

A Payload of Lies: False Data Injection Attacks on MQTT-based 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—In the ever-evolving landscape of Industrial Internet of Things (IIoT), security emerges as a critical concern. This paper delves into the realm of False Data Injection Attacks (FDIAs) within MQTT-based IIoT systems, specifically targeting the publisher-subscriber model. Our exploration unveils two distinct attack scenarios that exploit the vulnerabilities inherent in the communication fabric. In the first scenario, we demonstrate the potential chaos wrought by sending false data to subscribers, manipulating their perception and inducing actions that align with the attacker's whims. The second scenario ventures into the heart of the publisher, where the attacker injects false data – deceptive status updates from other publishers e.g., Programmable Logic Controllers (PLCs). The repercussions ripple through the entire industrial process, impacting operations based on fraudulent information. This showcases the cascading effects of FDIAs, illustrating the profound threat they pose to the reliability and integrity of IIoT systems. For real-world attack scenarios, Our attacks were conducted on a small MQTT-based IIoT system, using the Fischertechnik Lernfabrik 4.0 9V factory. Finally, we propose mitigation solutions to safeguard IIoT systems from the far-reaching consequences of false data manipulation. Our attack codes as well as a proof-of-concept are publicly available for further research.

Index Terms—FDIAs, IIoT, IoT, MQTT Protocol.

I. INTRODUCTION

In the contemporary tapestry of our technological landscape, the advent of Industrial Internet of Things (IIoT) systems has become inseparable from the fabric of our daily lives. Revolutionizing the industrial sector, these interconnected systems imbue processes with unprecedented efficiency and responsiveness [1], [2]. At the heart of this revolution lies the MQTT (Message Queuing Telemetry Transport) protocol [3], a lightweight and effective communication protocol that has streamlined the connectivity of IoT systems, particularly in the industrial domain. Its ability to facilitate real-time data exchange between devices, coupled with its low bandwidth requirements, has ushered in a new era of connectivity, enabling diverse industrial components e.g., Programmable Logic Controllers (PLCs) to communicate effortlessly. As a result, MQTT has become the linchpin of IIoT, acting as the conduit through which disparate devices, from sensors to PLCs to end-users, harmoniously exchange information.

Unfortunately, this privilege connectivity has not been without its challenges. As IIoT systems proliferate, they have

become enticing targets for malicious actors seeking to exploit vulnerabilities [4], [6]. Among the myriad threats faced by MQTT-based IIoT systems, one particularly insidious menace is the False Data Injection Attack (FDIA) [5]. This type of attack poses a significant challenge in the context of IIoT systems. Adversaries can manipulate devices to exhibit malicious behavior or inject false data into the communication, potentially leading to outcomes such as data leakage or system failure.

This paper delves into the consequences of FDIAs in the context of MQTT based IIoT systems, highlighting the significant impact of misleading data on the trustworthiness and stability of industrial processes. To this end, we demonstrate two FDIA scenarios. In the first scenario, we illustrate the vulnerability of the publisher-subscriber network model to FDIAs. Our demonstration involves attackers sending crafted data to subscribers, leading them to execute actions aligned with the attacker's objectives. This manipulation of data within the communication framework highlights the potential compromise of the entire industrial process. In the second attack scenario, we focus on manipulating publishers by injecting false data, specifically deceptive status updates, from other publishers within the network, such as other PLCs. The implications of this attack are significant, as PLCs and other IoT devices make operational decisions based on the false data they receive, thereby jeopardizing the integrity of the entire industrial process.

Both of our attack scenarios showcase the cascading effects of FDIAs, emphasizing the profound threat they pose to the reliability and integrity of IIoT systems. To provide real-world context, we implement our attacks on an MQTT-based IIoT system, utilizing the Fischertechnik Lernfabrik 4.0 9V factory¹ as the testing environment. In this setup, the user-dashboard functions as a subscriber, while the TXT controllers serve as PLCs. Finally, we propose mitigation solutions to safeguard IIoT systems from the far-reaching consequences of false data injection attacks. To facilitate further research, we have made our attack codes [17] and a proof-of-concept [18] publicly available, encouraging a collaborative effort in addressing and fortifying against these emerging threats in IIoT security.

The rest of the paper is structured as follows. Section

¹<https://www.fischertechnikwebshop.com/en-gb/trainingfactory-4-0-en-gb>

II discusses related works, while Section III presents the experimental setup we used in this paper. In Section IV, we provide an overview of both MQTT protocol, while Section V presents our attack approach. Afterwards, we suggest some security recommendations and mitigation solutions in Section VI, and Section VII concludes the paper.

II. RELATED WORK

Several prior studies have underscored vulnerabilities in MQTT-based IIoT systems regarding FDIAs. Xu et al. [7] conducted an FDIAs scenario on an IIoT system, specifically a Kalman filter system. They successfully injected malicious measurements, disrupting system operation and highlighting the impact on data integrity in IIoT networks. However, their study did not address the security of the MQTT protocol, which is the primary focus of our work. In another study, a different research group [8] demonstrated adversaries' capabilities to generate false reports, modify data, and tamper with sensor readings in IIoT systems. These actions pose challenges for end-users in ensuring device security. Additionally, a separate investigation [9] analyzed FDIAs against IIoT systems and proposed a fault diagnosis strategy to identify such attacks. However, their assumption of the attacker possessing full system knowledge is considered unrealistic for real-world attack scenarios. Al-Ani et al. [10] showcased that adversaries with access to IoT systems can manipulate MQTT packet payloads, altering sensor data transmitted in plaintext using the standard MQTT protocol. In alignment with our work, we manipulate MQTT packets containing sensitive data related to controller statuses, targeting the entire publisher-subscriber MQTT model. Our specific emphasis on addressing MQTT protocol security aims to enhance the resilience of IIoT systems against sophisticated FDIAs threats. Bostami et al. [11] conducted an analysis of various FDIAs targeting different IoT-based systems. Their study revealed that the implications of FDIAs on smart grids and other IIoT systems have not been sufficiently addressed. Therefore, our work concentrates on FDIAs scenarios against IIoT-based systems, especially those utilizing MQTT as a core protocol for transmitting sensitive data across the network. This focus enables us to contribute to the understanding and mitigation of FDIAs threats in IIoT systems, with a particular emphasis on the MQTT communication framework.

III. EXPERIMENTAL SETUP

To validate the attack approach outlined in this paper, we employed a Fischertechnik training factory, also known as Lernfabrik 4.0 9V. This factory is dedicated to the development of Industry 4.0 applications, as depicted in Figure 1. The factory facilitates the simulation, understanding, and application of automated industrial activities on a small scale before scaling up. Comprising four industrial workstations [13]:

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

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

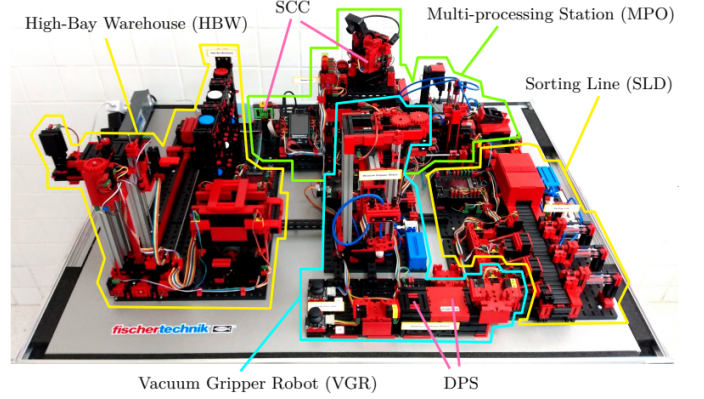


Fig. 1: Fischertechnik Lernfabrik 4.0 9V, adopted from [13].

Beyond these workstations, the factory integrates diverse modules, including 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 moving small workpieces of different colors (white, red, and blue) within the environment. Each workpiece is assigned a unique NFC identification tag, enabling the recording of color and production history, including date and time information. The factory configuration comprises six Fischertechnik TXT 4.0 controllers (refer to Figure 2).

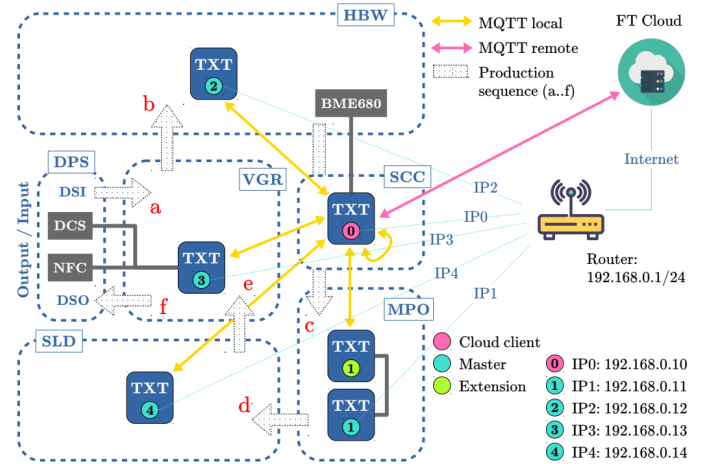


Fig. 2: Block Diagram of the Lernfabrik 4.0, adopted from [13].

The TXT controllers communicate bidirectionally with the central TXT-0, serving as an MQTT broker for controllers linked to a Wi-Fi network via a TP-Link Router. Configured as a cloud client, TXT-0 connects to Fischertechnik's cloud through a remote MQTT bridge. Users access a cloud

dashboard via a web browser to monitor and interact with specific factory production processes. TXT-0, acting as a cloud master, features a Universal Serial Bus (USB) camera and environmental sensor for air pressure, humidity, temperature, and gases. TXT-1 to TXT-4 controllers function as cloud masters, with TXT-1 having an extension. TXT-2 manages the warehouse, TXT-3 controls the DPS module, and TXT-4 oversees the SLD station. The block diagram outlines the standard manufacturing sequence from DSI to DSO, and each TXT controller runs compiled C++ code for its station.

IV. MQTT PROTOCOL

A. Overview

MQTT stands out as a lightweight messaging protocol explicitly crafted for deployment in scenarios characterized by low bandwidth, high latency, or unreliable network conditions [12]. Figure 3 depicts the structure of an MQTT packet.

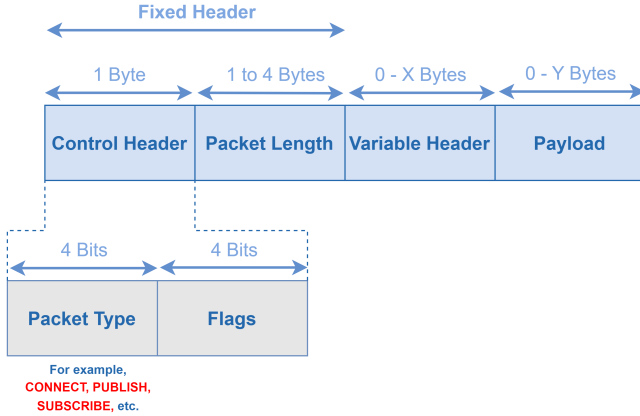


Fig. 3: MQTT Packet Structure

It consists of a fixed header and, in some cases variable headers, and payload. The format of the packet is determined by the packet type as follows:

1) **Fixed Header:** The fixed header is present in all MQTT packets and includes the following fields:

- **Packet Type** (4 bits): specifies the type of MQTT packet (*CONNECT*, *PUBLISH*, *SUBSCRIBE*, etc.).
- **Flags** (4 bits): Various flags that depend on the packet type, e.g., Quality of Service (QoS) level, Duplicate (DUP) flag, retain flag, etc.
- **Packet Length** (variable length): represents the length of the variable header and payload. It uses a variable-length encoding scheme with each byte using 7 bits for the length.

2) **Variable Header:** The variable header's structure depends on the packet type. It contains additional information about the packet.

3) **Payload:** The payload contains the actual data being transmitted. Its structure depends on the packet type. For a *CONNECT* packet, the payload contains client identifier, will topic, will message, username, and password. While for *PUBLISH* packet, the payload contains message topic,

message payload and list of topics filters and requested QoS levels for *SUBSCRIBE* packet.

The MQTT protocol operates on a publish-subscribe model, offering an efficient communication framework. Essential components that constitute the backbone of the MQTT protocol encompass the following key elements:

- **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 by utilizing the Transmission Control Protocol/Internet Protocol (TCP/IP). These devices are capable of functioning as publishers, subscribers, or both within the MQTT communication framework. The interaction between IoT devices is facilitated through MQTT topic names, which are designated by publishers in their messages and subsequently subscribed to by subscribers. Ensuring adherence to the prescribed convention in defining MQTT topic names is crucial for the seamless transmission of MQTT messages between IoT devices (e.g., PLCs) and cloud servers. Notably, publishers and subscribers are relieved from the necessity of being aware of the source and destination IP addresses for effective communication, as highlighted in [7].

Each MQTT message dispatched from a publisher to the broker is characterized by a specific topic and payload. The payload, in turn, encapsulates the substantive data being transmitted. For instance, a PLC might publish a message on the topic "sensor/temperature," providing the current temperature reading within the payload. This systematic structuring of MQTT messages enables efficient and targeted communication between IoT devices and their associated cloud servers.

B. Security Issues in MQTT

The MQTT protocol presents numerous security vulnerabilities, as outlined in [10], which could potentially give rise to various attack scenarios against IoT and IIoT systems. Here, we specifically highlight the security issues that render the MQTT protocol susceptible to FDAs.

1) **Lack of End-to-End Encryption:** MQTT, as a default, lacks inherent end-to-end encryption. Unencrypted communication between MQTT clients and brokers exposes messages to interception and manipulation. In our Lernfabrik scenario, an attacker intercepts and alters sensor readings transmitted from TXT controllers to the MQTT broker (TXT-0), introducing false information into the system.

2) **Unsecured Broker Configuration:** Improperly configured MQTT brokers create vulnerabilities, allowing attackers to exploit weaknesses. For instance, if a broker permits anonymous connections or lacks robust access controls, attackers can inject false data or compromise communication integrity. In our Lernfabrik example, an attacker leverages a misconfigured

broker to gain administrative privileges, enabling manipulation of critical settings, injection of false data, or disruption of MQTT model operations.

3) **Insufficient Authentication Mechanisms:** Weak authentication, like easily guessable passwords or the absence of robust methods (e.g., client certificates), opens avenues for unauthorized access. In our example, an attacker employs brute-force techniques to compromise the MQTT Broker, gaining unauthorized access to publishers and subscribers and manipulating packets in transit between IoT devices.

4) **Message Tampering:** Without integrity checks, MQTT messages are susceptible to tampering during transit. In our Lernfabrik, an attacker intercepts and alters control commands transmitted from a TXT controller to a mini motor through the MQTT broker, inducing Fischertechnik to execute incorrect actions.

5) **Topic-Based Authorization Issues:** MQTT's topic-based publish-subscribe model is vulnerable to insecure authorization mechanisms. In our factory context, an attacker may subscribe to a topic with critical data, intercept legitimate MQTT messages, and inject manipulated data into the same topic, resulting in the processing of false information.

6) **Misuse of Retained Messages:** MQTT's support for retained messages introduces a potential avenue for injecting false data if not configured and monitored properly. Without vigilance, retained messages can lead to unexpected behavior and compromise data integrity.

V. FALSE DATA INJECTION ATTACK APPROACH

In Figure 5, we present a high-level overview of the FDIAs we conducted on our Fischertechnik Lernfabrik 4.0 system presented in Figure 1. Our attack comprises four main phases:

- Breaking the authentication of the TP-Link Router.
- Compromising the authentication of the MQTT Broker.
- Identifying Targeted Topics (Sniffing Attack).
- Topics Substitution.

In this work, we omit the detailed depiction of the first two phases, which has been extensively covered in our previous paper [2]. Our emphasis in this paper is solely on phases 3 and 4, as elucidated in the following subsections.

A. Sniffing the MQTT Topics (MitM)

In MQTT-based IIoT systems, an attacker with access to the MQTT network, can employ various techniques to eavesdrop on MQTT topics, potentially compromising the confidentiality of sensitive information. To this end, an attacker can utilize a packet sniffing tool, such as Wireshark², to capture and inspect MQTT packets traversing the network. This allows him/her to scrutinize the content of messages related to targeted topics. Subsequently, armed with a detailed understanding of the legitimate data flow, the attacker can inject false or deceptive information into the MQTT topics during transmission. This manipulative approach seeks to sow confusion among legitimate participants in the IIoT network, potentially leading to

erroneous decision-making or compromising the integrity of critical processes relying on the exchanged MQTT messages.

Figure 6 depicts a *PUBLISH* message featuring a specific topic, "f/i/state/hbw." This message originates from the publisher, in this case, the TXT-2 controller, and is transmitted to the MQTT broker, subsequently reaching the Subscriber, represented in our case by the User-dashboard. The Subscriber actively subscribes to this topic, facilitating the retrieval of data from the message to update the dashboard accordingly.

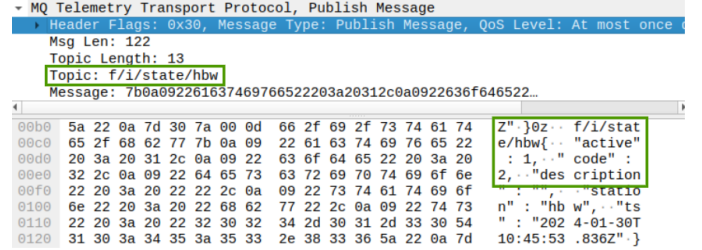


Fig. 6: MQTT *PUBLISH* Message showing the status of the HBW workstation to the User

The purpose of the "f/i/state/hbw" topic is to communicate the HBW status on the dashboard. The status indicators include whether the HBW is available (indicated by a green light), waiting (indicated by a yellow light), or busy (indicated by a red light). This information is extracted from the "active" and "code" fields within the *PUBLISH* packet. The dynamic updating of the dashboard based on these fields ensures that users are promptly informed about the current state of the HBW. Figure 7 shows the various topics we identified through the Wireshark software. These topics encompass sensitive data that could be exploited by an attacker, potentially resulting in significant damage to the entire system as illustrated in the next subsection.

B. Sending Manipulated MQTT Topics (FDIA)

Figure 8 depicts the both scenarios of our FDIA. In the following, we illustrate each scenario in more detail.

1) Infecting Subscribers:

In this scenario, the attacker creates deceptive MQTT topics that closely resemble the legitimate ones associated with the user-dashboard data. The goal is to deceive the user-dashboard into interpreting and displaying manipulated data as genuine. Specifically, for our Lernfabrik system, we designed a malicious topic named "f/i/stock", as shown in Figure 9. This topic aims to present a detailed overview of the materials stored within HBW, encompassing information such as quantity, unique identifiers, locations, colors, and conditions (indicating whether a material is in its *RAW* or *PROCESSED* state).

During the material-saving process, the DSO (NFC reader) reads the identifier of each material, and this intricate information is conveyed through the payload of the *PUBLISH* "f/i/stock" packet. However, a security vulnerability exists in the user-dashboard, as it neglects to validate the authenticity of material identifiers. This flaw allows potential attackers to manipulate the "id" field by inputting arbitrary characters,

²<https://www.wireshark.org/>

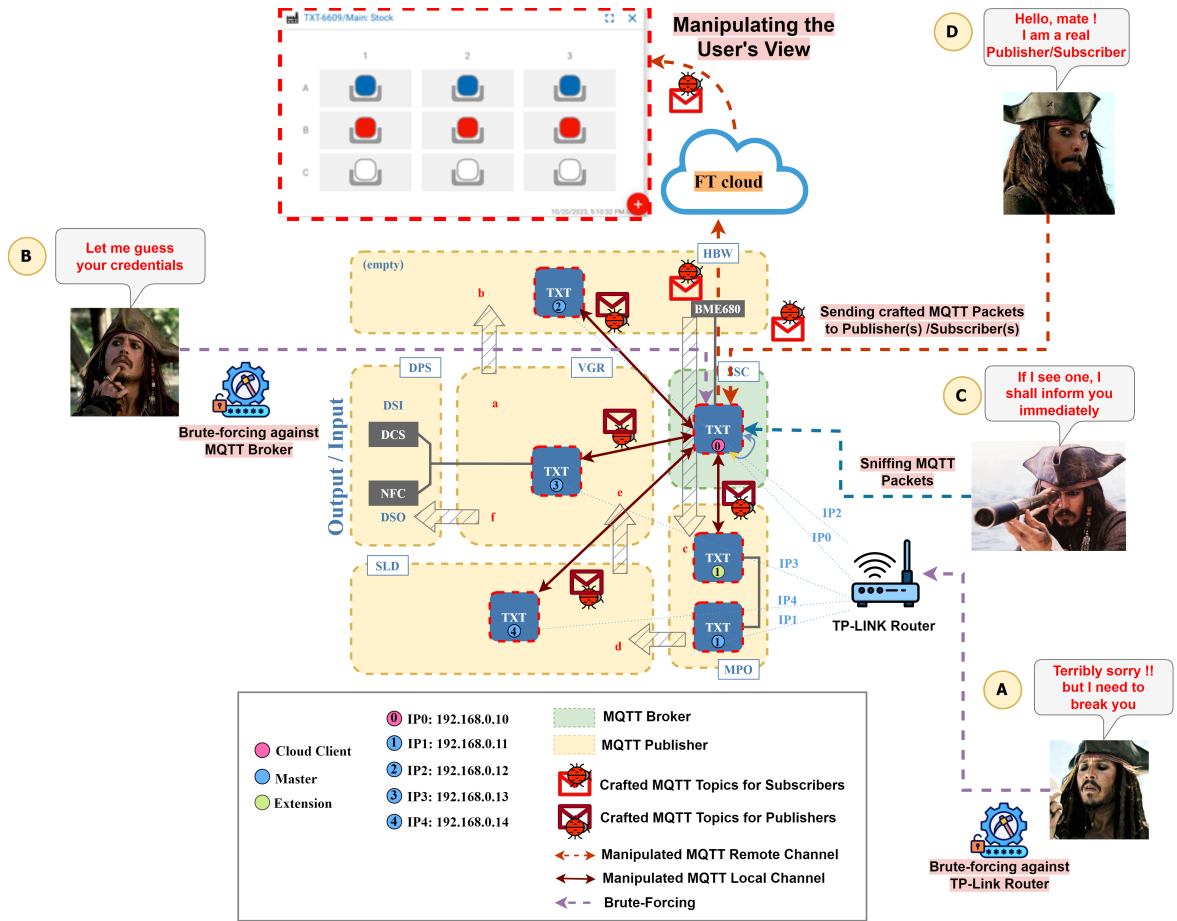


Fig. 5: High-level Overview of our False Data Injection Attack.

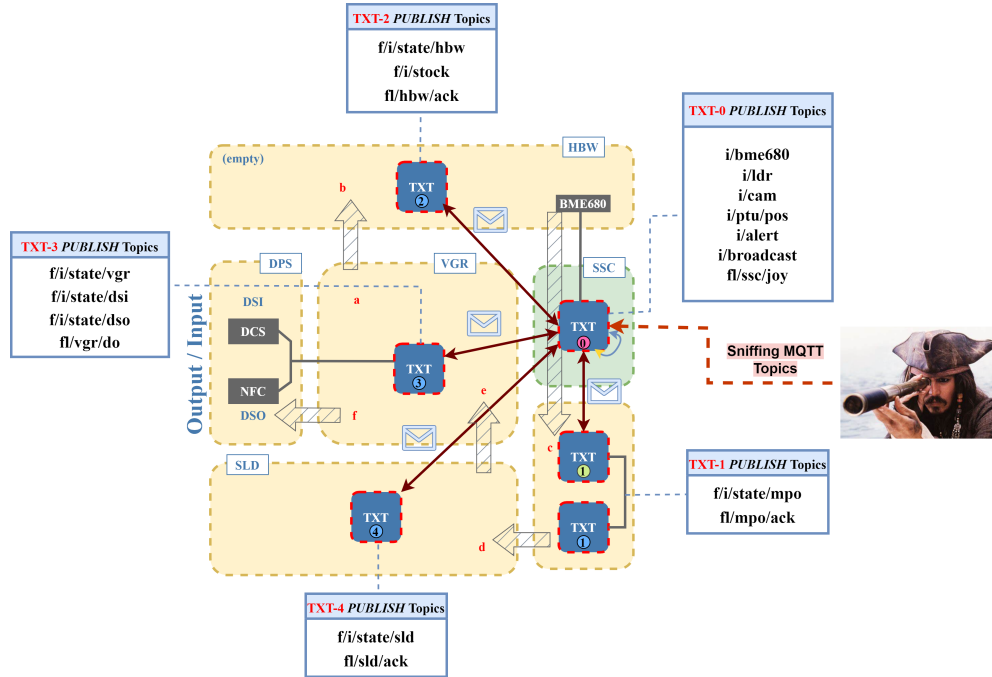


Fig. 7: MQTT *PUBLISH* Topics transmitted over the MQTT network of our Lernfabrik 4.0 9V.

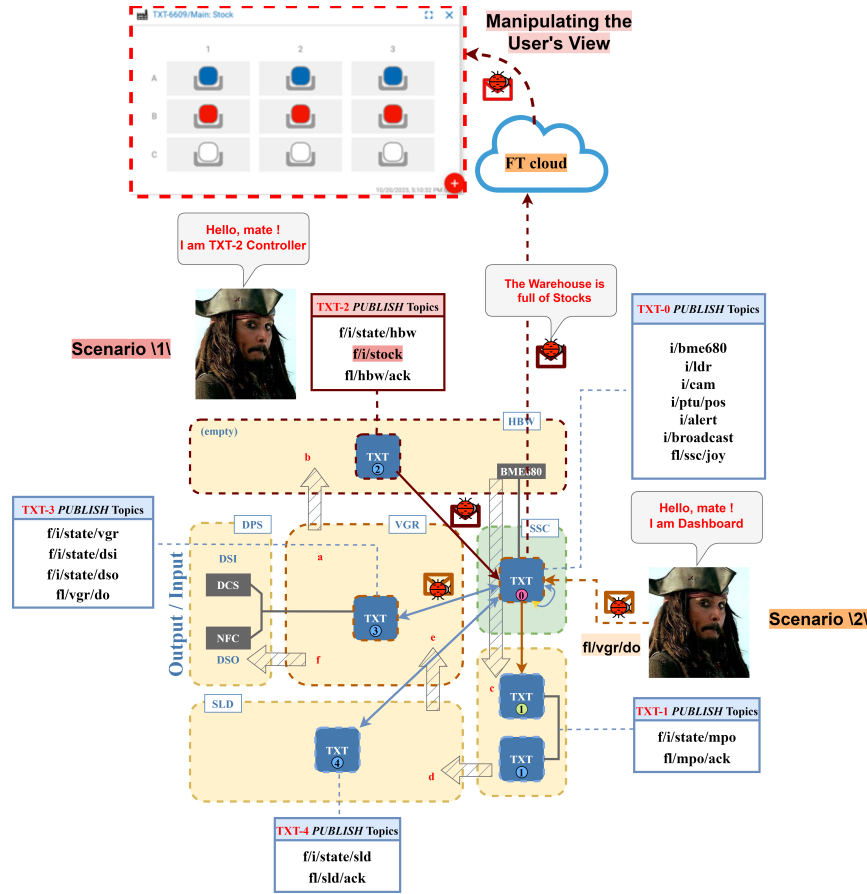


Fig. 8: High-level Overview of our Injection Scenarios: Scenario /1/ Infecting a subscriber with a manipulated Topic; Scenario /2/ Infecting a publisher with a manipulated Topic.

like "0000000000" or "abcdefghij," as presented in Figure 9b. Despite this security concern, the user-dashboard dynamically updates itself based on the information from these fields, ensuring that users receive real-time updates about the current status of stocks in HBW.

Once the attacker substitutes the genuine topic with manipulated ones, they inject the latter into the MQTT message stream. This ensures that the malicious topic infiltrates the user-dashboard of subscribed users, effectively replacing authentic data with manipulated information. In Figure 9, two screenshots of the user-dashboard are also presented. The one highlighted in green (Figure 9a) represents the original view, indicating that the HBW is empty. However, the user is presented with the view highlighted in red (Figure 9b), falsely suggesting that the HBW is full of stocks. What exacerbates this attack scenario is that, if the user proceeds to order a stock from the HBW, believing it to be full, the robot arm (VGR) will move to the specified location as requested by the user. Unfortunately, the user remains unaware that no actual stock is available. In real-world IIoT systems, such a scenario can lead to significant damages within the facility.

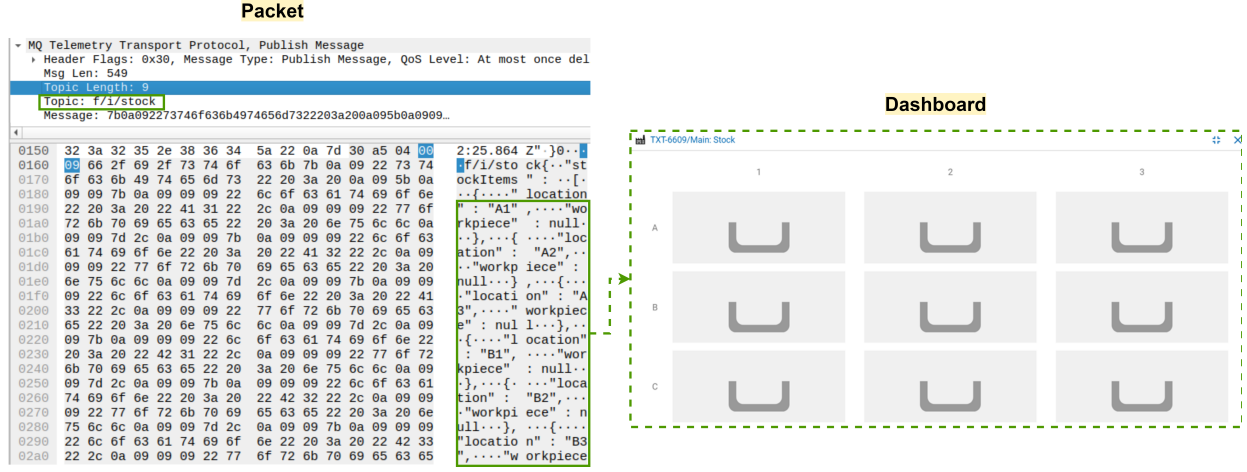
2) Infecting Publishers:

In this attack scenario, the adversary aims to compromise the system by injecting falsified data into the controllers. For

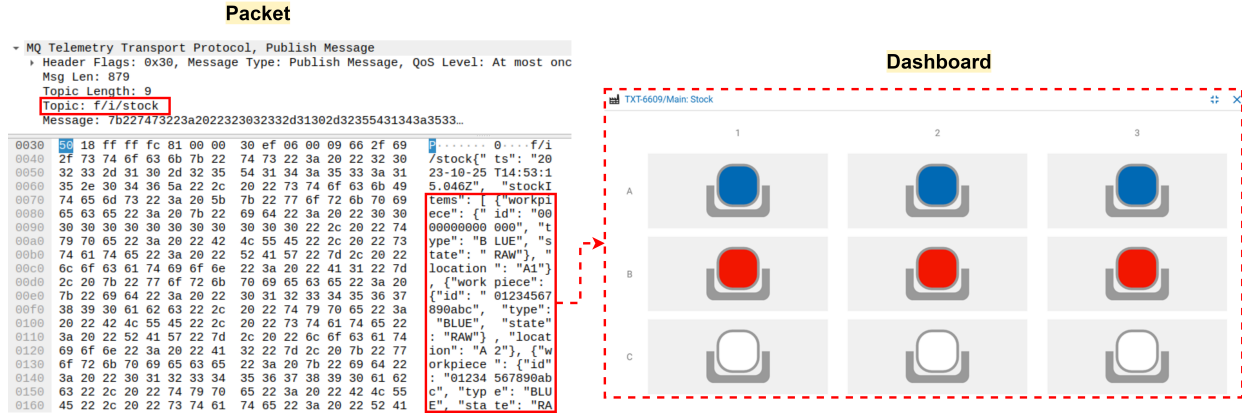
our system, the TXT-3 controller (VGR) utilizes MQTT packets with the 'fl/vgr/do' topic to orchestrate crucial movements for either the HBW or the MPO workstation. The payload within these packets includes a specific code, indicating the subsequent operation in coordination with other stations.

Upon receiving packets related to an order with the 'f/o/order' topic from the user-dashboard, VGR promptly confirms order reception by dispatching a packet to the 'f/i/order' topic. Following this, VGR publishes MQTT packets to the 'fl/vgr/do' topic, embedding the code '3,' instructing the robot arm to move towards the HBW for the retrieval of the designated workpiece. The code '3' directs the robot arm to procure the target workpiece from HBW. After successfully retrieving the workpiece, VGR facilitates its transfer to MPO through an MQTT packet to 'fl/vgr/do' with a unique code. In response, MPO initiates its production process. Upon completion, MPO releases the workpiece onto the conveyor belt, and the SLD efficiently categorizes the workpiece using color recognition algorithms.

In the event of a security breach, the attacker can craft an MQTT packet with the topic 'fl/vgr/do' and the code '7' (See Figure 10), illicitly initiating MPO's production process. This particular code functions as a misleading signal to MPO,



(a) Original "f/i/stock" MQTT Topic



(b) Crafted "f/i/stock" MQTT Topic

Fig. 9: Crafting "f/i/stock" MQTT Topic to fabricate the user view on dashboard

falsely suggesting that the workpiece is ready for processing, thereby triggering its operational sequence.

VI. MITIGATION SOLUTIONS

Mitigating False Data Injection Attacks in MQTT-based IIoT systems, particularly those targeting topics substitution, requires a combination of technical measures and best practices. In the following, we recommend some key mitigation strategies if applied, our presented attack would not be successful.

A. Message Validation and Filtering

1) **Topic Structure Validation:** This includes several aspects. For instance, Implementing stringent checks on the structure of MQTT topics. Defining and enforcing a clear pattern for topic naming conventions. Rejecting any messages with topics that deviate from the predefined structure.

2) **Regular Expression Patterns:** Utilize regular expressions to define and validate topic patterns. This allows for flexible yet controlled matching, ensuring that subscribed topics conform to expected formats.

B. 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 employ Transport Layer Security (TLS) or its predecessor, Secure Sockets Layer (SSL), to encrypt MQTT communication [14]. This prevents adversaries from intercepting and tampering with data in transit, adding an extra layer of security.

C. Topic Whitelisting

Implementing topic whitelisting restricts MQTT clients to subscribe only specific topics as proposed in [15]. This helps to control the flow of information and prevents unauthorized access.

D. 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

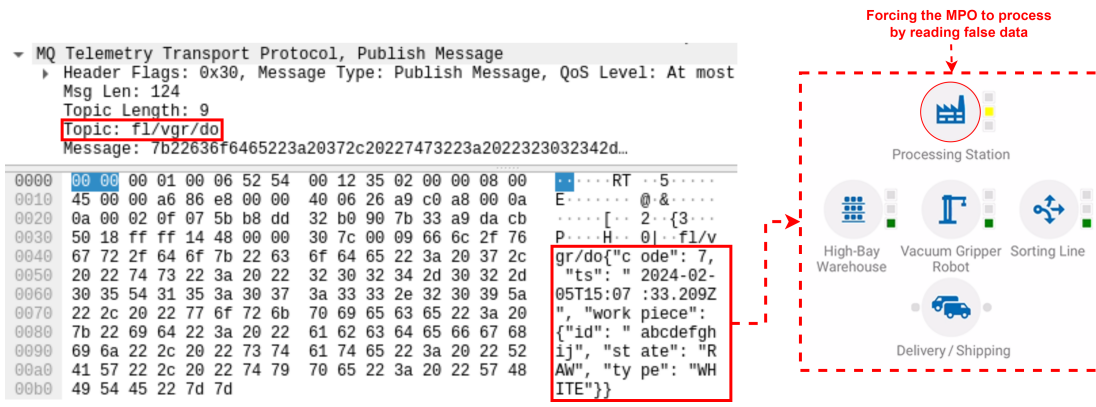


Fig. 10: Crafted *fl/vgr/do* Topic to manipulate the MPO operation.

manipulated. For all that, it is recommended to implement an integrity check mechanism e.g., [16] that adheres to the expected format and does not contain malicious content.

VII. CONCLUSION

This paper extensively explores FDIA scenarios in MQTT-based IIoT systems, exploiting vulnerabilities in the MQTT protocol that can lead to severe scenarios affecting both publishers and subscribers. Our experiments demonstrate the misinformation relayed to users about workstation status and the manipulation of MQTT topics directed at controllers, causing abnormal actions. The severity underscores potential harm if executed against real IIoT systems.

To mitigate these impacts, we propose security solutions, rendering our attack unsuccessful or enabling early detection by users. Demonstrating transparency and a commitment to further research, we have made the proof of concept and attack codes publicly accessible. This paper not only raises awareness of MQTT-based IIoT vulnerabilities but also provides actionable insights to fortify against threats. Sharing our findings and proposed solutions contributes to the ongoing dialogue on IIoT system security, fostering advancements in safeguarding critical infrastructure.

REFERENCES

- [1] W. Alsabbagh and P. Langendörfer, "Security of Programmable Logic Controllers and Related Systems: Today and Tomorrow," IEEE Open Journal of the Industrial Electronics Society, vol. 4, pp. 659-693, 2023, doi: 10.1109/OJIES.2023.3335976.
- [2] W. Alsabbagh, C. Kim, and P. Langendörfer, "Silent Sabotage: A Stealthy Control Logic Injection in IIoT Systems," Submitted at the 5th Silicon Valley Cybersecurity Conference (SVCC 2024), June 17 - 19, 2024, South Korea, doi: 10.13140/RG.2.2.33854.25925.
- [3] 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 (COM-SWARE), 2008, pp. 791-798.
- [4] A. C. Panchal, V. M. Khadse and P. N. Mahalle, "Security Issues in IIoT: A Comprehensive Survey of Attacks on IIoT and Its Countermeasures," 2018 IEEE Global Conference on Wireless Computing and Networking (GCWCN), Lonavala, India, 2018, pp. 124-130, doi: 10.1109/GCWCN.2018.8668630.
- [5] M. A. Rahman and H. Mohsenian-Rad, "False data injection attacks with incomplete information against smart power grids," 2012 IEEE Global Communications Conference (GLOBECOM), Anaheim, CA, USA, 2012, pp. 3153-3158, doi: 10.1109/GLOCOM.2012.6503599.
- [6] D. Serpanos, "Industrial Internet of Things: Trends and Challenges," in Computer, vol. 57, no. 1, pp. 124-128, Jan. 2024, doi: 10.1109/MC.2023.3331552.
- [7] H. Xu, W. Yu, X. Liu, D. Griffith and N. Golmie, "On Data Integrity Attacks against Industrial Internet of Things," 2020 IEEE Intl Conf on Dependable, Autonomic and Secure Computing, Intl Conf on Pervasive Intelligence and Computing, Intl Conf on Cloud and Big Data Computing, Intl Conf on Cyber Science and Technology Congress (DASC/PiCom/CBDCOM/CyberSciTech), Calgary, AB, Canada, 2020, pp. 21-28, doi: 10.1109/DASC-PiCom-CBDCOM-CyberSciTech49142.2020.00020.
- [8] W. G. Hatcher, C. Qian, F. Liang, W. Liao, E. P. Blasch and W. Yu, "Secure IoT Search Engine: Survey, Challenges Issues, Case Study, and Future Research Direction," IEEE Internet of Things Journal, vol. 9, no. 18, pp. 16807-16823, 15 Sept.15, 2022, doi: 10.1109/IJOT.2022.3153171.
- [9] Y. Jiang, J. Dong and S. Yin, "Improving the safety of distributed cyber-physical systems against false data injection attack by establishing interconnections," IECON 2020 The 46th Annual Conference of the IEEE Industrial Electronics Society, Singapore, 2020, pp. 2623-2628, doi: 10.1109/IECON43393.2020.9254328.
- [10] 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.
- [11] B. Bostami, M. Ahmed and S. Choudhury, "False Data Injection Attacks in Internet of Things," Al-Turjman, F. (eds) Performability in Internet of Things. EAI/Springer Innovations in Communication and Computing. Springer, Cham, doi: 10.1007/978-3-319-93557-7_4.
- [12] A. H. Eyeleko and T. Feng, "A Critical Overview of Industrial Internet of Things Security and Privacy Issues Using a Layer-Based Hacking Scenario," IEEE Internet of Things Journal, vol. 10, no. 24, pp. 21917-21941, 15 Dec.15, 2023, doi: 10.1109/IJOT.2023.3308195.
- [13] 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.
- [14] I. L. B. M. Paris, M. H. Habaebi and A. M. Zyoud, "Implementation of SSL/TLS Security with MQTT Protocol in IIoT Environment," Wireless Pers Commun 132, 163-182 (2023), doi: 10.1007/s11277-023-10605-y.
- [15] 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.
- [16] Ch. Patel, N. Doshi, "A Novel MQTT Security framework In Generic IIoT Model", Procedia Computer Science, v. 171, 2020, pp. 1399-1408, doi: 10.1016/j.procs.2020.04.150.
- [17] [Online]. Available: <https://github.com/rnrm0909/a-payload-of-lies>.
- [18] [Online]. Available: <https://www.youtube.com/watch?v=coOLEo1PNTY>.