PLC Access Control: A Security Analysis

Haroon Wardak
Information and Computer Science Department
KFUPM, Dhahran, 31261, KSA
Email: g201302150@kfupm.edu.sa

Sami Zhioua
Information and Computer Science Department
KFUPM, Dhahran, 31261, KSA
Email: zhioua@kfupm.edu.sa

Ahmad Almulhem
Computer Engineering Department
KFUPM, Dhahran, 31261, KSA
Email: ahmadsm@kfupm.edu.sa

***Abstract*—A Programmable Logic Controller (PLC) is a very common industrial control system device used to control output devices based on data received (and processed) from input devices. Given the central role that PLCs play in deployed industrial control systems, it has been a preferred target of ICS attackers. A quick search in the ICS-CERT repository reveals that out of a total of 589 advisories, more than 80 target PLCs. Stuxnet attack, considered the most famous reported incident on ICS, targeted mainly PLCs. Most of the PLC reported incidents are rooted in the fact that the PLC being accessed in an unauthorized way. In this paper, we investigate the PLC access control problem. We discuss several access control models but we focus mainly on the commonly adopted password-based access control. We show how such password-based mechanism can be compromised in a realistic scenario as well as the list the attacks that can be derived as a consequence. This paper details a set of vulnerabilities targeting recent versions of PLCs (2016) which have not been reported in the literature.**

***Keywords*-PLC; SCADA; Industrial Control Systems; Access Control; Passwords;**

I. Introduction

A Programmable Logic Controller (PLC) is an important component in an ICS system. It is a control device used to automate industrial processes via collecting input data from field devices such as sensors, processing it, then send commands to actuators devices such as motors. Being a pivotal device in ICS systems, PLCs are preferred target for cyber security attacks. ICS-CERT, the repository for ICS specific incidents, includes a large number of PLC related issues. A quick search performed in November 2016 reveals that out of a total of 589 advisories, 89 target directly PLCs and out of a total of 114 alerts, 17 involve PLCs. Another manifestation of the exposure of PLCs to cyber security attacks is the Stuxnet malware [**?**] which is designed to attack primarily PLCs of the Iranian nuclear facility.

PLC security issues range from simple DoS to sophisticated remote code execution vulnerabilities. Most of PLC attacks are possible because attackers could have access and compromise the PLC device. PLC Access Control can be implemented at different layers: network layer, physical access, firmware, etc. In this paper, we discuss the different access control models for PLCs, but we focus on the most commonly deployed access control mechanism, namely, password-based access control. Using recent PLC devices (2016) with updated firmware, we show how passwords are stored in PLC memory, how passwords can be intercepted in the network, how they can be cracked, etc. As a consequence of these vulnerabilities, we could carry out advanced attacks on ICS system setup, such as replay, PLC memory corruption, etc.

II. PLC Vulnerabilities

A PLC is a particular type of embedded devices that is programmed to manage and control physical components (motors, valves, sensors, etc.) based on system inputs and requirements. A PLC typically has three main components, namely, an embedded operating system, control system software, and analog and digital inputs/outputs. Hence, a PLC can be considered as a special digital computer executing specific instructions that collect data from input devices (e.g. sensors), sending commands to output devices (e.g. valves), and transmitting data to a central operations center.

PLCs are commonly found in supervisory control and data acquisition (SCADA) systems as field devices. Because they contain a programmable memory, PLCs allows a customizable control of physical components through a user-programmable interface.

The ICS-CERT repository, dedicated to ICS related security incidents, includes several reports involving PLC vulnerabilities and alerts. Most of the reports are relatively recent (2010 and later). The increase in ICS and PLC incidents coincides with the increasing interconnection of ICS and corporate networks which became a necessity to improve efficiency, minimize costs, and maximize profits. This, however, exposes ICS systems, and PLCs in particular, to various types of exploitation.

Most of PLC vulnerabilities can be grouped into three categories, namely, network vulnerabilities, firmware vulnerabilities, and access control vulnerabilities.

PLCs are increasingly required to be interconnected with corporate LANs, Intranets, and Internet. Due to their increasing connectivity, PLCs are expected to support mainstream network protocols. Such standard protocols (e.g. TCP, IP, ARP, etc.) facilitate interconnection, but bring their own vulnerabilities (e.g. Spoofing, Replay, MITM, etc.). However,

Table I
EXAMPLES OF PLC NETWORK VULNERABILITIES AS REPORTED IN ICS-CERT ADVISORIES

Advisory	Affected product	Vulnerability	Exploit
ICSA-11-223-01A	Siemens SIMATIC PLCs	Use of Open Communication Protocol	Execute unauthorized commands
ICSA-15-246-02	Shneider Modicon PLC Web Server	Remote file inclusion	Remote file execution
ICSA-12-283-01	Siemens S7-1200 Web Application	Cross-site Scripting	Run malicious javascript on Engineering station browser
ICSA-15-274-01	Omron PLCs	Clear text transmission of sensitive information	Password sniffing
ICS-ALERT-15-224-02	Schneider Electric Modicon M340 PLC Station	Local file inclusion	Directory traversal/file manipulation

the most common type of network vulnerabilities is related to ICS specific network protocols such as Modbus, profinet, DNP3, etc. which include lack of authentication, lack of integrity checking of data sent over the protocol. Table I lists a sample set of PLC network vulnerabilities as reported in ICS-CERT repository.

Firmware is the operating system of controller devices, in particular, PLCs. It consists in data and code bundled together with several features such as OS kernel and file system. As any software, a firmware is prone to flaws and security vulnerabilities. Vulnerabilities include buffer overflow, improper input validation, flawed protocol implementation, etc. More importantly, firmware and patches must be certified by vendors to make sure that they will not break system functionalities. Unfortunately, a large number of PLC vendors use weak firmware update validation mechanisms allowing unauthenticated firmware updates [?]. Table II lists a sample set of PLC firmware vulnerabilities as reported in ICS-CERT repository.

A PLC is a sensitive component of ICS systems and hence only authorized entities should be allowed to access it and any such access should be appropriately authenticated. The most common PLC access control vulnerabilities include poor authentication mechanism, lack of integrity methods, flawed password protection, and flawed communication protocols. For example, PLC vendors use hidden or hard coded usernames and passwords to fully control the device. Attackers setup a database of default usernames and passwords and can brute-force such devices. Once unauthorized access is performed, an adversary can retrieve sensitive data, modify values, manipulate memory, gain privilege, change PLC logic, etc.

III. PLC ACCESS CONTROL

A. Physical access control

Proper deployment and access control of PLC as well as other ICS controllers mitigate significantly security breaches either from internal or external adversaries. Access control vulnerabilities can be significantly reduced by implementing recommendations in established standards such as the ANSI/ISA-99 [?]. It is a complete security life-cycle program that define procedures for developing and deploying policy and technology solutions to implement secure ICS systems. ISA99 is based on two main concepts, namely, zones and conduits, whose goal is to separate various subsystems and components. Devices that share common security requirements have to be in the same logical or physical group and the communication between them take place through conduits. This way, network traffic confidentiality and integrity is protected, DoS attacks are prevented and malware traffic is filtered. In addition, control system administration must restrict physical and logical access to ICS devices to only those authorized individuals expected to be in direct contact with system equipments.

B. Network access control

ICS network access control is typically implemented in layers. The first layer is network logical segmentation achieved typically with security technologies such as firewalls and VPNs. All controller devices, in particular PLCs, must be located behind firewalls and not connected directly to corporate or other networks. Most importantly, critical devices should not be exposed directly to Internet. Remote access to all ICS devices should be through secure tunnels such as VPNs. It is important to note that firewall and VPN technologies used in ICS systems are different from mainstream firewall and VPN used in typical IT networks. Indeed, many vendors many vendors provide special appliances for securing ICS networks. For example, Siemens provides a special type of switch, namely, Scalance S, with firewall and VPN features to secure the communication from/to PLCs.

Finally, even with full deployment, these technologies may not block all breaches due to weak or inadequate configurations and filtering rules.

C. Password access control

Password based access control is by far the most commonly used type of access control. Most PLC devices have built-in password protection to prevent unauthorized access and tampering. For effective password access control, important requirements need to be satisfied. In particular, password protection:

- must be enabled whenever possible
- must be properly configured

Table II
EXAMPLES OF PLC FIRMWARE VULNERABILITIES AS REPORTED IN ICS-CERT ADVISORIES

Advisory	Affected product	Vulnerability	Exploit
ICSA-16-026-02	Rockwell MicroLogix 1100 PLC	Stack-based buffer overflow	Remote execution of arbitrary code
ICSA-13-116-01	Galil RIO-47100 PLC	Improper input validation (allowing repeated requests to be sent in a single session)	Denial of Service
ICSA-14-086-01	Shneider Modbus Serial Driver	Stack-based buffer overflow	Arbitrary code execution with user privilege
ICSA-12-271-02	Optimalog Optima PLC	Improper handling of incomplete packets	Denial of Service
ICSA-16-152-01	Moxa UC 7408-LX-Plus Device	Non-recoverable firmware overwrite	Permanently harming the device

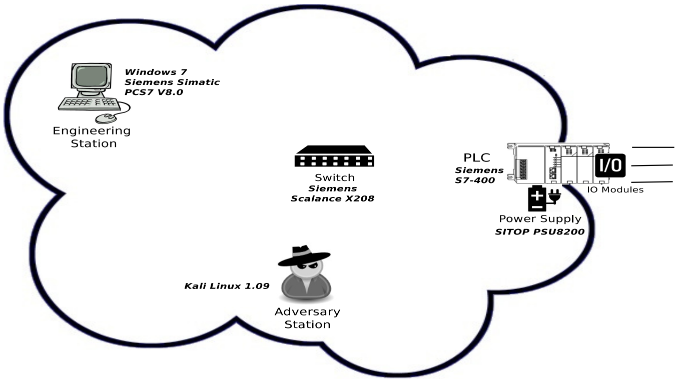

Figure 1. PLC Lab Setup

- must use strong encoding scheme
- must not need high processing operations
- must not use hardcoded credentials
- must be frequently and periodically changed.

In addition, it is highly recommended to delete default accounts or change default passwords. Unfortunately, not all vendors comply with and enforce these principles, therefore several password related incidents are reported.

IV. SECURITY ANALYSIS OF PLC PASSWORD ACCESS CONTROL

To carry out a realistic security analysis of PLC access control, we selected a commonly used PLC model, namely, Siemens S7-400, and setup a lab including common ICS configuration (Fig. 1).

Based on S7-400 documentation, several test cases have been performed which revealed three access control levels for the PLC, namely, no protection, write-protection and read/write-protection. The first level of access control, which is the default level, does not provide any form of access control. Using this level, any entity (device, station, etc.) can access the PLC processes and data without restriction. Access is possible provided that the remote entity "speaks" a PLC supported communication protocol (e.g. COTP, Modbus, Profinet). The second level, write-protection, provides as its name indicates a write protection on PLC data and processes. That is, any attempt to modify data or processes on the PLC (e.g. Load new program, clear data) is password authenticated. The third level, which is the most restrictive, is read/write-protection. Using that level, any interaction, that is, read from or write to the PLC is password authenticated.

A. Password policy

The configuration software, namely, SIMATIC PCS7 accepts any 8 ASCII characters password. If the password is less than 8 characters long, PCS7 pads it with white spaces. To set a PLC password, a user has to change the protection level and set the password in the PCS7 hardware configuration tool before loading the changes to the PLC. In addition to being loaded to the PLC memory, the password is stored locally in the engineering station's local files. In a normal scenario any command sent to the PLC (e.g. start, stop, clear memory) should be authorized by providing the password. However, since the password is stored locally in the engineering station, PCS7 software will ask for the password only one time after the new configuration is loaded to the PLC. In subsequent interactions, PCS7 will include automatically the password in the packet requests sent to the PLC.

B. PLC memory structure

As mentioned above, setting a password consists in changing the protection level, selecting a password and then loading the new configuration to the PLC memory. The latter is organized into labeled blocks. Each block holds a specific type of information (Fig. 2). Most of PLC blocks are used to organized the PLC program into independent sections corresponding to individual tasks. Function Block (FB) is a block that holds user-defined functions with memory to store associated data. Functions (FC) is used to keep frequently used routines in the PLC operations. Data Block (DB) stores user data. Organization Block (OB) is an interface between operating system and user program, used to determine the CPU behavior, for example, define error handling. System

Function Block (SFB) and System Functions (SFC) hold low level functions (libraries) that can be called by user programs such as handling the clock and run-time meters.

Therefore, information loaded to the PLC is divided into blocks as well. The password is communicated and stored in the System Data Block (SDB). SDB itself is divided into sub-blocks with different roles. The sub-blocks numbered from 0000 to 0999 and from 2000 to 2002 hold data that is updated in each download process. The rest of the sub-blocks are divided into two sets: sub-blocks from 1000 to 1005 should contain data and sub-blocks from 1006 to 1011 should contain configuration data. Loading a new program to the PLC yields to ovewriting all sub-blocks of the SDB block, except the 0000 sub-block which contains the password. If an adversary aims at updating the password, he needs to clear the 0000 block first with a dedicated command and then set a new password with another command.

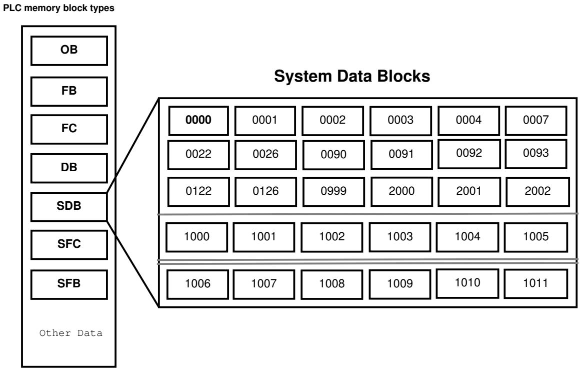

Figure 2. S7-400 PLC memory structure

C. PLC password sniffing

In order to evaluate the security of the password-based access control, a first step is to sniff the network packets containing the password. Typical network sniffing software is used to capture packets exchanged between the engineering station (PCS7) and the PLC during a password setting process (e.g. Wireshark, tcpdump). Since password setting is achieved through load configuration command sent to the PLC, the process is repeated several times with different passwords to collect a good number of samples. The captured traffic is first filtered to extract complete TCP streams. The streams are then compared using byte comparison tools (e.g. Burp Suite Comparator). These tools help finding similarities and differences between TCP streams. This allowed to identify the specific packets containing the password and the exact bytes shift for the password location inside the packets. It turned out that the 8 characters password is encoded in each packet. Hence configuration software in the engineering station uses an encoding scheme to encode the password before uploading it to the PLC.

It is important to note that when the PLC is configured with no-protection level, sniffed packets during load configuration have the same size as with the other levels of protection (read protection and read/write protection). Hence, packets are padded with random bits in place of the password in case of no-protection level.

D. Reverse engineering password encoding scheme

After locating the 8 bytes inside the network packets containing the password, the next step is to decode the bytes to retrieve the plain-text version of the password. The reverse-engineering started by trying typical encoding schemes, namely, URL encoding, ASCII Hex, Base64, variants of Xor (single-byte, multiple-byte, rolling, etc.). However, none of these typical schemes retrieved the plain text version of the password, pre-set in our samples. Full-fledged cryptographic (DES, AES, RC4, etc.) as well as hashing (MD5, SHA512, etc.) functions are excluded in the investigation because of three reasons. First, there is no key exchange stage involved before password communication[1]. Second, if cryptographic and hashing functions were used, the encoded password bytes would be completely shuffled compared to the plain text version, which is not the case here (the cipher text is encoded byte by byte). Third, cryptographic and hashing functions are too processing intensive for PLCs.

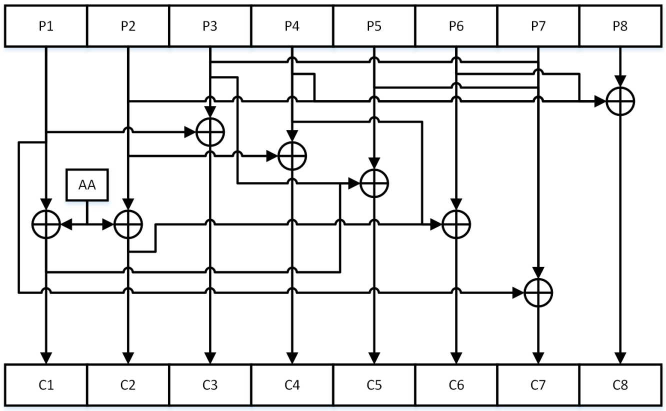

Figure 3. PLC Password Encoding

Xor is a very common encoding scheme that is suitable for resource limited hardware devices. As mentioned above,

[1] This holds for cryptographic functions.

the password encoding is not using typical Xor (single-byte, multiple-byte, etc.). Taking into consideration the fact that the encoding is done byte-by-byte and the requirement of a lightweight encoding algorithm, we focused on trying customized Xor transformations. To this end, a representative list of (plain-text password, encoded-text password) pairs have been sampled from the network. Then, using automated scripts to brute-force each byte, we could successfully reverse-engineer the Xor based encoding scheme. A graphical representation of the nested Xor based encoding scheme is shown in Fig. 3. It is important to note that the PLC is using two variants of the encoding scheme: one used to load a configuration to the PLC and the other is used during the authentication process. Both variants differ by the staic byte constant used: “0x55” and “0xAA”.

V. PLC Access Control Attacks

As a consequence of compromising the password based PLC access control, several concrete attacks can be carried out on the PLC ranging from simple replay to unauthorized password update attacks.

A. *Replay attack*

A replay attack on the PLC consists in recording a sequence of packets related to a certain legitimate command and then replaying it later without authorization. The attack consists of 3 steps: starting a given command (stop, start, load configuration, clear memory block, etc.), capturing the packets, and replaying the captured packets at a later time. The target PLC may or may not be password protected.

are accepted by the TCP/IP kernel at the PLC, We resorted to write a customized python script using scapy [**?**]. Scapy is a powerful packet manipulation program written in python and hence can be easily used in python scripts. It features a variety of packet manipulation capabilities including: sniffing and replaying packets in the network, network scanning, tracerouting, etc. However, the most useful scapy features for our replay attack are the ability to rewrite the sequence and acknowledgement numbers and to match requests and replies. Algorithm 1 shows the core of the python script using the scapy features.

Algorithm 1 Replay a sequence of captured packets using Scapy

```
function REPLAY(pcapfile, eth_interface, srcIP, srcPort)
    recvSeqNum ← 0
    SYN ← True
    for packet in rdpcap(pcapfile) do
        ip ← packet[IP]
        tcp ← packet[TCP]
        del ip.chksum                          ▷ Clearing the checksums
        ip.src ← srcIP                         ▷ Attacker’s machine IP
        ip.sport ← srcPort                     ▷ Attacker’s machine Port
        if tcp.flags == ACK or tcp.flags == RSTACK then
            tcp.ack ← recvSeqNum+1
            if SYN or tcp.flags == RSTACK then
                sendp(packet, iface=eth_interface)
                SYN ← False
                continue
            end if
        end if
        rcv ← srp1(packet, iface=eth_interface)
        recvSeqNum ← rcv[TCP].seq
    end for
end function
```

The above python program has been tested using two attack scenarios. In the first scenario, the replay attack was launched from the same host (IP address) used for the capture, that is, the engineering station with the configuration software. In the second scenario, the replay attack was launched from a different host on the same network, that is, the attacker machine with Kali. In each scenario, two types of commands are tried, namely, start and stop which require password authentication. The replay attack was successful in both scenarios for both types of commands. Hence, an unknown attacker machine (without appropriate configuration software) on the same network, can turn the PLC ON or OFF by simply replaying a start or stop command without knowing the PLC password. This clearly might cause significant damage to a SCADA system.

1) Password stealing: As detailed Section IV, packets between the engineering station and the PLC are sent in clear including the encoded passwords. Based on a representative set of samples, we could locate the password inside packets and reverse-engineer the password encoding scheme. This allowed us to retrieve the plain-text password from the network traffic between the engineering station and the PLC.

2) Unauthorized password setting and updating: In a legitimate scenario, the PLC password is set and updated from the configuration software in the engineering station. In case of password update, the old password should be supplied first. Due to the PLC access control vulnerability, an attacker can set and update the password by replaying malicious packets directly to the PLC.

When a password is written on the PLC, the SDB (System Data Block) is overwritten. The load process first checks the SDB to see if it’s clean or has a configuration already. If there is a configuration, the process checks if a password is set or not. Hence, there are two main cases: setting a configuration with a password for the first time and updating an old configuration that has already a password.

For the first case, setting a password for the first time requires to record a password setting packets sequence used in an old session and then replaying them. Since the goal is mainly to set the password, only packets in charge of overwriting block 0000 in the SDB, which contain the password, are kept (More details in Section IV-B).

For the second case, the goal of the attack is to set a password while the PLC is already protected by an existing password. Using the same procedure as the first case as-is did not work. After investigation it turned out that the block 0000 of the SDB holding the password cannot be overwritten by replaying packets. As a result, the PLC keeps sending a

FIN packet whenever an attempt is made to overwrite the SDB. To overcome this problem, we resorted to a two-stage procedure where initially we clear the content of 0000 block and then we replayed packets to overwrite only that block with a new password. Since there is no legal command to just clean 0000 block, we looked for a sequence of packets to delete a different block and we modified them to delete 0000 block. With this two-stage procedure, the password is successfully updated by a different workstation without the configuration software and without knowing the old password.

3) Clear PLC memory: The first stage of the unauthorized password updating attack consists in clearing the 0000 block of the SDB without a need for the password. This step can be generalized to clear other blocks. More importantly, in an extreme use case, all PLC memory blocks can be cleared. With this vulnerability, an attacker can launch a DoS attack by clearning all PLC memory and turning the PLC into unresponsive device.

VI. Related work

Very close to our work, in a BlackHat talk, Beresford demonstrated a number of vulnerabilities in Siemens Simatic PCS7 software including replay attacks, authentication bypass, fingerprinting and remote exploitation using Metasploit framework [**?**]. This paper deviates from Beresford's demonstrations since our attacks are more interactive and use the recent and more secure versions of the PCS7 software as well as the more uptodate firmware of Siemens PLC S7-400. As a generalization of Beresford's attacks, Milinkovic and Lazic reviewed a set of commercial Operating Systems running on PLCs of major vendors, highlighting serious vulnerabilities with some experiments of few attacks conducted on ControlLogix PLC [**?**].

Also close to our work, Sandaruwan et al. showed how to attack Siemens S7 PLCs by exploiting flaws in the ISO-TSAP (Transport Service Access Point) protocol used for data exchange between controllers and PLCs [**?**].

A significant body of work in the literature focuses on security solutions for ICS systems which yield several countermeasures to reinforce the security of such systems. These can be classified into communication protocols improvement [**?**], [**?**], and firewalls, filtering methods, DMZs [**?**], [**?**], [**?**]. However, unlike typical IT systems, it is impractical and cost-effective to embrace several layers of mitigations due to performance and availability considerations.

VII. Conclusion

PLCs are preferred target for cyber security attacks. PLC security issues range from simple DoS to sophisticated remote code execution vulnerabilities. Most of PLC attacks are possible because attackers could have access and compromise the PLC device. In this paper, we carried out a security analysis of the most common PLC access control mechanism, namely, password-based access control. Using recent PLC devices (2016) with updated firmware, we showed how passwords are stored in PLC memory, how passwords can be intercepted in the network, how they can be cracked, etc. As a consequence of these vulnerabilities, we could carry out advanced attacks on ICS system setup, such as replay, PLC memory corruption, etc. Although mitigating such vulnerabilities is relatively easy by placing a security module (e.g. Scalance S) between the PLC and other devices, such approach is not yet widely deployed for budget and practical considerations.

Acknowledgment

This research was supported by The National Science, Technology and Innovation Plan (NSTIP) grant, NSTIP 13-INF281-04 at King Fahd University of Petroleum and Minerals.

References

[1] N. Falliere, L. O. Murchu, and E. Chien, “”w32.stuxnet dossier”,” *White paper, Symantec Corp., Security Response*, 2011.

[2] A. Costin, J. Zaddach, A. Francillon, and D. Balzarotti, “A large-scale analysis of the security of embedded firmwares,” *23rd USENIX Security Symposium (USENIX Security 14)*, pp. 95–110, 2014.

[3] E. Byres, “Revealing network threats, fears: How to use ansi/isa-99 standards to improve control system security,” 2011.

[4] P. Biondi, “Scapy,” *see http://www. secdev. org/projects/scapy*, accessed on 2016-09-20.

[5] D. Beresford, “Exploiting siemens simatic s7 plcs,” *Black Hat USA*, 2011.

[6] S. A. Milinković and L. R. Lazić, “Industrial plc security issues,” *Telecommunications Forum (TELFOR)*, pp. 1536–1539, 2012.

[7] G. Sandaruwan, P. Ranaweera, and V. A. Oleshchuk, “Plc security and critical infrastructure protection,” *2013 IEEE 8th International Conference on Industrial and Information Systems*, pp. 81–85, 2013.

[8] M. Majdalawieh, F. Parisi-Presicce, and D. Wijesekera, “Dnpsec: Distributed network protocol version 3 (dnp3) security framework,” in *Advances in Computer, Information, and Systems Sciences, and Engineering*. Springer, 2007, pp. 227–234.

[9] J. Heo, C. S. Hong, S. H. Ju, Y. H. Lim, B. S. Lee, and D. H. Hyun, “A security mechanism for automation control in plc-based networks,” *2007 IEEE International Symposium on Power Line Communications and Its Applications*, pp. 466–470, 2007.

[10] R. E. Johnson, “Survey of scada security challenges and potential attack vectors,” *Internet Technology and Secured Transactions (ICITST), 2010 International Conference for*, pp. 1–5, 2010.