Contents lists available at ScienceDirect

Forensic Science International: Digital Investigation

journal homepage: www.elsevier.com/locate/fsidi

DFRWS 2021 USA - Proceedings of the Twenty First Annual DFRWS USA

JTAG-based PLC memory acquisition framework for industrial control systems

Muhammad Haris Rais [a, *], Rima Asmar Awad [b], Juan Lopez Jr. [b], Irfan Ahmed [a]

[a] Virginia Commonwealth University, Richmond, VA, 23284, USA
[b] Oak Ridge National Laboratory, Oak Ridge, TN, 37830, USA

A R T I C L E I N F O

Article history:

Keywords:
SCADA
PLC
ICS
Embedded devices
IoT forensics
Memory forensics
JTAG
Critical infrastructure protection

A B S T R A C T

In industrial control systems (ICS), programmable logic controllers (PLC) are the embedded devices that directly control and monitor critical industrial infrastructure processes such as nuclear plants and power grid stations. Cyberattacks often target PLCs to sabotage a physical process. A memory forensic analysis of a suspect PLC can answer questions about an attack, including compromised firmware and manipulation of PLC control logic code and I/O devices. Given physical access to a PLC, collecting forensic information from the PLC memory at the hardware-level is risky and challenging. It may cause the PLC to crash or hang since PLCs have proprietary, legacy hardware with heterogeneous architecture. This paper addresses this research problem and proposes a novel JTAG (Joint Test Action Group)-based framework, Kyros, for reliable PLC memory acquisition. Kyros systematically creates a JTAG profile of a PLC through hardware assessment, JTAG pins identification, memory map creation, and optimizing acquisition parameters. It also facilitates the community of interest (such as ICS owners, operators, and vendors) to develop the JTAG profiles of PLCs. Further, we present a case study of Kyros implementation over Allen-Bradley 1756-A10/B to help understand the framework's application on a real-world PLC used in industry settings. The sample PLC memory dumps are shared with the research community to facilitate further research.

1. Introduction

Industrial control systems (ICS) are used to automate a broad range of physical processes in critical infrastructure such as electrical grid-stations, steel mills, water supply and treatment systems, oil and gas facilities, and nuclear power plants (Stouffer et al., 2011; Ahmed et al., 2016). They have an essential component, programmable logic controllers (PLC), which are embedded devices directly connected to a physical process through input/output components (such as sensors and actuators), and execute a control-logic program to control the process (Ahmed et al., 2012).

PLCs are a common target of cyberattacks to sabotage a physical process. The attacks over PLCs can have severe adverse effects, such as shutting down of an electrical power grid, draining out of a city's water supply arteries, or sabotaging a nuclear reactor facility (Qasim et al., 2021; Yoo et al., 2019a, 2019b; Kush et al., 2011; Kalle et al., 2019; Bhatia et al., 2018; Senthivel et al., 2018). As PLCs have volatile and non-volatile memory components, memory forensics may help understand the attacks and ultimately contribute to the improvement in security of the critical infrastructure (Ahmed et al., 2017).

Current forensic efforts mostly focus on acquiring memory contents remotely using PLC debugging tools (Wu and Nurse, 2015) and ICS protocols such as GE-SRTP protocol (Denton et al., 2017), Modicon M221 protocol (Yoo et al., 2019a; Control logic forensics f, 2020; Qasim et al., 2019; Ayub et al., 2021) and Allen-Bradley's PCCC protocol (Senthivel et al., 2017). Given physical access to a PLC, these approaches are less effective since they cannot acquire the entire PLC memory and are limited to the memory contents within a PLC's protocol address space. JTAG (Joint Test Access Group) ports on a circuit board have been explored in the past to acquire PLC memory at the hardware-level. However, in the literature, JTAG is utilized mostly for modifying PLC firmware to demonstrate attacks such as Harvey (Garcia et al., 2017), and device exploitation (Firmware, 2013).

There is no forensics framework that provides guidelines for reliable memory acquisition of a PLC using JTAG. This work is an

* Corresponding author.
E-mail address: raismh@vcu.edu (M.H. Rais).

https://doi.org/10.1016/j.fsidi.2021.301196

effort to fill this void. Collecting forensic information from a PLC's memory is a risky and challenging task and can cause a PLC to crash or hang. PLCs are developed to live much longer than a traditional IT system without updates or upgrades, and may contain legacy hardware. PLC vendors use their proprietary hardware architectures, firmware, and programming applications. The vendors typically do not share the implementation details. In some instances, even the specifications of the integrated circuits (ICs) used in the PLC circuit boards are not available. These factors make the memory forensics of PLCs a challenging task.

This paper presents a novel JTAG (Joint Test Action Group)-based forensics framework, Kyros for PLC memory acquisition. Kyros supports a diverse set of PLC architectures by systematically creating their JTAG profiles containing essential information for JTAG setup and reliable memory acquisition. It also facilitates the community of interest (mainly, ICS owners, operators, vendors, and consultants) to contribute to developing the JTAG profiles of PLCs.

Specifically, Kyros consists of two phases: 1) Profile creation of a suspect PLC and then 2) memory acquisition. Profile creation starts with the circuit board's inspection to identify the JTAG pins and their working status, followed by an extensive exercise of memory-map creation to exclude unacquirable and redundant address-space regions. For the heterogeneous memory elements typically used in PLCs, Kyros discovers a set of optimized acquisition parameters for each memory-map entry to finalize the acquisition profile.

With the help of the acquired profile, Kyros recovers the memory contents of a suspect PLC. The paper also presents a case study of Kyros using Allen-Bradley's 1756-A10/B hosting 1756-L61 controller to understand the framework's application on a real-world PLC used in industry settings. The process is documented in detail, highlighting the caveats, challenges, and workarounds for the reproducibility of the work.

The contributions of this work are as follows:

1. We propose a forensics framework, Kyros for reliable acquisition of PLCs' memory through JTAG interface.
2. Using Kyros, we present a case study over Allen-Bradley 1756-L61 that not only outlines the process and the challenges, but also share the final "acquisition profile" that can be readily used by the community.

The rest of the paper is organized as follows. Section 2 covers the background and related work. Section 3 describes the proposed framework. Sections 4 explains the implementation case study for the acquisition framework over Allen-Bradley 1756-L61, followed by the limitations, the future work, and the conclusion.

2. Background and related work

2.1. Background

Critical infrastructure is defined by the Department of Homeland (DHS) as 16 sectors that compose the assets, systems, and networks (physical or virtual) vital to the United States (Critical infrastructure sectors; Rais et al., 2021). Industrial Control Systems (ICS) and supervisory control and data acquisition (SCADA) systems are the underpinning technologies ensuring proper operation of critical infrastructures. PLCs are a critical component of ICS.

PLC Architecture. Fig. 1 shows a simplified view of the PLC architecture (Ahmed et al., 2017). The IO modules communicate with the physical process via sensors and actuators. Once the PLC starts, the CPU loads & executes the firmware available in non-volatile memory. The firmware fetches the control logic downloaded from the control PC and runs it in an infinite loop. Depending upon the latest sensor values and the control logic, the PLC updates the output values sent to the actuator during each cycle.

JTAG Overview. Joint Test Action Group (JTAG), commonly referred to as boundary-scan and defined by the Institute of Electrical and Electronics Engineers (IEEE) 1149.1, was developed for validating proper assembly of components on PCB boards in the 1990s (Vishwakarma and Lee, 2018). In the ICS industry, most PLC manufacturers use JTAG at the developing and testing stage to access the internal subsystem of ICS or for breakdown examination, updating and storing firmware on the chip, and debugging. Because there are many JTAG debuggers available in the market that range in cost from ten to a few hundred dollars, JTAG can provide a "backdoor" that can be exploited by attackers and allow access to the internals of a chip and firmware. For that reason, vendors sometimes entirely remove the JTAG pins before putting the control system in production, or hide JTAG interface points from users.

JTAG consists of a Test Access Port (TAP) controller, which is a 16-state machine that gives easy access to test functions built into a component and can be programmed through five JTAG pins:

- Test Clock (TCK): Clock signal provides timing data to the boundary scan system. Input data is read at the rising edge from TMS and TDI pins, while data is output to TDO pin at the falling edge.
- Test Mode Select (TMS): Signal received on TMS pin controls the test logic's operational mode.
- Test Data Input (TDI): Connection onto which the test instructions data stream is passed.
- Test Data Output (TDO): Provides data from the boundary scan registers, i.e., test data shifts out on this pin.
- Test Reset (TRST) (optional): Optional active low test reset pin on the JTAG interface that permits asynchronous TAP controller initialization without affecting other device or system logic.

Although the JTAG port is mostly used for hardware fault debugging, forensic investigators also use it to access the content of Random Access Memory (RAM) following cyber-attacks or system faults.

2.2. Related work

This section covers a body of literature for JTAG-based memory acquisition on embedded devices for offensive and defensive

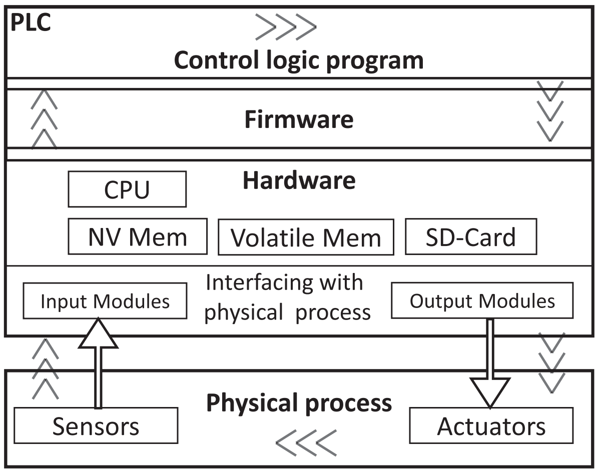


Fig. 1. PLC architecture.

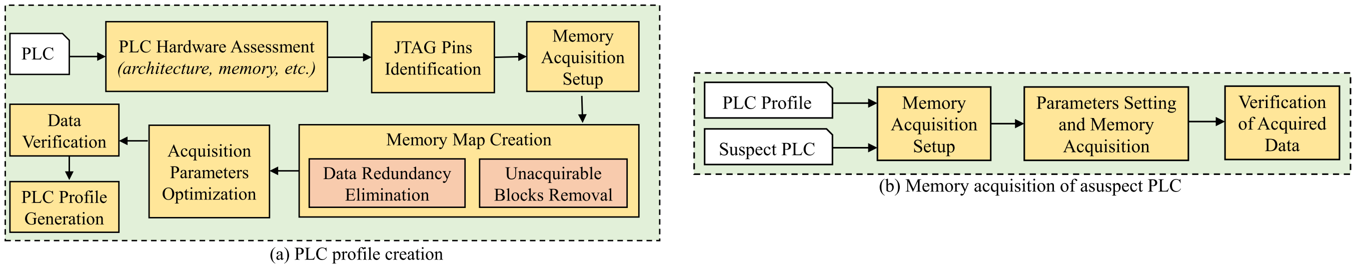

PLC
PLC Hardware Assessment (architecture, memory, etc.)
JTAG Pins Identification
Memory Acquisition Setup
Memory Map Creation
Data Redundancy Elimination
Unacquirable Blocks Removal
Acquisition Parameters Optimization
Data Verification
PLC Profile Generation
(a) PLC profile creation
PLC Profile
Suspect PLC
Memory Acquisition Setup
Parameters Setting and Memory Acquisition
Verification of Acquired Data
(b) Memory acquisition of asuspect PLC

Fig. 2. JTAG-based memory acquisition framework for programmable logic controllers.

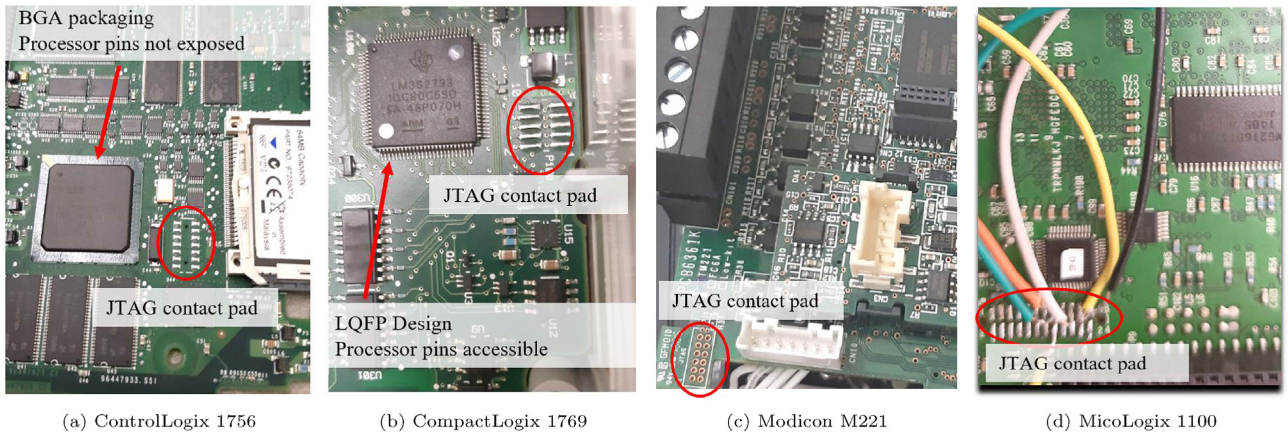


Fig. 3. JTAG contact pads identified in various PLCs.

purposes. Most literature focuses either on PLC firmware attacks or targets smartphones. The work closest to ours are Harvey (Garcia et al., 2017) and Majeric et al. (Fabien et al., 2016). They use the JTAG interface to acquire PLC's firmware for launching a firmware modification attack. However, none of them discusses acquiring the complete addressable memory of a PLC and proposing any forensics framework. On the other hand, our proposed framework, `Kyros`, focuses on discovering and acquiring the entire acquirable address space in a PLC, explained through a detailed case study.

JTAG-based Cyber Defense. Breeuwsma (2006) describes a broad set of steps for accessing the JTAG port and acquiring raw memory dumps of embedded systems. The author also discusses some practical challenges, but does not present any framework or a detailed case study.

Konstantinou et al. (2018) implement PHYLAX, a JTAG based online monitoring and detection mechanism that connects to a JTAG enabled embedded device. PHYLAX is evaluated on a power grid HITL testbed, where it attaches to the JTAG interface of the embedded circuit board controller. PHYLAX consists of three parts: 1) A reader module to fetch device, process, and application-specific parameters such as memory size, memory, or register access rate; 2) an adaptive module that compares the immutable section of the firmware to a baseline model to detect any modification and check for suspicious executable instruction; and 3) a detector module to check violations detected by the adaptive module to generate alerts.

Guri et al. (2015) propose a framework, JoKER (JTAG observe Kernel), for detecting stealthy rootkits in the Android OS kernel by using the JTAG interface. The mobile device is halted using JTAG commands, providing read, write, and other debugging functionality to the RAM and Flash memory. Raw memory is extracted from selected regions for analysis by the scripts written by the authors. The framework obtains a trusted snapshot of the device's memory and effectively detects kernel rootkits.

Willsassen (Willassen, 2005) uses JTAG test pins to retrieve a device's internal memory. This method is verified on the Nokia 5110 model.

JTAG-based Cyber Attacks. Schuett et al. (Schuett et al., 2014) perform firmware modification attacks on PLC using JTAG interface. The authors first extract the firmware image and perform static and dynamic analysis of the code. JTAG is used to perform dynamic analysis to identify execution paths and generate memory dumps. The firmware is repackaged with a malicious attack that triggers a denial of service attack with a combination of CIP protocol commands by writing a sentinel value to an unused flash memory area.

Basnight et al. (Firmware, 2013) use JTAG and PLC firmware reverse-engineering to demonstrate the possibilities of firmware modification attacks. They extract the firmware from an update package. The firmware is then reverse engineered, and the validation algorithm is analyzed to uncover vulnerabilities that allow firmware modification.

Harvey (Garcia et al., 2017) is a PLC rootkit at the firmware level that can evade operators viewing the HMI. Using JTAG and firmware disassembly, the authors managed to locate the subroutines needed for their rootkit. Harvey manipulates input and output values of the PLC without changing the PLC's control logic and eventually causing damage to the physical system.

Majeric et al. (Fabien et al., 2016) perform a hardware/software combined attack to get root privilege in Android. The authors successfully demonstrate the privilege escalation attack by 1) finding the system call address that manages the privilege allocation to processes and 2) modifying the system call and injecting the

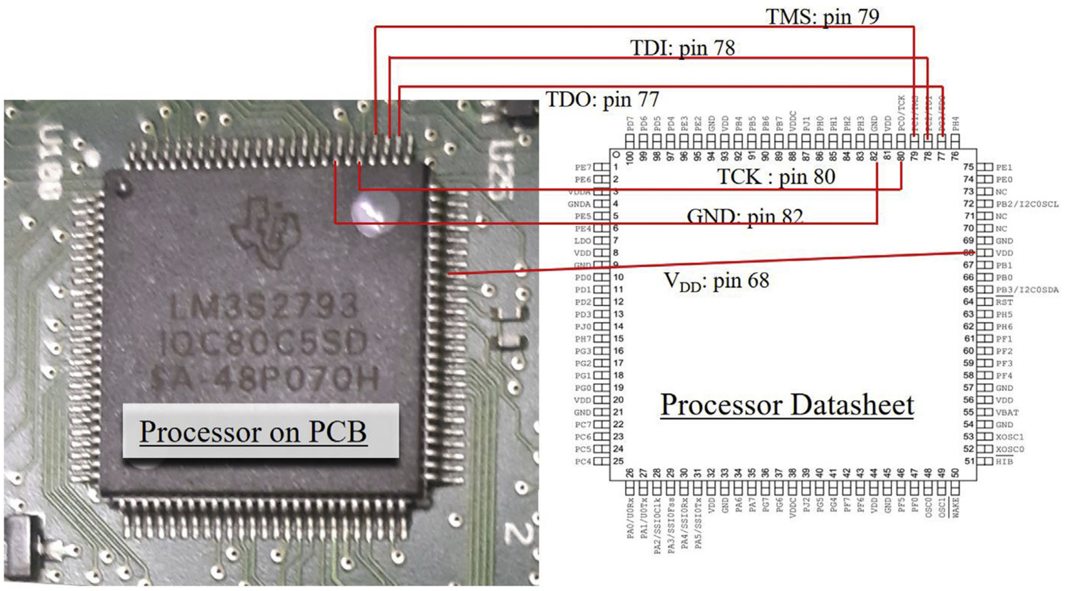


Fig. 4. JTAG pins accessible on the processor with LFQP packaging design for PLC CompactLogix 1769

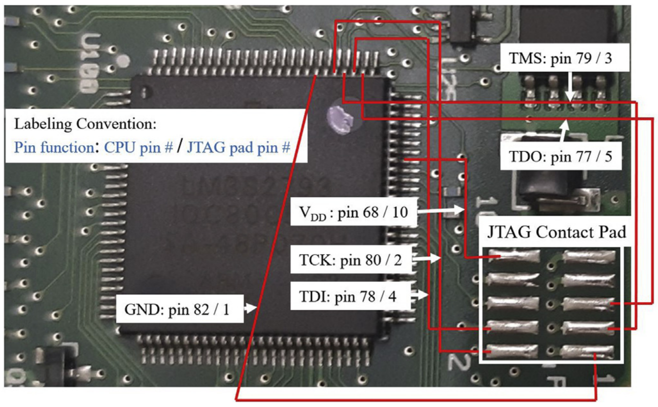


Fig. 5. Identifying JTAG pins on the contact pads through connectivity test.

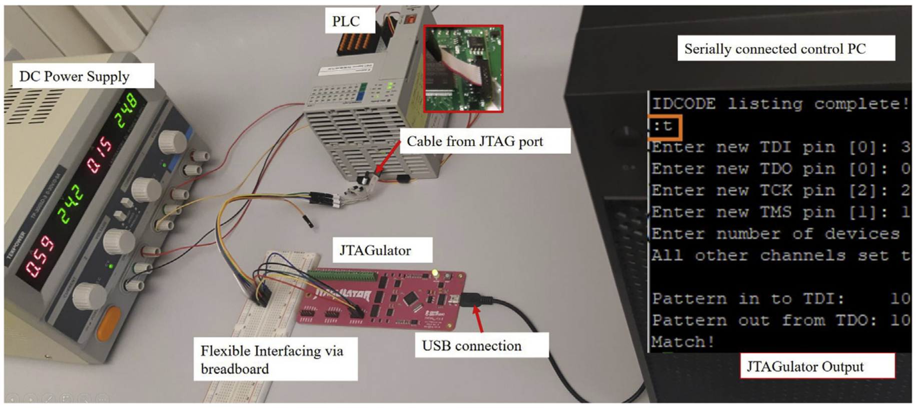
fault at an identified location using the JTAG port to request root permission from the previously identified system call.

3. JTAG-based memory acquisition framework

Fig. 2 presents the proposed framework, `Kyros`, consisting of two distinct phases: 1) Creating a PLC profile for JTAG acquisition and 2) acquiring a suspect PLC's memory if the PLC's profile is available.

3.1. JTAG acquisition profile creation

PLCs are proprietary, legacy hardware with heterogeneous architecture, which may cause a PLC to crash or hang during memory acquisition using JTAG. Thus, we propose to create a JTAG profile of a target PLC containing essential information for a reliable memory acquisition, such as the location of JTAG ports on a PLC's circuit board, identification of unacquirable memory blocks, and JTAG parameter settings. The profile can be provided by the PLC vendors, obtained through PLC datasheets and other vendor documents, or created by obtaining and analyzing a sample PLC. `Kyros` focuses on the latter, identified in Fig. 2a to help the community of interest (mainly, ICS owners, operators, vendors, and consultants) to contribute to developing the JTAG profiles of PLCs.



```
IDCODE listing complete!
:t
Enter new TDI pin [0]: 3
Enter new TDO pin [0]: 0
Enter new TCK pin [2]: 2
Enter new TMS pin [1]: 1
Enter number of devices
All other channels set t

Pattern in to TDI:    10
Pattern out from TDO: 10
Match!
```

Fig. 6. Confirming JTAG pinout using JTAGulator device.

Kyros provides a procedure for setting up a PLC for memory acquisition through JTAG, generates the memory map of the PLC, and discovers the JTAG optimized parameters to create a JTAG profile. Note that the process involves hardware interference and trials that may result in a PLC crash. Thus, it should be carried out on a test PLC of the same model as a suspect PLC.

PLC Hardware Assessment. The first step in profile creation is PLC hardware assessment to attain the required information about processor and memory elements in a PLC. Relevant information about the processor includes the architecture, internal volatile and non-volatile memory, and the processor's memory map. The processor's internal memory (if a microcontroller is used) is not enough, and the PLC circuit board usually hosts volatile and non-volatile memory elements. As JTAG-based acquisition can be slow, the memory elements set an expectation of total memory size. A removable SD-card may also be available for storage containing useful information related to the PLC programs and firmware running. However, SD-card is not part of the internal memory and can be acquired through an SD-card reader.

The hardware information search should start with the PLC vendor's hardware manuals. Generally, the vendors do not provide architectural information or the memory elements used in the PLC. However, the data like microcontroller ID, code memory size, IO data memory size, backup memory options, etc., do provide certain clues that help in drawing the complete picture in due course. This incomplete information can be augmented through a physical inspection of the circuit boards by disassembling the PLC. In the case of a modular PLC, the focus should be on the controller module.

The circuit boards on a PLC can be categorized into three groups as per their functionality: mainboard, communication board, and power board. A quick search of the ICs present on the main and communication boards helps identify the processor and the memory chips. The data sheets can provide the specifications of commercially available ICs.

JTAG Pins Identification. This step identifies the physical location of the JTAG port on the circuit board, its operational status, and the JTAG pinout. If the processor packaging style allows for access to the pins (e.g., quad flat package design), the JTAG pins can be physically accessed on the processor. Surface mount (SMD) test clips and micro-grabber clips can be used for this purpose. However, they have practical limitations, including *fragile connections* that may pop out or may even short-circuit the processor pins, and *hindrance* in the PLC's reassembly, without which the acquisition process cannot continue. In some cases, the processor IC packaging does not allow physical access to the pins (such as ball grid array design).

JTAG Interface Pinout on Circuit-board. Most vendors remove the header from the JTAG interface on a circuit-board, and only the contact pad is visible. We observe the pads with 12–24 pins organized in 2 rows in different PLCs. Fig. 3 shows the contact pads identified in 4 different PLCs. If the processor pins are accessible and datasheets available, the JTAG pins can be identified on the processor (shown in Fig. 4). Through connectivity tests between the processor-designated pins and the candidate contact-pad pins, the JTAG-pinout can be confirmed. Fig. 5 shows an example of the connectivity tests performed on PLC CompactLogix 1769.

For the processors with non-accessible pins, the connectivity test is not possible. JTAG pins finder device, such as JTAGulator (GrandJtagulator, 2013) can be used to find the JTAG pinout. JTAGulator works by assigning different roles to the pins through permutation to arrive at the correct pinout. Fig. 6 shows a typical JTAGulator setup. The cable routed out of the PLC is connected to random connecting pins on the JTAGulator through flexible interfacing via a breadboard. A clipped output of JTAGulator CLI shows the pins used for various JTAG signals. Some vendors permanently disable the JTAG port after the circuit testing (An12419, 2019). Thus, the successful pinout identification by JTAGulator also confirms the active status of the JTAG interface.

To use JTAGulator, the PLC has to be reassembled and powered up. It is advisable to install a header by soldering on the contact pad and route a cable out of the PLC before the reassembly. This one-time hardware interference is a limitation of JTAG-based memory acquisition.

Memory Acquisition Setup. We can use any readily available JTAG debugger devices supporting the processor's architecture to utilize the JTAG port. Fig. 7 illustrates a typical JTAG setup. Debuggers can connect to a computer running a JTAG debugger software suite through USB or over the network. If the processor datasheets are not available, testing through multiple debuggers may be required. Unfortunately, there are no standard interfaces for the debugging devices and the PLC's JTAG port. Thus, Kyros recommends a flexible interfacing platform (using breadboard or pin converter circuits) to switch between debuggers and PLCs for testing conveniently.

Device Memory Map Creation. The memory acquisition through JTAG is a slow and risky process (Afonin and Katalov, 2016). This step obtains a memory-map of address space ranges for reliable and efficient acquisition by eliminating redundant data and unacquirable address spaces.

Data Redundancy Elimination. A processor's address space is often sparsely populated in PLCs. By identifying the unused bits, redundant data acquisition can be (optionally) avoided reducing the acquisition time. For instance, in our experience, a 32-bit processor of a PLC may be using less than 200 million unique addresses from the possible 4 Gig space. Multiple address combinations may point to a single physical location. Due to slow acquisition speed, it helps identify this type of duplication. However, care must be taken to distinguish it with the same data at different unique memory addresses. One way of discrimination is to write on the memory address via JTAG debugger on one location and see the impact on the other location. We highlight both types of examples in the case study.

Table 1
List of equipment and software used in the case-study.

Ser	Equipment/Software	Description
1	PLC	Allen-Bradley 1756 A10 with modules L61 ControlLogix 5561, HSC/A, IF8, 2x OB32/A, DNB B, ENBT A
2	PLC Control PC	core i7-8700, 16 GB RAM, Windows 10 for PLC control software PC connected to PLC over Ethernet
3	Project PC	core i7-7800. 16 GB RAM, Windows 10 connected to JTAG debugger over USB
4	JTAG Interface Finder	JTAGulator by Parallax Inc
5	JTAG Debugger	Segger J-Link ARM V8.00
6	Power Supply	Tekpower TP-3005D-3 Digital Variable DC Power Supply (any 24 V DC 3 A power supply will work)
7	SMT header for PCB	2 × 7 pins unshrouded header, 2.54 mm pitch
8	IDC cable for header	Digilent, JTAG 2 × 7 pin cable
9	Miscellaneous	Breadboard, connecting cables, soldering iron, soldering tips (for SMT), Multimeter, Magnifying glass
10	PLC control software	RSLogix 5000 version 20.05.00 (CPR 9 SR 10) FactoryTalk Activation Manager v 4.05.01
11	JTAG Debugger software	SEGGER -J-Link V6.80 d suite
12	Python library	PyLink Python library by Square Inc. for Segger Debugger

Unacquirable Blocks Removal. The unacquirable addresses can hang the PLC. They can be identified using mapping data from the vendor. Otherwise, identifying unacquirable memory will involve "crash and learn" exercise that should be conducted on a test-PLC.

Acquisition Parameters Optimization. As JTAG interaction is a low-level process, and the PLC firmware is not designed to cater for JTAG based memory acquisition, the use of arbitrary JTAG parameter settings can timeout certain watchdog timers crashing the PLC, the debugger or both. This step is performed to avoid such occurrences. Parameters optimization is also useful for speeding up the memory acquisition process. The memory ICs used on the circuit board have different response-times, refresh rates, usage, and other specifications, warranting an optimization effort on a per chip or address block basis. The optimization parameters include the speed of acquisition, block size, buffer size, and wait-time between consecutive read operations. These parameters should be synchronized with the hardware acquisition tool (such as the JTAG debugger module and the driver software).

IC datasheets indicate the maximum time available between consecutive refresh operations. Using communication buffers capacity and the acquisition speed, we can obtain the maximum memory block to read in one go. However, in the absence of the documentation, it is not simple (and not directly sought-after information from a forensic perspective) to map each IC with the address region. An alternate approach focuses on the populated memory regions and tries a range of combinations of these parameters for the optimal result. However, this scheme may crash the PLC and hang the debugger resulting in restarting the PLC and debugger. Since the debugger typically runs on a desktop computer, it can restart automatically via a shell script. However, power-cycling the PLC is a challenging task and requires restarting the power source unit. As the memory chips and their roles are limited in number, manual PLC resets may also work.

Profile Generation and Verification. By this stage, we achieve a set of acquirable, redundant, unacquirable, and unused memory addresses. Aligning these address blocks with the corresponding optimization parameters results in a "JTAG Acquisition Profile" for the PLC. The first two fields of the profile constitute the address block's starting and ending address. The remaining fields pertain to optimization parameters, including the block-size, speed, and wait-time. "Block-size" represents the data-size acquired by issuing a single read-command through the debugger. "Speed" is the JTAG clock rate, and the "Wait-time" is the idle time between two consecutive read-commands issued.

3.2. PLC memory acquisition

Fig. 2b outlines `Kyros` for utilizing a JTAG acquisition profile of a suspect PLC for reliable memory acquisition.

Acquisition Setup for a Suspect PLC. They can replicate the acquisition setup of a test-PLC for a suspect PLC, involving header installation and cable routing out of PLC.

Parameters Setting and Memory Acquisition. Similarly, investigators can use JTAG parameter settings in the profile for the acquisition. Although the acquisition can be performed manually, it can be automated using the debugger APIs. The simple process involves parsing the profile-set and invoking the memory-read calls. The attained memory can be saved in any format of choice. As the acquired address blocks will be disjoint, the most straightforward format could save the acquired memory in multiple files by embedding the metadata in the filename.

Verification of Acquired Data. The acquired memory should be verified by using the indicators of a valid memory dump. Some indicators include finding the firmware obtained from the vendor's website, ASCII symbols e.g., device catalog name, modules inventory, IP address, etc.

Table 2
Memory elements on Allen-Bradley ControlLogix 5561.

Ser	Type	Part No	Vendor	Qty	Total Size
1	NOR Flash	28F640J3D75	Intel	01	8 MB
2	Static RAM	CY7C1041CV33	Cypress	04	2 MB
3	SDRAM	48LC4M16A2	Micron	03	24 MB
4	NAND Flash	K9F1G08U0A	Samsung	01	128 MB
5	SD Card	1784-CF64	Allen- Bradley	01	64 MB

4. Case study: Allen-Bradley PLC 1756-A10 with ControlLogix 5561

Allen-Bradley's modular chassis 1756-A10 B is used as an implementation case study of `Kyros`. The chassis can hold up to ten modules, including controller, IO, communication, and counter modules. The controller installed in the chassis is ControlLogix 5561 (catalog number 1756-L61 Series B), running the latest firmware 20.019.

4.1. Case study goals

The case study achieves the following three goals: 1) Demonstrates a practical example of `Kyros`; 2) presents the memory-acquisition profile and a ready-to-use tool for acquiring the memory contents of Allen-Bradley ControlLogix 5561 controller; 3) discusses and shares a dataset of acquired memory contents for Allen-Bradley ControlLogix 5561 for a range of programs to facilitate research on PLC memory analysis.

4.2. Implementation setup

The PLC is connected over the Ethernet interface to a computer running engineering (programming) software. This channel is used to download projects to the test-PLC. Another computer is used to read the PLC memory contents using the JTAG debugger software suite. The computer is connected to the PLC via the JTAG interface. Table 1 shows the list of equipment used.

4.3. JTAG profile creation

As the acquisition profile for the PLC does not exist, we go through the profile creation phase using `Kyros`.

4.3.1. Hardware assessment

Architecture. Allen-Bradley 1756-A10 chassis used in our case-study is populated with the modules mentioned in Table 1. We disassembled the controller module (1756-L61). There are two processors installed on the main circuit board. Atmel AT56J05-UQ3T located adjacent to the socket connecting the controller module to the chassis, is used for communication among the modules. Centrally located Philips VY22575 is the main processor. No official documentation is available about the Philips processor. Upon inquiry from the vendor (then Philips, now NXP), the information was regretted being "customer specific" product. The part number shows that it is an ARM processor.

Memory Elements. Memory elements on the circuit board are listed in Table 2 that includes nine ICs and one SD-card. Four Static RAM ICs have a total capacity of 2 MB, matching the "Data and Logic" memory size mentioned in the controller's properties in the engineering software. SDRAM, NOR, and NAND Flash ICs are also located onboard. Fig. 8 marks the processors, memory ICs, and the JTAG contact-pad (confirmed in due course of study) on the main circuit board.

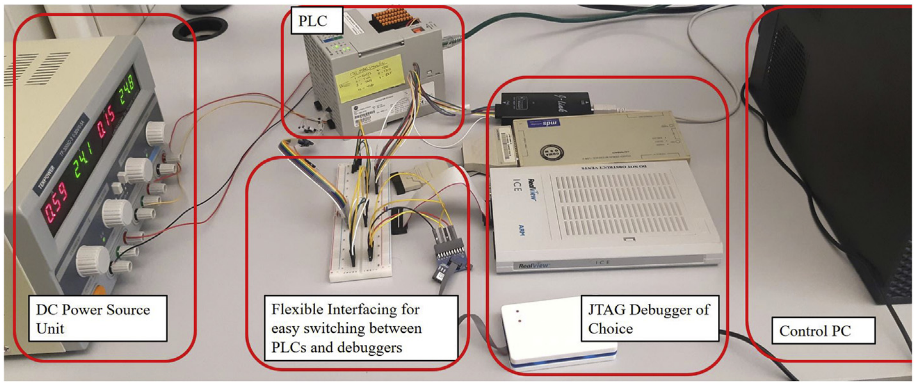

PLC
DC Power Source Unit
Flexible Interfacing for easy switching between PLCs and debuggers
JTAG Debugger of Choice
Control PC

Fig. 7. JTAG-based memory acquisition setup.

4.3.2. JTAG pins identification

Though the likely JTAG candidate is a 2 × 7 pins contact-pad available in vicinity of the processor, confirmation through connectivity test is not possible due to processor's BGA packaging design offering no access to the pins. Even if the pins were accessible, manual search without the data sheets would be a laborious and inefficient exercise.

JTAG Interface Pinout. To confirm about the operational status and the JTAG pinout, we utilized JTAGulator device. To use JTAGulator, the PLC has to be powered up, and thus to be reassembled.

Header Installation and Cable Routing. Before reassembly, we install a header on the candidate-contact pad. The pitch and inter-row spacing of the contact-pad shown in Fig. 9a are measured to be 2 mm and 5 mm, respectively. Headers with less inter-row spacing can also be used. We install a surface-mount breakaway header with 2 mm pitch and 2 mm inter-row spacing. Fig. 9b shows the header soldered on the contact-pad using the SMD soldering technique. A flat ribbon 2 × 7 cable extends the JTAG connectivity out of the PLC as visible in Fig. 9c. The soldering and cable installation tasks are verified through a connectivity-test between the contact-pad and the cable's remote-end. For routing the cable out, PLC's porous metallic wall is slightly punctured, as highlighted in Fig. 11.

After extending the pins out, the JTAGulator setup is created similar to Fig. 6 except for the PLC model. The JTAG pins are successfully identified as shown in Fig. 10.

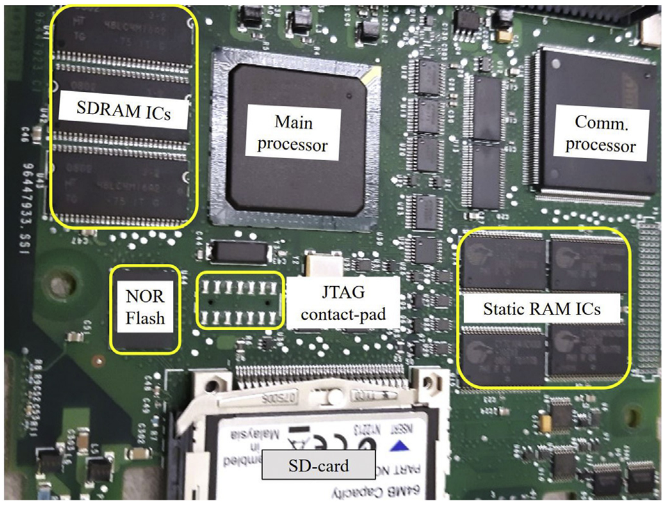


Fig. 8. Allen-Bradley 1756-L61 Main board.

4.3.3. Memory acquisition setup

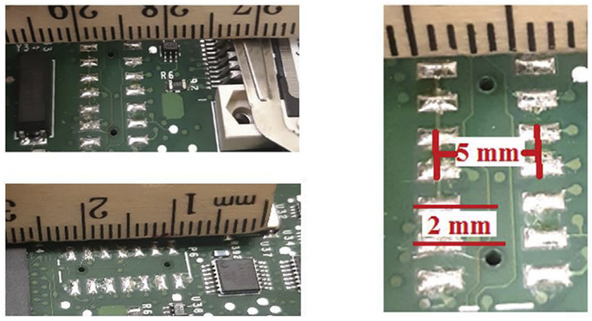
JTAG debugger devices are readily available for different processor architectures and series. We use Segger's JLink debugger device for this study. The debugger has prior JTAG related settings for a wide range of processors. The physical setup is presented in Fig. 11. The controller module of the PLC connects to the project PC via the JTAG interface, and the EtherNet/IP module is connected to the PLC Control-PC via Ethernet. Though the datasheet for our target-processor is not available, the search for the vendor's commercial products belonging to the same manufacturing era point towards the LPC series to be a likely candidate for the processor family. Using the LPC-2xxx series profile, the JLink debugger successfully connected to the target processor.

4.3.4. Device Memory Map Creation and optimization

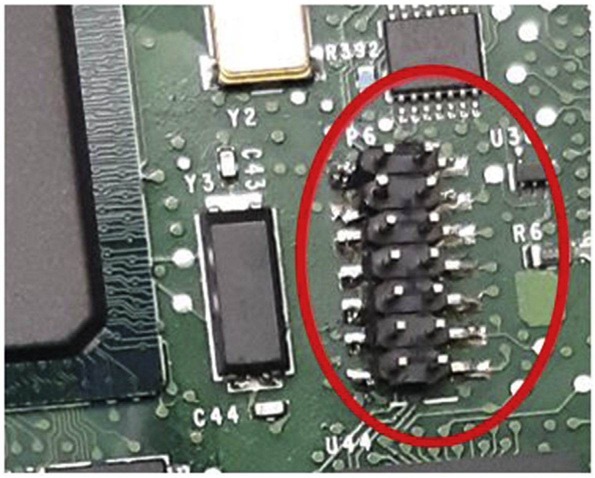
In the absence of a processor's datasheets, a memory-map has to be created from scratch. To avoid missing out on valid data, we resort to a complete address-space scan. The memory acquisition through the JTAG debugger is a slow process. While trying to retrieve information faster than a threshold value, undesirable incidents like the hanging of PLC, disconnection from engineering software, hanging of debugger, or incomplete data retrieval can occur. These issues happen more frequently if the PLC is in “Run” mode. We expected these events to address ranges allocated to peripherals. Therefore, we resort to a slow acquisition exercise with an offline PLC to acquire the complete address-range. After reducing the data size for a single read command to 8192 bytes and increasing the wait-time between two consecutive reads to 4 s, the data is acquired successfully after a few manual skips and restarts at non-acquirable addresses.

Redundant and unacquirable Address Blocks. The above exercise took over two weeks and revealed useful information, including the following: there is no data in the upper half of the address-space; some blocks are still unreadable; multiple copies of the same data blocks are observed at different addresses. We identify two possible reasons for multiple copies of the same data blocks: 1) the data blocks may exist twice at different memory locations, and 2) multiple addresses may point to the same memory location. An example of the first case is the firmware, which is loaded from non-volatile memory to the RAM on startup. From a forensics point of view, both copies are independent and should be acquired. In the second case, multiple addresses point to the same location as illustrated in Fig. 12 through an example.

When A_{31-27} address bits are set to “0000 0”, A_{26-24} are “Don't Care” bits, while A_{23-0} points to unique memory location. It implies that acquiring 256 MB for data for all address combinations of A_{26-0} provides eight copies of 16 MB data. Interestingly, the same data



(a) Header measurement - pitch and Inter-row distance


(b) Header installation

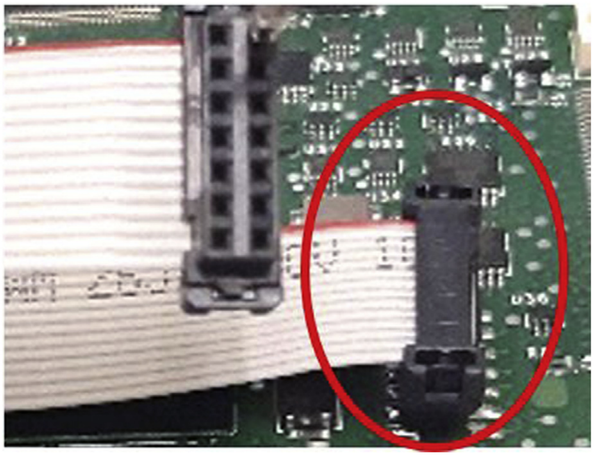

(c) IDC cable installed over header

Fig. 9. Header and cable installation over contact-pad.

reappears when A_{31-28} address bits are “0101” with four “‘Don't care” bits from A_{27-24}, while A_{23-0} pointing to the actual memory. The “Don't Care” bits are address pins neither used to indicate a specific device nor required for the device offset. After conducting a manual analysis, we classified duplicate data cases. Address blocks under case-1 must be kept, while case-2 (duplicate instances) is optional and may be ignored to speed up the acquisition process.

Address ranges resulting in undesirable incidents mentioned in the preceding paragraph are excluded from the acquirable memory list. For example, data acquisition attempt from “0C080000” to “0C081F0 F” immediately crashes the PLC. This learning is done on the test-PLC to keep interference with the suspect PLC to a minimum. The memory acquisition attempt on addresses not pointing to any memory element does not return any data. We observe no data for this PLC after address “68000000”. After removing the redundant, unacquirable, and unused address-blocks, we formulate a list of the address-ranges to be acquired. The first two columns of Fig. 13 presents a sample of the finalized acquirable address-blocks list.

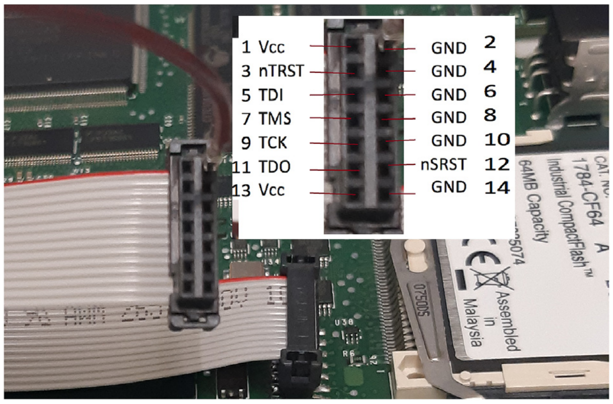


Fig. 10. JTAG pins on the contact pad.

4.3.5. Acquisition parameters optimization

Certain memory blocks allow slower data recovery than others, and any attempt to speed up the process crashes the PLC, presumably, due to different specifications and IC refresh-rates. We observe that more caution is required when the PLC is connected to the engineering software and is in “Run” mode. We carried out the exercise of finding optimal data block size and the wait-time between 2 consecutive read operations, suitable for all address-ranges. Fig. 13 shows a sample output in a list data structure of Python programming language hosting all acquirable address-blocks and suitable acquisition parameters. “0” represents that the default value of the parameter works well for that entry. Another optimization parameter, named as “marker-based acquisition”, is discussed under section 4.4.

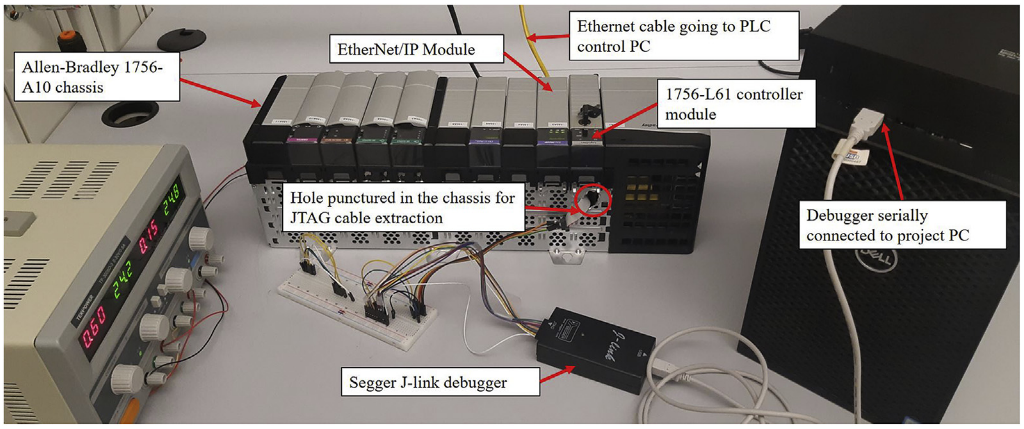

Allen-Bradley 1756-A10 chassis
EtherNet/IP Module
Ethernet cable going to PLC control PC
1756-L61 controller module
Hole punctured in the chassis for JTAG cable extraction
Debugger serially connected to project PC
Segger J-link debugger

Fig. 11. PLC experimental setup for the case study.

4.3.6. *Profile Generation and Verification*

The consolidated knowledge acquired in the previous steps constitutes the JTAG-Acquisition profile. The profile is verified before using on the suspect PLC.

Data Verification. The verification process confirms the correctness of (1) the acquired data, (2) the claimed redundant, unused, and unacquirable address blocks, and (3) the optimization parameters.

The largest file in the memory is the firmware (2.7 MB). We download the firmware from the vendor's website and match it with all firmware instances of the firmware found in the dump. The match is 100% for all the tests. Although these instances are not acquired from independent memory blocks, they serve as separate copies to verify the memory acquisition process. As a second verification test for the acquisition process's consistency, the strings found in the memory dump (such as controller name, project name, and filenames) are repeatedly acquired and matched and found to be correct.

To confirm the redundant address spaces' identification, one byte is modified in one address-range, and the change is visible in all the redundant blocks.

The optimization parameters are used to speed up the acquisition process. Less optimization will lead to more acquisition time, but over-optimization results in PLC or debugger crashes and disconnections. During verification, we focus on ensuring that the parameters are not over-optimized.

These tests are successfully repeated five times over five days, after restarting the PLC and the debugger and downloading different programs.

4.4. *Memory acquisition of the suspect PLC*

When the profile is created and verified, it is used for the memory-acquisition of a suspect PLC.

Acquisition Setup. Due to the practical limitation of unavailability of the same model's PLC, we put the test-PLC in a conveyor-belt test setup and re-used it as the suspect PLC.

Parameters Settings. After loading the profile, the memory is acquired and verified. The memory dump is organized in multiple files - one file for each address range.

Acquisition Modes. To facilitate the memory-acquisition, we write a python program that uses Pylink library (Pylink) to interface with Segger's JLink debugger. The program allows for three modes: complete memory acquisition, customized range acquisition, and signature-based acquisition. Signature-based or marker-based acquisition mode is developed after observing the case study's PLC memory structure. For instance, 2 MB memory block for "Data and Logic" is populated from the start and the end with a precise pattern indicating the start and end of each part. Marker-based acquisition (if suits the requirement) can quickly provide the in-use "Data and Logic" memory valid at the time of acquisition.

As one of the case-study goals is to generate the PLC memory data sets, we use all the modes to acquire the memory contents.

Verification of Acquired Data. A reasonable level of confidence in the acquisition setup is already attained after the detailed verification in the profile creation phase. For the suspect PLC's data verification, we search for the known strings (such as controller catalog-name, filenames, etc.) and verify them in the memory-

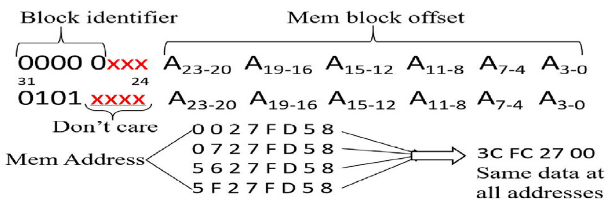
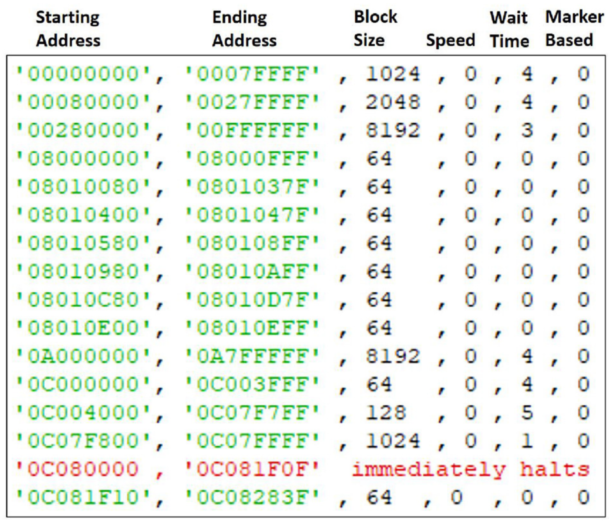


Fig. 12. Example of duplicate data due to Don't Care bits and multiple copies.



Starting Address	Ending Address	Block Size	Speed	Wait Time	Marker Based
'00000000',	'0007FFFF'	, 1024	, 0	, 4	, 0
'00080000',	'0027FFFF'	, 2048	, 0	, 4	, 0
'00280000',	'00FFFFFF'	, 8192	, 0	, 3	, 0
'08000000',	'08000FFF'	, 64	, 0	, 0	, 0
'08010080',	'0801037F'	, 64	, 0	, 0	, 0
'08010400',	'0801047F'	, 64	, 0	, 0	, 0
'08010580',	'080108FF'	, 64	, 0	, 0	, 0
'08010980',	'08010AFF'	, 64	, 0	, 0	, 0
'08010C80',	'08010D7F'	, 64	, 0	, 0	, 0
'08010E00',	'08010EFF'	, 64	, 0	, 0	, 0
'0A000000',	'0A7FFFFF'	, 8192	, 0	, 4	, 0
'0C000000',	'0C003FFF'	, 64	, 0	, 4	, 0
'0C004000',	'0C07F7FF'	, 128	, 0	, 5	, 0
'0C07F800',	'0C07FFFF'	, 1024	, 0	, 1	, 0
'0C080000 ,	'0C081F0F'	immediately halts			
'0C081F10',	'0C08283F'	, 64	, 0	, 0	, 0

Fig. 13. Sample of address ranges and the optimization parameters; default value for zero.

dump. To test the process's repeatability, we randomly picked 20 addresses holding string-data across acquirable memory-ranges, and repeatedly acquire those addresses five times. No anomaly is observed, confirming the correct acquisition.

5. Limitations and future work

Utilization of Kyros requires the establishment of a small one-time setup comprising of items mentioned in Table 1. A significant limitation of Kyros is to require a test-PLC of the same model as the suspect PLC for the creation of "JTAG Acquisition Profile". However, as the community of interest involves supporting Kyros, more profiles will be available, reducing the requirement for "Acquisition Phase". We intend to share the memory dumps and the profiles for more PLCs of different vendors in the future.

6. Conclusion

This paper presented Kyros, the first memory acquisition framework for PLC forensics using the JTAG interface. Dealing with the practical challenge of proprietary architecture and customized ICs with no access to the datasheets, Kyros can guide a forensic investigator through the required tasks from the hardware setup to the memory acquisition, highlighting the potential obstacles and workarounds. The resultant profile comprises the memory-map of the PLC, eliminating unused, redundant, and unacquirable address-spaces. The acquisition parameters optimization process in the framework compensates for the slow speed of JTAG-based acquisition. The paper also presented a practical case-study of the memory-acquisition of Allen-Bradley 1756-A10 hosting 1756-L61 controller. The memory acquisition tool and sample memory dumps are shared publicly for further research by the community interested in PLC memory-forensics.

Acknowledgement

This work was supported, in part, by the Virginia Commonwealth Cyber Initiative, an investment in the advancement of cyber R&D, innovation, and workforce development. For information, visit www.cyberinitiative.org.

References

Afonin, O., Katalov, V., 2016. Mobile Forensics—Advanced Investigative Strategies. Packt Publishing Ltd.
Ahmed, I., Obermeier, S., Naedele, M., Richard III, G.G., 2012. SCADA systems: challenges for forensic investigators. Computer 45 (12), 44–51.
Ahmed, I., Roussev, V., Johnson, W., Senthivel, S., Sudhakaran, S., 2016. A SCADA system testbed for cybersecurity and forensic research and pedagogy. In: Proceedings of the 2nd Annual Industrial Control System Security Workshop (ICSS).
Ahmed, I., Obermeier, S., Sudhakaran, S., Roussev, V., 2017. Programmable logic controller forensics. IEEE Secur. Privacy 15 (6), 18–24.
An12419, 2019. Secure jtag for i.mxrt10xx. https://www.nxp.com/docs/en/application-note/AN12419.pdf.
Ayub, A., Yoo, H., Ahmed, I., 2021. Empirical study of plc authentication protocols in industrial control systems. In: 15th IEEE Workshop on Offensive Technologies (WOOT). IEEE.
Bhatia, S., Behal, S., Ahmed, I., 2018. Distributed denial of service attacks and defense mechanisms: current landscape and future directions. In: Versatile Cybersecurity, vol. 72. Springer International Publishing, Cham.
Breeuwsma, M., 2006. Forensic imaging of embedded systems using jtag (boundary-scan). Digit. Invest. 3 (1), 32–42.
CISA USA, Critical Infrastructure Sectors. (Accessed 15 Mar 2021).
Control logic forensics framework using built-in decompiler of engineering software in industrial control systems. Forensic Sci. Int.: Digit. Invest. 33, 2020, 301013.
Denton, G., Karpisek, F., Breitinger, F., Baggili, I., 2017. Leveraging the srtp protocol for over-the-network memory acquisition of a ge fanuc series 90-30. Digit. Invest. 22, S26–S38.
Fabien, M., Bossuet, L., Gonzalvo, B., 2016. Jtag combined attack. In: 8th IFIP International Conference on New Technologies, Mobility & Security (NTMS). IFIP.
Firmware, 2013. Modification attacks on programmable logic controllers. Int. J. Critical Infrastruct. Protect. 6 (2), 76–84.
Garcia, L., Brasser, F., Cintuglu, M., Sadeghi, A.-R., Mohammed, O., Zonouz, S., 2017. Hey, My Malware Knows Physics! Attacking Plcs with Physical Model Aware Rootkit.
Grand, J., Jtagulator, 2013. https://github.com/grandideastudio/jtagulator.
Guri, M., Poliak, Y., Shapira, B., Elovici, Y., 2015. Joker: trusted detection of kernel rootkits in android devices via jtag interface. In: 2015 IEEE Trustcom/BigDataSE/ISPA, vol. 1. IEEE, pp. 65–73.
Kalle, S., Ameen, N., Yoo, H., Ahmed, I., 2019. CLIK on PLCs! Attacking control logic with decompilation and virtual PLC. In: Proceeding of the 2019 NDSS Workshop on Binary Analysis Research (BAR).
Konstantinou, C., Chielle, E., Maniatakos, M., 2018. Phylax: snapshot-based profiling of real-time embedded devices via jtag interface. In: 2018 Design, Automation & Test in Europe Conference & Exhibition (DATE). IEEE, pp. 869–872.
Kush, N.S., Foo, E., Ahmed, E., Ahmed, I., Clark, A., 2011. Gap analysis of intrusion detection in smart grids. In: Valli, C. (Ed.), Proceedings of 2nd International Cyber Resilience Conference. secau-Security Research Centre, Australia, pp. 38–46.
Pylink library for Segger jlink debugger, by Square Inc.. (Accessed 15 Mar 2021).
Qasim, S.A., Lopez, J., Ahmed, I., 2019. Automated reconstruction of control logic for programmable logic controller forensics. In: Information Security. Springer International Publishing, Cham, pp. 402–422.
Qasim, S.A., Ayub, A., Johnson, J., Ahmed, I., 2021. Attacking the iec-61131 logic engine in programmable logic controllers in industrial control systems. In: Staggs, J., Shenoi, S. (Eds.), Critical Infrastructure Protection XV. Springer International Publishing, Cham.
Rais, M.H., Li, Y., Ahmed, I., 2021. Spatiotemporal G-code modeling for secure FDM-based 3D printing. In: Proceedings of the ACM/IEEE Twelfth International Conference on Cyber-Physical Systems, ICCPS '21. Association for Computing Machinery, New York, NY, USA.
Schuett, C., Butts, J., Dunlap, S., 2014. An evaluation of modification attacks on programmable logic controllers. Int. J. Critical Infrastruct. Protect. 7 (1), 61–68.
Senthivel, S., Ahmed, I., Roussev, V., 2017. SCADA network forensics of the PCCC protocol. Digit. Invest. 22 (S), S57–S65.
Senthivel, S., Dhungana, S., Yoo, H., Ahmed, I., Roussev, V., 2018. Denial of engineering operations attacks in industrial control systems. In: Proceedings of the Eighth ACM Conference on Data and Application Security and Privacy, CODASPY '18, ACM, New York, NY, USA, pp. 319–329.
Stouffer, K.A., Falco, J.A., Scarfone, K.A., 2011. Sp 800-82. Guide to Industrial Control Systems (Ics) Security: Supervisory Control and Data Acquisition (Scada) Systems, Distributed Control Systems (Dcs), and Other Control System Configurations Such as Programmable Logic Controllers (Plc). Tech. rep., Gaithersburg, MD, USA.
Vishwakarma, G., Lee, W., 2018. Exploiting jtag and its mitigation in iot: a survey. Future Internet 10 (12), 121.
Willassen, S., 2005. Forensic analysis of mobile phone internal memory. In: IFIP International Conference on Digital Forensics. Springer, pp. 191–204.
Wu, T., Nurse, J.R., 2015. Exploring the use of plc debugging tools for digital forensic investigations on scada systems. J. Digital Forensics, Secur. Law 10 (4), 7.
Yoo, H., Kalle, S., Smith, J., Ahmed, I., 2019a. Overshadow plc to detect remote control-logic injection attacks. In: Perdisci, R., Maurice, C., Giacinto, G., Almgren, M. (Eds.), Detection of Intrusions and Malware, and Vulnerability Assessment. Springer International Publishing, Cham, pp. 109–132.
Yoo, H., Ahmed, I., 2019b. Control logic injection attacks on industrial control systems. In: Dhillon, G., Karlsson, F., Hedström, K., Zúquete, A. (Eds.), ICT Systems Security and Privacy Protection. Springer International Publishing, Cham, pp. 33–48.