Real-Time Intrusion Detection of Insider Threats in Industrial Control System Workstations Through File Integrity Monitoring

Bakil Al-Muntaser[1], Mohamad Afendee Mohamed[2], Ammar Yaseen Tuama[3]
Faculty of Informatics and Computing, Universiti Sultan Zainal Abidin, Kuala Terengganu, Malaysia[1, 2]
College of Computer Science and Information Technology, University of Kirkuk, Iraq[3]

***Abstract*—Industrial control systems (ICS) play a crucial role in various industries and ensuring their security is paramount for maintaining process continuity and reliability. In ICS, the most damaging cyber-attacks often come from trusted insiders rather than external threats or malware. Insiders have the advantage of bypassing security measures and staying undetected. This research focuses on developing a real-time intrusion detection system for ICS workstations that effectively detects insider threats while prioritizing user privacy. The approach employs file integrity monitoring to identify suspicious activities, particularly file violations such as data tampering and destruction. The model presented in this research demonstrates low system resource consumption by utilizing an event-triggered approach instead of continuous polling of file data. The model leverages built-in operating system functions, eliminating the need for third-party software installation. To minimize disruptions to the ICS network, the model operates at the supervisory level within the ICS architecture. Through extensive testing, the model achieves a high level of accuracy, detecting insider intrusions with a high true positive rate. This reliable detection capability contributes to enhancing the security of ICS and mitigating the risks associated with insider threats. By implementing this real-time intrusion detection system, organizations can effectively protect their control systems while preserving user privacy.**

***Keywords*—Industrial control system; insider threats; intrusion detection; file integrity monitoring; SCADA security**

I. Introduction

The industrial control system is very important for various industries in our life. As with any other system, security is now a top priority, and it must be achieved without affecting process continuity and reliability. ICS faces different kinds of threats from outside the system as well as threats from insiders. The security threat from within can be even more powerful than many external attacks [1], [2]. This is particularly the case with ICS networks, which manage critical infrastructure and manufacturing plants. A smart, motivated, perhaps disgruntled employee or ex-employee with knowledge of a plant and access to the network, can cause a variety of disruptions that may result in information breaches, financial losses, equipment damages, and even threaten human lives [3]. Industrial control systems should be very secure to ensure plant resource availability and integrity without any disturbance or production loss. The system should imply a very secure means of protection including fast detection of insider intrusion, to avoid exploitation getting even worse.

An insider threat occurs when someone close to a company with authorized access misuses that access to harm the company's key information or systems [1]. This individual does not have to be an employee; third-party vendors, contractors, and partners may represent a threat as well. privileged employees, such as IT team members, superusers, knowledge workers, resigned or dismissed people, and managers are all possibilities to be insiders [4].

Several recent security surveys report a high increase in insider threats. In the report [1] more than half of organizations have experienced an insider threat in the last year. Based on the survey, 74% of insider attacks have become more frequent over the last 12 months. In the report of [5], about 4700 reported attacks were subjected to analysis and it showed that 23% of attacks were attributed to malicious insiders while 63% were attributed to employee and contractor negligence. In [4] survey report The negligent insider is the root cause of most incidents that come from insider threats. According to the Kaspersky 2019 Status of industrial cybersecurity study, staff mistakes and accidental activities were responsible for 52 percent of incidents affecting operational technology (OT) and industrial control system (ICS) networks in 2018 [5]. Sometimes, unaware or negligent employees can unintentionally cause security breaches without knowledge of doing so [1]. According to a recent Ponemon Institute (2022) research sponsored by ObserveIT and IBM Insider, risks have increased in the last two years [4]. As a result, enhancing the approaches of early identification of any source of insider attack requires more security controls and solutions.

Network intrusion detection, Firewall, and antivirus systems have been shown to be ineffective to detect attacks coming from insiders. Large security operations centers have started to implement endpoint-based sensors that give their organizations broader visibility into low-level occurrences [6]. Therefore, employing techniques and tools specifically designed to address the threat of insiders will be more effective. Furthermore, these tools should consider the phenomena and requirements of the industrial system.

In the field of (ICS), applying traditional IT countermeasures and solutions blindly is not recommended due to several differences between the two environments. While IT systems prioritize confidentiality, ICS focuses on availability

as its major priority. Additionally, the expected reaction time in ICS should be below a millisecond, compared to a few seconds in IT systems [7]. Outages in ICS can have severe consequences, including production stoppage and financial losses. In ICS, security will include ensuring safety for company assets and personnel as well as the environment [8].

II. Literature Review and Related Works

Industrial control system (ICS) is a term used to describe different control systems and related instrumentation, including the devices, networks, and controls used to operate and automate industrial manufacturing processes [7]. It includes a distributed control system (DCS), supervisory control and data acquisition system (SCADA), Safety Instrumented Systems (SIS), Emergency Shutdown Systems (ESD), and programmable logic controllers (PLC) [9], [7]. They are core parts of every technical infrastructure globally, ranging from the small controller used in air conditioning in our cars and homes to the extensive control networks used in factories [10]. These factories include power production and distribution, oil and gas, chemical production, water distribution systems, and even nuclear plants[3]. These systems help control and automate industrial operations, providing remote monitoring and recording for different data and parameters on the field side.

A typical IACS design seen in any modern facility might have a DCS as the primary control system, with interfaces to additional systems such as PLCs and SCADA System. The plant data is sent to a central control room to be used for monitoring and controlling purposes [9]. These data are saved on a workstation known as the Historian to help the panel operator view historical trends. Panel operator workstations are used by plant operators to monitor the process. Besides operator workstations, Engineering Workstations are used to configure the DCS controllers and maybe the subsystems such as PLCs, as well as the associated systems such as SIS [11].

The architecture of ICS is hierarchical and has several operational levels as in Fig. 1 including Process Level, Basic Control Level, Supervisory Control Level, Process Management, and Corporate Network Levels by ISA 99 (control system automation security and safety standard) [8]. To guarantee greater security, security measures should be implemented at every level. The intrusion might have taken place at any level of this hierarchy or on multiple levels at once. Understanding the components of these levels enables more professional security measurement implementation.

The first level of the architecture is called Process Level, and it interfaces with the physical process through actuators and sensors instrumentation. The second level, known as Basic Control Level, is where the system's overall control takes place. The primary goal of the basic control level is to use controllers to regulate the physical process that interfaces with instrumentation components [8]. The supervisory level is the following level in the hierarchy. This level is in charge of interacting and gathering data from the process and control levels so that the operator workstations can monitor and view the control state and field reading of the process [8]. In this level, engineering workstations also exist which are used to access controllers setting and programs. Supervisory level workstations have a good supply of memory and processors in comparison with two lower levels, which make them more suited to deploy a security intrusion detector than the previous two levels.

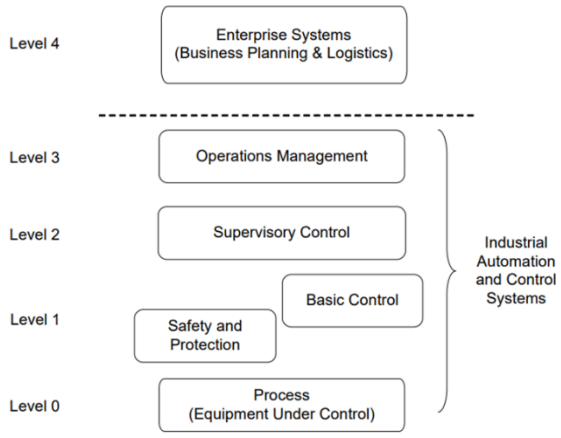

Fig. 1. Reference model for ISA99 standards.

Source: [8]

There is a lack of research and studies done on the field of ICS insiders, because of the gap between IT security researchers and operating technology [8]. In this part, we discuss the research which discusses insider intrusion detection solutions in IT system and ICS system as well. In the last few years, many ICS-oriented intrusion detection approaches have been explored in the research community. Several ICS-oriented IDS taxonomies with various categorizations are found in the literature. Some approaches attempt to detect insider threats by using machine and deep learning approaches and other approaches which depend on monitoring internal logs and system commands.

Some research on insider threat detection shows that potential insider threats can be proactively identified through psychological changes and language habits before the execution of malicious activities. These researchers argue that variations in an individual's behavior and communication patterns can serve as early warning signs of potential threats. The studies that focus on psychological changes and language habits are primarily designed to identify potential malicious insiders such as traitors. They can be very effective at detecting individuals who might be planning to intentionally harm an organization [12]. However, when it comes to negligence or unintentional insiders, these psychological and linguistic-based detection methods may not be as effective.

Another research discusses insider intrusion detection through user behavior monitoring. By monitoring user activities during using the system like login and logoff beside other activities during doing normal task and then deploying these data by using deep leaning to detect insider [13]. Keystroke and mouse dynamics have a lot of information about how a user operates the host [14] . Because this type of data source contains information that can be used to identify legitimate users using behavioral biometrics, it is best suited for detecting masqueraders [15]. Although such a log may be

verbose at times due to a significant number of duplicated entries, the information contained within it may still be worth investigating. A particularly extended sequence of authentication failures, for example, may indicate a brute-force password attack [15].

In the paper [16], the authors propose a method that combines deep learning techniques with mouse biobehavioral features to detect insider threats. This approach achieves both rapid user authentication and high accuracy. The study focuses on five fundamental mouse actions: click, move, drag, stay, and scroll. These actions, along with authentication events, are used to extract features that capture the user's unique behavioral characteristics. Support Vector Machine (SVM) classification methods are then employed to categorize and analyze these features, enabling effective insider threat detection.

As per the survey of 2023 insider threats most businesses possess end-to-end user activity monitoring, which includes both access logging and automatic user behavior monitoring and is even more alarming. One big obstacle to this technology is that it violates user privacy and does not comply with the EU's general data protection law (GDPR), which leads to many businesses avoiding utilizing it [1]. The survey shows also that cybersecurity units within organizations are intensifying their employment of User Behavior Analytics (UBA) tools. These sophisticated instruments are utilized to identify, categorize, and raise alerts in response to aberrant behavior. An impressive 86% of organizations are reported to actively observe user behavior via one methodology or another [1].

Another research discusses using audit log windows detection of fraudulent behavior. The research suggests that Windows audit logs provide sufficient information to enable the reliable detection of fraudulent behavior while generating manageable data streams on endpoints. Audit logs offer a cost-effective and efficient alternative to more expensive breach detection systems that rely on agent-based approaches [6]. Another research explores the utilization of Multi-Source Logs for detecting insider behaviors. The security logs are converted into text format and compiled as a corpus. By training a model using Word2vec with the corpus, the researchers were able to approximate the posterior probabilities associated with insider behaviors. The proposed approach proves to be effective and scalable for practical applications in insider threat detection [17]. The problem with this method is detecting malware incidents that will happen after they have bypassed perimeter security layers. Audit logs, like any other endpoint software, can be modified or destroyed once malware has administrative access to the target endpoint. This issue is reduced in part because most enterprise setups store audit logs on a remote server, and the audit logs' integrity can be verified using cryptographic protocols [18].

Another research discusses creating a USB-Watch which is a Generalized Hardware Assisted Insider Threat Detection. The statement suggests that the framework utilizes hardware to capture real-time USB traffic, enabling the collection of data before advanced attackers have the opportunity to tamper with it within a compromised operating system. Additionally, the framework employs a decision tree anomaly detection classifier, which is implemented in the hardware itself. This classifier analyses the behavioral patterns of connected USB devices, allowing for the detection of anomalous behavior [19]. However, this method is not effective with insider threats which do not need USB connections.

A new research focuses on the proactive detection of insider threats through the application of graph learning and psychological context [20]. The MEWRGNN utilizes graph neural networks to capture the contextual relationship of user behaviors and achieve accurate anomaly identification. It ranks the contribution of different edge-representation features, providing interpretability and understandable insights for security analysts. Experimental results show that the MEWRGNN can learn from limited sample data sets and achieve quick and accurate insider threat detection [20].

There are many researches which try to implement machine learning in intrusion detection such as in [21], [22]. With respect to ICS, studies based on machine learning normally faced the difficulty of finding a real dataset from an industrial control system. The industrial company always tries to reveal any data related to industrial control as a kind of security [23]. In our research, our objective was to create a model that does not rely on machine learning but rather utilizes signature-based anomaly detection, primarily functioning at the host side. This model is designed to operate independently of the operating system security logs file. Our focus is specifically on detecting unauthorized changes that occur in the monitored files.

In our model, we will try to utilize an important point related to industrial workstations which is: these workstations will be provided by a system vendor with the required installed software. The vendor will ensure its security configuration and hardness. It is not allowed to install new software or change configuration filled by any unauthorized person [24]. Many application software activities depend on the configuration which is stored on known paths and directories. Any change in working application configuration will affect the working of related services and could have a bad effect on some process working set points or on the panel operators monitoring screens. Determining the underlying cause of the intrusion or any file system problem is critical, but manual analysis extends the time it takes to resolve threats [25]. Finding a model which can detect any changes for these files on any one of the operator and engineer workstations will help to detect if there are some suspicious activities from an insider.

There are Many Files Integrity Monitoring (FIM) tools available today developed by private companies in response to these industry needs. Some examples include Tripwire [26], Samhain [27], and OSSEC [28]. FIM tools can help in detecting file integrity intrusions in monitored host computers [29]. These tools continuously monitor the file system for changes, including changes to file attributes, content, and timestamps. They can generate alerts when unauthorized or unexpected modifications occur [30], [26].

Monitoring file integrity in ICS environments can be challenging due to the unique requirements and constraints of these systems. For example, ICS often use specialized hardware and software that may not be compatible with standard FIM tools. Additionally, ICS typically have strict

performance requirements, making it important to minimize the performance impact of monitoring activities [31]. FIM solution has its own drawbacks and issues such as Delays in detection, and Complex deployment. Current FITs work off-line pattern which means these tools will monitor the files at scheduled times to check the integrity of the system. Delay in detection is the biggest issue because this will create an opportunity for the intruder to take advantage of the system [30].

One of the main challenges when building a FIM is real-time detection, which should have minimal performance overhead to ensure it can be used effectively in real production environments. High-performance overhead can cause system slowdowns, reduced productivity, and user frustration, which could lead to the tool being disabled or ignored [30]. Implementing a FIM in an ICS poses specific challenges due to the constraints of computing resources and the need for continuous system operation. The installation of third-party software may also introduce compatibility issues and potentially disrupt the system's functionality. Therefore, it is crucial to carefully evaluate and minimize any adverse effects on system performance during the implementation of any intrusion detection tools.

A good File Integrity Tracking (FIT) tool should collect comprehensive information about file changes, enabling administrators to make informed decisions and use the tool effectively. Detailed information can help identify the root cause of an issue, track unauthorized changes, and mitigate potential security risks. In the process of selecting files for monitoring, priority should be given to those that are integral to your system and applications. Emphasis should be placed on files that are not anticipated to undergo changes without premeditated scheduling. Opting to monitor files that are frequently altered by applications or the operating system, such as log files and text files, could generate a surplus of data, subsequently obscuring the identification of a potential attack [32].

III. Research Methodology

We propose an intrusion detection approach that centers around the automated identification of unauthorized alterations in Monitored File lists. These modifications may occur intentionally or unintentionally, involving employees or contractors, and can stem from both legitimate and malicious intents. Whether resulting from human actions or software interventions, any such changes are regarded as potential security concerns. To address this, our model generates alerts to promptly designated personnel to review and assess these alterations for suspicious activity.

The supervisory level In ICS architecture is the best place to implement the Intrusion detection model. This is the primary level of interest in our research. At this level, operational operators and engineers interact with physical system programming and configuration. This study aims to improve the detection of intrusions at the level of operator and engineering workstations. The other higher layers are often managed by IT and should be kept separate from the ICS system. This isolation is already accomplished with a hardware firewall and DMZ, or with a data diode.

At the supervisory level, the operators' and engineers' workstations and servers existed [8]. These workstations are equipped with a group of certain software and tools which are already provided by the system vendors. A limited number of services should be working with a minimum number of listening open ports [8]. Many working software depends on configuration files to define their working environment as well as their responses. In addition to that, many system engineers depend on written text file scripts to automate daily tasks with the help of operating system task schedules. Improving a technique to monitor the integrity of these files and services and providing a mechanism to notify the right people in case of suspicious changes, and providing data to help cybersecurity teams take the best possible course of action is very important. Such tools will contribute to enhancing and maximizing the ability to stop incidents from occurring or getting worse. The data collected by this model will provide a good data source for other troubleshooting activities in case of a security incident happening, because it will save a history record for the monitored file and the changes that happened to them.

The following flowchart in Fig. 2 describes how the philosophy of the model works to detect any malicious activities on monitored files. The initial phase of our methodology involves establishing a baseline by creating a comprehensive list of crucial files that need to be monitored for potential malicious alterations in the targeted industrial workstations. This process requires identifying the critical systems and software applications operating on these workstations, as compromising them could have significant operational consequences. When selecting which files to monitor, it is important to consider the files that are vital for the proper functioning of the system and applications. These are the files that you expect to remain unchanged unless planned modifications are made. On the other hand, monitoring files that are frequently changed by applications or the operating system, such as log files, can introduce unnecessary noise and make it more challenging to detect an actual attack.

When selecting files related to industrial control systems, it is important to include operational and configuration files of the main component such as SCADA software, PLC programming tools, and other specialized applications. To effectively understand the critical components of the system and the primary software, a thorough examination of the OT system vendor's documentation is necessary. This documentation provides insights into the system's architecture and the essential software components, including vital configuration files, and data files that are crucial for the system's proper functionality. To gain further insights, engaging with system administrators, operators, and other experts within the organization is highly valuable. These individuals possess specialized knowledge and experience with the system, allowing them to provide valuable input on the most crucial files that should be included in the monitored list.

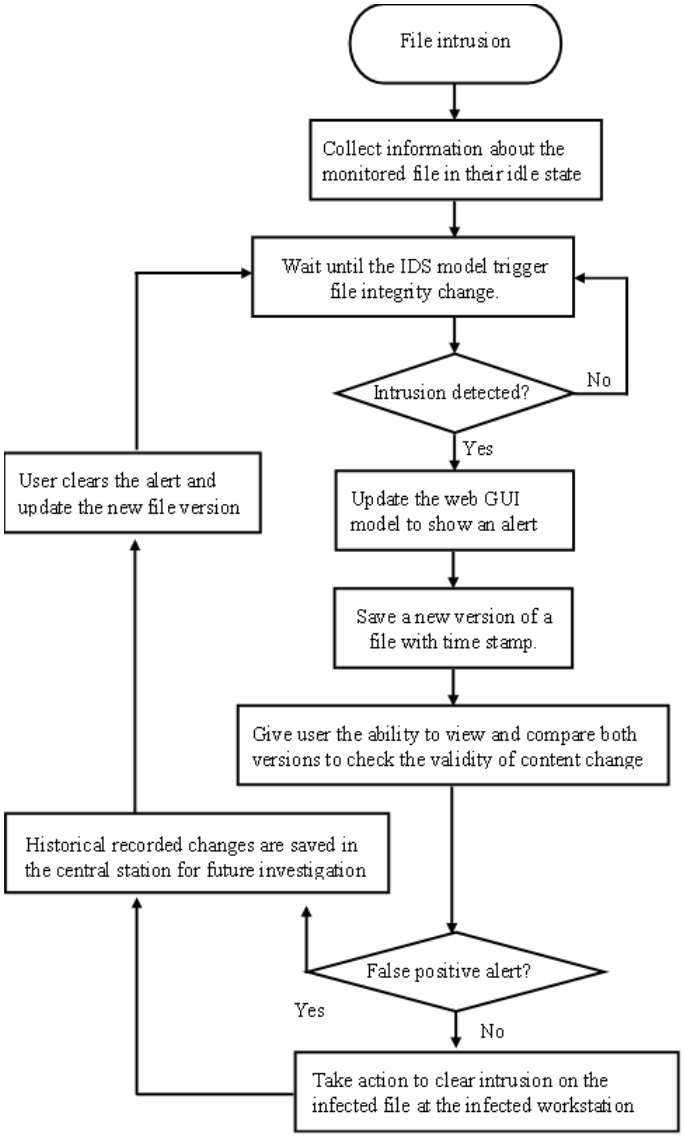

Fig. 2. Flowchart describes the methodology.

Various types of data can be collected for files to monitor their integrity and detect potential malicious activities. Some of these data attributes encompass file metadata such as filename, file path, file size, file extension, file permissions, file owner, file attributes, file content hash, and file timestamps. While the date-modified timestamp is a valuable data attribute for detecting file changes, its reliability can be compromised due to timestamp tampering. Timestamps can be effortlessly manipulated or spoofed, enabling an attacker to alter the timestamp following modifications to conceal their activities. Consequently, timestamps alone become an unreliable source for detecting unauthorized changes. To address this issue, file content hashes can be employed in conjunction with timestamps. A file content hash is a unique, fixed-length string generated from a file's contents using a cryptographic hash function. This hash functions as a digital fingerprint of the file, facilitating the effortless identification and verification of the file's contents. Cryptographic hash functions, such as SHA-256, SHA-1, or MD5, accept an input (in this instance, the file contents) and generate a fixed-length hash value. Moreover, a file content snapshot of the monitored value can be stored in a secure location, which can be utilized in the event of intrusion detection to revert to a healthy condition. A file content snapshot is a copy or representation of the file's content at a specific moment in time, which can be employed for comparison with subsequent versions of the file.

The focus of the model developed in this research is on insider attacks involving file content changes or file deletions within industrial control systems (ICS). These attacks encompass several categories, including data tampering, data destruction, unauthorized access with modification, malicious code injection, accidental data modification, negligence, and accidental file deletion. Data tampering refers to the deliberate alteration of critical ICS configuration files, program files, or databases by insiders with the intent of manipulating process data or creating falsified records. Data destruction involves insiders intentionally deleting or corrupting critical ICS files or backups. Unauthorized access with modification occurs when insiders misuse their legitimate access credentials to gain unauthorized entry to sensitive ICS files or systems and then make unauthorized changes to the content. Malicious code injection involves insiders introducing malicious code or scripts into ICS files where this malicious code is designed to compromise the system, extract sensitive process data, or disrupt industrial operations. Accidental data modification can occur when insiders inadvertently modify the content of ICS files due to human error or a misunderstanding of the system or processes; these unintentional modifications can have adverse effects on process control, safety systems, or data integrity. Accidental file deletion refers to instances where insiders unintentionally delete important ICS files, resulting in data loss or disruptions in industrial processes. By focusing on these various types of insider attacks involving file content changes or file deletions, the model aims to enhance the detection and mitigation of such threats within industrial control systems, thereby improving overall system security and integrity.

To ascertain the efficacy of our model, we conducted tests in an environment that mirrored the workgroup network for workstations at the supervisory level. This was achieved either by creating a small, real local network or constructing a virtual network using VMware Workstation virtual machines. In this research, we utilized a group of workstations operating on the Windows operating system.

For the real-time testing of the intrusion detection system model in a Windows OS environment, we employed the Register-ObjectEvent cmdlet in PowerShell. The Register-ObjectEvent cmdlet is designed to monitor events on .NET objects with minimal system resource usage. It employs an event-driven model, triggering an action when a specified event transpires, as opposed to continuous polling for changes, which can be more resource-intensive. Despite the comparatively low resource usage of the Register-ObjectEvent cmdlet, especially when contrasted with continuous polling methods, it is crucial to manage event subscriptions meticulously and unregister them when no longer required, to avert unnecessary resource consumption.

To illustrate the operation of our model, we predefined a list of files requiring monitoring. Initially, the model collects data such as the last modified time and file content hash value of these files, storing this information in a remote station. This data serves to confirm file integrity violations in case of intrusion detection and is preserved for subsequent incident investigation. To gauge the model's effectiveness, we devised a comprehensive list of 100 potential scenarios that could compromise file integrity and observed the model's capacity for intrusion detection. These scenarios encompass all conceivable insider threats, whether originating from employees or contractors, and whether they are intentional or unintentional. Concurrently, we recorded the model's response to these scenarios to measure its performance. In the event of intrusion detection, the data is transmitted to a remote station that maintains a database to record file information and the history of recorded attacks. This station also manages the display of alarms in a Graphical User Interface (GUI) form. This GUI can be accessed from any network station using a standard web browser.

IV. Result and Discussion

After conducting 100 simulated insider intrusion scenarios targeting file integrity violations, the model demonstrated a high true positive score, as depicted in Table I. These results indicate its high sensitivity in detecting this type of intrusion. This outcome aligns with expectations within the context of industrial control system workstations, where configuration files should be modified under a management of change (MoC) process, thereby minimizing the risks associated with these changes. Any modification outside of the MoC will be detected.

On occasion, the model might flag false positives. However, this is an expected occurrence, as these false alarms may be generated during an authorized modification of any of the monitored files. In the context of industrial control systems, alterations to any configuration file or settings are ideally conducted within a structured (MoC) process. Consequently, the occurrence of such false alarms is anticipated. Moreover, the model possesses the ability to enter a state of inhibition during approved changes, thus preventing the triggering of unnecessary alarms. Significantly, the model reported a rarely false-negative rate. This result is consistent with the operational characteristics of the model, which is designed to raise an alarm whenever there is a change or deletion in the file content. In all other instances, the model remains inactive, thus ensuring no false negatives are reported.

By analyzing the results of the model test, it can be observed from the data presented in Table I that the model demonstrates a notable level of efficiency and precision when it comes to detecting alterations in file content. The percentage of true positive detections is significantly high, indicating the model's ability to accurately identify instances of insider intrusion in real-time. However, it is important to note that there were occasional occurrences of false positives during the testing phase. Despite this, the model's overall performance remains strong, highlighting its capability to effectively identify unauthorized changes in file content and mitigate the risk of insider threats. This successful detection and prompt response contribute to enhancing the overall security of the system.

TABLE I. Intrusion Detection Performance

Scenario	Actual Intrusion	Model Detection	Result
Scenario 1	Yes	Yes	True Positive
Scenario 2	Yes	Yes	True Positive
Scenario 3	Yes	Yes	True Positive
…..	Yes	Yes	True Positive
Scenario 99	Yes	Yes	True Positive
Scenario 100	No	Yes	False Positive

By focusing on file integrity monitoring, the model avoids the need to scrutinize user behavior, offering a distinct advantage for organizations that prioritize privacy. This approach ensures that employees' personal habits and actions remain confidential while still effectively safeguarding the system. This respect for privacy, combined with robust intrusion detection, supports a balance between security and individual privacy rights, aligning with best practices for ethical workplace monitoring. Therefore, this model possesses an advantage over other research approaches that rely solely on monitoring user behavior and actions.

In the initial phase of the methodology, identifying the files to be monitored in their optimal state allows for the establishment of a signature-based detection approach specific to those files. The model can readily detect intrusions by identifying deviations from the expected state of the files, without the need for implementing machine learning algorithms. Therefore, this approach offers the advantage of effective intrusion detection with low false positive rates, in contrast to research that relies on anomaly algorithms and data training. However, it is important to note that the model may encounter challenges in detecting new or unknown attacks that do not match any existing signatures, as well as attacks that do not involve file content violations. In the context of machine learning research, acquiring specialized datasets for (ICS) also poses significant challenges due to the complexities involved in accessing data from industrial environments. These challenges are primarily driven by security concerns, which restrict the availability and sharing of such datasets.

The model works based on an event-driven approach. This means it only springs into action when a specific event - in this case, a change to a file - occurs. This is different from a continuous polling approach where the system would constantly check the files for changes. The benefit of the event-driven response model is its remarkable efficiency in utilizing system resources, while also enabling real-time detection of intrusions as they occur. There's no need to wait for the next round of checks or polling cycle. As soon as a file changes, the model knows about it and can respond immediately. This real-time detection is crucial for catching and responding to intrusions as quickly as possible.

Leveraging the inherent capabilities of the PowerShell Register-ObjectEvent cmdlet, the model confines its operation

to the monitored workstation, thus eliminating the need for the installation of additional third-party software. The model's avoidance of third-party software installation is indeed a substantial advantage, particularly given the specific needs and constraints of industrial control systems. The introduction of third-party systems can lead to compatibility issues with existing software and configurations. Typically, the installation of any new software on industrial workstations requires explicit vendor permissions, thus adding another layer of complexity. Moreover, the introduction of new software necessitates the implementation of patching and updating plans, potentially imposing additional burdens on maintenance operations. Hence, a model that operates effectively without the need for additional software installations alleviates these potential challenges, thereby enhancing the model's applicability and ease of use in an industrial control system environment.

Storing file information and the history of file changes in a remote station negates the need to use the local storage of the workstation under normal conditions. The storage of file modification time and file hash value in a remote station enhances security by mitigating the risk of file metadata tampering by an attacker. The secure remote storage of file information enhances accessibility, enabling the review of history from any network station. This feature simplifies the investigation of file event history in the event of incident response following attack detection. Moreover, the storage of file content snapshots effectively supports system continuity by facilitating the restoration of files to a healthy condition.

The decision to implement the intrusion detection model at the supervisory level of the Industrial Control System (ICS) architecture offers numerous benefits. This level, in contrast to the control and field levels beneath it, has an abundance of system resources. Crucially, the operation of the model at this level does not burden the bandwidth of the control and field-level networks, which are known for their resource limitations. Further, the supervisory level is uniquely positioned as the sole access point to both the control and field levels. These lower levels are particularly sensitive to disruptions due to their resource limitations, and any interference could lead to significant production process impacts. By placing the intrusion detection model at the supervisory level, we ensure minimal intrusion, optimal use of resources, and maintain the integrity of the lower levels, safeguarding the overall production process.

V. Conclusion

The implementation of file integrity monitoring has proven to be a highly effective and accurate method for detecting insider intrusion involving file violations in control system workstations. These violations include data tampering, data destruction, unauthorized modifications, malicious code injection, accidental data modifications, negligence, and accidental file deletion. The model also facilitates the detection of other types of attacks that involve file content alteration, such as ransomware and remote code execution. The detection process is performed without the necessity of monitoring user behavior and actions, thereby upholding user privacy.

While the model examined in this research demonstrates its efficacy in detecting insider threats related to file integrity violations, it does not possess the capability to identify other forms of insider threats that do not involve modifications to file content, such as Intellectual Property Theft and Unauthorized Data Access. Exploring and addressing these additional categories of insider threats could serve as a promising area for future research. Another crucial aspect to consider is that the model solely focuses on detecting intrusions and does not possess the capability to independently prevent their consequences. Therefore, to achieve maximum effectiveness, it is essential to seamlessly integrate a file integrity monitoring strategy with an organization's incident response plan. This integration ensures that timely and appropriate measures are taken to mitigate the impact of file integrity intrusions, thereby preventing further unwanted consequences.

References

[1] H. Schulze, “2023 Report Insider Threat,” 2023.

[2] Q. Chen, M. Zhou, Z. Cai, and S. Su, “Compliance Checking Based Detection of Insider Threat in Industrial Control System of Power Utilities,” in 2022 7th Asia Conference on Power and Electrical Engineering (ACPEE), 2022, pp. 1142–1147.

[3] T. Alladi, V. Chamola, and S. Zeadally, “Industrial Control Systems: Cyberattack trends and countermeasures,” Comput. Commun., vol. 155, pp. 1–8, 2020.

[4] Ponemon Institute, “2022 Cost of Insider Threats Report Global Report,” 2022.

[5] T. Menze, “The State of Industrial Cybersecurity,” 2019.

[6] K. Berlin, D. Slater, and J. Saxe, “Malicious behavior detection using windows audit logs,” AISec 2015 - Proc. 8th ACM Work. Artif. Intell. Secur. co-located with CCS 2015, pp. 35–44, 2015.

[7] K. Stouffer, V. Pillitteri, S. Lightman, M. Abrams, and A. Hahn, “Guide to Industrial Control Systems (ICS) Security NIST Special Publication 800-82 Revision 2,” Gaithersburg, MD, Jun. 2015.

[8] E. Kronfuss, “Industrial cyber security standard-IEC 62443,” 2018.

[9] E. Colbert and A. Kott, Cyber-security of SCADA and Other Industrial Control Systems, vol. 66. Cham: Springer International Publishing, 2016.

[10] D. Bhamare, M. Zolanvari, A. Erbad, R. Jain, K. Khan, and N. Meskin, “Cybersecurity for industrial control systems: A survey,” Comput. Secur., vol. 89, p. 101677, Feb. 2020.

[11] E. Pricop, J. Fattahi, N. Dutta, and M. Ibrahim, Recent Developments on Industrial Control Systems Resilience, vol. 255. Springer, 2020.

[12] P. J. Taylor et al., “Detecting insider threats through language change.,” Law Hum. Behav., vol. 37, no. 4, p. 267, 2013.

[13] R. Nasir, M. Afzal, R. Latif, and W. Iqbal, “Behavioral Based Insider Threat Detection Using Deep Learning,” IEEE Access, vol. 9, pp. 143266–143274, 2021.

[14] T. Katerina and P. Nicolaos, “Mouse behavioral patterns and keystroke dynamics in End-User Development: What can they tell us about users’ behavioral attributes?,” Comput. Human Behav., vol. 83, pp. 288–305, 2018.

[15] L. Liu, O. De Vel, Q. L. Han, J. Zhang, and Y. Xiang, “Detecting and Preventing Cyber Insider Threats: A Survey,” IEEE Commun. Surv. Tutorials, vol. 20, no. 2, pp. 1397–1418, 2018.

[16] T. Hu, W. Niu, X. Zhang, X. Liu, J. Lu, and Y. Liu, “An Insider Threat Detection Approach Based on Mouse Dynamics and Deep Learning,” Secur. Commun. Networks, vol. 2019, 2019.

[17] L. Liu, C. Chen, J. Zhang, O. De Vel, and Y. Xiang, “Insider Threat Identification Using the Simultaneous Neural Learning of Multi-Source Logs,” IEEE Access, vol. 7, pp. 183162–183176, 2019.

[18] A. V. Artem Storozhuk, “Audit logs security: cryptographically signed tamper-proof logs,” Cossack Labs, 2020. [Online]. Available: https://www.cossacklabs.com/blog/audit-logs-security/. [Accessed: 01-

Jun-2023].

[19] K. Denney, L. Babun, and A. S. Uluagac, "USB-Watch: a Generalized Hardware-Assisted Insider Threat Detection Framework," J. Hardw. Syst. Secur., 2020.

[20] J. Xiao, L. Yang, F. Zhong, X. Wang, H. Chen, and D. Li, "Robust Anomaly-based Insider Threat Detection using Graph Neural Network," IEEE Trans. Netw. Serv. Manag., p. 1, 2022.

[21] D. C. Le and N. Zincir-Heywood, "Anomaly Detection for Insider Threats Using Unsupervised Ensembles," IEEE Trans. Netw. Serv. Manag., vol. 18, no. 2, pp. 1152–1164, 2021.

[22] B. Nagabhushana Babu and M. Gunasekaran, "An Analysis of Insider Attack Detection Using Machine Learning Algorithms," in 2022 IEEE 2nd International Conference on Mobile Networks and Wireless Communications (ICMNWC), 2022, pp. 1–7.

[23] B. A. & H. H. Moghadam M. H., "Anomaly Detection Dataset for Industrial Control Systems.," LawArXiv. /abs/2305.09678, 2023.

[24] A. Ribeiro, "ICS system hardening required to improve operational resilience, boost overall cybersecurity posture," Industrialcyber, 2023. [Online]. Available: https://industrialcyber.co/features/ics-system-hardening-required-to-improve-operational-resilience-boost-overall-cybersecurity-posture/. [Accessed: 21-May-2023].

[25] H. K. Sharma, I. Khanchi, N. Agarwal, P. Seth, and P. Ahlawat, "Real time activity logger: A user activity detection system," Int. J. Eng. Adv. Technol., vol. 9, no. 1, pp. 1991–1994, 2019.

[26] Tripwire, "What Is FIM (File Integrity Monitoring)?," 2023. [Online]. Available: What Is FIM (File Integrity Monitoring)? [Accessed: 10-May-2023].

[27] SAMHAIN LABS, "THE SAMHAIN FILE INTEGRITY / HOST-BASED INTRUSION DETECTION SYSTEM," 2023. [Online]. Available: https://www.la-samhna.de/samhain/. [Accessed: 21-May-2023].

[28] OSSEC, "Server Intrusion Detection for Every Platform Server Intrusion Detection for Every Platform," 2023. [Online]. Available: https://www.ossec.net/. [Accessed: 21-May-2023].

[29] Crowdstrike, "WHAT IS FILE INTEGRITY MONITORING?," 2023. [Online]. Available: https://www.crowdstrike.com/cybersecurity-101/file-integrity-monitoring/. [Accessed: 10-May-2023].

[30] S. K. Peddoju, H. Upadhyay, and L. Lagos, "File integrity monitoring tools: Issues, challenges, and solutions," Concurr. Comput. Pract. Exp., vol. 32, no. 22, pp. 1–8, 2020.

[31] Nandini Raghvendra, "SCADA System – Components, Hardware & Software Architecture, Types," electricalfundablog, 2023. [Online]. Available: https://electricalfundablog.com/scada-system-components-architecture/?utm_content=cmp-true. [Accessed: 01-Mar-2023].

[32] 'Microsoft, "File Integrity Monitoring in Microsoft Defender for Cloud," 2023. [Online]. Available: https://learn.microsoft.com/en-us/azure/defender-for-cloud/file-integrity-monitoring-overview. [Accessed: 10-May-2023].