitly stated in this page, but they are part of the subitem's payload information. Second, Building Acquisition Packages with Minimal Interference: According to the findings of the aforementioned protocol analysis, it is necessary to send multiple request packets to the Focas API in order to simultaneously gather the position coordinates, spindle speed, and feed speed of CNC systems. There are sometimes unconnected and useless subitems in these request packets as well. in order to identify attacks. These superfluous subitems drain the CNC system's and the network's computational resources under high-frequency collection, which in turn lowers detection efficiency. We overcame this issue by merging many request packets containing detection-related subitems into a single packet. This packet was subsequently transmitted to the CNC system in order to simultaneously gather various machining statuses. The CNC system's status data is retrieved using the Focas protocol frame format upon receiving the response packet. In order to minimize interference with the CNC system and the network overhead of status gathering, low-interference acquisition packages were developed by reverse-engineering the proprietary protocol. In addition, the results demonstrate that the article's methodology may be applied to the S7comm-nck protocol, which is utilized by SINUMERIK 828 and 848 CNCs, in order to generate low-interference request packets. Section V-D contains the detailed experimental data. Active Detection of Intrusions Approach The methods used to identify assaults on manufacturing processes based on detection criteria are outlined in this section [see Fig. 3]. To avoid interfering with the regular CNC

machining process, our method makes use of a runtime active intrusion detection mechanism with low interference. Two major obstacles were successfully overcome throughout the execution of this module. 1) Circuit instability, mechanical jitter, and equipment aging are common phenomena that can happen during milling. The CNC's execution time for each instruction may become inconsistent due to these difficulties. This makes it hard to quickly and precisely match the detected runtime machining condition with the recorded data. 2) The detection program's false alarm rate might be boosted by jitter and other interference caused by ordinary faults; thus, it is necessary to differentiate between regular mistakes and cyber assaults. 1) Rules for Detecting Selections: As with the first challenge, we used two algorithms: dynamic temporal warping (DTW) and k-nearest neighbors (KNNs). DTW is a dynamic programming method that compares time series, especially ones with different durations, to find out how similar they are [30]. Its suitability to temporal data makes it a popular choice in speech recognition, gesture identification, and information retrieval, among other fields. For this research, we used the DTW technique to chronologically align a reference state sequence with a rule label with a captured runtime processing state sequence that had timestamps. When it comes to supervised learning, KNN is a nonparametric technique [31]. KNN operates on an easy-to-understand principle: a sample is likewise classified as belonging to a specific category if the majority of the k-most comparable samples in its feature space also belong to that category. When deciding, the algorithm simply takes into account the category of the samples that are geographically closest to it. To categorize points in runtime processing, our study utilized the KNN method. Thanks to the reference state sequences that were tagged with the rules, we were able to choose the proper detection rules. 2) Consistency Checker-Reliant Detection Boundaries and Windows: Our strategy for tackling difficulty 2 included the use of a detection window and an alert threshold. As part of the detection procedure, we record the CNC machine tool's continuous machining runtime state throughout the whole window period, and then we add up the

discrepancy between each runtime machining state and the machining process regulations. When the total mistake goes beyond a certain point, an alert is sent out. If the cumulative error is less than the threshold when the window expires, the cumulative error is reset to zero and a new inspection window is started. Here, we used (4) to perform a consistency check, which entails adding up the distance in geometric units between the machining process rule and the runtime machining state inside the inspection window and comparing it to the inspection threshold. In particular, using the formula in (5), the real error value is determined by subtracting the actual feed rate F and spindle speed S from the machining process rule, as well as the distance D between the actual location and the machining trajectory. To further guarantee key parameter consistency, we used dissimilarity verification to compare the runtime key parameter matrix $C_{3 \times n}$ with the key parameter rule $K_{3 \times n}$. In order to detect assaults on the CNC system, such as machining code injection, parameter injection, and instruction injection, we conduct a machining process consistency verification based on Equation (3).

$$\left\{ \sum^{W \text{ size}} \|y(t) - r(t)\| \leq \delta(t) \right\} \wedge \{C_{3 \times n} \oplus K_{3 \times n} = [0]_{3 \times n}\} \quad (4)$$

$$\|y(t) - r(t)\| = \sqrt{D^2 + (F - F_r)^2 + (S - S_r)^2}. \quad (5)$$

In Algorithm 1, we can see the essential components of the active detection engine, which are responsible for actively and minimally interfering with the CNC's ability to detect attacks. After connecting to the CNC (line 1), it builds the active acquisition packet (lines 2-4), simulates the machining path by parsing the detection rules (lines 5-16), detects the attack (lines 5-16), and finally disconnects (line 17) when detection is complete. Initializing the detection window and detection threshold is the first step in the assault detection phase (line 6). After then, lines 7-10 detail the steps taken to get processing status values from the CNC's return data after sending a low-interference request packet. The second step is to match the detection criteria for the data in the current detection window using the DTW method. Lastly, lines 12-14 are used to check if the detection rules

are satisfied by doing a consistency check on the machining state and key parameters. If you want your CNC to have less interference, you need to specify the necessary frequency (line 15).

V. EVALUATION

The following research question is the main emphasis of this section.

Algorithm 1: Algorithm of Active Detection

Input : Key Parameter Rule
Machining Process Rule
Detection Window Size
Output: Cyber-Attack Alerts

```

1 Connect (cncIP, cncPORT);
2 RuleDB ← LoadRules (KeyParameter, MachiningProcess);
3 PathSim ← PathSimulation (RuleDB);
4 AcquisitionPKG ← PackageConstructor (RuleDB);
5 while ProcessFlag do
6   InitDetectionWindow();
7   for 0 to DetectionWindowSize do
8     Send (AcquisitionPKG);
9     ProcessingStatus ← Receive();
10  end
11  FlaggedStatus ← DTW (ProcessingStatus, PathSim);
12  if ConsistencyChecker (FlaggedStatus, RuleDB) is
    False then
13    Alarm ("Illegal Processes: cncIP, RuleNo.");
14  end
15  Sleep (t);
16 end
17 Disconnect();
```

RQ1: What is the time cost of MPI-CNC to generate rules?

RQ2: What is the effectiveness of MPI-CNC detection against cyber attacks on CNC systems?

RQ3: Does MPI-CNC affect CNC machine tool machining efficiency?

We used 3589 lines of C code and 2492 lines of Python code to implement MPI-CNC. MPI-CNC is deployed on a ThinkPad P15Gen2 with an 8 cores Intel Core i9-11950H CPU and 64 GB of RAM. MPI-CNC was evaluated using Fanuc 0i-md CNC and Fanuc 0i-td CNC.

Part A. The Field of Experiment Designing the Experiment: Experiments were carried out utilizing real-world machining settings with the FANUC CNC system to address the three research topics mentioned above. In order to address the dearth of real-world cyber-attack data for the FOCAS CNC system, the three attack techniques outlined in Section II-B were put into action and their consequences were shown on actual equipment. The correctness of the automated parsing of NC code for rule creation was

checked by analyzing many NC programs from actual machining scenarios that are suitable to the FANUC CNC system. Detection rules were then developed. Secondly, in order to assess how well the suggested intrusion detection strategy uncovered network threats during NC machining, it was compared to existing detection methods. Lastly, we compared the network resource utilization between using low-interference data collection requests and using the FOCAS standard interface to collect the CNC system's machining status. We monitored the variations in machining time during the detection phase to show that our solution has minimal impact on the machining process. In this study, we used a FANUC 0i-tf CNC system [Fig. 6(a)] and a FANUC 0i-md CNC system [Fig. 6(b)] for our tests. as seen in Figure 6.

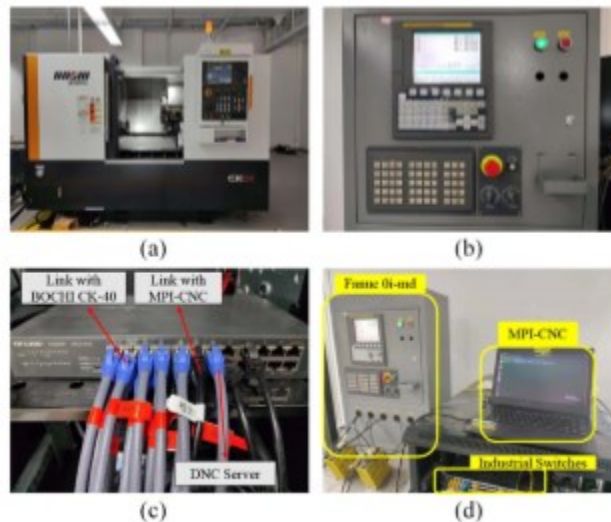


Fig. 6. MPI-CNC experimental environment. (a) BOCHI CK-40 with Fanuc 0i-tf. (b) Fanuc 0i-md. (c) Industrial switches. (d) MPI-CNC deployment environment.

To ensure normal communication between the CNC and the MPI-CNC, we ensure that it is configured to be in the same network segment as the other devices and connect it to the industrial switch that is linked to the CNC. Bochi CK-40, DNC server, and MPI-CNC connection status to each industrial switch port is partially shown in Fig. 6(c). You can see the CNC, MPI-CNC, and industrial switch in Fig. 6(d), which also illustrates the site configuration of the Fanuc 0i-md intrusion detection testing environment. Scenario of a Cyberattack: In this paper, we focus on three

attack methods that were used on FANUC CNCs because there weren't any other assaults accessible for examination. The machining code injection assault is the initial strategy; it entails inserting harmful machining instructions and routes into the NC code. To test how well the approaches might identify tampering, we used ten distinct machining codes and created twenty tamperings, such as creep assaults. The second technique involves manipulating critical CNC parameters through request packets transmitted to the FANUC CNC according to the findings of the Focas protocol inversion. This is known as the remote parameter injection attack. Malicious command injection is the third type of attack. It entails sending request packets to the FANUC CNC over the allocated ports of the PMC, using the results of the Focas protocol inversion and the CNC's interface manual as guidelines. With this vulnerability, malicious instructions like remote start/stop and coolant on/off commands may be introduced. B. RQ1—Processing Detection Rules: How Much Time and How Accurate Is It? From actual machining situations and online resources like Github and Traceparts, we compiled 57 NC codes that are compatible with FANUC CNC systems. A total of 8354 instructions, including those for linear machining, circular arc machining, milling, and turning, make up these codes.

TABLE II

TIME COST OF GENERATING DETECTION RULES

NC Code	Code Lines	Number of Rules	Times Cost(ms)
O5665-NC	134	90	1.004
O6383	150	93	0.805
NCViewer.nc	5780	5753	0.962
NCtest26.NC	64	61	0.768
7190.3-1A.nc	255	222	0.792
...	...	...	...
Number of NC Code: 57	Total Code Lines: 8354	Total Rules: 7671	Total Time Cost: 53.579ms

controls for coolant and tool changes, among other things. We retrieved attack detection criteria by examining the NC programs' machining instructions. With just a few G codes and M codes, NC programs are relatively simple compared to other high-level languages. We represented the program size by the number of G codes that regulate the machining trajectory. Based on the link between G codes and M codes, we developed multiple key parameter rules and one machining process rule per G code. In order

to create a mapping table of G codes, M codes, and system parameter addresses, we first studied the FANUC CNC system user handbook and interface manual. Then, we incorporated the findings from reverse engineering the FOCAS proprietary protocol. Next, we created key parameter rules and chose critical machining parameters using the semantic information of the NC codes' G and M codes. We then combined data such as the beginning and ending points of the machining, spindle speed, feed rate, and tool number to produce machining process rules. We abstracted the machining trajectory's curve equation using its semantic information. We documented all of the produced rules for critical parameters and machining processes after analyzing all of the acquired NC codes, as seen in Table II. There were 8354 lines of machining instructions in all of the 57 NC codes that we gathered. We identify critical machining parameters and construct 6056 machining process rules for these 57 NC codes that we gathered. For each NC code, we documented the time it took to develop the rules, the amount of inspection rules generated, and the number of lines of machining code. According to Table II, it takes an average of 0.94 ms to examine the creation of inspection rules for a single NC code, and 0.007 ms for a single inspection rule. We picked ten NC codes that were typical of the whole set and used all of the rules for the important parameters and the machining process to manually check that the detection rules that had been automatically created were accurate. Findings shown that, in a small subset of NC program samples, the detection criteria achieved a perfect score. Response to Research Question 1: This paper proposes an approach that eliminates the need for past production data by automatically analyzing NC code and generating complete detection criteria. The generation of precise inspection rules takes 0.94 milliseconds for every NC code.

TABLE III

COMPARING THE ACCURACY AND TIME COST OF DTW AND KNN ALGORITHMS IN SELECTING DETECTION RULE

Algorithms	KNN				DTW
	k=3	k=5	k=7	kd-tree	
Accuracy %	96.95	97.82	96.83	97.32	99.38
Times Cost(ms)	273	321	326	137	352

This article's solution has a very low time cost and may be applied rapidly to new machining settings. C. Results of the Cyber-Attack Detection Survey (RPMS) Section II-B summarizes three typical attack strategies and offers a thorough analysis of the present state-of-the-art research on assaults against industrial processes. Keep in mind that there aren't any generic CPMS attacks out there just yet; the ones that are publically accessible tend to be quite particular to certain machining circumstances, equipment, and processing procedures. Hence, we use three attack strategies to assess the MPI-CNC's detection efficacy in this study. Detection Rules for Selection End result: We used 25,283 data points from actual machining settings to do studies comparing the speed and accuracy of KNN and DTW algorithms for rule selection. Table III displays the outcomes. In the rule selection tests, we used a kd-tree data structure and varied the values of k to evaluate the KNN algorithm's performance. According to the data, the KNN algorithm achieved its best rule selection accuracy of 97.82% with a value of k equal to 5. Note that rule selection for 25283 data points took just 137 ms, a significant reduction in time, thanks to the usage of a kd-tree data structure. Because of this, it can meet the needs of complicated machining situations as well as those involving real-time detection. With a time cost of 352 ms and a selection accuracy of 99.38%, the DTW algorithm was able to choose rules with sufficient precision to satisfy real-time alert requirements. To summarize, MPI-CNC uses the DTW method to pick rules, which improves detection accuracy. The kd-tree data structure-based KNN method is recommended for complicated machining situations requiring high-real-time detection. 2) Findings on Cyber-Attack Detection: Table IV displays the findings of our trials that compared the efficacy of various detection windows in detecting attacks. The machining code injection, key parameter injection, and malicious instruction injection attacks were all identified by the active detection engine. The machining process and the critical parameters of the CNC system are immediately affected by these assaults, which disrupt the regular machining process

and alter the machining trajectory and state. In order to create a map of the FANUC CNC's real machining state and critical parameters, the active detection engine constantly interacts with it. It is challenging for the assaults to be detected using this active detection strategy.

TABLE IV COMPARISON OF CYBER-ATTACK DETECTION RESULTS IN DIFFERENT DETECTION WINDOWS AND DETECTION THRESHOLD CASES

Window size /Threshold	Detection accuracy	False alarm rate	Missing alarm rate	Alarm delay(s)
10/0.1	99.15%	2.89%	1.83%	1.45
50/0.5	98.81%	1.09%	2.42%	2.45
100/1	98.68%	0.75%	6.17%	3.71
200/2	96.88%	2.61%	7.15%	6.32
500/5	94.87%	4.44%	6.25%	13.75

concealed because it necessitates the adversary to acquire knowledge of the actual machining process and then alter the CNC firmware, the network communication module, or use other complex ways to change the network data so that it follows the rules of the machining process and the rules of critical parameters. Table IV shows that throughout the active detection studies, we used detection window widths of 10, 50, 100, 200, and 500. Using the CNC machine tool's machining precision of 0.01 mm multiplied by the window size, we determined the alert threshold in millimeters. Based on the results of the tests, the approach suggested in this article has a detection accuracy of 99.15% with a detection window of 10, and the accuracy drops to 94.87% as the detection window increases. The missed alarm rate and detection accuracy both drop as the detection window becomes larger because the detection threshold gets higher. At a detection window of 10, the missed alarm rate is 1.83% in Table IV. The missed alarm rate is 6.25 percent when the detection window is 500, and it increases as the window size increases. It should be noted that when the detection window is either too big or too tiny, the false alarm rate increases. For the most part, the DTW algorithm needs more time to match more data, hence an expanding detection window causes the alert delay to grow. Taking everything into account, we find that 50 is the sweet spot for this method's detection window size, which yields a detection accuracy of 98.81%, a false alarm rate of 1.09%, a missed alarm rate of 2.42%, and an alert latency of 2.45 s. 3) Evaluation Against Other CPMS IDS: against prove

that our detection method worked, we compared MPI-CNC against other CPMS ICS models. To be more precise, we contrasted it with popular models for identifying assaults on manufacturing processes, such as those based on digital twins [7], side-channel analysis [4], Long Short-Term Memory [3], LSTM-AE [6] and machining code analysis [8]. In order to fit digital twin models, which are used for intrusion detection, previous processing state data is required. Anomaly classification models may be learned using KCAD and LSTM-AE by collecting audio data produced by processing equipment in manufacturing. Processing video classification anomalies with LTDT analysis. The intrusion detection system that relies on machining code analysis uses support vector machines (SVMs) to train on NC code characteristics for off-line anomalous code classification. We introduced 20 tampering points (such as G02 replacement) into 10 distinct machining situations (machining codes) and conducted 200 machining code injection attacks against them.

TABLE V COMPARISON WITH OTHER CPMS IDS

Attack Type	Machining Code Injection	Parameter Injection	Instruction Injection
Digital Twins[7]	N/A	93.25%	93.25%
KCAD[4]	81.39%	81.39%	N/A
LTDT[3]	95.55%	95.55%	N/A
LSTM-AE[6]	94.79%	94.79%	N/A
Machining Code Analysis[8]	98.38%	N/A	N/A
MPI-CNC	98.81%	100.00%	100.00%

and G03 in conjunction with G01; including the introduction of projections or depressions; and altering the ends to induce deformation. To further evaluate the MPI-CNC's detection capabilities, 100 assaults including parameter injection and command injection were used. The accuracy, miss rate, and false alarm rate of MPI-CNC are determined by collecting detection data and attack logs, much as in.

$$\text{Accuracy} = \frac{TP+TN}{TP+TN+FP+FN} \quad (6)$$

$$\text{Missing Alarm} = \frac{FN}{TP+FN} \quad (7)$$

$$\text{False Alarm} = \frac{FP}{TN+FP} \quad (8)$$

In Table V, we can see that the approaches have successfully used state-of-the-art methodologies to obtain high detection accuracy. When it comes to

detecting assaults that include injecting parameters or instructions, the digital twin-based detection technique achieves an accuracy of 93.25 percent [7]. The KCAD has an accuracy of 81.39% when it comes to detecting attacks that include instruction injection or processing code manipulation [4]. Furthermore, the high-detection accuracies of the LTDT and LSTMAE models are 95.55% [3] and 94.79% [6], respectively. But it's not easy to swiftly adapt to new situations since the side channel data characteristics act differently in various processing contexts. Although it lacks realtime detection capacity during processing, code analysis-based intrusion detection directly examines processing code for fault identification with a detection accuracy of 98.38% [8]. When it comes to assault situations, MPI-CNC is more versatile than the other approaches mentioned. We are able to identify parameter injection attacks and instruction injection attacks with a 100% success rate because we actively detect the critical parameter information during milling and evaluate the critical parameters of the CNC system. The article's technique also generates thorough detection criteria by analyzing the NC code, which leads to a greater detection accuracy (98.81%), comparable to that of the machining code analysis method. Response to inquiry 2: The experimental results show that MPI-CNC is more robust than the conventional historical data-based detection model, and it can handle three different attack strategies. D. RQ3—Low-Interference Experimental Results Moreover, MPI-CNC significantly outperforms other detection algorithms with a 98.81% accuracy rate. Compared to conventional personal computers, CNC systems do not have very good computational capability. So, we have to be careful not to disrupt NC systems' regular functioning using active detection techniques. We use a polling-based, low-interference strategy to get the data presented in this paper.

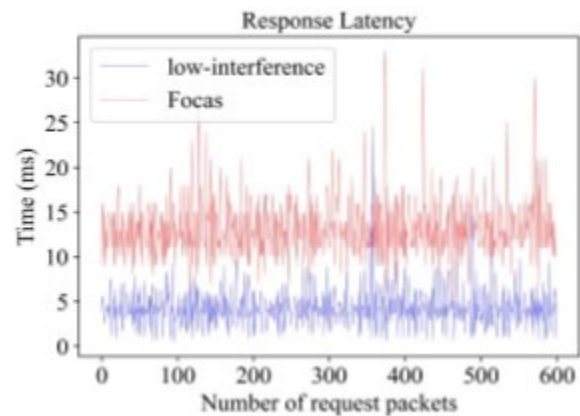


Fig. 7. Comparison of response latency for low interference and Focas.

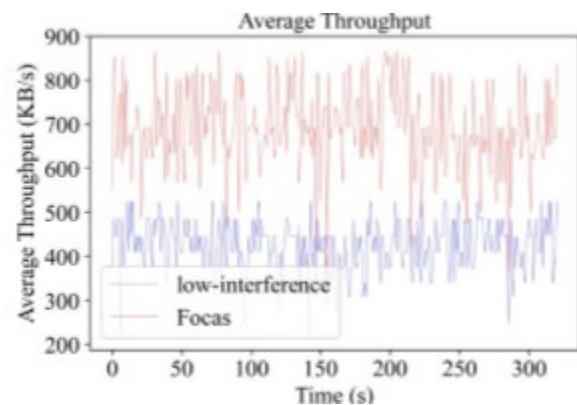


Fig. 8. Comparison of average throughput for low interference and Focas.

remotely identify attack behaviors by processing the data on the machining state of NC systems. Here, we evaluate the FOCAS standard interface for collecting CNC system machining status data against the low-interference collection approach suggested in this article in terms of acquisition latency, network resource use, and effect on machine tool processing times. 1) Latency in Acquiring Data and Network Performance: In Figure 7, it is shown that the typical FOCAS collection technique has an average reaction time of 13.240 ms and a collection frequency of 75.53 times/second. The paper proposes a low-interference collection approach that, when implemented, results in an average reaction time of 4.368 ms and a collection frequency of 228.94 times per second. A 67 percent decrease in reaction time

and a 203.12 percent increase in sample frequency are both achieved. Figure 8 shows that the network throughput dropped by 38.41% when the NC system used the low-interference collection method suggested in this article, compared to when the CNC system used the conventional FOCAS collection method (692.04 KB/s). Results from experiments demonstrate that the article's suggested self-assembled packet method—based on private protocol reverse engineering—reduces disturbance to CNC systems and improves collection frequency when used to gather information on their machining state.

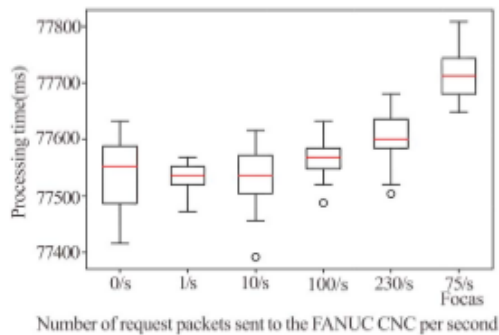


Fig. 9. Influence of network traffic on FANUC CNC.

2) Processing Times Under CPMS IDS: We ran trials with varying request frequencies during active detection to see how machining time changed, to prove that our detection strategy doesn't significantly affect CNC systems. We connected the CNC system directly to the network using Ethernet cables, delivered 1, 10, 100, and 230 low-interference request packets per second, and 75 Focas standard interface request packets per second, to circumvent the forwarding latency of industrial switches. Fig. 9 shows that the CNC system's machining time was about 1 minute and 17.545 seconds under typical circumstances with no outside interference. Our research shows that the machining time of the CNC system is quite sensitive to the amount of external interference it is exposed to. There was no packet loss and the CNC system's machining time increased by only 0.076% while delivering 230 packets per second. This means that the FANUC CNC system can continue to function normally while processing at least 230 packets per second. The CNC system's machining time rose by 0.217 percent while employing Focas standard interface request packets at a maximum rate of 75 requests per second; this

type of low-interference data collecting was far more effective. In addition, there is a 75 ms limit for the FANUC system. Our detection technique has a negligible effect on CNC machining times, according to the testing findings. Reverse engineering specialized acquisition protocols and tailored packet acquisition significantly enhance acquisition frequency and efficiency with MPI-CNC, which in turn reduces CNC network resource utilization and increases machining time by just 0.076% (RQ3). Standard machining is unaffected by MPI-CNC.

VI. RELATED WORK

In addition to endangering human lives, cyberattacks on CPMS have a direct impact on production efficiency. So, IDS studies in CPMS have recently exploded in popularity among scholars. The primary goal of CPMS intrusion detection systems is to prevent cyberattacks that target CPMS with the intention of influencing or interfering with the manufacturing process. Through the examination of study findings in this We have divided the most recent CPMS intrusion detection systems (IDS) into three groups according on the methodologies used for detection. Digital Twin-Based CPMS IDS: By analyzing past data, digital twin technology creates a control model that mimics real-world operations [32]. Balta et al. [7] built controller digital twin models for the CNC system of a 3D printer by collecting and analyzing previous machining data from the device. They determined that cyber-attacks had altered the temperature settings of the 3-D printer's nozzle heaters by comparing the controller digital twin model's simulated machining state with the real machining state. Manufacturing equipment produces a mountain of measurement channel data when cutting, which may provide an indirect reflection of the machining condition; this data is used to build CPMS IDS systems that rely on side-channel analysis. In the CPMS IDS field, side-channel analysis detection approaches are very prevalent. To train detection models for identifying machining path manipulation assaults, Chhetri et al. [4] first suggested using auditory data around production equipment. To ensure product consistency, Bayens et al. [33] analyzed acoustic aspects of the machining equipment, features of the machining site, and features of production waste. By comparing the 3-D printer stepper motors' audio characteristics to audio fingerprints, Belikovetsky et al. [5] were able to identify the printing process. Using video stream

analysis, Mamun et al. [3] were able to identify changes in the processing trajectory of 3-D printers. Yoginath et al. [2] used the Bayesian model to identify creep assaults by analyzing the current values of 3-D printer power lines. Using the LSTM-autoencoder technique, Shi et al. [6] retrieved features from vibration sensors' side-channel data. They then used the OCSVM classification algorithm to identify anomalies. The CPMS IDS system According to the Analysis of Machining Codes: As a global standard for NC programming languages, the ISO-6983-1 [29] standard was established. The NC code, which consists of G codes and M codes, has its syntax and lexical rules defined by this standard. A CNC system will automatically regulate the machining process in accordance with the instructions in the NC code, which provides the most thorough and exhaustive control information of the machining process. It is possible to successfully identify outliers by examining the NC code. A variety of statistical characteristics were retrieved from NC code by Beckwith et al. [8]. These features included the frequency of XYZ values, the quantity of G codes and M codes, and so on. In order to detect outliers, they performed an optical examination of NC code and constructed a machine learning anomaly classification model. In order to create 3-dimensional models, Tsoutsos et al. [34] simulated pressure tests using NC code that had been reverse-engineered. While doing so, they found security holes in the NC code.

VII. DISCUSSION

These are some of the drawbacks of the article's suggested intrusion detection approach. 1) Attacks that use firmware manipulation of the CNC system to pose as a man-in-the-middle might be challenging for the approach to handle. This article proposes an intrusion detection approach that is successful against these assaults, although they need highly sophisticated attackers. 2) CNC machines with two or three axes may benefit from the procedure, but 5-axis machine centers may be out of its league when it comes to generating detection rules. 3) CNC systems that include authentication procedures for their interfaces would not be able to use the active detection method suggested in this article. To be clear, most CNC systems do not limit remote access to machining status information at this time. 4) CNCs from many manufacturers may be acquired via low-interference ways, but reversing protocols requires a high level of technical expertise. It is important to

tackle the significant job of automated protocol reversal.

VIII. CONCLUSION

An innovative method for CPMS cyber-attack runtime detection, MPI-CNC, is presented in this research. We use the FANUC CNC machine tools to demonstrate how to create a prototype system. In instance, MPI-CNC does automated analysis of NC programs, derives invariants from the machining process, and creates rules for attack detection, which include rules for the machining process and rules for critical parameters. After then, MPI-CNC interacts with the CNC system using low-interference request packets to gather process status and critical parameters, and it sets detection windows and thresholds to identify malicious actions. Finally, we test MPI-CNC with a FANUC CNC machine to see how it performs in actual milling situations. By successfully detecting a wide range of cyber-attacks without interfering with the CNC system's regular operation, experimental findings show that MPI-CNC has outstanding adaptability performance. Our method has better detection and adaptation performance when compared to existing top-tier detection algorithms.

REFERENCES

- [1] A. Kusiak, "Smart manufacturing," *Int. J. Prod. Res.*, vol. 56, nos. 1-2, pp. 508–517, 2018.
- [2] S. Yoginath et al., "Stealthy Cyber anomaly detection on large noisy multi-material 3-D printer datasets using probabilistic models," in *Proc. ACM CCS Workshop Addit. Manuf. Secur.*, New York, NY, USA, 2022, pp. 25–38.
- [3] A. A. Mamun, C. Liu, C. Kan, and W. Tian, "Securing cyber-physical additive manufacturing systems by in-situ process authentication using streamline video analysis," *J. Manuf. Syst.*, vol. 62, pp. 429–440, Jan. 2022.
- [5] S. R. Chhetri, A. Canedo, and M. A. Al Faruque, "KCAD: Kinetic Cyber-attack detection method for cyber-physical additive manufacturing systems," in

Proc. IEEE/ACM Int. Conf. Comput. Aided Design (ICCAD), 2016, pp. 1–8.

[6] S. Belikovetsky, Y. A. Solewicz, M. Yampolskiy, J. Toh, and Y. Elovici, “Digital audio signature for 3-D printing integrity,” *IEEE Trans. Inf. Forensics Security*, vol. 14, pp. 1127–1141, 2019.

[6] Z. Shi, A. A. Mamun, C. Kan, W. Tian, and C. Liu, “An LSTMautoencoder based online side channel monitoring approach for cyber–physical attack detection in additive manufacturing,” *J. Intell. Manuf.*, vol. 34, no. 4, pp. 1815–1831, Apr. 2023.

[7] E. C. Balta, M. Pease, J. Moyne, K. Barton, and D. M. Tilbury, “Digital twin-based cyber-attack detection framework for cyber–physical manufacturing systems,” *IEEE Trans. Autom. Sci. Eng.*, early access, May 25, 2023, doi: 10.1109/TASE.2023.3243147.

[8] C. Beckwith et al., “Needle in a haystack: Detecting subtle malicious edits to additive manufacturing G-code files,” *IEEE Embed. Syst. Lett.*, vol. 14, no. 3, pp. 111–114, Sep. 2022. [9] H. Pearce, K. Yanamandra, N. Gupta, and R. Karri, “FLAW3D: A trojan-based cyber attack on the physical outcomes of additive manufacturing,” *IEEE/ASME Trans. Mechatron.*, vol. 27, no. 6, pp. 5361–5370, Dec. 2022. [10] M. Yampolskiy, L. Graves, J. Gatlin, J. T. McDonald, and M. Yung, “Crypto-steganographic validity for additive manufacturing (3D printing) design files,” in *Proc. Int. Conf. Inf. Secur.*, 2022, pp. 40–52. [11] T. Le et al., “Physical logic bombs in 3-D printers via emerging 4-D techniques,” in *Proc. 37th Annu. Comput. Security Appl. Conf.*, New York, NY, USA, 2021, pp. 732–747. [12] M. McCormack, S. Chandrasekaran, G. Liu, T. Yu, S. DeVincent Wolf, and V. Sekar, “Security analysis of networked 3-D printers,” in *Proc. IEEE Security Privacy Workshops (SPW)*, 2020, pp. 118–125. [13] B. Shadow. “Industrial security exploitation framework.” 2020. [Online]. Available: <https://github.com/w3h/isf> [14] R. R. Maiti, C. H. Yoong, V. R. Palleti, A. Silva, and C. M. Poskitt, “Mitigating adversarial attacks on data-driven invariant checkers for cyber–physical

systems,” *IEEE Trans. Depend. Secure Comput.*, vol. 20, no. 4, pp. 3378–3391, Jul./Aug. 2023. [15] J. Liu et al., “ShadowPLCs: A novel scheme for remote detection of industrial process control attacks,” *IEEE Trans. Depend. Secure Comput.*, vol. 19, no. 3, pp. 2054–2069, Jun. 2022. [16] P. Mahesh et al., “A survey of cybersecurity of digital manufacturing,” *Proc. IEEE*, vol. 109, no. 4, pp. 495–516, Apr. 2021. [17] Y. Pan et al., “Taxonomies for reasoning about cyber–physical attacks in IoT-based manufacturing systems,” *Int. J. Interact. Multimedia Artif. Intell.*, vol. 4, no. 3, pp. 45–54, Jul. 2017. [18] T. Lin et al., “Susceptibility to spear-phishing emails: Effects of Internet user demographics and email content,” *ACM Trans. Comput. Human Interact.*, vol. 26, no. 5, pp. 1–28, Jul. 2019. [19] N. Karsten and L. Jakob, “BadUSB—On accessories that turn evil,” presented at Blackhat Conf., Las Vegas, NV, USA, 2014, pp. 1–28. [20] M. Conti, N. Dragoni, and V. Lesyk, “A survey of man in the middle attacks,” *IEEE Commun. Surveys Tuts.*, vol. 18, no. 3, pp. 2027–2051, Mar. 2016. [21] S. B. Moore, W. B. Glisson, and M. Yampolskiy, “Implications of malicious 3-D printer firmware,” in *Proc. 50th Hawaii Int. Conf. Syst. Sci.*, 2017, pp. 1–10. [22] Y. Liu, Y. Peng, B. Wang, S. Yao, and Z. Liu, “Review on cyber–physical systems,” *IEEE/CAA J. Automatica Sinica*, vol. 4, no. 1, pp. 27–40, Jan. 2017. [23] G. Y. Dayanikli, S. Sinha, D. Muniraj, R. M. Gerdes, M. Farhood, and M. Mina, “Physical-layer attacks against pulse width modulation controlled actuators,” in *Proc. 31st USENIX Secur. Symp.*, Boston, MA, USA, 2022, pp. 953–970. [24] J. Gatlin et al., “Encryption is futile: Reconstructing 3D-printed models using the power side-channel,” in *Proc. 24th Int. Symp. Res. Attacks, Intrusions Defenses*, New York, NY, USA, 2021, pp. 135–147. [25] L. J. Wells, J. A. Camelio, C. B. Williams, and J. White, “Cyber–physical security challenges in manufacturing systems,” *Manuf. Lett.*, vol. 2, no. 2, pp. 74–77, 2014. [26] T. Zinner, G. Parker, N. Shamsaei, W. King, and M. Yampolskiy, “Spooky manufacturing: Probabilistic sabotage attack in metal AM using shielding gas flow control,” in *Proc. ACM CCS Workshop Additive Manuf. (3D Printing) Security*, New York, NY, USA, 2022, pp. 15–24. [27] S. R.

Chhetri, A. Barua, S. Faezi, F. Regazzoni, A. Canedo, and M. A. Al Faruque, “Tool of spies: Leaking your IP by altering the 3-D printer compiler,” *IEEE Trans. Depend. Secure Comput.*, vol. 18, no. 2, pp. 667–678, Apr. 2021. [28] H. Choi et al., “Detecting attacks against robotic vehicles: A control invariant approach,” in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, New York, NY, USA, 2018, pp. 801–816. [29] *Automation Systems and Integration—Numerical Control of Machines—Program Format and Definitions of Address Words—Part-1: Data Format for Positioning, Line Motion and Contouring Control Systems*, International Organization for Standardization, ISO Standard 69831:2009, 2009. [30] H. Sakoe and S. Chiba, “Dynamic programming algorithm optimization for spoken word recognition,” *IEEE Trans. Acoust., Speech, Signal Process.*, vol. 26, no. 1, pp. 43–49, Feb. 1978.

[31] E. Fix and J. L. Hodges, “Discriminatory analysis. nonparametric discrimination: Consistency properties,” *Int. Statist. Rev./Revue Internationale de Statistique*, vol. 57, no. 3, pp. 238–247, 1989. [32] R. Minerva, G. M. Lee, and N. Crespi, “Digital twin in the IoT context: A survey on technical features, scenarios, and architectural models,” *Proc. IEEE*, vol. 108, no. 10, pp. 1785–1824, Oct. 2020. [33] C. Bayens, T. Le, L. Garcia, R. Beyah, M. Javanmard, and S. Zonouz, “See no evil, hear no evil, feel no evil, print no evil? malicious fill patterns detection in additive manufacturing,” in *Proc. 26th USENIX Secur. Symp.*, Vancouver, BC, Canada, 2017, pp. 1181–1198. [34] N. G. Tsoutsos, H. Gamil, and M. Maniatakis, “Secure 3-D printing: Reconstructing and validating solid geometries using Toolpath reverse engineering,” in *Proc. 3rd ACM Workshop Cyber Phys. Syst. Secur.*, New York, NY, USA, 2017, pp. 15–20.