future internet

MDPI

Article

A Deception-Based Access Control Mechanism for Protecting PLCs from ModbusTCP Brute-Force Attacks in IIoT Environments †

Mohammad AbdulJawad [1,2,*], Mohammad Z. Masoud [3], Álvaro Álesanco [2] and José García [2]

[1] Software Engineering Department, Al-Zaytoonah University of Jordan, Amman 11733, Jordan
[2] Aragon Institute of Engineering Research (I3A), University of Zaragoza, 50018 Zaragoza, Spain; alesanco@unizar.es (Á.Á.); jogarmo@unizar.es (J.G.)
[3] Electrical Engineering Department, Al-Zaytoonah University of Jordan, Amman 11733, Jordan; m.zakaria@zuj.edu.jo
* Correspondence: m.abduljawad@zuj.edu.jo
† This paper is an extended version of our paper published in 2025 12th International Conference on Information Technology (ICIT), A Distributed Brute-Force Attack and Misleading Countermeasure for Securing ModbusTCP Implementations in PLCs, Amman, Jordan, 27–30 May 2025.

Abstract

Industrial control systems (ICSs) increasingly rely on legacy communication protocols such as ModbusTCP, which lack built-in security mechanisms and remain widely exposed to network-based attacks. This paper investigates the security limitations of authentication mechanisms in ModbusTCP-enabled programmable logic controllers (PLCs) and demonstrates how plaintext credential transmission and limited connection handling capabilities can be exploited to perform brute-force and denial-of-service (DoS) attacks. An experimental testbed based on two industrial Delta PLC families (DVP-13SE and DVP-311SV3) was developed to systematically evaluate these vulnerabilities under realistic conditions. The results show that authentication credentials can be easily captured through network sniffing, while the PLC communication stack supports a maximum of 16 concurrent connections and can process up to approximately 8600 Modbus operations per second, making it susceptible to resource exhaustion and performance degradation under distributed attack scenarios. To address these limitations, this paper proposes a lightweight deception-based protection mechanism, termed the PLC misleading algorithm (PMA), which is implemented directly within the PLC ladder logic. Unlike traditional network-level defenses, PMA operates at the device level and dynamically misleads attackers by generating controlled randomized responses while preserving consistent behavior for legitimate clients. Experimental results demonstrate that PMA significantly mitigates brute-force effectiveness by preventing reliable password extraction while introducing minimal overhead (2.2% memory usage) and maintaining acceptable communication latency. Additionally, the proposed approach significantly reduces observable attack traffic, with only 0.246 Modbus operations per second observed during the attack phase, thereby limiting the effectiveness of automated exploitation tools. These findings highlight the potential of in-device deception mechanisms as a practical and deployable security layer for legacy industrial systems, and provide new insights into the resilience of PLC-based infrastructures against network-level attacks. This work bridges the gap between lightweight PLC-level protections and the growing need for robust cybersecurity mechanisms in industrial IoT environments.

Keywords: ModbusTCP; Industrial Internet of Things (IIoT); brute-force attack; denial of service (DoS); programmable logic controller (PLC); obfuscation; authentication

Academic Editors: Wei Zong, Yang-Wai Chow and Gianluigi Ferrari

Received: 25 March 2026
Revised: 30 April 2026
Accepted: 11 May 2026
Published: 14 May 2026

1. Introduction

The rapid expansion of the Internet of Things (IoT) has transformed modern technological infrastructures by enabling billions of devices to communicate, exchange data, and automate decision-making processes [1]. This transformation is particularly evident in industrial environments through the emergence of the Industrial Internet of Things (IIoT) and the concept of Industry 4.0 [2], where sensors, controllers, and machines are interconnected to enable smart manufacturing, predictive maintenance, and the real-time monitoring of industrial processes. It has been shown in industrial reports and news that more than 65% of manufacturing companies have integrated IIoT technologies into their production systems and that approximately 70% of industrial processes rely on connected devices for monitoring and automation [3]. While these technologies significantly improve efficiency and productivity, they also introduce major cybersecurity challenges [4]. Industrial systems have traditionally been designed by control engineers who prioritize reliability and real-time performance rather than cybersecurity practices, resulting in many industrial networks lacking strong authentication, encryption, and intrusion detection mechanisms [5].

Among the various components used in industrial automation systems, the programmable logic controller (PLC) is considered the most widely deployed control device in industrial control systems [6]. PLCs are responsible for monitoring sensors, executing control logic, and actuating industrial equipment in real-time. It has been reported that the global PLC market was valued at more than USD 14 billion and is expected to exceed USD 22 billion by 2030 [7]. Furthermore, more than 41 million PLC units were deployed worldwide in 2023 [8]. PLCs are widely used in several industrial sectors, including manufacturing [9], energy [10], transportation [11], and water treatment [12]. In addition, PLCs are the core components of distributed control systems and SCADA frameworks [13]. The widespread adoption of PLCs makes them a critical component of industrial infrastructures, and consequently, an attractive target for cyber attackers seeking to disrupt industrial operations [14].

With the evolution of IIoT architectures, PLCs are increasingly connected to enterprise networks, and in some cases, directly to the Internet to enable remote monitoring, diagnostics, and maintenance [15]. In many industrial environments, PLC communication over Ethernet relies on the widely used ModbusTCP protocol due to its simplicity, vendor interoperability, and ease of integration with supervisory control systems and industrial software platforms [16]. As a result, ModbusTCP has become one of the most commonly used protocols in industrial communication networks. However, the protocol was originally designed without strong security features such as encryption or robust authentication mechanisms. Consequently, PLC devices communicating over ModbusTCP may become vulnerable to various attacks, including brute-force authentication attempts, packet sniffing, and denial-of-service attacks, especially when deployed in IIoT environments with increased network exposure.

To address these challenges, this work introduces a deception-based protection mechanism implemented directly within the PLC logic. Unlike traditional industrial honeypots that simulate PLC behavior on external computers or virtual systems, the proposed PLC misleading algorithm (PMA) operates inside the real PLC and dynamically generates misleading responses to unauthorized authentication attempts. This approach allows the PLC to maintain normal operation for legitimate clients while disrupting automated brute-force attacks by returning randomized authentication outcomes. Moreover, the proposed mechanism is designed to be lightweight and suitable for resource-constrained PLC environments, requiring only a small portion of the available PLC memory and no additional hardware or external security infrastructure. As a result, PMA provides a practical and deployable

security enhancement for IIoT environments where many legacy PLCs lack strong built-in cybersecurity mechanisms.

The main contributions of this work are summarized as follows:

1- Design and implementation of the PLC misleading algorithm (PMA): a lightweight deception-based protection mechanism embedded directly into PLC that generates misleading authentication responses to disrupt automated brute-force attacks without modifying the ModbusTCP protocol.
2- A comprehensive empirical analysis of ModbusTCP authentication weaknesses, including vendor-specific function exploitation.
3- A fully embedded deception-based authentication mechanism implemented in PLC ladder logic without external components.
4- A multi-scenario experimental evaluation across different PLC families and distributed attack conditions.
5- A system-level analysis of PLC resource limitations and their security implications.
6- A comparative discussion positioning PMA relative to protocol-level and network-level defenses.

The rest of this paper is organized as follows. Section 2 reviews related work on ModbusTCP security and industrial honeypots. Section 3 introduces the PMA algorithm and describes its design and operation. Section 4 presents the experimental setup, while Section 5 discusses the obtained results. Finally, Section 6 concludes the paper and outlines future research directions.

2. Related Work

Existing research on Modbus and ICS security can be broadly categorized into four main directions: intrusion detection systems, deception-based honeypots, protocol-level security enhancements, and publicly available datasets and experimental platforms. Table 1 show a comparison of all related works.

2.1. Intrusion Detection Systems and Machine Learning Approaches

The lack of built-in authentication and encryption in ModbusTCP has led to extensive research on intrusion detection mechanisms for industrial control systems. Early Modbus studies focused on identifying concrete attack behaviors and designing rule-based detection mechanisms tailored to industrial protocols. Gao and Morris described 28 attacks targeting Modbus-based ICS environments and developed state-based intrusion detection rules, validating them using laboratory pipeline and tank systems [17]. Their efforts remain a cornerstone in the design of protocol-aware IDSs for industrial networks, subsequently yielding four public datasets derived from Modbus ICS test platforms [18].

In addition to rule-based detection approaches, machine learning techniques have been widely explored to model anomalous patterns in Modbus traffic. Duque Antón et al. evaluated several algorithms, including support vector machine (SVM), random forest, k-nearest neighbors (k-NN), and k-means clustering, using synthetic ModbusTCP traffic traces. Their results showed strong supervised detection performance, particularly for SVM and k-NN models in realistic traffic scenarios [19].

Other research has explored statistical traffic analysis methods. Ghosh et al. combined univariate and bivariate entropy features with tree-based classifiers, achieving high accuracy in detecting denial-of-service (DoS) attacks, although their approach showed lower performance when detecting more complex man-in-the-middle (MITM) attack stages [20]. Similarly, Varol and İskefiyeli proposed a Modbus-focused intrusion detection system evaluated on hybrid datasets using both machine learning and deep learning techniques, reporting competitive accuracy for critical infrastructure traffic [21].

Although these approaches demonstrate strong detection capabilities, they typically rely on external monitoring components that analyze network traffic rather than mechanisms embedded directly within industrial controllers.

2.2. Deception and ICS Honeypot Systems

Deception technologies have also been utilized to study and thwart adversaries targeting industrial devices. Liu et al. designed a scalable ICS honeypot that mimics highly interactive and deployable Modbus devices [22]. More advanced deception frameworks have been proposed to increase realism. NeuralPot, for example, extends traditional honeypots by learning device behavior through autoencoder–GAN architectures, allowing the system to generate responses that more closely resemble real Modbus endpoints, thereby complicating attacker fingerprinting and reconnaissance activities [23]. Operational case studies further illustrate the risks associated with Modbus-based infrastructures. A study conducted on a wastewater treatment plant (WWTP) analyzed the operational impact of ModbusTCP attacks and evaluated potential risk mitigation strategies in a real ICS environment [24].

In addition to high-interaction deception platforms, low-interaction honeypot frameworks such as Conpot have become widely used tools in ICS cybersecurity research. Conpot provides configurable templates capable of emulating several industrial protocols, including Modbus, enabling researchers to analyze reconnaissance and probing activities in industrial networks [25]. However, these honeypot systems typically operate as external deception mechanisms deployed on separate hosts or virtual machines. As a result, they primarily function as observational platforms and do not directly protect the authentication mechanisms of operational PLC devices.

2.3. Protocol-Level Security Enhancements

Several studies have explored improvements to the security of the ModbusTCP protocol itself. The Modbus Security specification introduces transport layer security (TLS) encapsulation and role-based authorization mechanisms, representing a significant evolution of the protocol toward modern security practices and enabling certificate-based authentication mechanisms [26]. Nevertheless, the traditional protocol documentation, including the Messaging Implementation Guide and the Application Protocol Specification, remains essential for understanding the Modbus Application Protocol (MBAP) header structure, function code semantics, and the client–server communication model used in ModbusTCP networks [27,28]. These specifications provide the foundation for modeling both legitimate communication patterns and potential attack vectors in industrial control networks.

Beyond protocol specifications, several researchers have proposed additional protection mechanisms that can operate within existing industrial infrastructures. For example, kernel-level security mechanisms based on eBPF have been proposed to introduce transparent security transformations for ModbusTCP traffic without modifying legacy industrial applications [29]. Similarly, Zero Trust–oriented authentication and authorization mechanisms implemented over Modbus/TLS have been experimentally validated using Node-RED prototypes, demonstrating the feasibility of integrating modern access control principles into industrial communication protocols [30].

In addition to these software-based strategies, several researchers have proposed practical protection mechanisms that can be easily integrated into existing industrial infrastructures. Dao et al. developed a lightweight hardware module capable of detecting and mitigating ARP spoofing attacks affecting ModbusTCP traffic in Siemens S7 environments, providing a practical method to harden legacy industrial installations without modifying

PLC logic [31]. Similarly, Fagundes and da Cunha proposed the HB-MP mechanism, an application-layer master/slave authentication scheme designed to identify unauthorized devices and prevent identity spoofing attempts during ModbusTCP communications [31].

However, these approaches typically require either additional hardware modules, protocol extensions, or modifications to client–server communication logic, which may limit their deployment in legacy industrial environments.

2.4. Datasets and Experimental Platforms for Modbus Security Research

The development of reliable intrusion detection and security mechanisms requires realistic datasets and experimental testbeds. Several platforms and datasets have therefore been developed to support reproducible research in Modbus security. The MOCAST platform extended the SMOD penetration testing tool and introduced an anomaly-based Modbus DoS detection mechanism, which was validated using traffic collected from a hydroelectric power plant environment [32]. A number of public datasets have also become available for Modbus security research. These include the Modbus SYN flood dataset available on Kaggle [33], the CIC Modbus2023 dataset, which contains multiple attack scenarios and baseline industrial traffic [34], and the IEEE DataPort Modbus dataset, which includes 609,934 real Modbus packets suitable for byte-level modeling and GAN-based traffic synthesis [35]. These datasets and experimental platforms provide valuable resources for evaluating IDS models and security mechanisms, but they do not address the problem of protecting PLC authentication mechanisms directly within industrial controllers.

2.5. Research Gap and Positioning of the Proposed Approach

Although prior research on Modbus and ICS security includes intrusion detection systems, protocol-level defenses, public datasets, and deception-based platforms, most existing approaches address the problem from an external monitoring or simulation perspective. Honeypot systems—ranging from high-interaction deception environments to low-interaction frameworks such as Conpot—primarily function as observational tools used to study attacker behavior rather than mechanisms that actively protect operational PLC devices. Consequently, these external systems do not intervene when brute-force or credential-guessing attacks are directed at real controllers.

The approach proposed in this work differs fundamentally from these strategies by embedding the defensive logic directly within the PLC itself. By integrating the deception mechanism into the controller's authentication process, the proposed PLC misleading algorithm (PMA) allows the device to manipulate an attacker's perception of authentication outcomes and disrupt automated password-guessing attempts without modifying the Modbus protocol or requiring additional hardware. This design moves deception from the network perimeter to the authentication path of the PLC, providing a lightweight and deployable protection mechanism for industrial environments where legacy controllers often lack built-in cybersecurity capabilities.

Table 1. Comparison of related work in Modbus/ICS security.

Ref.	Focus	Method/Contribution	Data/Testbed	Attacks Covered	Key Result	Limitations
[17]	Modbus IDS rules	28 Modbus ICS attacks, signature & state-based rules	Pipeline & tank lab ICS	Recon, injection, DoS, etc.	Validated detection rules, per-class rates reported	Lab scope, rule coverage
[18]	Public datasets	4 datasets from 2 Modbus ICS testbeds	Gas pipeline & water tank	Multiple classes	Common benchmark for IDS	Legacy scenarios
[19]	ML anomaly detection	SVM, RF, k-NN, k-means on synthetic Modbus traffic	Synthetic ModbusTCP	Multiple	SVM/k-NN strong supervised accuracy	Synthetic bias
[20]	Entropy + ML	Univariate/bivariate entropy + J48/RF	ICS traces, Modbus + TCP features	DoS, MITM	High DoS accuracy, weaker for early-stage MITM	Reconnaissance under-detected

Table 1. *Cont.*

Ref.	Focus	Method/Contribution	Data/Testbed	Attacks Covered	Key Result	Limitations
[21]	Modbus IDS (CI)	AI-based IDS, supervised & ensemble models	Hybrid datasets	Common ICS attacks	High accuracy on CI traffic	Details behind paywall
[22]	ICS honeypot	High-interaction, extendable Modbus honeypot	Multi-agent, Shodan scoring	Recon, probing	High interaction vs. Conpot	Limited protocol scope
[23]	Honeypot (ML)	NeuralPot (GAN/AE-driven behavior cloning)	Lab Modbus	Recon, probing	Realistic response generation	Training coverage
[24]	Real-plant impact	WWTP case study of ModbusTCP attacks	Real WWTP	Various Modbus threats	Operational impact assessment	Site-specific
[25]	ICS Honeypot	Conpot: widely-used open-source ICS/SCADA honeypot supporting Modbus and multiple ICS protocols	Simulated ICS templates	Reconnaissance, probing	Provides realistic protocol emulation for attacker analysis, widely adopted in ICS research	External decoy only, does not protect real PLC authentication or disrupt brute-forcing attempts
[26]	Protocol spec	ModbusTCP Security (TLS, mbaps RBAC)	Specification	N/A	Defines secure Modbus profile	Requires PKI ops
[27]	Protocol guide	Messaging on TCP/IP (MBAP, client/server)	Specification	N/A	Reference for correct state/frames	—
[28]	Protocol spec	Application Protocol (function codes)	Specification	N/A	Canonical semantics	—
[29]	Kernel hardening	eBPF-based transparent Modbus security	OS-level	N/A	No app changes required	Preprint
[30]	Zero-Trust add-ons	AuthZ/AuthN over Modbus/TLS	Node-RED prototype	MITM-aware setup	Feasible lab validation	Prototype scope
[36]	ARP-spoofing Modbus defense	Hardware-based inline ARP-spoofing detector	Siemens S7-1500/1200	ARP spoofing	Protects legacy PLCs without code changes	Hardware cost and deployment overhead, scope limited to ARP spoofing
[31]	Master/slave authentication	HB-MP (application-layer Modbus/TCP authentication)	Custom Modbus network	Device impersonation	Identifies unauthorized nodes & blocks spoofing attempts	Requires modifying client/server logic, adds handshake overhead.
[32]	Attack platform	SMOD enhancements, Modbus IDS (DoS)	Hydropower plant data	DoS	81% Acc/77% F1 on real data	Focused on DoS
[33]	Dataset	Modbus SYN-flood dataset	Kaggle	SYN flood	Public benchmark	Narrow attack type
[34]	Dataset	CIC-Modbus2023 (attacks & benign)	Docker substation	Recon, flood, FDI, etc.	Multi-scenario PCAPs and logs	Simulated env.
[35]	Dataset	609,934 real Modbus packets	Real ICS	Broad	Byte-level realism for ML/GANs	Access terms apply
[37]	Your prior work	Distributed brute-force + misleading countermeasure	PLC/HMI LAN, Arduino swarm	Credential brute-force	Deception disrupts brute-forcing & aids attribution	Lab evaluation

Our previous work [37] demonstrated the feasibility of distributed brute-force attacks against ModbusTCP authentication mechanisms and introduced an initial deception-based response strategy. Building upon that foundation, the present study substantially extends the concept by implementing the deception logic entirely within PLC ladder code, introducing a structured session-based authentication workflow, and evaluating the mechanism across multiple PLC families under diverse attack scenarios. In addition, this work provides a detailed characterization of PLC communication constraints, connection scalability, and protocol-specific vulnerabilities. These enhancements transform the original proof-of-concept into a practical, device-level protection mechanism suitable for deployment in legacy industrial environments. Furthermore, the proposed implementation incorporates a session management and exchange protocol that was not addressed in [37]. Collectively, these contributions transform our earlier proof-of-concept into a practical, device-resident security mechanism specifically designed for protecting legacy ModbusTCP industrial controllers.

Our earlier survey of machine-learning-based intrusion detection systems for IIoT environments [38] also highlighted persistent challenges related to dataset availability, feature engineering, and model generalization. These findings further motivate the need

for lightweight, controller-resident security mechanisms that can operate effectively without reliance on external monitoring infrastructure.

While this subsection positions PMA relative to existing Modbus and ICS security research, the following subsection clarifies its relationship with protocol-level and network-level security mechanisms.

2.6. Comparison with Protocol-Level and In-Network Security Approaches

While PMA introduces a lightweight deception-based mechanism, it is not intended to replace standards-based security solutions such as Modbus/TCP Security (TLS) or Zero Trust architectures. These approaches provide strong guarantees for confidentiality, integrity, and authentication but often require certificate infrastructures, firmware modifications, or significant changes to legacy systems.

PMA does not aim to provide confidentiality, integrity, or formal authentication guarantees; instead, it increases attacker uncertainty and operational cost in legacy industrial environments where stronger security mechanisms cannot be readily deployed.

In contrast, PMA is designed for environments where such upgrades are not feasible, enabling a device-level security enhancement implemented directly in PLC logic without modifying the communication protocol. Similarly, programmable data plane approaches (e.g., P4-based in-network security) offer powerful line-rate protection capabilities but require specialized network infrastructure that is not commonly available in industrial deployments. Therefore, PMA should be considered as a complementary mechanism within a defense-in-depth strategy, particularly suited for legacy and resource-constrained systems.

3. The PLC Misleading Algorithm (PMA) Architecture

3.1. Overview and Design Principles

PMA is a lightweight algorithm implemented in ladder (LAD) logic, adding approximately 50 rungs to existing PLC programs with negligible impact on ModbusTCP latency. It integrates authentication and deception directly within the controller to protect against brute-force and denial-of-service (DoS) attacks.

Unlike traditional honeypots deployed on external systems, PMA operates inside the real PLC. It does not block unauthorized access but instead detects abnormal behavior and returns controlled misleading responses. This embedded design significantly increases the difficulty of distinguishing between legitimate and deceptive interactions. The overall logic of PMA is summarized in Algorithm 1.

3.2. Attack Detection and Deception Mechanism

As shown in Algorithm 1, PMA uses hidden registers, timers, and counters to detect abnormal password attempts. Passwords are stored in internal registers that are not accessible via ModbusTCP, while temporary registers store captured values during attack attempts.

Algorithm 1: PLC Security Algorithm for Password Attempts and Access Control

Input: ModbusTCP Read/Write requests
Internal Registers: DI00, DI01 (not externally readable)
Counters and Timer: C2—Received Values
C3—Lock Timer Counter, incremented every second
C4—Password Attempt Counter
T1—1-s timer (T1 = 1-s time limit each Modbus Write (password attempt))

1: Increment C3 (Lock Timer Counter) by **1 every second.**
2: On every **Modbus Write** (password attempt), **increment C4** (**C4** (Password Attempt Counter) by 1.
3: If C2 ≠ 0, save the temporary XOR result of the received values (RN XOR T1, X = RN1 XOR T3) to the challenge response registers (R25, R26) and record the received client's IP address in D200, D201 only if C4 is equal to 1 or C4 is between 20 and 3 (2 ≤ C4 ≤ 3).

On Modbus Read/Write Request:	
26: **IF** IP in [D200 to D201] THEN Access Temp Registers 27: **Else** TEMP XOR Result T1 and D100: Temporary XOR Result RN1 and New 32-bit	 Access Temp Registers TEMP XOR Result of T1 and → D100 Temporary XOR Result of RN1 and T2 → R25, R26

D100, D101: Hidden/Secret Registers

C2	Counter for Received Values
C3	Lock Timer Counter; Incremented on each Modbus Write (in sec.)
C4	Password Attempt Counter, incremented on each Password Attempt

On Modbus Read/Write Request:	
26: **IF** IP in [D200 to D201] THEN Access Temp Registers 27: **Else** TEMP XOR Result of T1 and D100: Tempo-rary XOR Result RN1 and Timer IP: Received Client's IP Ad-dress	C2: Temporary XOR Result of T1 and D100 R2 (Temporary XOR Result of RN1 and T2) → Temporary XOR Result of RN1 and New 32-bit D200 to D210, Temporary Range IP: Received Client's IP Address

The detection mechanism in PMA is based on monitoring the rate of authentication attempts using counters and timers. An attack is detected when the number of attempts exceeds a predefined threshold within a specific time window. Therefore, the detection latency (time-to-detection) is directly influenced by the attack rate. Higher-rate attacks trigger detection rapidly, while lower-rate attacks may delay activation of the deception mechanism. PMA is primarily designed to detect high-rate automated authentication attacks. Legitimate human-operated sessions do not normally generate authentication attempts at rates sufficient to trigger the deception threshold.

Once detected, the PLC generates a session number and redirects all subsequent Modbus operations associated with that session to temporary (fake) registers. The attacker's IP address is also recorded. This approach preserves normal system operation while isolating malicious interactions, preventing resource exhaustion and unintended control actions.

To clarify the randomization mechanism, PMA generates challenge-response values using XOR operations between the received password fragments, hidden registers (D100, D101), and a dynamically generated random number (RN), as detailed in Algorithm 1.

For legitimate users, consistency is ensured because the same RN and session context are maintained across the authentication sequence, leading to deterministic verification outcomes. In contrast, for suspicious or malicious sessions, RN values and responses are intentionally altered or decoupled from valid register states, resulting in inconsistent and misleading outputs that prevent attackers from inferring correct credentials.

3.3. Authentication and Lightweight Obfuscation

The authentication process, illustrated in Figure 1, follows a lightweight XOR-based challenge–response mechanism. The client encodes a 32-bit password using a random number (RN) across three Modbus write operations. The PLC verifies correctness through XOR operations with hidden registers and consistency checks across the received values.

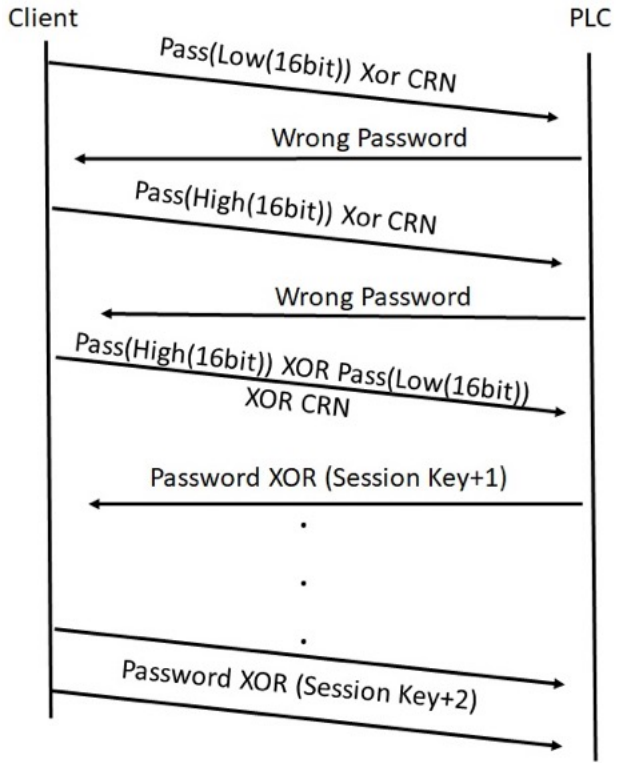

Figure 1. Message sequence of the proposed PMA authentication mechanism between the client and the PLC. The client transmits password fragments combined with a challenge random number (CRN) using XOR operations, while the PLC validates each authentication attempt and responds with updated session-based values.

Upon successful authentication, the PLC generates a session number, encrypts it using the extracted RN, and returns it to the client. This session number is then used for lightweight obfuscation. For each session, the sequence number is incremented based on Modbus operations and XORed with register values before transmission. This prevents the reuse of observed data while maintaining low computational overhead. For attacker sessions, session numbers are still generated (see Algorithm 1) but only applied to fake data to preserve behavioral consistency.

This mechanism is not intended to replace cryptographic protocols such as TLS, but to provide a practical protection layer for resource-constrained PLCs. Passwords are never transmitted in plaintext, and all operations rely on simple instructions, such as XOR, MOV, and comparisons, ensuring deterministic execution with minimal overhead.

This design ensures that legitimate clients observe consistent authentication behavior, while attackers receive statistically unpredictable responses due to the controlled randomization process embedded in the PLC logic. However, PMA does not provide confidentiality,

integrity, or formal authentication guarantees comparable to cryptographic protocols. Instead, its primary objective is to increase attacker uncertainty and operational cost in legacy industrial environments.

3.4. Threat Model and Security Assumptions

PMA assumes an adversary with network-level access to the PLC who can establish ModbusTCP connections, capture traffic, replay messages, and perform automated password-guessing attacks. The attacker is assumed not to possess physical access to the controller, firmware-level privileges, or knowledge of the PLC's internal hidden registers and ladder-logic implementation.

PMA is specifically designed to mitigate online brute-force, session exhaustion, and unauthorized Modbus interaction attempts. It does not protect against firmware compromise, physical tampering, or attacks targeting vendor-specific programming interfaces that transmit credentials outside the PMA-controlled authentication workflow.

As with any deception-based mechanism, PMA assumes that the attackers cannot reliably distinguish deceptive responses from legitimate ones within a limited observation window. Long-term statistical analysis by highly adaptive adversaries may reduce the effectiveness of the deception strategy, which represents an inherent limitation of this class of defenses.

4. Experimental Setup

This section describes the experimental setup and attack scenarios used to evaluate the proposed PMA mechanism. This setup was designed to evaluate the effectiveness of the proposed PMA mechanism under realistic industrial network conditions and to assess its behavior under different attack scenarios targeting ModbusTCP communications.

4.1. Experimental Testbed

The testbed emulates a small industrial control network composed of programmable logic controllers (PLCs), sensors, actuators, monitoring stations, and multiple attacker nodes. To implement PMA and test its operations, two Delta PLCs from different families were utilized: Delta DVP-13SE and Delta DVP-311SV3. Each PLC is connected to two output relays and two digital proximity sensors. The PLCs are programmed to open and close the relays after a random timer expires or when an object is detected by the proximity sensors. The PLCs are connected to a LAN using Ethernet cables. Four Raspberry PIs 4 model B devices with 4 GB RAM are connected to the network. The official Raspberry PI OS (Raspbian) was installed on these devices. Static IP addresses and SSH connections were configured on these devices. These devices are inexpensive, support Python 3.17, and provide multithreading capabilities; therefore, they can replace the Arduino kits utilized in our previous work. In addition, their outputs and errors can be easily monitored through SSH sessions. In this setup, the Raspberry Pi devices emulate distributed attacker nodes capable of launching coordinated attacks against the PLCs.

Two laptops were utilized in the experiment. The first laptop, equipped with an Intel i7 14th-generation processor and 16 GB of RAM, acts as the network sniffer. It runs Windows 11 and Wireshark. The switch port connected to this laptop was configured as a mirror port to forward network traffic for capture and analysis. The second laptop, equipped with an Intel i7 12th-generation processor and 16 GB of RAM, runs Kali Linux. This laptop was utilized for three operations. First, it acts as an attacker attempting to overrun the number of ModbusTCP sessions supported by the PLCs. Second, it was used to read, program, and reconfigure the IP addresses of the PLCs in order to capture their authentication credentials. Finally, it was used to monitor, start, and stop the Raspberry attacks through SSH sessions.

4.2. Attack Implementation

Python was used to perform the brute-force attacks on the PLCs. Several libraries, including PyModbus, PyShark, Scapy, OS, Threading, and Pydump, were leveraged to implement ModbusTCP communication and attack logic. However, one popular ModbusTCP function, Read-Write-Multi-Reg, is not implemented in PyModbus; therefore, this function was implemented in Scapy to accelerate the brute-force attack process.

In addition, Scapy was combined with the OS library to change the source IP address every three ModbusTCP operations. This technique simulates distributed attack behavior and helps bypass potential rate-limiting mechanisms that may restrict repeated authentication attempts from the same IP address.

4.3. Evaluation Scenarios

Four different scenarios were tested in this environment in order to evaluate the behavior of the PLCs and the effectiveness of the PMA mechanism under different attack conditions:

- Scenario 1—Stress Testing: The first scenario consists of a stress test designed to evaluate the maximum number of client sessions that can be simultaneously handled by the PLCs.
- Scenario 2—Distributed Brute-Force Attack: The second scenario involves brute-force attacks launched from the Raspberry Pi devices in parallel in order to test whether authentication credentials can be discovered through repeated ModbusTCP requests.
- Scenario 3—Session Exhaustion Attack: The third scenario attempts to exceed the maximum number of supported ModbusTCP sessions in order to determine whether attackers can exhaust the PLC resources and prevent legitimate clients from establishing connections.
- Scenario 4—Configuration and Programming Attacks: Finally, the fourth scenario involves loading PLC programs and modifying device configurations in order to evaluate the default authentication mechanisms implemented in these PLCs and determine whether unauthorized reconfiguration attempts can be performed.

Figure 2 illustrates the experimental environment and the interaction between the PLCs, attacker nodes, monitoring stations, and network infrastructure.

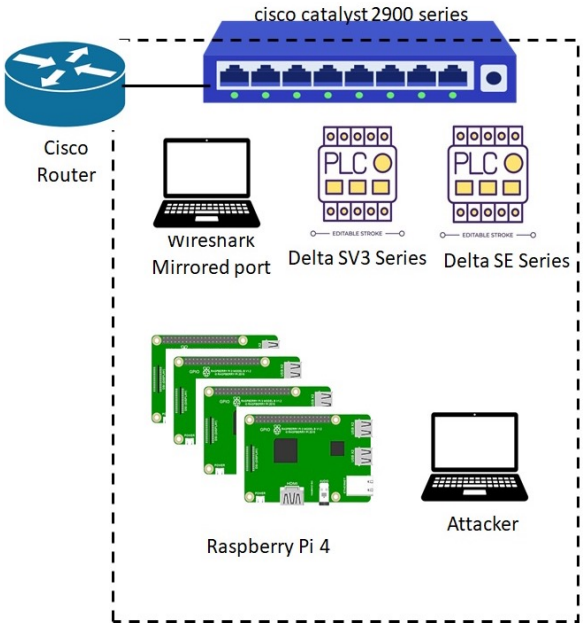

Figure 2. Experimental testbed used to evaluate the PMA. The setup consists of two Delta PLCs (DVP-13SE and DVP-311SV3), sensor and relay interfaces, four Raspberry Pi 4 devices acting as distributed attack nodes, an attacker workstation running Kali Linux, and a monitoring workstation capturing mirrored ModbusTCP traffic using Wireshark. All components communicate through a switched LAN enabling controlled brute-force, stress testing, and DoS attack experiments.

5. Results and Discussion

The Results section is divided into several subsections. Each subsection describes one of the evaluated scenarios and presents the corresponding experimental observations and analysis.

5.1. Default PLC Security Mechanism

Before implementing PMA, it is necessary to demonstrate that the default PLC security mechanism is insufficient. To do this, a password was configured on both PLCs using a configuration software from Delta named DCI-soft. This software is responsible for configuring security parameters such as PLC access passwords, IP addresses, and memory mapping. The password used in this section consists of four digits.

To program the PLC, Delta's IDE, named ISP-Soft, was used. This IDE has the ability to program, compile, download and upload PLC codes, as well as monitor PLC outputs during execution. The PLCs were connected to the network using an Ethernet cable, which was also used to allow the IDE to communicate with the PLCs. ModbusTCP is the default protocol used to connect over Ethernet. A password was also configured in the PLC program to prevent unauthorized read/write operations. The program was then uploaded from the PLC after entering the required password. All of these operations were captured using Wireshark on the monitoring laptop. This means that passwords were configured for two critical functions. The first password protects the PLC configuration interface, which is responsible for IP address configuration and memory mapping. The second password protects the program management functions, including code upload and download operations.

To isolate the experiment, all devices on the network were powered off except for the Wireshark monitoring laptop, the configuration laptop, and the two PLCs, in order to reduce the size of the captured network traces. Finally, the captured dumps were analyzed using Python and the PyShark library.

The packet inspection process revealed that the PLC uses a vendor-specific ModbusTCP function (function code 100) for password authentication and configuration-related operations. Analysis of the packet payload showed that the password is transmitted in plaintext within the ModbusTCP frame. For example, the byte sequence "31 32 33 34" corresponds to the ASCII representation of the password "1234". In our experiment, the plaintext password was observable in every authentication exchange captured by Wireshark. This observation indicates that authentication credentials are not encrypted during transmission, which exposes the system to potential interception and credential theft attacks.

Although configuration access and PLC reprogramming are typically performed only during installation or maintenance, which may explain the limited security mechanisms implemented for these operations, this behavior still introduces a potential vulnerability. For instance, an attacker could intentionally disrupt the network by overwhelming the DHCP server. If the PLC subsequently fails to obtain an IP address, a maintenance engineer may attempt to manually reconfigure the PLC's network settings, which requires entering the configuration password. During this process, an attacker monitoring the network could capture the plaintext password and later use it to gain unauthorized access to the PLC.

Furthermore, even if a ladder logic program implements an additional protection mechanism requiring a password to access the PLC, the password must still be transmitted with each Modbus read or write operation. As a result, the authentication credentials repeatedly appear in the network traffic, making it possible for an attacker to capture the programmed password using a simple network sniffer.

These observations highlight a fundamental limitation of the default ModbusTCP authentication model and motivate the need for additional protection mechanisms such as the proposed PMA approach.

It is important to emphasize that the proposed PMA mechanism cannot protect these configuration-level authentication processes, as they are implemented at the firmware level and rely on vendor-specific Modbus functions. Addressing these vulnerabilities requires protocol-level security mechanisms such as Modbus/TCP Security or firmware modifications by the device manufacturer.

5.2. PLCs Stress Test Results

Before executing the brute-force attack, a stress test was performed (using the first laptop) to determine the maximum number of simultaneous client connections that the PLC could handle. The PyModbus library was used in combination with Python's threading library to generate multiple concurrent Modbus requests. Different numbers of threads were tested to evaluate their impact on password-cracking time. Figure 3 shows the time required to crack a 20-bit password using different numbers of threads. A 20-bit password space was selected because a 16-bit password can be cracked very quickly, while a 32-bit password would require several days under the same testing conditions.

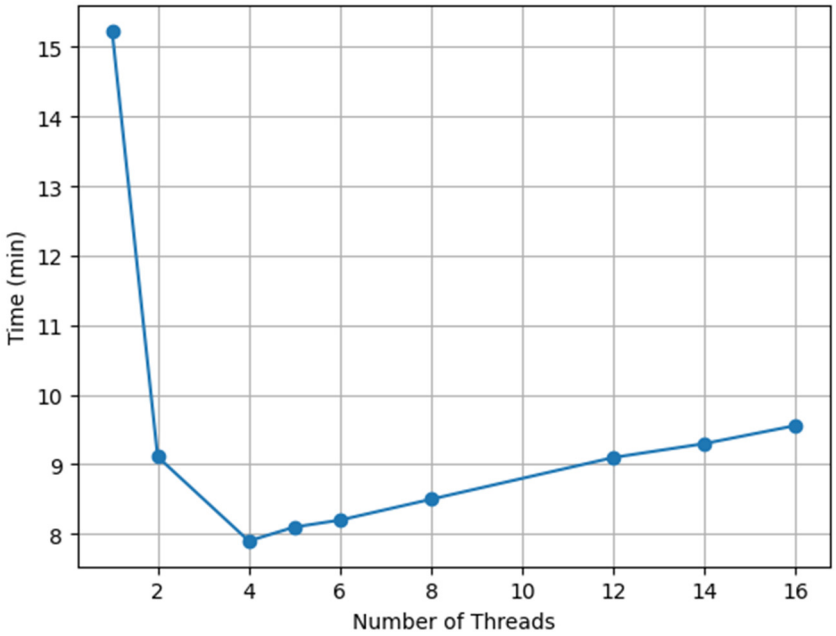

Figure 3. Password cracking time (seconds) as a function of the number of concurrent threads. The results illustrate the impact of thread-level parallelism on brute-force attack efficiency.

The results show that the maximum number of effective concurrent threads supported by the PLC is approximately 16 simultaneous connections, which was confirmed for both PLC models. As illustrated in the figure, increasing the number of threads initially reduced the cracking time. However, beyond four concurrent threads, the total cracking time began to increase. The experimentally observed peak at four concurrent threads appears to be related to the internal scheduling and request-processing characteristics of the PLC communication stack. This behavior suggests that the PLC processes incoming requests using an internal queue when the maximum processing capacity is reached. Therefore, the device can efficiently process a limited number of simultaneous requests before queuing and contention effects dominate. Once the PLC becomes fully utilized, additional requests experience delays while waiting to be processed. This result indicates that there exists an

optimal level of concurrency for brute-force attacks, beyond which additional parallelism leads to performance degradation rather than improvement.

Despite this queuing effect, the cracking time does not increase significantly when the maximum number of threads is reached. Similar results were observed for the SV3 family PLC. However, the cracking time was approximately 50% lower compared to the SE family, which is likely due to the faster execution capabilities of the SV3 PLC.

To further accelerate the brute-force process, the implemented attack performs two Modbus requests for each password attempt. The first request writes the candidate password to the PLC registers, and the second request reads the session or verification value to determine whether the authentication attempt was successful. To reduce the number of network transactions, Modbus function code 23 (0 x17) and read/write multiple registers can be used to perform both operations within a single request–response cycle. Using this approach reduced the cracking time by approximately 50% for both PLC models.

However, this function is not directly supported in the PyModbus library. Therefore, the function was manually implemented using the Scapy library to construct the required ModbusTCP packets. Experimental results showed that this approach worked successfully with the SE family PLC, but it did not function with the SV3 PLC, likely due to differences in the supported Modbus function set.

The brute-force attack was distributed across four Raspberry Pi devices, with four threads per device, resulting in a total of 16 concurrent connections. Figure 4 illustrates the cracking time for different password sizes when using the optimized single-request approach on the SE family PLC. For the SV3 PLC, the cracking time remained approximately equal to the time required when using the modified two-request method. These findings highlight the importance of understanding PLC processing limits when designing both attack strategies and defensive mechanisms.

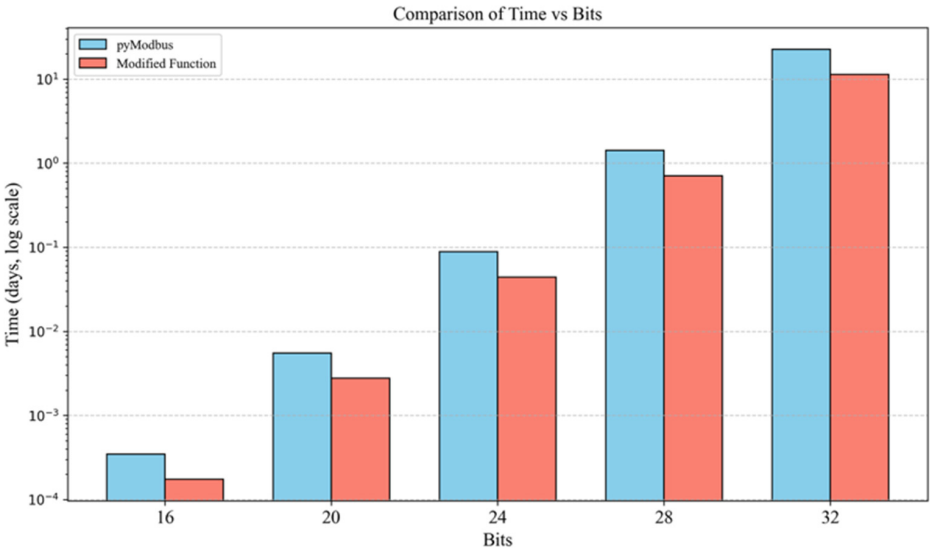

Figure 4. Password cracking time (seconds) for different password sizes under optimized Modbus request conditions.

5.3. PLC Brute-Force DoS Attack

As shown in Figure 3, the maximum number of simultaneous client connections supported by both tested Delta PLC families is 16 connections. However, it is important to examine how the PLC behaves when additional connection attempts are made beyond this limit. Different PLC vendors implement different strategies to handle excess connections. For example, some vendors terminate existing connections and accept new ones using a round-robin scheduling mechanism to serve incoming requests. Although this approach

improves fairness among clients, it may also expose the system to Modbus-based denial-of-service (DoS) attacks, since attackers can continuously generate new connections to force the PLC to repeatedly close and reopen sessions.

In contrast, the Delta PLCs tested in this study reject additional connections once the maximum number of clients has been reached. However, an interesting behavior was observed: although the PLC refuses to establish new sessions, incoming Modbus requests from these extra clients can still be successfully transmitted over the network, but the PLC does not generate any responses. This behavior creates an asymmetric communication condition in which the PLC continues to receive and process incoming traffic at the network level but does not provide application-layer responses, leading to inefficient resource utilization.

To investigate the impact of this behavior, the four Raspberry Pi devices maintained persistent connections to the PLC using four threads each, resulting in 16 active connections, which saturates the PLC's connection capacity. Meanwhile, the first laptop attempted to initiate additional connections and send Modbus requests.

In this test, the laptop repeatedly attempted to perform two operations: write multiple registers and read holding registers, using a randomly varying number of threads generated every 100 ms. The results show that the write requests were successfully transmitted, but the read operations did not receive any responses from the PLC. Packet inspection using PyShark confirmed that all Modbus requests originating from the laptop were captured in the network traffic, yet no corresponding responses from the PLC were observed. This confirms that the PLC does not fully reject excess traffic at the network level, but instead silently drops responses at the application layer.

This behavior increases the load on the PLC and consumes additional network bandwidth, as illustrated in Figure 5. The figure shows that the number of successful Modbus operations decreased significantly under these conditions, dropping from approximately 7200 operations per second to about 3300 operations per second. This degradation demonstrates that the PLC communication stack becomes partially saturated, even though it does not explicitly accept additional client sessions. This result demonstrates that a brute-force attack targeting PLC authentication mechanisms can unintentionally evolve into a DoS condition, severely degrading the PLC's ability to process legitimate Modbus requests.

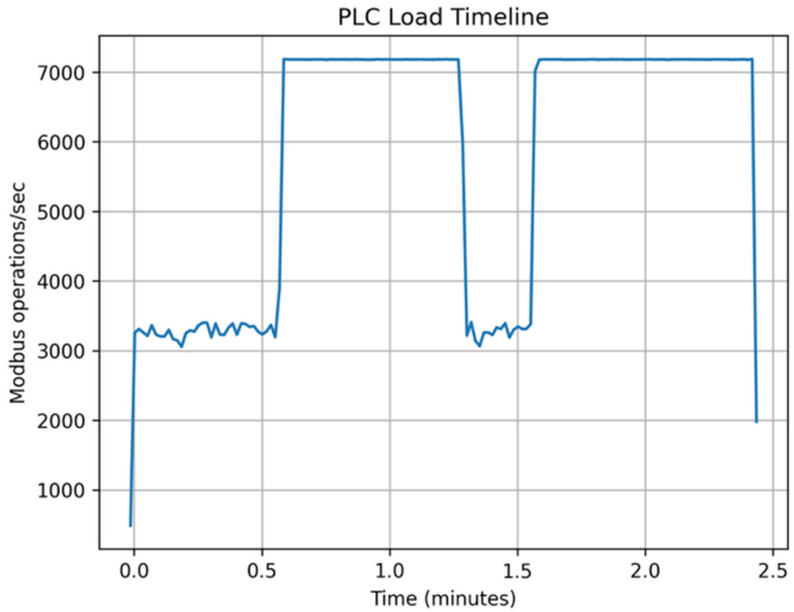

Figure 5. Impact of excessive concurrent ModbusTCP connection attempts on PLC communication performance, showing the reduction in successful operations per second under high-load conditions.

These findings highlight a critical weakness in the PLC connection handling mechanism, where partial processing of unauthorized traffic can still lead to resource exhaustion and service degradation. This behavior suggests that effective protection mechanisms must not only limit connection establishment, but also control the processing of incoming requests beyond the connection threshold.

5.4. The Underlying Network Behavior

To study the underlaying LAN network behavior, four different traffic dumps were analyzed, each one containing four ModbusTCP sessions, where one additional Raspberry Pi device is introduced in each dump. These captures were analyzed to study the Modbus operation rate, bandwidth usage, and latency. All reported metrics represent averaged values over segmented traffic captures, with bandwidth expressed in Mbps, latency in milliseconds (ms), and throughput in operations per second.

Table 2 shows the traffic characteristics of the SE PLC. The dump files were divided into smaller segments of approximately 150 s, and the reported values corresponded to the average across these segments. This approach was used to reduce memory usage and improve processing efficiency during analysis. During the capture sessions, from one to four Raspberry Pi devices attempted to perform brute-force attacks against the PLCs. Each Raspberry Pi established four concurrent threads per PLC. The results show significant variation in network load depending on the number of concurrent connections. The generated traffic reached a maximum bandwidth of approximately 9.18 Mbps when the PLC was handling a smaller number of concurrent flows. As the number of flows increased, the average bandwidth slightly decreased due to communication overhead and resource sharing among multiple clients.

Table 2. Network traffic characteristics of the SE PLC under varying numbers of concurrent Raspberry Pi clients.

Scenario	Upload (Bytes/s)	Download (Bytes/s)	Total (Bytes/s)	Avg Bandwidth (Mbps)
Single Client (4 Threads)	562,753	586,981	1.15 M	9.18
Two Clients (8 Threads)	523,719	547,706	1.07 M	8.571
Three Clients (12 Threads)	488,235	511,911	1 M	8.010
Four Clients (16 Threads)	469,551	491,989	961,541	7.699

Table 3 presents the Modbus operations statistics. The results indicate that the PLC achieved a maximum processing rate of approximately 8566 Modbus operations per second. As the number of concurrent connections increased, the total throughput slightly decreased due to internal scheduling and processing limitations within the PLC communication stack.

Table 3. SE PLC Modbus operations under varying numbers of concurrent Raspberry Pi clients.

Scenario	Modbus Packets/s	Modbus Operations/s	Requests/s	Responses/s
Single Client (4 Threads)	16.8 k	8566	8.396 k	8.55 k
Two Clients (8 Threads)	16.28 k	7995	8.14 k	8.14 k
Three Clients (12 Threads)	14.73 k	7472	7.365 k	7.36 k
Four Clients (16 Threads)	14.1 k	7182	7.06 k	7 k

Table 4 presents the flow-level statistics. The number of active flows increased during the experiments to simulate multiple clients accessing the PLC simultaneously. However, while the number of flows increased, the total number of packets slightly decreased, indicating reduced per-flow activity. Since each client creates a pair of TCP streams (request

and response), the number of flows corresponds to approximately half the number of TCP connections. Therefore, the experiments represent up to 16 concurrent ModbusTCP client sessions communicating with the PLC.

Table 4. SE PLC flows under varying numbers of concurrent Raspberry Pi clients.

Scenario	Total Flows	Flows/s	Packets/s
Single Client (4 Threads)	8	0.042	17.13 k
Two Clients (8 Threads)	16	0.077	15.99 k
Three Clients (12 Threads)	24	0.141	14.9 k
Four Clients (16 Threads)	32	0.207	14.36 k

Table 5 summarizes the observed response latency of the PLC during Modbus communication. As shown in Table 5, the average response time increases progressively with the number of concurrent connections, indicating growing contention within the PLC communication stack.

Table 5. SE PLC latency under varying numbers of concurrent Raspberry Pi clients.

Scenario	Avg RTT (ms)	Min RTT (ms)	Max RTT (ms)
Single Client (4 Threads)	0.75	0	36.7
Two Clients (8 Threads)	1.791	0	21.935
Three Clients (12 Threads)	2.969	0	18.401
Four Clients (16 Threads)	4.203	0	26.7

The results demonstrate that the average response time increases as the number of active connections increases. This behavior indicates resource contention within the PLC communication stack as more clients attempt to access the device simultaneously.

The experimental results reveal a clear PLC saturation trend. When the number of concurrent connections increases, the PLC must distribute its processing capacity among all active sessions. As a result, the overall Modbus throughput decreases while the response latency increases. The maximum throughput observed during the experiments was approximately 8600 Modbus operations per second, representing the practical processing capacity of the evaluated PLC. Beyond this level, additional traffic would likely lead to communication delays and potential service degradation.

These findings demonstrate that the PLC communication stack can be saturated through relatively low-cost distributed traffic generation, highlighting a potential attack surface for ModbusTCP-based denial-of-service (DoS) attacks.

5.5. The Underlying Network Behavior of SV3 PLC

For the SV3 PLC, the computational capability was approximately twice that of the SE family, which directly impacts its performance characteristics. As a result, the number of processed packets and overall throughput approximately doubled, while the latency remained comparatively lower, as also observed during the password-cracking experiments, where the cracking time was reduced by approximately 50%. This confirms that the SV3 PLC can sustain a higher processing rate under identical attack conditions.

However, despite this improved computational performance, the maximum number of concurrent client connections remains limited to 16, which is the same as in the SE family. This limitation indicates that the bottleneck is not purely computational, but rather related to the internal design of the PLC communication stack or firmware constraints.

5.6. The DDoS Attack Impact

A final experiment was conducted to evaluate the impact of a massive semi-distributed denial-of-service (semi-DDoS) attack using four Raspberry Pi devices. In this scenario, 20 concurrent threads were initiated from each Raspberry Pi, resulting in a total of 80 connection attempts.

The attack was launched after 16 connections had already been successfully established from a laptop to the SE PLC, corresponding to the maximum number of supported concurrent client connections. One of these connections was maintained by the PLC development environment (IDE) to monitor and visualize the process of writing data to the PLC registers. All 80 attack threads attempted to establish connections simultaneously without any delay.

The results showed that although a complete denial of service was not achieved, the system experienced severe performance degradation. In particular, the process of writing data to the PLC registers became significantly slower, requiring several seconds to update values, indicating a substantial increase in communication latency and processing delay caused by excessive connection attempts.

The same experiment was conducted on the SV3 PLC. In this case, the number of attack threads was increased to 200 concurrent threads. Under these conditions, the IDE became unresponsive, and no updates were initially observed. However, after approximately 20 to 30 s, delayed updates were eventually processed.

These results demonstrate that even if a complete DoS condition is not immediately achieved, high-rate connection attempts can severely degrade system responsiveness, effectively leading to a functional denial of service from an operational perspective.

5.7. PMA Implementation

To implement PMA, one of the main design requirements was to ensure that the algorithm remains lightweight and compatible with resource-constrained PLC environments. In Delta PLCs, program memory is measured in steps, where each ladder logic rung requires between 3 and 15 steps, depending on its complexity. The PMA algorithm was implemented using 52 rungs, with an average of 7 steps per rung. This number can be further optimized.

The total memory capacity of the evaluated PLC families is approximately 16 k steps, which means that PMA occupies only 2.2% of the available memory. This demonstrates that PMA introduces minimal memory overhead, making it suitable for deployment in legacy industrial systems.

PMA was implemented on the Delta SE PLC, and brute-force attacks were launched from the four Raspberry Pi devices using different numbers of threads. The password-cracking process on each raspberry generated different passwords and random data patterns in the read/write operations. However, threads originating from the same client consistently obtained the same password, since PMA associates authentication behavior with the client IP address.

The system was evaluated over a period of 300 s during active attack conditions. This part of the experiment was repeated 10 times and the average value of these attempts is reported. During this time, only 73.5 Bytes/s of traffic was captured, corresponding to an average bandwidth of approximately 0.001 Mbps. Additionally, only 208 Modbus packets were observed, with a rate of 0.246 operations per second. This represents a drastic reduction in observable attack traffic compared to the baseline scenario, demonstrating the effectiveness of PMA in mitigating brute-force attempts.

It should be noted that the average response time (RTT) increased due to the additional computation and obfuscation introduced by PMA. The observed average RTT was approx-

imately 3.6 ms. This increase represents a trade-off between security and performance, which remains acceptable within industrial communication constraints.

For configuration code download/upload passwords associated with ModbusTCP function codes 100 and 108, PMA cannot provide protection against brute-force attacks targeting these mechanisms. These functions require modifications at the PLC firmware level rather than application-layer mitigation.

To analyze the impact of attack rate on detection performance, we conceptually distinguished between high-rate and low-rate attacks. In high-rate scenarios, the detection threshold is exceeded within a short interval, resulting in near-immediate activation of deception. In contrast, low-rate password-guessing attacks may remain below the threshold for longer periods, increasing detection latency. Slow-rate brute-force attacks were not explicitly evaluated, since their operational impact is negligible and the time required to exhaust a 32-bit credential space is prohibitively large.

Without upgrading of the framework, the only viable protection mechanisms are external security solutions, such as firewalls configured to block unknown Modbus function codes. Additionally, IDS/IPS systems can be deployed to monitor and filter malicious LAN traffic in the case of system compromise. These findings highlight that protocol-level limitations cannot be fully mitigated through lightweight PLC-side defenses alone.

This also indicates that connecting PLCs directly to the Internet should be avoided. Instead, a secure intermediary server architecture, as proposed in [22], should be used. Overall, the simplicity and lack of built-in security in ModbusTCP make it an attractive attack vector for network penetration in IIoT environments.

These results validate PMA as a practical and lightweight mitigation mechanism against brute-force attacks while also highlighting the need for complementary network-level defenses to address protocol-level vulnerabilities. Although the experiments were conducted on Delta PLCs, the identified vulnerabilities and attack vectors are inherent to the ModbusTCP protocol and are expected to affect other vendors implementing similar authentication mechanisms.

5.8. Adversarial Considerations

An advanced attacker may attempt to detect the presence of deception by systematically analyzing the behavior of the PLC across multiple interactions. For example, instead of relying on a single brute-force attempt, the attacker can perform multi-session probing, where the same credential guesses are repeated across different sessions and the resulting responses are compared with any inconsistency in authentication outcomes. This attack can be eliminated by saving only one wrong password for all heavy traffic that hit the system in a small window. In addition, an attacker can perform timing analysis by measuring response delays associated with authentication attempts and Modbus operations. If deceptive responses introduce different processing times compared to legitimate ones due to additional logic such as randomization, redirection to fake registers, or session management, these timing variations may be statistically distinguishable over a large number of observations. However, this attack is not valid in an IIoT environment since reading the data from different devices connected to PLCs may encounter different delay and latency values. Furthermore, replay and desynchronization attacks may be used to probe the consistency of the PMA mechanism. An attacker could capture valid communication sequences and replay them at different times or from different source addresses to observe whether the PLC produces consistent responses. If session-dependent values or obfuscation keys evolve in a predictable manner, the attacker may exploit this behavior to identify valid sessions or bypass certain aspects of the deception logic. This attack requires

small enhancement of the algorithm code to add another decoy layer to create different session numbers.

Finally, long-term traffic analysis can reveal statistical differences between legitimate and deceptive interactions. Even if individual responses appear plausible, aggregated metrics such as response frequency, value distributions, register access patterns, or session lifetimes may deviate from normal PLC behavior. Over time, these discrepancies can provide sufficient evidence for an attacker to distinguish between real and deceptive system states.

While PMA incorporates randomized responses and session-based behavior to increase attacker uncertainty, it does not provide formal guarantees of indistinguishability under all adversarial models. Instead, its effectiveness relies on increasing the cost and complexity of attacks, particularly for automated brute-force tools, rather than achieving cryptographic security. Addressing these limitations would require integrating PMA with stronger security primitives or formally analyzing its resistance against adaptive adversaries.

6. Conclusions

This paper investigated the security limitations of PLC authentication mechanisms in ModbusTCP-based industrial environments and proposed a lightweight deception-based protection mechanism, called the PLC misleading algorithm (PMA). Experimental analysis conducted on two Delta PLC families demonstrated that authentication credentials are transmitted in plaintext during configuration and program management operations, allowing attackers to capture passwords through simple network sniffing techniques. In addition, stress testing revealed that the tested PLCs support a maximum of 16 simultaneous ModbusTCP sessions, making them vulnerable to brute-force attacks and resource-exhaustion scenarios. Furthermore, distributed brute-force attacks generated from multiple Raspberry Pi devices significantly degraded the PLC communication performance and increased response latency.

The proposed PMA mechanism was implemented directly inside the PLC ladder program with minimal resource consumption, occupying only 2.2% of the available PLC program memory. Unlike conventional protection techniques that simply block unauthorized access attempts, PMA introduces a deception-based strategy that misleads attackers by returning randomized yet controlled responses while maintaining consistent behavior for legitimate clients.

Experimental results showed that brute-force attacks were effectively neutralized, as attackers obtained inconsistent and unusable password values, while legitimate clients maintained stable and correct authentication responses. Furthermore, network traffic analysis indicated a drastic reduction in observable Modbus operations during attack conditions, demonstrating the effectiveness of PMA in limiting the capabilities of automated password-cracking tools.

These findings highlight that security mechanisms embedded directly within PLC logic can provide an effective first-line of defense without requiring modifications to existing industrial infrastructure. However, the study also revealed that certain vulnerabilities, particularly those associated with specific ModbusTCP function codes (e.g., configuration and firmware-related operations), cannot be mitigated solely through application-layer solutions and require firmware-level or network-level protections.

Overall, the results demonstrate that deception-based security mechanisms integrated at the device level can significantly improve the resilience of industrial control systems against brute-force and network-based attacks. This work contributes toward bridging the gap between lightweight PLC-level protections and the growing need for robust cybersecu-

rity mechanisms in legacy industrial systems. While PMA effectively mitigates brute-force attacks targeting application-level authentication implemented within PLC logic, it does not address vulnerabilities in vendor-specific configuration and programming interfaces. These require complementary protocol-level or firmware-based security mechanisms.

Future work will focus on extending PMA to heterogeneous industrial environments involving multiple PLC vendors and communication protocols. In addition, integrating PMA with AI-based anomaly detection techniques will be investigated to enable adaptive and context-aware attack detection. Another promising direction is the deployment of PMA within distributed industrial architectures, where multiple PLCs cooperate to share threat intelligence and coordinate deception strategies. Furthermore, formal security analysis will be conducted to evaluate the robustness of PMA against adaptive adversaries and advanced traffic analysis techniques.

Author Contributions: Conceptualization, M.Z.M. and J.G., methodology, Á.Á. and J.G., software, M.A. and M.Z.M., validation, Á.Á. and J.G., formal analysis, M.Z.M. and M.A., investigation, M.A., resources, M.Z.M. and M.A., data curation, Á.Á. and J.G., writing—M.Z.M. and M.A., review and editing, Á.Á., J.G. and M.Z.M., visualization, M.A., supervision, M.Z.M. and J.G., project administration, J.G., funding acquisition, M.Z.M. and M.A. All authors have read and agreed to the published version of the manuscript.

Funding: This research was partially supported by grant PID2022-136476OB-I00 funded by MICIU/AEI/10.13039/501100011033/FEDER EU, ERDF/EU and Gobierno de Aragón (reference group T31_20R).

Data Availability Statement: Any data used in this work is available upon requested.

Acknowledgments: In the preparation process of this manuscript ChatGPT (chatgpt 4.0) has been utilized to edit, rephrase and correct grammar and typo mistakes.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Masoud, M.; Jaradat, Y.; Manasrah, A.; Jannoud, I. Sensors of smart devices in the internet of everything (IoE) era: Big opportunities and massive doubts. *J. Sens.* **2019**, *2019*, 6514520. [CrossRef]
2. Khan, N.; Solvang, W.D.; Yu, H. Industrial Internet of Things (IIoT) and other Industry 4.0 technologies in spare parts warehousing in the oil and gas industry: A systematic literature review. *Logistics* **2024**, *8*, 16. [CrossRef]
3. Industry 4.0 Statistics. Available online: https://www.globalgrowthinsights.com/market-reports/industrial-iot-platform-market-107217?utm_source=chatgpt.com (accessed on 12 March 2026).
4. Zhukabayeva, T.; Zholshiyeva, L.; Karabayev, N.; Khan, S.; Alnazzawi, N. Cybersecurity solutions for industrial internet of things–edge computing integration: Challenges, threats, and future directions. *Sensors* **2025**, *25*, 213. [CrossRef] [PubMed]
5. Cecílio, J.; Souto, A. Security issues in industrial Internet-of-Things: Threats, attacks and solutions. In *2024 IEEE International Workshop on Metrology for Industry 4.0 & IoT (MetroInd4.0 & IoT)*; IEEE: New York, NY, USA, 2024; pp. 458–463.
6. Masoud, M.; Jaradat, Y.; Manasrah, A.; Taleb, B. Designing of a General Purpose Soft Programmable Logic Controller (PLC) for the Internet of Things (IoT) Era. *Int. Rev. Autom. Control (IREACO)* **2020**, *13*, 153–161. [CrossRef]
7. PLC Industry Statistics. Available online: https://wifitalents.com/plc-industry-statistics/ (accessed on 12 March 2026).
8. PLC MArket Size. Available online: https://www.industryresearch.biz/market-reports/plc-market-103877?utm_source=chatgpt.com (accessed on 12 March 2026).
9. Mo, F.; Querejeta, M.U.; Hellewell, J.; Rehman, H.U.; Rezabal, M.I.; Chaplin, J.C.; Sanderson, D.; Ratchev, S. PLC orchestration automation to enhance human–machine integration in adaptive manufacturing systems. *J. Manuf. Syst.* **2023**, *71*, 172–187. [CrossRef]
10. Gnana Swathika, O.V.; Karthikeyan, A.; Karthikeyan, K.; Sanjeevikumar, P.; Thomas, S.K.; Babu, A. Critical review of SCADA and PLC in smart buildings and energy sector. *Energy Rep.* **2024**, *12*, 1518–1530. [CrossRef]
11. Ahmed, B.; Shehzad, Q.; Ullah, I.; Zahoor, N.; Tayyab, H.M. An Effective Combination of PLC and Microcontrollers for Centralized Traffic Control and Monitoring System. *Eng. Proc.* **2022**, *12*, 71.

12. Kavaliauskas, Ž.; Blažiūnas, G.; Šajev, I.; Iljinas, A.; Gimžauskaitė, D. Development and Optimization of an Automated Industrial Wastewater Treatment System Using PLC and LSTM Neural Network. *Appl. Sci.* **2025**, *15*, 8990. [CrossRef]
13. Waqas, M.; Jamil, M. Smart IoT SCADA system for hybrid power monitoring in remote natural gas pipeline control stations. *Electronics* **2024**, *13*, 3235. [CrossRef]
14. Mughaid, A.; AlJamal, M.; Issa, A.L.-A.; AlJamal, M.; Alquran, R.; AlZu'bi, S.; Abutabanjeh, A.A. Enhancing cybersecurity in scada IoT systems: A novel machine learning-based approach for man-in-the-middle attack detection. In *2023 3rd Intelligent Cybersecurity Conference (ICSC)*; IEEE: New York, NY, USA, 2023; pp. 74–79.
15. Mohammad, Z.M.; Jaradat, Y.; Alhawawsheh, M.A.; Jannoud, I.; Alsakarnah, R. Smartphone Communication Strategies for Legacy Industrial Systems: A Study of ModbusTCP and Local Server Solutions. In *2025 12th International Conference on Information Technology (ICIT)*; IEEE: New York, NY, USA, 2025; pp. 546–551.
16. Maniraj, R. Comparative Analysis of Real-Time Industrial Ethernet Protocols: ModbusTCP Protocol Implementation. Master's Thesis, KTH Royal Institute of Technology, Stockholm, Sweden, 2022.
17. Gao, W.; Morris, T. On Cyber Attacks and Signature Based Intrusion Detection for Modbus Based Industrial Control Systems. *J. Digit. Forensics Secur. Law* **2014**, *9*, 37–56. [CrossRef]
18. Morris, T.; Gao, W. Industrial Control System Traffic Data Sets for Intrusion Detection Research. In *Critical Infrastructure Protection VIII, IFIP AICT 441*; Springer: Berlin/Heidelberg, Germany, 2014; pp. 65–78. [CrossRef]
19. Duque Antón, S.; Kanoor, S.; Fraunholz, D.; Schotten, H.D. Evaluation of Machine Learning-Based Anomaly Detection Algorithms on an Industrial Modbus/TCP Data Set. In *ARES 2018*; ACM: New York, NY, USA, 2018; pp. 1–9. [CrossRef]
20. Ghosh, T.; Bagui, S.; Bagui, S.; Kadzis, M.; Bare, J. Anomaly Detection for Modbus over TCP in Control Systems Using Entropy and Classification-Based Analysis. *J. Cybersecur. Priv.* **2023**, *3*, 895–913. [CrossRef]
21. Varol, M.; İskefiyeli, M. An Intrusion Detection System for Critical Infrastructures: Modbus Approach. *Eng. Appl. Artif. Intell.* **2025**, *162*, 112410. [CrossRef]
22. Liu, I.H.; Lin, J.H.; Lai, H.Y.; Li, J.S. Extendable ICS Honeypot Design with Modbus/TCP. *Proc. Int. Conf. Artif. Life Robot. (ICAROB 2022)* **2022**, *27*, 287–290. [CrossRef]
23. Siniosoglou, I.; Efstathopoulos, G.; Pliatsios, D.; Moscholios, I.D.; Sarigiannidis, A.; Sakellari, G.; Loukas, G.; Sarigiannidis, P. NeuralPot: An Industrial Honeypot Implementation Based on Deep Neural Networks. In *IEEE ISCC 2020*; IEEE: New York, NY, USA, 2020. [CrossRef]
24. Machaka, V.; Figueroa-Lorenzo, S.; Arrizabalaga, S.; Elduayen-Echave, B.; Hernantes, J. Assessing the Impact of Modbus/TCP Protocol Attacks on Critical Infrastructure: WWTP Case Study. *Comput. Electr. Eng.* **2025**, *126*, 110485. [CrossRef]
25. Conpot: Open-Source ICS/SCADA Honeypot Framework. Available online: https://github.com/mushorg/conpot (accessed on 12 March 2026).
26. Modbus Organization. MODBUS/TCP Security Protocol Specification (v36). 2021. Available online: https://modbus.org/file/secure/modbussecurityprotocol.pdf (accessed on 13 March 2026).
27. Modbus Organization. MODBUS Messaging on TCP/IP Implementation Guide (V1.0b). Available online: https://modbus.org/file/secure/messagingimplementationguide.pdf (accessed on 13 March 2026).
28. Modbus Organization. MODBUS Application Protocol Specification (V1.1b3). Available online: https://modbus.org/file/secure/modbusprotocolspecification.pdf (accessed on 13 March 2026).
29. Jhan, J.Y.; Sun, H.M. Enhancing Modbus TCP Protocol Security with eBPF Technology. *arXiv* **2023**. [CrossRef]
30. Martins, T.; Garcia Oliveira, S.V. Enhanced Modbus/TCP Security Protocol: Authentication and Authorization Functions Supported. *Sensors* **2022**, *22*, 8024. [CrossRef] [PubMed]
31. Fagundes, F.D.; da Cunha, M.J. Industrial Network Security: HB-MP as an Authentication Technique for Modbus TCP. *J. Control Autom. Electr. Syst.* **2022**, *33*, 1177–1187. [CrossRef]
32. Radoglou-Grammatikis, P.I.; Siniosoglou, I.; Liatifis, T.; Kourouniadis, A.; Rompolos, K.; Sarigiannidis, P. Implementation and Detection of Modbus Cyberattacks. In *MOCAST 2020*; IEEE: New York, NY, USA, 2020. [CrossRef]
33. Kaggle. Modbus TCP SYN Flood Dataset for Intrusion Detection. Available online: https://www.kaggle.com/datasets/zoltandobrady/modbus-tcp-syn-flood-ds-for-intrusion-detection (accessed on 12 March 2026).
34. Canadian Institute for Cybersecurity (UNB). CIC-Modbus. 2023 Dataset. Available online: https://www.unb.ca/cic/datasets/modbus-2023.html (accessed on 12 March 2026).
35. Badawy, W.; Tawfik, M. A Large Dataset of 609,934 Real Modbus Packets. In *IEEE DataPort*; IEEE: New York, NY, USA, 2025. [CrossRef]
36. Dao, Q.T.; Nguyen, L.T.; Ha, T.K.; Nguyen, V.H.; Nguyen, T.A. Investigation of Secure Communication of Modbus TCP/IP Protocol: Siemens S7 PLC Series Case Study. *Appl. Syst. Innov.* **2025**, *8*, 65. [CrossRef]

37. Abdul Jawad, M.; García, J.; Masoud, M. A Survey of Network Intrusion Detection Systems (NIDS) Based on Machine Learning Algorithms for IIoT. In *Artificial Intelligence in Business (SICB 2025), LNNS 1503*; Springer: Berlin/Heidelberg, Germany, 2025; pp. 158–167. [CrossRef]
38. Abdul Jawad, M.A.; Masoud, M.Z.; Alesanco, Á.; García, J. A Distributed Brute-Force Attack and Misleading Countermeasure for Securing ModbusTCP Implementations in PLCs. In *2025 12th International Conference on Information Technology (ICIT)*; IEEE: New York, NY, USA, 2025. [CrossRef]