Journal of Cyber Security
DOI:10.32604/jcs.2020.010045
Review

Review of PLC Security Issues in Industrial Control System

Xiaojun Pan, Zhuoran Wang and Yanbin Sun*

Cyberspace Institute of Advanced Technology, Guangzhou University, Guangzhou, 510006, China
*Corresponding Author: Yanbin Sun. Email: sunyanbin@gzhu.edu.cn
Received: 06 February 2020; Accepted: 28 June 2020

Abstract: Programmable Logic Controllers (PLC), core of industrial control systems, is widely used in industrial control systems. The security of PLC is the key to the security of industrial control systems. Nowadays, a large number of industrial control systems are connected to the Internet which exposes the PLC equipment to the Internet, and thus raising security concerns. First of all, we introduce the basic principle of PLC in this paper. Then we analyze the PLC code security, firmware security, network security, virus vulnerability and Modbus communication protocol by reviewing the previous related work. Finally, we make a summary of the current security protection methods.

Keywords: Industrial control system; PLC security issues; PLC safety protection

1 Introduction

Industrial Control Systems (ICS) are widely deployed in basic industries closely related to civil life, such as energy, transportation, water conservancy, electric power, oil and so on. With the rapid development of the Internet, more and more industrial control systems that used to be physically isolated from the outside world are now connected to the Internet via universal protocols [1]. However, as the industrial control systems are initially designed without taking being connected to the Internet into consideration, the protocols it adopted are nowadays found lacking encryption, authorization, authentication and other mechanisms. And there are many vulnerabilities in PLC's firmware and software. The previous methods used to penetrate traditional networks can now be applied to industrial systems. An increasing number of cyber-attacks on SCADA and DCS, which adopted these protocols and PLC devices, has become the status quo. And such cyber-attacks are causing more and more serious issues.

The increasing number of cyber-attacks makes countries and organizations worry more about security of industrial control systems and invest more into the related research [2].

In 2011, Stuxnet infected more than 45,000 network systems around the world and caused about 20% of the centrifuges at Bushehr nuclear power plant in Iran to fail or even be scrapped. In 2014, a German steel plant was hit by an ART cyberattack that forced production to stop. In June 2015, Polish Airlines suffered a hacker attack, which forced about 10 flights to be cancelled. On December 23, 2015, at least three regional power systems in Ukraine were attacked by malware, causing about 1.4 million people to lose power. In 2017, ransomware was able to use PLC's own loopholes to fully control it.

In recent years, the vulnerabilities of industrial control systems have increased dramatically. Until May 2020, according to the data of CHINA NATIONAL VUENERABILITY DATABASE, the number of vulnerabilities in the industrial system industry is 2574.

Because PLC devices do not have an identity verification mechanism, and the protocols adopted by PLCs do not have encryption, identity authentication and other security methods, if an attacker get the PLC's IP address, he can upload or download the malicious programs and send the malicious instructions. In 2011, McLaughlin et al. [3] proposed a method to dynamically generate malicious payloads of PLC,

which greatly increased the risk of PLC being attacked. In 2015, Klick et al. [4] inserted a backdoor in the S7-300, enabling local scanning and other functions. In 2016, Spenneberg et al. [5] realized a worm existing in S7-1200 that was able to self-propagate.

2 Introduction to Industrial Control System and the Principle of PLC

PLC and its accessories are the core of industrial control system. For a better review of industrial control systems, the architecture of industrial control system, the equipment and working mechanisms of PLC, this section mainly introduces the industrial system architecture, its working mode, programming language and firmware of the PLC.

2.1 The Working Principle of Industrial Control System

A typical industrial control system is composed of field equipment layer, field control layer, control network and process monitoring layer. The field equipment layer is mainly composed of field actuators, field sensors, etc.; the field control layer is mainly composed of PLC and related components; the process monitoring layer is mainly composed of PLC related configuration software and server.

There is a large number of universal protocols, such as Modbus and profibus, adopted between PLC and the host computer or between PLC and PLC for data transmission. Due to the lack of protection, these protocols can be easily exploited by attackers. The system structure is shown in Fig. 1.

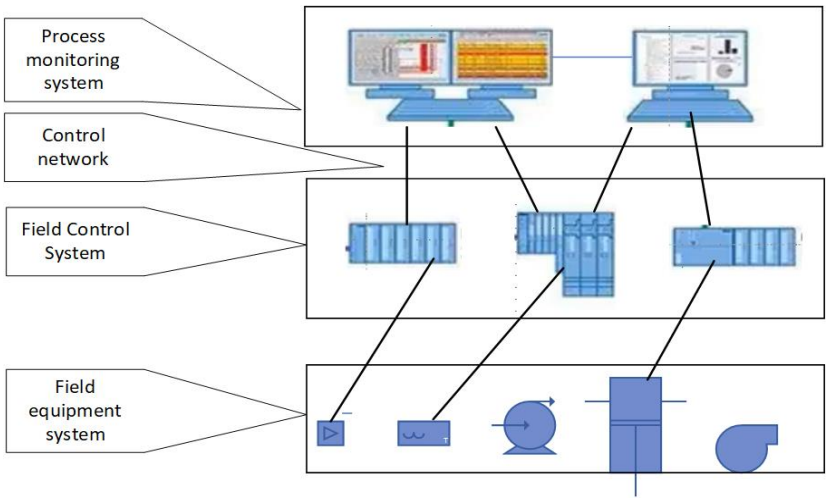

Figure 1: The architecture of industrial control system

2.2 The Working Principle of PLC

2.2.1 The Hardware Composition of PLC

As shown in Fig. 2, PLC is composed of CPU, power supply, I/O modules, memory and expansion modules [6]. Its' hardware composition and structure are basically the same as those of traditional computers.

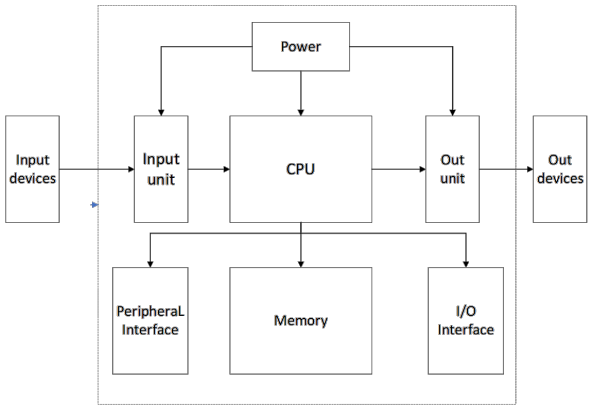

Figure 2: The structure of PLC

The CPU is responsible for executing user programs and system diagnostics, etc. The power module is responsible for powering modules such as CPU, memory, and I/O. The memory is used to store user programs and system programs. The I/O module is responsible for inputting external input signals and outputting control signals to control the actuators.

2.2.2 The Working Mechanisms of PLC

The PLC adopts the cyclic scanning working mode. When the PLC is in the STOP state, it only responds to communication and internal processing calls. In the RUN state, during the phase of input scanning, the PLC scans the external input successively and stores it in the input image register, then it executes the user program and outputs the result to the output image register. Finally, during the phase of output scanning, the contents of output image register are sent to the output latch, and then the latch produces an output in a certain way, such as a relay, a thyristor or the like, to control the execution module.

The time that a cyclic scan takes is called the scan period [7]. As is shown in Fig. 3, Since the PLC adopts the sequential execution and cyclic scanning flow, it will refresh the input area and the storage area every time the program is executed, leading to that attackers have time and means to hack the PLC device, for example, they may use worms.

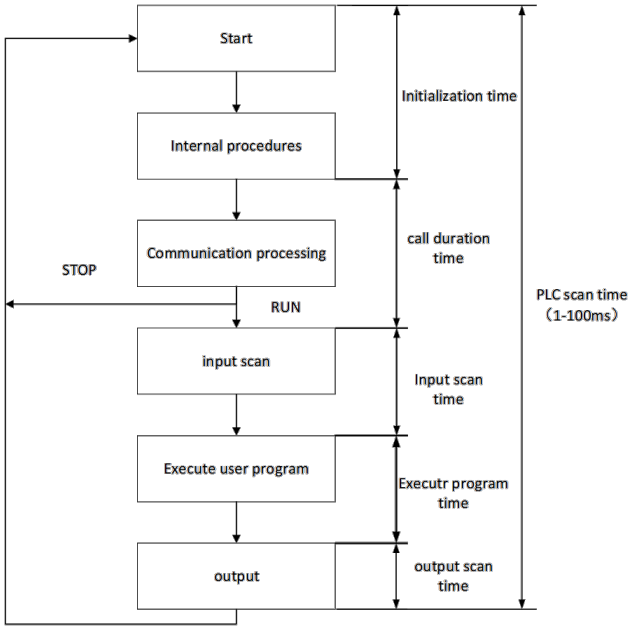

Figure 3: Scan period of PLC

2.2.3 PLC Programming Language

PLC programming languages include Ladder Language (LD), Instruction List Language (IL), Function Block Diagram Language (FBD), and Sequential Function Flow Chart Language (SFC). Please refer to the official programming reference manual. Fig. 4 is a simple Siemens PLC ladder diagram.

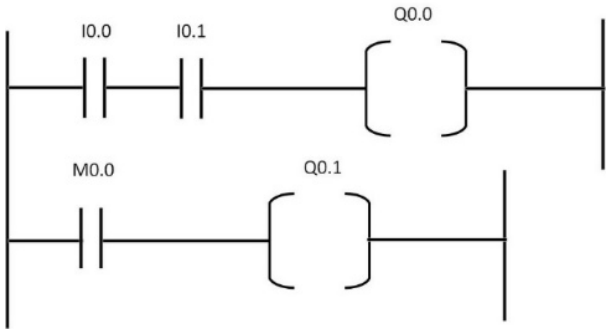

Figure 4: PLC Ladder Language

The left side of the PLC is the start line, and the right side is the end line. Ladder Language is scanned and executed from left to right and top to bottom. Each component symbol has a label indicating its memory address. In Fig. 4, I0.0, I0.1 represents the external input point, and Q0.0, Q0.1 represents the output point, and M0.0 represents the intermediate state of Q0.0. When we input high level to I0.0 and I0.1, Q0.0 outputs a high level, and M0.0 also switches to high level, then Q0.1 outputs high level. The symbols and syntax of PLC ladder diagrams from different manufacturers are different. Please refer to the official programming reference manual. Due to the function module language and the programming methods it adopts, PLC is susceptible to security issues such as logical loopholes.

2.2.4 PLC Firmware

Firmware is the infrastructure of the system [8], serving the fundamental functions, such as controlling the hardware and other communication equipment. Operating system and software rely on firmware to implement specific functions. For example, the basic I/O system (BIOS) of a computer is a firmware. The firmware of the PLC generally supports online update, whereas its integrity check is generally not strict, which makes the firmware easily to be tampered by hackers.

PLC system generally consists of hardware, firmware, and software. Firmware simultaneously provides interfaces for hardware and software. As is shown in Fig. 5.

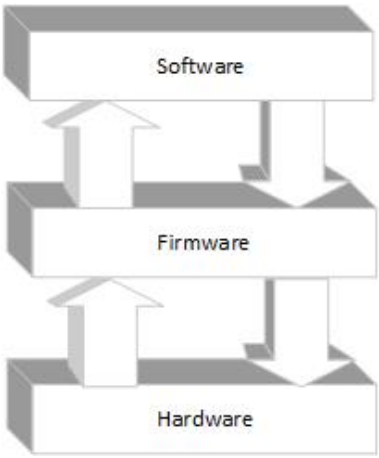

Figure 5: PLC system structure

3 PLC Security

The essence of PLC is a simple computer, following the instructions from the host computer, processing the user program, producing an output to control the operation of the field device, and transmitting the data of the field device to the host computer. In this execution process, there are security concerns like user code loopholes, firmware tampering and cyber-attacks. This section introduces the safety problems which PLC confronts with and also introduces some research on the protection methods.

3.1 Code Security Issue

3.1.1 Code Security Classification

During the ladder diagram drawing process, there are some potential security loopholes because of negligence of personnel. This section introduces these security issues. Tab. 1 shows a classification and description of code security issues [9].

Table 1: Code security issues

Classification	Security issues	Consequences
Logic errors	Timer condition competition	May cause timers to oscillate
	Unconditional transfer	Cause some branches to be unreachable
	Code unreachable	Some code cannot be reached through the normal control flow
	Infinite loop	Resulting in a loop timeout error
Syntax errors	Comparator hardcoded	Easy to be attacked and changing the process
Scope and linkage errors	Missing jumps and links	May be embedded in a malicious program
Hidden jumpers	Hidden jumpers	Can change the process of program execution
Duplicate objects	Object repeat reference	Multiple inputs independently control the same output, causing output failure
Unused objects	Unused objects	Easy to be exploited by attackers

3.1.2 Code Security Formal Analysis Method

PLC code security formal analysis is to detect the security problems in the PLC. Model detection is currently the most widely used PLC formal analysis method. It translates the language like PLC ladder diagram or instruction list language into intermediate language, which is convenient for model builder. It then takes environment and other factors into consideration to construct the model. The constructed model will be clipped to solve the path space explosion problem and then leverage symbolic execution to solve the space state explosion problem. Finally, it will check various code defects. Technical routine is shown in Fig. 6 [10–11].

PLC code security formal analysis uses symbolic execution, which consumes a lot of time and memory, to solve the problem of space explosion, so once the program to be detected is too large, it will cause a longer detection time. And because it uses a static analysis method, it has no detection capability for the fake sensor value assigned to the PLC.

Intermediate language translation is to convert the programming language of PLC into a specific language to facilitate the modeling detection technology. In order to better solve the problem of intermediate language translation, Daravas proposed an intermediate model method based on NuSMV [12], and McLaughlin [13] proposed a Vine-based approach.

NuSMV is divided into the following steps. The first step is to parse the PLC ST source code. The second step is to convert the parsed ST source code into an intermediate automaton model. The third step is to create a NuSMV official model from the intermediate automaton model. However, in the process of converting the ST source code to NuSMV, the order of operations will be highly affected by the output format, making the conversion difficult. Incremental syntax construction may cause problems such as code reconstruction.

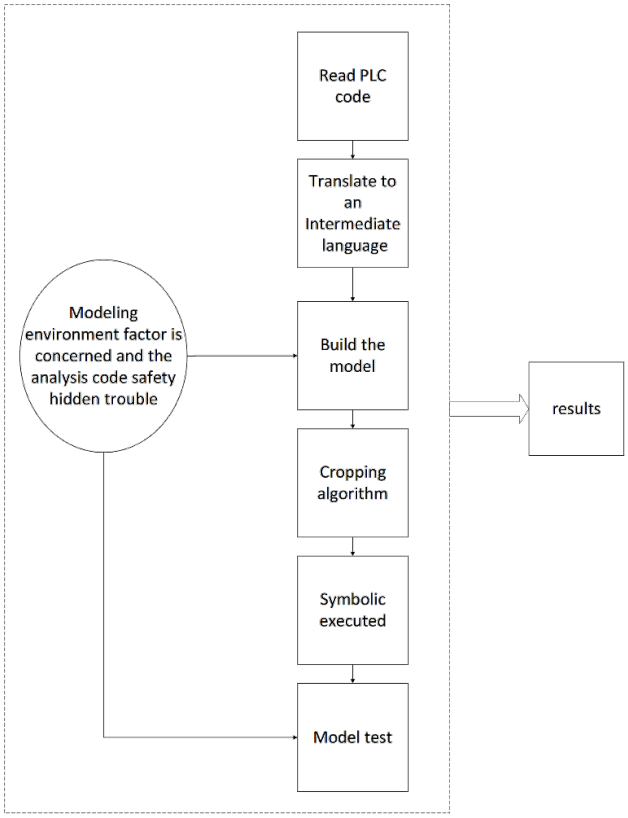

Figure 6: PLC code security formal checking

Model detection can build an appropriate model to auto-inspect the system. Currently SMV, UPPALL, SPIN, Verilog, TSV and so on are used in model checking. In order to solve the problem of state space explosion, GUO et al. [14] developed SymPLC. SymPLC is an automatic symbol detection tool, which can convert the PLC source code into C before symbolic execution. Symbolic execution can solve the problem of state space explosion. But symbolic execution may consume a lot of time and memory, which is not suitable for detecting programs that are too large.

3.2 PLC Firmware Security

3.2.1 PLC Firmware Modification Attack

Many PLCs allow users to download firmware updates remotely, and the firmware verification adopts CRC checksum algorithm. However, such algorithm cannot detect if the firmware has been intentionally modified. Carl Schuett et al. [15] implemented the attack on the PLC by repackaging the modified firmware.

The experimental platform structure is shown in Fig 7.

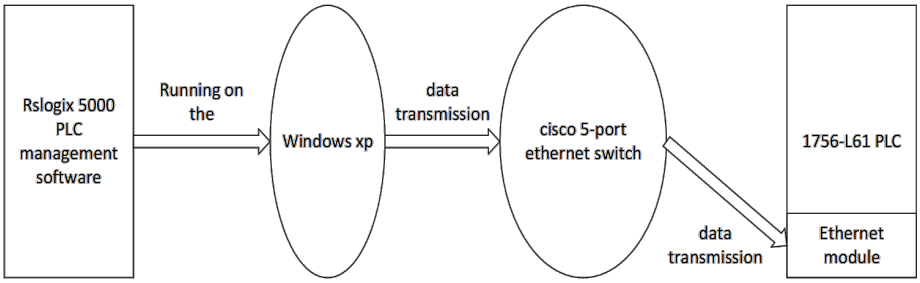

Figure 7: Experimental platform architecture

The test equipment consists of a 1756-L61ENBT Ethernet module and a 1756PA75-B chassis. The attack steps are as follows:

The first step is Hardware analysis. Hardware analysis primarily uses the readview ICE debugger to identify CPU. The ControlFlash update only captures incompletely transferred data rather than fully corrupted firmware images. So ControlFlash or locally developed programs are also used to handle updates. PLC will check the firmware twice by CRC32 checksum algorithm [16].

The second step is Firmware analysis. The firmware analysis mainly modifies and repackages the firmware image to execute the program modified by attacker. It is typically statically analyzed by IDA PRO and tested by the JTAG structure of the device.

The last step is identifying modifiable firmware section. An attack could happen by modifying the modifiable section of firmware.

Because of PLC firmware verification defects, the CRC verification process for firmware is trusted by the PLC, but CRC can only detect if the firmware is damaged, not if it is maliciously tampered. Once the manufacturer changes the verification method of PLC firmware, this method will be invalid. Thus this method is not universal when different PLC manufacturers use different firmware verification methods. Moreover, manufacturers need to master a large number of assembly and other low-level knowledge, which also raises the threshold of this attack method.

3.2.2 PLC Firmware Security Detection Approach

Qiao et al. [17] proposed a method for designing trusted PLC, which uses a trusted chip and trust metric root, combined with hash algorithm, to implement the trusted start of PLC through the transfer of the trust chain during the system startup phase.

The fitst method is Authentication mechanism. The PLC system startup file includes BOOT.BIN, devicetree. Dtb, uImage and uramdisk.image. The trust chain relies on the firmware code BL0 as trust foundation. The start-up link measures the security of the next start-up and extends the metric value into the PCR. Only if the next startup link is secured, the control will be transferred to it.

The second method is Integrity check. When the device is powered on, the hash code segment in each file is called by system startup order to perform integrity check. If the hash value is not the same as the previously stored standard value, the file will be deleted directly. All startup files will be verified in turn to make sure that once one startup file is modified, the device cannot be started, guaranteeing the security of PLC firmware.

However, it does not check the integrity of files that are unnecessary during the startup process but only the files necessary for the PLC system to start. The attacker can tamper with these files to achieve the purpose of the attack.

Adelstein et al. [18] Introduced a detection method based on firmware signatures, allowing the firmware to be checked at runtime. The verification technology is based on Efficient code certification

(ECC) technology. When loading the firmware module, it will be verified according to the standard security policy. ECC is implemented based on a certified compiler. However, this technology is only for malicious firmware while malicious hardware and denial of service attacks cannot be detected. On one hand, this method increases the credibility of the firmware during PLC startup and operation. On another, it cannot verify non-open firmware security as the technology is based on open firmware.

3.3 PLC Network Security

3.3.1 PLC Network Security Threats

One of the important components of the industrial system is the SCADA system which is responsible for field data acquisition and monitoring. As PLC equipment being the core controller of SCADA, the interference with the operation of the PLC may cause irreversible loss. There are numerous security issues in the communication between SCADA and field PLC. Once the SCADA system is connected to the Internet [19–20], this potential threat to the SCADA system is more likely to be exploited by attackers.

Gao et al. [21] launched attacks including response injection, command injection, man-in-the-middle attacks, replay attacks, and denial of service attacks, on several protocols adopted by SCADA such as Modbus, DNP3 and Alien Bradley. Morris et al. [22] attacked on Modbus by replay attacks, response injection, command injection and denial of service attacks. It can be seen that SCADA is very vulnerable. Ghaleb1 et al. [23] described replay attacks, man-in-the-middle attacks, and stealth command modification attacks on SCADA. Andres et al. [24] analyzed the network attacks on the PLC input and output memory, which can be used to interfere with system operation. This paper specifically introduces cyber-attack on Siemens PLC.

Replay attack. The replay attack can be launched by the same host address for capturing or by another host on the same network. Both replay attacks are effective and capable of starting and stopping the PLC.

Man-in-the-middle attack. TCP/IP and COTP protocols are used between the PCS7 host computer and the PLC, while the ARP address resolution protocol is adopted by Ethernet. This communication is susceptible to MITM (Man-in-the-middle attack). Man-in-the-middle attack can be either active or passive. Passive mode is to monitor the traffic of PLC communication and obtain data; active mode is to tamper with data and commands in order to control the PLC and interfere with the normal system operations.

Stealth command modification attack. The stealth command modification attack is a combination of replay attack and man-in-the-middle attack. It is used to obtain communication between PCS7 and PLC and replay other commands in stealth mode to interfere with PLC operation.

PLC worm virus. PLC adopts the cyclic scanning, and a complete scanning cycle is 1-100 ms. Every time after the execution of a program, PLC will refresh the input area and storage area, which offers an easy loophole for the attacker to modify the user program content. The attacker can leverage the communication protocol to modify the user execution program. The operation flow chart of PLC virus is shown in Fig. 8. The PLC virus calls system functions, then takes advantage of the proprietary protocol to connect with other PLCs, and then forges the communication, finally injecting itself into the PLC through the datagram.

The PLC virus hides its own program by storing the program in the OB block and the data required for communication in the DB block. When infected, the PLC will run in the order of the OB block (Fig. 9). During propagation, the virus needs to scan the target and inject itself using the protocol. And the virus will actively connect to the C and C server which remotely controls the PLC and can infect other PLCs connected to it via the Sokcs4 proxy.

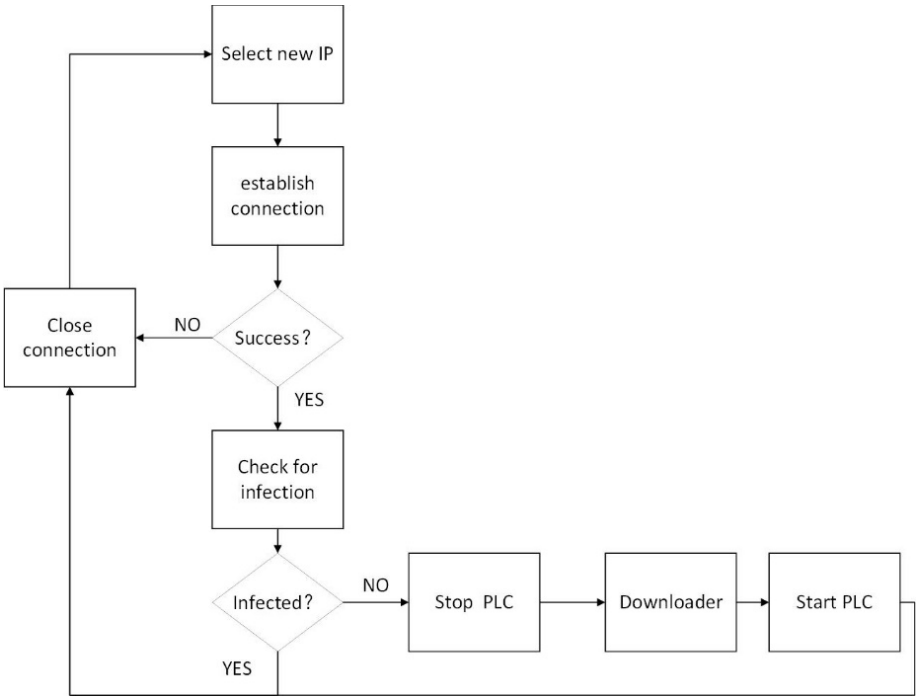

Figure 8: PLC virus infection process

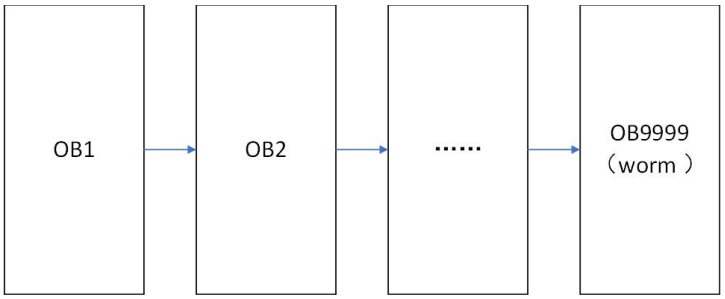

Figure 9: Organization block run sequence

Melaughlin et al. [25] designed a malware for PLCs that can generate dynamic packet payloads. They also built Sbot [26], which can automatically generate malicious code, facilitating PLC attacks.

3.3.2 PLC Network Security Protection

Chen et al. [27] proposed a security supervision program for industrial control networks. The program adopts three data acquisition schemes (Fig. 10).

The network collection utilizes the switch image to collect the traffic passing through the switch; the protocol direct collection means direct communication with the device to obtain data; the collection agent exploits the agent module installed on the device or the operating system to collect log and other data.

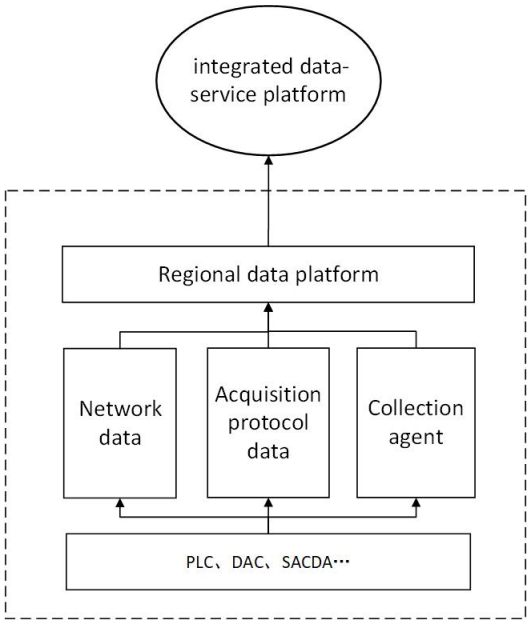

Figure 10： Network security supervision system structure

All of three data acquisition schemes could go deep to the field control equipment of control system, and also to monitor the original network data or the control equipment. It achieves external network intrusion detection, supervision of network layer of the industrial control system, and monitoring of field devices. Approaches shown as follows:

External intrusion detection [28]: the supervisory system monitors whether the network host activates the remote connection service or adds unauthorized processes or ports to determine if the system has been intruded by the external network.

The monitoring system can obtain the data traffic in the industrial system. If an OPC read command from an unknown source is detected, the supervisory network layer may have been invaded.

As the "write" command is sensitive, the consistency detection method could be adopted: the supervisory system obtain the 'write' command data from the device on the underlying layer, and then compare it with the detected value from monitoring software to decide if it has been tampered.

The system can detect the increased process and port of the industrial network host of the industrial control system; detect read commands and write commands from unknown sources, then compare it with the parameter value form the monitoring software to verify and intercept these commands. However, the system is based on comparative verification under the condition that the monitoring software is credible. Once the monitoring system is attacked and controlled, and the malicious program could hide itself, its verification of the read and write instructions would become invalid, and the check instruction would be intercepted, which would affect the real-time requirements of industrial control systems to a certain extent.

Siemens PLC uses a fixed memory space for input and output. The monitoring software can access these memory spaces to obtain or modify the PLC's operating data by using network, but an attacker can also access these data and even modify them remotely by using network. The PLC updates its memory every cycle (usually in units of 1 millisecond). Andres et al. [24] responded to intrusion by detecting PLC memory changes. The purpose of the attacker is to cover the PLC memory distributed to the input, thus this method can be used to detect attacks on inputs and outputs, but it is based only on theoretical analysis.

First, this memory detection has no way to prevent the input register from being modified, because after detecting the attack, the PLC needs to take the value from the input register, which results in the value obtained by the PLC being changed throughout the cycle. And second, the detection of this program will

increase the delay time of the control system. Moreover, the detection mechanism relies on the memory optimization function of Siemens s7-1200 PLC, therefore it has no detection capability for the old PLC.

In order to detect malicious behaviors such as worms that maliciously modify PLC code, Stephen McLaughlin et al. [13] proposed to convert the PLC code to intermediate language, using model checking and symbolic execution to generate the PLC time execution graph, and finally verify the security attributes in each state to determine whether the code is safe. The method is shown as Fig. 11.

However, this method cannot intercept data injection attacks. And because the method of symbolic execution is used to solve the problem of space explosion, for larger programs, it takes more time to detect. Besides, there is no way to detect some precise time security attributes, such as traffic light control time, etc.

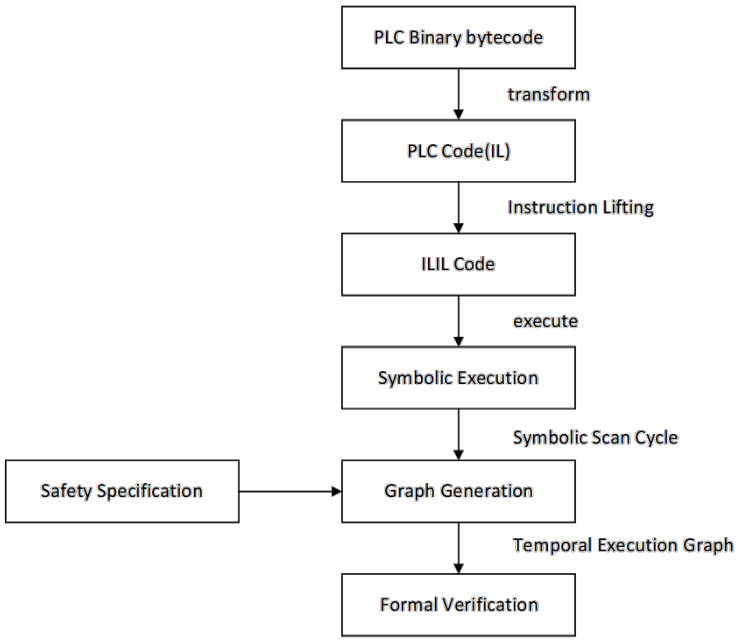

Figure 11: PLC code detection

3.4 Modbus Protocol Security

3.4.1 Modbus Protocol Security Issue

The Modbus protocol is a widely used protocol in SCADA systems. Following this protocol, between PLC and PLC, or PLC and host computer, data communication is accessible. The Modbus protocol includes two transmission modes: RTU and ASCII. Modbus TCP/IP is designed to make the Modbus protocol transmission over Ethernet encapsulated. The Modbus TCP message format is shown as follows (Tab. 2):

Table 2: Modbus TCP message format

Interactive identification	Protocol identifiers	Packet length	Device identification	Function code	Data
2 bytes	2 bytes	2 bytes	1 byte	1 byte	N bytes

The Modbus protocol has the following security issues [29]:

Lack of authentication. It can be seen from the table above that it lacks identity authentication in Modbus protocol. An attacker could forge a legitimate session to control a field device.

Lack of authorization. In Modbus protocol, there is no access control demarcation. This could lead to that any user can launch any code, which is an easy access for attackers.

Lack of encryption. Modbus protocol adopts no encryption protection on the data, then the attacker can easily eavesdrop on the communication and obtain confidential data.

3.4.2 Modbus Protocol Protection

Given numerous security issues in Modbus, many various studies have come out to increase the security of the Modbus protocol. At present, research on the security enhancement of Modbus is mainly carried out in three directions: one is to add on security mechanism to the gateway, the another one is to add on security mechanism to the terminal device, and the third is to encrypt the protocol data. Igor et al. [29] designed a modbus security architecture based on socket communication (Fig. 12).

In this system, sockets are used for communication between the computer operating system and Modbus devices. TCP/IP only uses TCP to provide streaming sockets.

Modbus Stream Builder is responsible for extracting the Modbus data packet and sending it to the RSA encryption/decryption unit to obtain the authenticity of the SHA2 digest and send the real digest to the SHA2 verifier to verify the integrity of the data packet. Finally, send the timestamp to the timestamp analyzer to verify the security of the data transmission time. Modbus ADU Builder/Reader is responsible for Modbus application data unit, also it collaborates with SHA2 and RSA unit to verify the identity of the data packet.

Although it can intercept various attacks with minimal overhead, this method unconditionally trusts the master device. If the master device is controlled by the attacker, this method cannot solve the malicious Modbus message sent by the master device; and if the private key of the main control device is mastered by the attacker, this method cannot achieve interception.

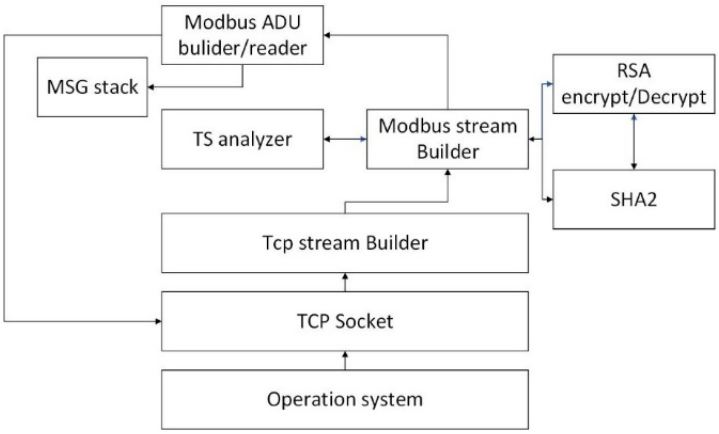

Figure 12: Modbus security architecture

Roberto et al. [30] proposed a formal modeling of the Modbus protocol and made it automatically converted to the Promela model by adopting the layered state extension process. And the model achieves locating susceptible attribute by model checking method. In order to avoid erroneous behaviors like low-level grammar and formalism, it adopts state-based advanced formalization (DSTM) to describe the protocol as a state aggregator, and then export symbols, which can be checked by the model checker (SPIN), derived from the source model using model transformation. The method is as Fig. 13.

Promela is a text verification model. The Promela model of the Modbus protocol adds the time logic formula obtained from the data traffic. Then SPIN model checker will check this model.

Although this model can detect security vulnerabilities in the Modbus protocol and prove that there is a man-in-the-middle attack in the Modbus-based system. But at present, the system cannot be used for the detection and interception of Modbus traffic in industrial control systems. Its main usage in industrial

control systems is to analyze the security vulnerabilities of Modbus and establish a security attribute database to capture network attacks.

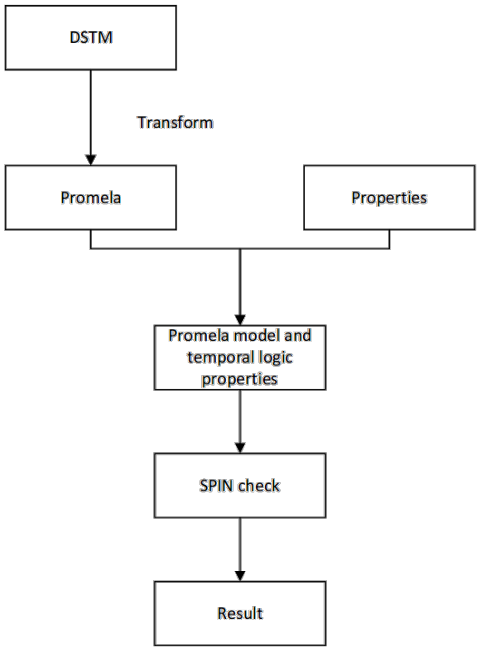

Figure 13: Approach overview

Promela is a text verification model. The Promela model of the Modbus protocol adds the time logic formula obtained from the data traffic. Then SPIN model checker will check this model.

Although this model can detect security vulnerabilities in the Modbus protocol and prove that there is a man-in-the-middle attack in the Modbus-based system. But at present, the system cannot be used for the detection and interception of Modbus traffic in industrial control systems. Its main usage in industrial control systems is to analyze the security vulnerabilities of Modbus and establish a security attribute database to capture network attacks.

Aamir et al. [31] encrypted Modbus protocol sessions using AES, RSA, and SHA-2 advanced encryption algorithms to enhance their security. They designed and configured a new password buffer (CB) in the protocol message stack. They used the secret key generated by the AES algorithm to encrypt the opcode and data segments in the Modbus protocol and stored them in M1, and used the SHA-2 pair opcode and data segments to calculate the byte hash number and store it as M2, and the RSA algorithm is used to carry out private key processing to generate M3. Then they used the public key to encrypt M1 and M3, and sent them to the target client. The CB, used with the Modbus protocol in user connections and transfer, integrates function codes with messages. The CB field structure is shown in Tab. 3.

But this method has some drawbacks. The first is that once the host is controlled, the PLC will unconditionally receive commands and programs from the host, and the encryption will be invalid. The second is that the encryption and decryption process consumes a lot of time and resources, leading an increase in the time delay of the industrial control system. When the time delay becomes too large, it is not suitable for industrial production.

Table 3: CB field structure

Source address	Bytes selector	Key sequence	Dynamic storage	Padding	Non-critical (Critical)	Select method (Acknowledge)	Destination address
2 bytes	2 bytes	2 bytes	(16-30) bytes	2(1) bytes	1(1) bytes	1(1) bytes	2 bytes

4 Summary

PLC equipment, the core of industrial control systems, of which the security is related to the security of the entire industrial control system. The design of PLC equipment and protocols is relatively primitive as the requirement that connect the industrial control networks to internet is not considered, resulting in a sharp increase in the security threat to industrial control networks once it is connected to the Internet. This paper starts from the illustration of PLC device itself, and introduces the code security, firmware security, network attack, modbus protocol security and some protection research of PLC. The security of industrial control systems requires a joint effort of various research institutes.

Funding Statement: This work is funded by the National Key Research and Development Plan (Grant No. 2018YFB0803504), the National Natural Science Foundation of China (Nos. 61702223, 61702220, 61871140, U1636215), the Opening Project of Shanghai Trusted Industrial Control Platform.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] Z. H. Tian, S. Su, W. Shi and X. Yu, "A data-driven method for future Internet route decision modeling," *Future Generation Computer Systems*, vol. 95, 212–220, 2019.

[2] Q. F. Tan, Y. G, J. Q. Shi, X. B. Wang, B. X. Fang, and Z. H. Tian, "Toward a comprehensive insight to the eclipse attacks of tor hidden services," *IEEE Internet of Things Journal*, vol. 6, no. 2, pp. 1584-1593, 2019.

[3] S. E. McLaughlin, "On dynamic malware payloads aimed at programmable logic controllers," in *Usenix Workshop on Hot Topics in Security*, 2011.

[4] J. Klick, S. Lau, D. Marzin, J. O. Malchow and V. Roth, "Internet-facing PLCs-a new back orifice," *Blackhat*, USA, pp. 22–26, 2015.

[5] R. Spenneberg, M. Brüggemann and H. Schwartke, "PLC-blaster: a worm living solely in the PLC," *Blackhat*, Asia, Marina Bay Sands, Singapore, pp. 1–16, 2016.

[6] E. Hrynkiewicz and M. Chmiel, "Programmable logic controller-basic structure and idea of programming," *Electrical Review*, vol. 88, pp. 98–101, 2012.

[7] Y. L. Yuan, "Realization of PLC control system based on configuration software," *Process Automation Instrumentation*, vol. 5, pp. 57–58, 2006.

[8] Z. H. Tian, M. H. Li, M. K. Qiu, Y. B. Sun and S. Su, "Block-DEF: a secure digital evidence framework using blockchain," *Information Sciences*, vol. 491, pp. 151–165, 2019.

[9] S. E. Valentine Jr, "PLC code vulnerabilities through SCADA systems," Ph.D. dissertation, University of South Carolina, 2013.

[10] G. Canet, S. Couffin, J. J. Lesage, A. Petit and P. Schnoebelen, "Towards the automatic verification of PLC programs written in instruction list," in *SMC Conference IEEE International Conference on System*, vol. 4, pp. 2449–2454, 2002.

[11] O. Pavlovic, R. Pinger and M. Kollma*nn,* "Automated formal verification of PLC programs written in IL," *Conference on Automated Deduction*, pp. 152–163, 2007.

[12] D. Darvas, B. Fernández and E. Blanco, "Transforming PLC programs into formal models for verification purposes," *Internal Note*, *CERN*. CERN-ACCNOTE-2013-0040, 2013.

[13] S. E. McLaughlin, S. A. Zonouz, D. J. Pohly and P. D. McDaniel, "A trusted safety verifier for process controller code," *NDSS*, vol. 14, 2014.

[14] S. J. Guo, M. Wu and C. Wang, "Symbolic execution of programmable logic controller code," in *Proc. 2017 11th Joint Meeting on Foundations of Software Engineering*, pp. 326–336. 2017.

[15] C. Schuett, J. Butts and S. Dunlap, "An evaluation of modification attacks on programmable logic controllers," *International Journal of Critical Infrastructure Protection*, vol. 7, pp. 61–68, 2014.

[16] Z. Basnight, J. Butts, J. Lopez Jr and T. Dube, "Firmware modification attacks on programmable logic controllers," *International Journal of Critical Infrastructure Protection*, vol. 6, pp. 76–84, 2013.

[17] Q. S. Qiao, S. Y. Xing, W. L. Shang, J. M. Zhao and X. F. Zhao, "Design and implementation of the trusted PLC," *Process Automation Instrumentation*, vol. 12, pp. 21, 2016.

[18] F. Adelstein, M. Stillerman and D. Kozen, "Malicious code detection for open firmware," in *18th Annual Computer Security Applications Conference*, pp. 403–412, 2002.

[19] Z. H. Tian, X. S. Gao, S. Su, J. Qiu, X. J. Du and M. Guizani, "Evaluating reputation management schemes of internet of vehicles based on evolutionary game theory," *IEEE Transactions on Vehicular Technology*, vol. 68, pp. 5971–5980, 2019.

[20] Z. H. Tian, X. S. Gao, S. Su and J. Qiu, "Vcash: a novel reputation framework for identifying denial of traffic service in internet of connected vehicle," *IEEE Internet of Things Journal*, vol. 7, pp. 3901–3909, 2019.

[21] W. Gao, T. Morris, B. Reaves and D. Richey, "On SCADA control system command and response injection and intrusion detection," *ECrime Researchers Summit (ECrime)*, pp. 1–9, 2010.

[22] T. H. Morris and W. Gao, "Industrial control system cyber attacks," in *1st International Symposium for ICS & SCADA Cyber Security Research*, pp. 22–29, 2013.

[23] A. Ghaleb, S. Zhioua and A. Almulhem, "On PLC network security," *International Journal of Critical Infrastructure Protection*, vol. 22, pp. 62–69, 2018.

[24] A. Robles-Durazno, N. Moradpoor, J. McWhinnie, G. Russel and I. Maneru-Marin, "PLC memory attack detection and response in a clean water supply system," *International Journal of Critical Infrastructure Protection*, vol. 26, pp. 100300, 2019.

[25] S. E. McLaughlin, "On dynamic malware payloads aimed at programmable logic controllers," in *Usenix Workshop on Hot Topics in Security*, 2011.

[26] S. McLaughlin and P. McDaniel, "SABOT: specification-based payload generation for programmable logic controllers," in *Proc. 2012 ACM Conference on Computer and Communications Security*, pp. 439–449, 2012.

[27] X. B. Chen, K. Chen, Z. Xu and L. M. Wang, "Security supervisory scheme for industrial control networks," *Netinfo Security*, vol. 7, 2016.

[28] Z. H. Tian, W. Shi, Y. H. Wang, C. S. Zhu and X. J. Du, "Real time lateral movement detection based on evidence reasoning network for edge computing environment," *IEEE Transactions on Industrial Informatic*, vol. 15, pp. 4285–4294, 2019.

[29] I. N. Fovino, A. Carcano, M. Masera and A. Trombetta, "Design and implementation of a secure modbus protocol," *Critical Infrastructure Protection III*, Springer Berlin Heidelberg, vol. 311, pp. 83–96, 2009.

[30] R. Nardone, R. J. Rodriguez and S. Marrone, "Formal security assessment of modbus protocol," in *11th International Conference for Internet Technology and Secured Transactions*, pp. 142–147, 2016.

[31] A. Shahzad, M. Lee, Y. Lee, S. Kim, N. X. Xiong, J. Choi and Y. Cho, "Real time MODBUS transmissions and cryptography security designs and enhancements of protocol sensitive information," *Symmetry*, vol. 7, pp. 1176–1210, 2015.