DOI: 10.1109/SVCC61185.2024.10637363

Silent Sabotage: A Stealthy Control Logic Injection in IIoT Systems

Wael Alsabbagh[1,2], Chaerin Kim[1,2] and Peter Langendörfer[1,2]
[1]*IHP – Leibniz-Institut für innovative Mikroelektronik, Frankfurt (Oder), Germany*
[2]*Brandenburg University of Technology Cottbus-Senftenberg, Cottbus, Germany*
E-mail: (Alsabbagh, Kim, Langendoerfer)@ihp-microelectronics.com

***Abstract*—As Industrial Internet of Things (IIoT) continues to revolutionize industrial processes, the security of these systems becomes paramount. This work contributes to ongoing efforts to uncover security issues in IIoT-based systems, specifically those that allow attackers to conduct stealthy control logic injections. To achieve this, we designed an attack aimed at compromising IIoT-based systems by exploiting vulnerabilities within the MQTT and SFTP protocols. The attack specifically targets the interaction between IoT devices and Programmable Logic Controllers (PLCs) through MQTT communication, while utilizing the SFTP protocol for file transfer, with the goal of infecting and manipulating PLCs discreetly. We initially demonstrate that attackers can leverage weaknesses in the SFTP protocol to inject malicious control logic into IoT-PLCs. Subsequently, attackers exploit the lack of robust security mechanisms in the MQTT protocol to conceal their injection, orchestrating a deception by falsifying the user-dashboard view. This manipulation ensures that the injected control logic remains hidden, presenting a fabricated operational status to the end-user. Finally, we propose a set of security recommendations and mitigation solutions to safeguard IIoT systems against such a severe attack scenario. Our attack codes, along with a proof-of-concept, are publicly available.**

***Index Terms*—component, formatting, style, styling, insert**

I. Introduction

The Industrial Internet of Things (IIoT) has ushered in a trans-formative era for industrial processes, with IoT-enabled Programmable Logic Controllers (IoT-PLCs) emerging as a revolutionary force. These devices seamlessly integrate traditional industrial systems with advanced digital technologies, fostering unprecedented levels of efficiency and connectivity.

IoT-PLCs act as the nerve center, collecting and transmitting real-time data from machinery and sensors [1]. This influx of data allows for predictive maintenance, minimizing downtime and optimizing operational performance. Moreover, IoT-PLCs facilitate remote monitoring and control, empowering industries to manage operations from anywhere in the world, enhancing responsiveness and reducing operational costs.

In IIoT Systems, the Message Queuing Telemetry Transport (MQTT) [2] and Secure File Transfer Protocol (SFTP) [3] play pivotal roles in ensuring seamless communication and secure data exchange. MQTT facilitates efficient real-time messaging between IoT devices and PLCs, enabling swift and reliable data transmission. Its lightweight, publish-subscribe architecture optimizes bandwidth usage, crucial for resource-constrained IoT environments. On the other hand, SFTP ensures the secure transfer of critical data files between devices and servers in IoT-PLC based systems. With encryption and authentication mechanisms, SFTP safeguards sensitive information, addressing the paramount need for data integrity and confidentiality in industrial settings. Together, MQTT and SFTP establish a robust communication infrastructure, fostering the scalability, reliability, and security essential for the successful integration and operation of IIoT Systems. However, their adoption introduces notable security challenges. For instance, MQTT may lack robust built-in security features, making devices vulnerable to unauthorized access and data breaches. Additionally, the simplicity of SFTP, while efficient for file transfers, poses also security risks such as unauthorized access and potential data interception during transmission. Therefore, it becomes imperative to investigate security issues in IoT-PLC based systems, precisely those rely on the security of both MQTT and SFTP protocols.

In this paper, we meticulously exploit vulnerabilities within the MQTT and SFTP protocols, and conduct a control logic injection attack tailored to compromise the integrity of IoT-PLC device. To further amplify the potency of our attack, we adeptly manipulate the user interface by fabricating a deceptive view for the user-dashboard. Operating covertly within the MQTT publisher-subscriber network model, we simulate the transmission of legitimate MQTT packets, meticulously concealing the malicious payload from the user's purview. This strategic manipulation ensures that the user-dashboard consistently displays falsified information, camouflaging the ongoing subversion of control logic with the IoT-PLC.

In response to the presented attack scenario, our paper proffers targeted mitigation strategies aimed at neutralizing the threat vectors introduced through MQTT and SFTP vulnerabilities. Our recommendations include implementing secure authentication and authorization measures, introducing a strong encryption mechanism to secure communication between MQTT clients and brokers, integrating a topic whitelisting to restrict clients to subscribe only certain topics, setting proper file permission on SFTP servers, configuring the SFTP server securely, among many others. As we navigate the intricate interplay between IIoT technologies and evolving cyber threats, this paper seeks to contribute to the

978-1-7281-9023-5/21/$31.00 ©2021 IEEE

collective understanding of potential vulnerabilities and, more importantly, to inspire proactive measures that will secure the foundation of our industrial interconnected future. To validate our findings, we have published our proof-of-concept of the attack we conducted in this paper, as well as all codes in [30], and [29], respectively.

The rest of the paper is organized as follows. In Section II, we delve into related works, and Section III outlines the experimental setup employed in this study. Moving on to Section IV, we offer an overview of both the MQTT and SFTP protocols, and in Section V, we elaborate on our chosen attack approach. Following that, we propose security recommendations and mitigation solutions in Section VI, and Section VII serves as the conclusion of the paper.

II. Related Work

Stealthy control logic injection attacks in IIoT systems aim to modify the programs running in IoT-PLCs without alerting the system operators [4], [5]. Alladi et al. [6] showed that adversaries with unauthorized access to the IIoT network can inject malicious code into the compromised device. Another research group [7] demonstrated a malicious injection attack in IIoT systems where attackers aim to steal data from the compromised nodes and then generate unauthorized actions in the target system. Andy et al. [8] demonstrated different attack scenarios based on the lack of security features in the MQTT protocol. In one of these scenarios, the authors proved that attackers who are able to sniff the traffic in IoT systems, would be also able to modify the payload of the MQTT packets, precisely the topics. Barua et al. [9] presented an attack that can exploit both availability and integrity of an industrial IoT-PLC settings. The authors managed successfully to disclose a vulnerability in different MQTT protocol variants, which allows adversaries to exploit the memory de-duplication feature of the cloud. Their attack approach caused disruption to the physical process.

All the aforementioned attack scenarios could be detected by ICS operators as they were not designed to be stealthy. In contrast, our attack approach is more severe and entirely concealed, ensuring the ICS operator sees only what he/she expects, making detection challenging.

III. Experimental Setup

To test our attack approach presented in this paper, we used a Fischertechnik training factory[1] (known also as Lernfabrik 4.0 9V) which is used for developing Industry 4.0 applications see figure 1. The factory enables automated industrial activities to be simulated, understood, and applied on a small scale before being implemented on a larger scale. It is comprised of four industrial workstations [10]:

- Vacuum Gripper Robot (VGR): a station featuring a robot equipped with a vacuum gripper.
- High-Bay Warehouse (HBW): a station housing an automated rack warehouse with 3x3 slots.

[1]https://www.fischertechnikwebshop.com/de-DE/lernfabrik-4-0-de-de

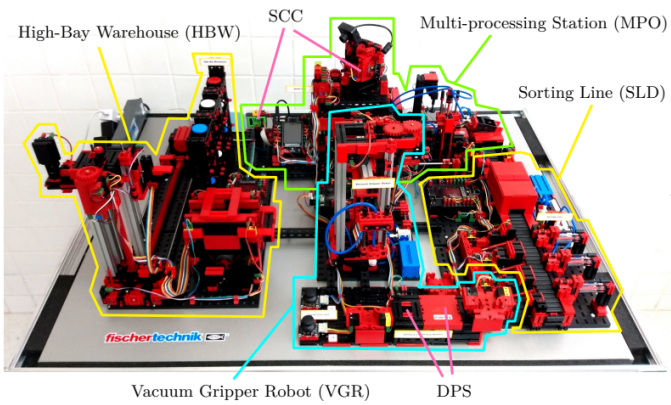

Fig. 1. Fischertechnik Lernfabrik 4.0 9V, adopted from [10].

- Multi-Processing Station (MPO): simulating an industrial oven and machining bench.
- Sorting Line and Color Detection (SLD): a station equipped with a conveyor, color detection system, and part separator.

In addition to these workstations, the factory incorporates various modules, such as a Delivery and Pickup module (DPS), a Near-Field Communication (NFC) reader/writer, an RGB camera module, and a Sensor Station with Camera (SSC) designed to capture supplementary data. The production simulation involves the movement of small workpieces of different colors (white, red, and blue) within the environment. Each workpiece is assigned to a unique NFC identification tag, enabling the recording of color and production history, inclusive of date and time information. The configuration of the factory comprises six Fischertechnik TXT 4.0 controllers (see Figure 2).

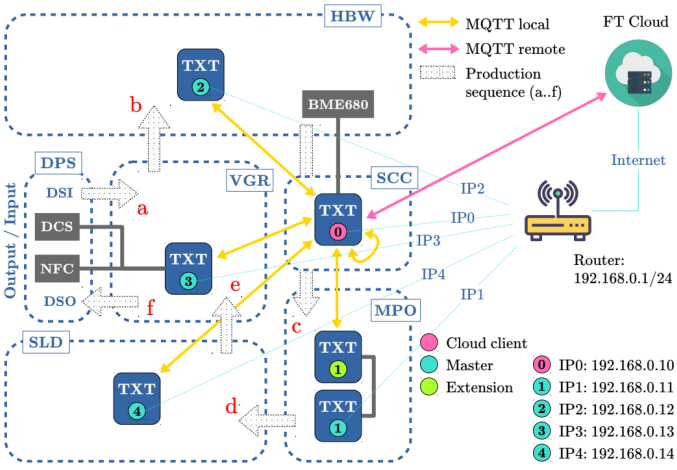

Fig. 2. Block Diagram of the Lernfabrik 4.0, adopted from [10].

The TXT controllers establish bidirectional communication exclusively with a central TXT controller denoted as TXT-0. Acting as an MQTT broker, TXT-0 facilitates the exchange of messages with other controllers connected to a Wi-Fi network

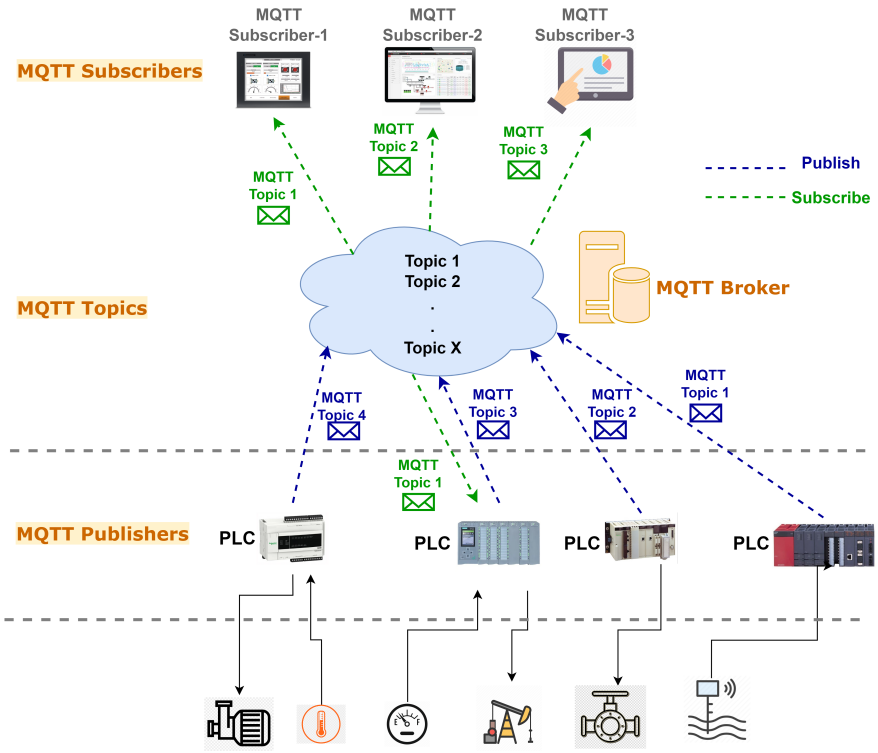

Fig. 3. MQTT Publish-Subscribe model in IIoT Systems.

through an TP-Link Router. Configured as a cloud client, TXT-0 establishes a connection to Fischertechnik's cloud via a remote MQTT bridge. Users can access a dashboard through a web browser in the cloud, enabling them to view information and interact with specific production processes within the factory. TXT-0 is equipped with a Universal Serial Bus (USB) camera and an environmental sensor, providing data on air pressure, humidity, temperature, and gases. Four controllers, TXT-1 to TXT-4, function as cloud masters within the MPO station, two TXT controllers (TXT-1) are present, with one configured as an extension under the control of the TXT-1 master. TXT-2 manages the automated warehouse for the HBW station, while the TXT-3 controller controls the DPS module in the VGR station. TXT-4 oversees operations in the SLD station.

The standard sequence of work-piece manufacturing, starting from DSI and concluding at DSO (considering storage and production), is illustrated in the block diagram by broad arrows labeled in alphabetical order from 'a' to 'f'. Each TXT controller receives an executable file containing compiled C++ code to govern its respective station.

IV. Overview of MQTT & SFTP

A. *MQTT Protocol*

MQTT is a lightweight messaging protocol designed for low-bandwidth, high-latency or unreliable networks. It follows a publish-subscribe model (see Figure 3), where clients communicate through a central broker [11]. The key components include:

- Publisher: sends messages to a specific topic on the broker.
- Subscriber: subscribes to one or more topics and receives messages from the broker.
- Broker: manages the communication between publishers and subscribers.
- Topics: act as communication channels or categories to which messages are published and from which subscribers receive messages.

IoT devices establish MQTT connections with the broker using Transmission Control Protocol/Internet Protocol (TCP/IP). They can act as publishers, subscribers, or both. Connection between them occurs through MQTT topic names declared by publishers in messages and subscribed by subscribers. To transmit MQTT messages between IoT devices and cloud servers, correct topic names following the convention are essential. Publishers and subscribers don't need to know source and destination IP addresses for communication [12]. Each MQTT message from a publisher to the broker includes a specific topic and payload. The payload holds the actual data, such as a PLC publishing a message on "sensor/temperature" with the current temperature reading.

B. *Security Issues in MQTT*

The MQTT protocol exhibits several security issues [13] that eventually might lead to several attack scenarios against IoT and IIoT Systems. In the following, we illustrate these security issues.

1) Confidentiality: The MQTT protocol lacks inherent encryption as it relies on the TCP protocol. While the MQTT

broker provides authentication security, restricting connections and governing node access, the protocol lacks default encryption and various protection mechanisms [13]. Consequently, MQTT security relies on non-encrypted authentication [8], [14], posing a risk of sensitive data extraction through traffic analysis. For instance, attackers on the same network can sniff "CONNECT" packets, extracting plaintext credentials [15]. Figure 4 shows a captured CONNECT packet in transit between a publisher (TXT-3 controller) and the MQTT broker (TXT-0 controller) in our experimental setup .

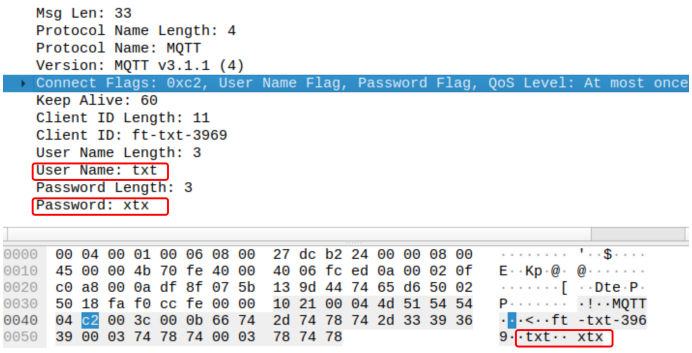

Fig. 4. MQTT CONNECT Packet: Client Credentials are sent in Plaintext.

2) Authentication: By default, MQTT lacks client authentication, enabling potential attackers to anonymously subscribe to and publish messages using the broker's IP address and port. Malicious publishers could exploit this to remotely disrupt IoT devices by sending commands like "OFF," "FORMAT," or "RESET." Similarly, nefarious subscribers could compromise system confidentiality by subscribing to all topics using wildcards, intercepting and disclosing sensitive information to unauthorized entities [16]. For instance, an attacker acting as a malicious publisher might send disruptive commands like "RESET," causing a fleet of IoT devices to unexpectedly reboot. As a malicious subscriber, the attacker subscribes to all topics using wildcards, allowing them to intercept and collect all messages within the MQTT network. This could lead to the disclosure of sensitive information, compromising the confidentiality and integrity of any IIoT System.

3) Integrity: When untrusted or anonymous MQTT clients connect to the broker, there's a risk of message data manipulation, especially in unencrypted connections, making it susceptible to Man-in-the-Middle (MitM) attacks. In IoT-PLC scenarios, attackers might exploit this vulnerability during firmware update, leading to unauthorized control over PLCs.

Due to the the aforementioned vulnerabilities, an attacker could inject malicious MQTT messages, exploiting weaknesses in encryption or broker configuration. For instance, by manipulating the payload of a message, they might trick a subscriber into executing unauthorized commands or altering the normal flow of operations.

C. SFTP Protocol

SFTP is a network protocol that provides secure file access, file transfer, and file management functionalities over an encrypted connection. The key characteristics of SFTP include:

- Authentication: utilizes cryptographic keys or username/password for user authentication.
- Encryption: encrypts data during transmission, preventing unauthorized access.
- Integrity: ensures data integrity through checksum and hashing.
- Portability: platform-independent and runs over the Secure Shell (SSH) protocol.

SFTP is commonly used for secure file transfers, particularly in scenarios where data confidentiality and integrity are paramount, such as in business data exchanges or server backups.

D. Security Issues in SFTP

1) Weak Authentication Mechanisms: If SFTP is configured with weak authentication mechanisms, such as simple passwords or outdated cryptographic keys, attackers could gain unauthorized access. Once inside, they might inject malicious logic into files being transferred.

2) Insufficient Logging and Monitoring: Inadequate logging and monitoring capabilities can make it challenging to detect suspicious activities. Attackers could exploit this weakness to inject control logic stealthily without triggering alarms.

3) Brute Force Attacks: SFTP servers may be vulnerable to brute force attacks where an attacker repeatedly tries to guess the username and password combination [17].

4) SSH Vulnerabilities: SFTP relies on SSH for its security. Any vulnerabilities in the underlying SSH implementation could potentially affect SFTP [18].

5) Permissions and Access Controls: Improper file and directory permissions may lead to unauthorized access.

In an SFTP client-server model, an attacker might exploit weak authentication, insufficient logging, server default configurations, etc. to inject malicious commands or scripts into files being transferred.

V. Control Logic Injection Attack

In Figure 5, we present a high-level overview of the control logic injection attack conducted on our Fischertechnik Lernfabrik 4.0 system (see Figure 1). The attack consists of four main phases:

- Breaking the Authentication of the TP-Link Router.
- Breaking the Authentication of MQTT Broker.
- Infecting the Control Logic Program.
- Concealing the Infection.

In the following sub-sections, we provide a detailed explanation of each phase.

A. Breaking the Authentication of the TP-Link Router

The initial step for an attacker is to compromise the security of the Wi-Fi router, specifically in our scenario, the TP-Link router. This is typically achieved by exploiting vulnerabilities in its firmware or configuration settings. Such exploits may involve tactics like leveraging default credentials or using

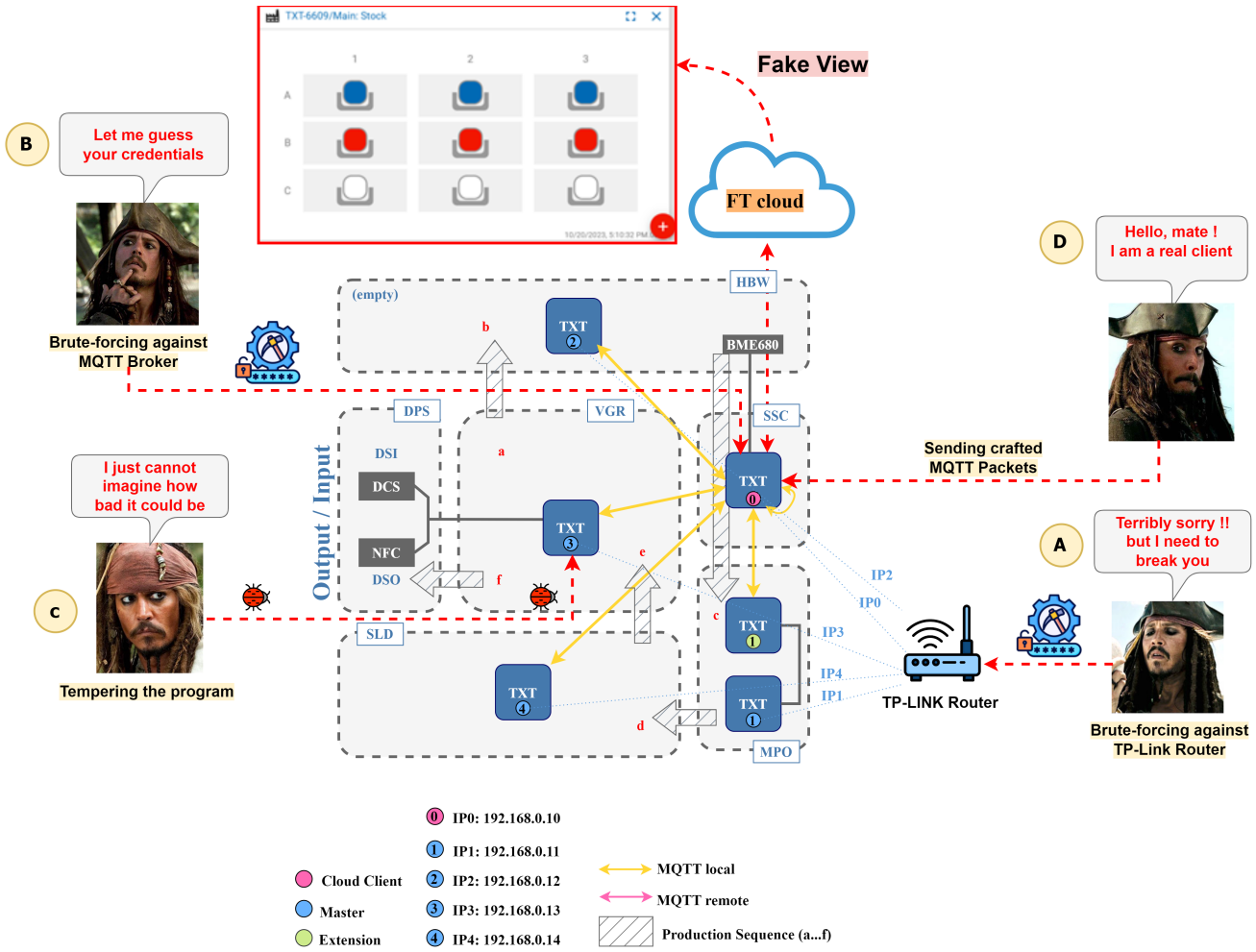

Fig. 5. High-level Overview of our Stealthy Control Logic Injection Attack.

brute-force attacks, as described in [19], [20]. In our work, we also employ a common brute-force attack [28] to breach the TP-Link router. Depending on the password complexity, the guessing process may range from a few seconds (for an 8-character password) to 30 minutes (for a 16-character password). Once the router is compromised, unauthorized access is obtained, enabling us to manipulate network settings, intercept traffic, or potentially launch further attacks on TXT controllers connected to the network.

B. *Breaking the Security of MQTT Broker*

Following the compromise of the TP-Link router, our focus shifts to infiltrating the MQTT broker's security. The MQTT broker (in our case TXT-0 controller) acts as a central hub for managing communication between devices in an IIoT ecosystem. Security breaches at this level can be particularly concerning as they may enable unauthorized access to sensitive data and control commands exchanged between the MQTT broker and other TXT controllers (IoT devices). For our experimental setup, we can break the authentication of the MQTT broker in different ways as follows.

- **Brute-force Attack:** an attacker can use automatic tools to produce multiple sequential guesses to compromise the ciphertext [19], or user credentials can be retrieved through running possible tries by the attacker during the authentication phase [20].
- **Sniffing Attack:** here, attacker has access the MQTT network and can sniff MQTT packets, precisely CONNECT packets that contain the user credentials sent in plaintext see figure 4. However, this method has limitations as the attacker needs to wait until a CONNECT packet is sent from an IoT node to the broker.
- **Retrieving User Credentials:** To break the security of MQTT broker, an attacker could check the MQTT configurations in the TXT-0 controller with PuTTY[2] terminal. If the configuration file is not well protected i.e., not access restricted to root-users, the attacker can get credentials to establish an MQTT connection with TXT controller(s). The MQTT configuration file may contain credentials

[2]PuTTY is a free software that allows a remote connection with IoT devices (TXT controllers) via Secure Shell Access (SSH), particularly via port 22. It is available online: https://www.chiark.greenend.org.uk/ sgtatham/putty/

for local broker or remote connection to work as MQTT bridge as shown in figure 6.

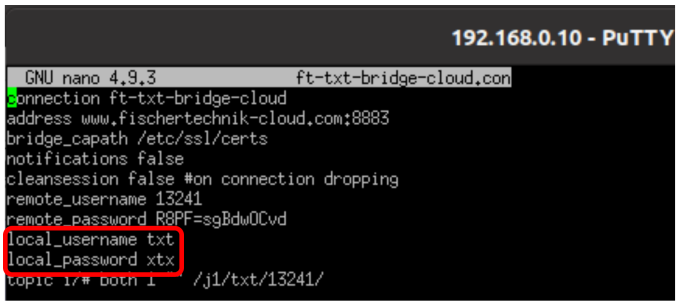

Fig. 6. Retrieving User Credentials using PuTTY tool

In our example, the MQTT broker (TXT-0 Controller) is set up with default credentials, such as 'txt' as username and 'xtx' as password. Nevertheless, even if the controller's credentials were different, PuTTY software can consistently retrieve them. This provides attackers with the capability to intercept, manipulate, and inject malicious payloads into the communication stream between IoT nodes (e.g., publishers - TXT controllers and subscribers - dashboards.).

C. *Infect the Control logic Program*

The TXT controllers are integrated with an SFTP interface, allowing secure file transfer from and to the remote user. Please note that TXT controllers simulate real-hardware PLCs (e.g., Siemens S7-1500 PLCs, Rockwell Automation PLCs, Beckhoff TwinCAT PLCs) which also utilize the SFTP protocol to transfer files over the IIoT network.

In this work, to inject the TXT controllers (in our case the TXT-3 controller), we used an open-source software called FileZilla[3] that allows to establish a secure channel with the FTP server on the target TXT controller, transmitting malicious executable files i.e., control logic programs to compromise the integrity of the targeted controller.

To do so, attackers need only to know the IP address of the target, the username and password used for the SSH connection, as well as the port which has always a fix value '22' (see Figure 7). Please note that for our experimental setup, the SSH connection credentials are default i.e., 'ROBOpro'. However, even if the SSH connection was configured with different credentials, attackers can still retrieve the credentials used for the SSH connection, utilizing our Brute-Force attack mentioned in Section V-A. For our experimental setup, the attacker's program disrupts the Fischertechnik system operation, causing inappropriate movements in the VGR workstation (See the proof-of-concept referenced in [29]).

The unidirectional authentication mechanism allows the attacker to establish an SFTP connection without breaching target IoT devices' authentication measures. By exploiting the SFTP protocol, the attacker not only uploads malicious files but also exfiltrates critical data (e.g., control logic programs, files, certificates) from the victim controller, escalating the potential impact of the breach.

[3]https://filezilla-project.org/

D. *Concealing the Injection (Fake View)*

To conceal our injection, it is imperative to meticulously fabricate the user's view, presenting a fake view that aligns seamlessly with the user's expectation. The user-dashboard serves as a dedicated subscriber within the MQTT publisher-subscriber paradigm. In this model, MQTT packets, emanating from various publishers (TXT Controllers) at regular intervals, relay real-time operational status updates for each workstation. The user-dashboard subscribes to all relevant topics, ensuring comprehensive coverage of the operational landscape.

Exploiting the inherent openness of the MQTT protocol, attackers can surreptitiously intercept and record these packets during prior sessions. Subsequently, the adversary pushes the recorded MQTT packets into the stream destined for the unsuspecting victim subscriber (user-dashboard). It is noteworthy that the subscriber, adhering to the MQTT protocol, lacks robust authentication and checksum mechanisms to validate the authenticity of packet sources, as expounded in Section IV-B. Consequently, the subscriber remains inherently susceptible to the reception and processing of injected packets, perpetuating the illusion of genuine operational statuses to the end user. The absence of source authentication in MQTT exacerbates the challenge of discerning between legitimate and injected packets, underscoring the effectiveness of our stealthy control logic injection approach. Figure 8 shows the user view under two cases: 1) when we conduct a normal control logic injection attack (highlighted in green), 2) when we conceal our control logic injection attack by fabricating the view for the user i.e., shows him always what he expects to see (highlighted in red).

VI. Mitigation Solutions

Mitigating our stealthy control logic injection attack scenario in IIoT Systems, particularly when involving MQTT and SFTP, requires a combination of technical and procedural measures as follows.

A. *MQTT-specific Mitigation*

*1) **Authentication and Authorization**:* It is recommended to implement strong authentication mechanisms for MQTT, such as MQTT-Auth [21], MQTT with Application Programming Interface (API) key authentication [22], and MQTT with X.509 Certificates [23]. Such mechanisms ensure that only authorized devices can publish and subscribe to relevant topics.

*2) **TLS/SSL Encryption**:* The channels between MQTT clients and brokers expose IIoT systems to various network based attacks e.g., MitM, replay, injection, etc. Thus, we need to secure those channels by implementing Transport Layer Security (TLS) or Secure Sockets Layer (SSL) for encrypted communication between MQTT clients and brokers as suggested in [24]. This helps to prevent malicious eavesdropping and tampering activities.

Fig. 7. Establishing an SFTP Channel with the Victim TXT Controller using Filezilla Software

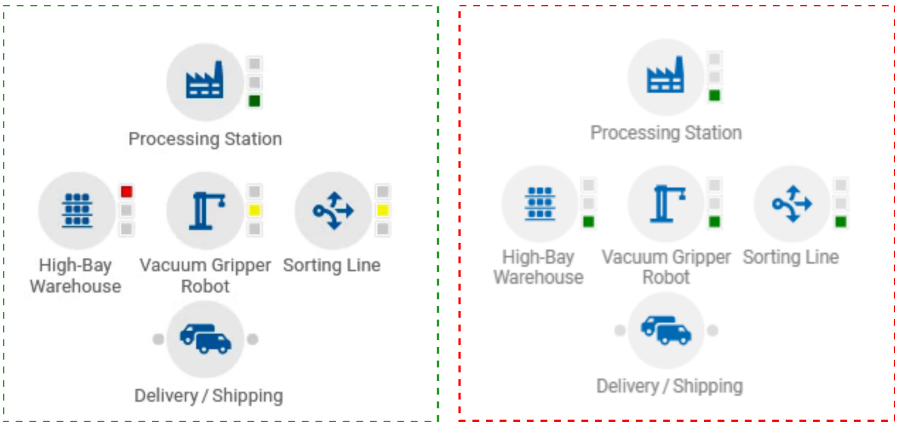

Fig. 8. Dashboard shows the user the operation statuses of the workstations under our attack in two cases: 1) when we perform a normal control logic injection attack (highlighted in green); 2) when we perform a stealthy control logic injection attack (highlighted in red)

*3) **Topic Whitelisting**:* Implementing topic whitelisting restricts MQTT clients to subscribe only specific topics as proposed in [25]. This helps to control the flow of information and prevents unauthorized access.

*4) **Payload Validation**:* The success of our injection was due to the fact that MQTT payloads are not checked and validated by any integrity check mechanism. Thus attackers could easily manipulate the payload of MQTT packets and the victim MQTT publisher/subscriber will not realized that the payload was manipulated. For all that, it is recommended to implement an integrity check mechanism e.g., [26], that adheres to the expected format and doesn't contain malicious content.

B. SFTP-Specific Mitigation Strategies

*1) **Secure File Permissions**:* To enhance the security of SFTP servers, it is imperative to establish meticulous control over file permissions similar to what [27] suggested. By configuring access rights judiciously, the system can restrict file access exclusively to authorized users, thereby safeguarding sensitive data from unauthorized entities.

*2) **Secure Configuration**:* A crucial aspect of fortifying SFTP servers involves the secure configuration of the system. Disabling unnecessary services is paramount in mitigating potential vulnerabilities and thwarting attempts of unauthorized access by malicious actors. This step significantly contributes to the overall robustness of the SFTP server, reinforcing its resilience against potential security breaches.

*3) **Security Pairing and Mutual Authentication**:* The implementation of robust security measures such as security pairing and mutual authentication is instrumental in fortifying the security posture of the Industrial Internet of Things (IIoT) environment. These measures specifically address the risk of control logic injection attacks through MQTT and SFTP channels. By enforcing stringent authentication processes, the system ensures that only legitimate entities are granted access, thereby minimizing the risk of unauthorized infiltration.

Regularly reviewing and updating security measures is paramount to staying abreast of evolving threats. This proactive approach ensures that the mitigation strategies remain effective in the face of emerging security challenges, maintaining the integrity and security of the IIoT environment over time.

VII. Conclusion

In this paper, we delve into the intricacies of control logic injection attacks, emphasizing their significance within the context of Industrial Internet of Things (IIoT)-based systems. Our investigation focuses on exploiting vulnerabilities in both MQTT and SFTP protocols, demonstrating a severe control logic injection attack scenario. We conceal our injection in such a way that users will consistently see what they expect, heightening the severity of this scenario in IIoT-based systems, particularly when IoT-PLCs are utilized and employ MQTT and SFTP without implementing necessary security practices.

To alleviate the impact of our attack, we propose security solutions. If implemented, our attack would either be unsuccessful, or at the very least, users could detect it at an early stage. We have made the proof of concept, along with the attack codes, publicly available for further research.

REFERENCES

[1] W. Alsabbagh and P. Langendörfer, "Security of Programmable Logic Controllers and Related Systems: Today and Tomorrow," in IEEE Open Journal of the Industrial Electronics Society, vol. 4, pp. 659-693, 2023, doi: 10.1109/OJIES.2023.3335976.

[2] U. Hunkeler, H. L. Truong, and A. Stanford-Clark, "MQTT-S—A publish/subscribe protocol for wireless sensor networks," in Proc. IEEE 3rd Int. Conf. Commun. Syst. Softw. Middleware Workshops (COMSWARE), 2008, pp. 791–798.

[3] [Online]. Available: https://www.rfc-editor.org/rfc/rfc913.html.

[4] W. Alsabbagh, S. Amogbonjaye, D. Urrego and P. Langendörfer, "A Stealthy False Command Injection Attack on Modbus based SCADA Systems," 2023 IEEE 20th Consumer Communications & Networking Conference (CCNC), Las Vegas, NV, USA, 2023, pp. 1-9, doi: 10.1109/CCNC51644.2023.10059804.

[5] W. Alsabbagh and P. Langendörfer, "A Stealth Program Injection Attack against S7-300 PLCs," 2021 22nd IEEE International Conference on Industrial Technology (ICIT), Valencia, Spain, 2021, pp. 986-993, doi: 10.1109/ICIT46573.2021.9453483.

[6] T. Alladi, V. Chamola, B. Sikdar and K. -K. R. Choo, "Consumer IoT: Security Vulnerability Case Studies and Solutions," in IEEE Consumer Electronics Magazine, vol. 9, no. 2, pp. 17-25, 1 March 2020, doi: 10.1109/MCE.2019.2953740.

[7] S. Khanam, I. B. Ahmedy, M. Y. Idna Idris, M. H. Jaward, and A. Q. Bin Md Sabri, "A survey of security challenges, attacks taxonomy and advanced countermeasures in the internet of things," IEEE Access, vol. 8, pp. 219 709–219 743, 2020.

[8] S. Andy, B. Rahardjo and B. Hanindhito, "Attack scenarios and security analysis of MQTT communication protocol in IoT system," 2017 4th International Conference on Electrical Engineering, Computer Science and Informatics (EECSI), Yogyakarta, Indonesia, 2017, pp. 1-6, doi: 10.1109/EECSI.2017.8239179.

[9] A. Barua, L. Pan and M. A. Al Faruque, "BayesImposter: Bayesian Estimation Based.bss Imposter Attack on Industrial Control Systems," ASAC'22: Proceedings of the 38th Annual Computer Security Applications Conference, Dec. 2022, pp. 440-454, doi: 10.1145/3564625.3564638.

[10] A. L. de Sousa and A. S. de Oliveira, "Order-Controlled Production Employing Multi-Agent and Flexible Job-Shop Scheduling on a Physical Simulation Platform," 2022 Latin American Robotics Symposium (LARS), 2022 Brazilian Symposium on Robotics (SBR), and 2022 Workshop on Robotics in Education (WRE), São Bernardo do Campo, Brazil, 2022, pp. 229-234, doi: 10.1109/LARS/SBR/WRE56824.2022.9995868.

[11] A. H. Eyeleko and T. Feng, "A Critical Overview of Industrial Internet of Things Security and Privacy Issues Using a Layer-Based Hacking Scenario," in IEEE Internet of Things Journal, vol. 10, no. 24, pp. 21917-21941, 15 Dec.15, 2023, doi: 10.1109/JIOT.2023.3308195.

[12] X. Liu, T. Zhang, N. Hu, P. Zhang, and Y. Zhang, "The method of internet of things access and network communication based on mqtt," Computer Communications, vol. 153, pp. 169–176, 2020, doi: 10.1016/j.comcom.2020.01.044.

[13] D. Dinculeană, and X. Cheng, "Vulnerabilities and limitations of MQTT protocol used between IoT devices," Appl. Sci, Vol. 9, no. 5, p. 848, 2019.

[14] J. J. Anthraper, and J. Kotak, "Security, Privacy and Forensic Concern of MQTT Protocol," SSRN Electron. J., no. December, 2019.

[15] A. Al-Ani, W. K. Shen, A. K. Al-Ani, S. A. Laghari and O. E. Elejla, "Evaluating Security of MQTT Protocol in Internet of Things," 2023 IEEE Canadian Conference on Electrical and Computer Engineering (CCECE), Regina, SK, Canada, 2023, pp. 502-509, doi: 10.1109/CCECE58730.2023.10288857.

[16] N. F. Syed, Z. Baig, A. Ibrahim, and C. Valli, "Denial of Service Attack Detection through Machine Learning for the IoT," Journal of Information and Telecommunication, vol. 4, no. 4, pp. 482–503, 2020.

[17] E. D. Saputro, Y. Purwanto and M. F. Ruriawan, "Medium Interaction Honeypot Infrastructure on The Internet of Things," 2020 IEEE International Conference on Internet of Things and Intelligence System (IoTaIS), BALI, Indonesia, 2021, pp. 98-102, doi: 10.1109/IoTaIS50849.2021.9359711.

[18] I. Ahmović, H. Hažiahmetović, R. Blažević and M. Alispahić, "Statistical analysis of hydrological parameters in the Bosna River basin," Proceedings of the 34th DAAAM International Symposium, 2023, Vienna, Austria, doi: 10.2507/34th.daaam.proceedings.xxx.

[19] K. Jackson, "Hacker's Choice Top Six Database Attacks," 2008. [Online]. Available: https://www.darkreading.com/ risk/hackers-choice-top-six-database-attacks/d/d-id/112 9481.

[20] I. Vaccari, G. Chiola, M. Aiello, M. Mongelli, and E. Cambiaso, "MQTTset, a new dataset for machine learning techniques on MQTT," Sensors (Switzerland), Vol. 20, no. 22, pp. 1–17, 2020.

[21] M. Calabretta, R. Pecori, M. Vecchio and L. Veltri, "MQTT-Auth: a Token-based Solution to Endow MQTT with Authentication and Authorization Capabilities," in Journal of Communications Software and Systems, vol. 14, no. 4, pp. 320-331, October 2018, doi: 10.24138/jcomss.v14i4.604.

[22] H. Chien et al., "A MQTT-API-compatible IoT security-enhanced platform," International Journal of Sensor Networks, Vol. 32, No. 1, 2020, doi: 10.1504/IJSNET.2020.104463.

[23] M. N. Dhaval, G. R. Ashwin, "A Study on MQTT Protocol Architecture and Security Aspects Within IoT Paradigm," In: Balas, V.E., Semwal, V.B., Khandare, A. (eds) Intelligent Computing and Networking. Lecture Notes in Networks and Systems, vol 632. Springer, Singapore, 2023, doi: 10.1007/978-981-99-0071-8_6.

[24] I. L. B. M. Paris, M. H. Habaebi and A. M. Zyoud, "Implementation of SSL/TLS Security with MQTT Protocol in IoT Environment," Wireless Pers Commun 132, 163–182 (2023), doi: 10.1007/s11277-023-10605-y.

[25] E. G. Ch. Zhi et al. "MQTT Traffic Collection and Forensic Analysis Framework," Digital Forensics and Cyber Crime: 13th EAI International Conference, ICDF2C 2022, Boston, MA, November 16-18, 2022, Proceedings. Vol. 508. Springer Nature, 2023.

[26] Ch. Patel, N. Doshi, "A Novel MQTT Security framework In Generic IoT Model", Procedia Computer Science, v. 171, 2020, pp. 1399-1408, doi: 10.1016/j.procs.2020.04.150.

[27] S. S. Sokolov, O. M. Alimov, L. E. Ivleva, E. Y. Vartanova and V. G. Burlov, "Using unauthorized access to information based on applets," 2018 IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (EIConRus), Moscow and St. Petersburg, Russia, 2018, pp. 128-131, doi: 10.1109/EIConRus.2018.8317046.

[28] [Online]. Available: https://codepen.io/AfroDev/pen/emezKV.

[29] [Online]. Available: https://github.com/rnrn0909/yohohomqtt.git.

[30] [Online]. Available: https://youtu.be/TcjYQrdDYVU.