

Empirical Evaluation of Suitability of Existing Forensics Solutions to Support Live Forensics on SCADA Machinery Control Systems

Avinash Srinivasan
Computer and Information Sciences
Temple University
Philadelphia, PA 19121
Email: avinash@temple.edu

Frank Ferrese
Naval Surface Warfare Center
Philadelphia, PA 19121
Email: frank.ferrese@navy.mil

Abstract—SCADA-based Machinery Control Systems (SCADA-MCSs) from leading vendors today are more vulnerable than ever before. To further complicate the secure and reliable operations of these systems, their cyber-physical environments open a whole new world of vulnerabilities which can be readily exploited, many of which are freely available on the Internet. SCADA-MCS being mission critical, must be continuously operational with zero downtime. Consequently, a forensic investigator cannot shut down the system to capture evidentiary data and analyze them. In such scenarios where there is zero downtime tolerance, live forensics is the only viable solution for conducting a digital investigation. Live forensics involves performing data acquisition and analysis on a running system. This will allow for a complete and detailed analysis while the system is still up and running, therefore having little to no effect on the mission.

In this paper, we evaluate some of the most popular and powerful forensics tools and frameworks to determine their suitability for porting to SCADA environment, specifically for MCS. We design and develop a rudimentary prototype of a live forensics framework for SCADA-MCS that can readily integrate the existing tools and techniques on to a common platform. Our prototype is designed conforming to *Open Systems Architecture* (OSA) to support cross vendor and cross platform solutions. Our empirical analysis highlights the strengths and weakness of tools studied proposes other requisite capabilities for tools to support a comprehensive live forensics on SCADA-MCS.

Keywords—Confidentiality, cyber-physical systems, data leak, fieldbus, integrity, live forensics, networks, SCADA, security.

I. INTRODUCTION

Vulnerabilities of such MCSs with majority proprietary system components and protocols can have vulnerabilities ranging from the very basic issues such as systems without passwords or with hard-coded passwords known to everyone to configuration issues and other software bugs.

Control systems domain has slowly yet steadily transitioned from the traditional all-analog domain to a more cyber-driven domain popularly known as the Cyber Physical Systems (CPS). Despite a rapid adoption of the new cyber-driven technology, the physical realm of the CPS is predominantly comprised of legacy and proprietary hardware. Consequently, such proprietary aspects of control systems environments present new challenges hindering the translation

and integration of modern state-of-the-art forensics analysis tools and techniques into the control systems domain.

Supervisory Control and Data Acquisition (SCADA) systems are a type of Industrial Control System (ICS) that monitor and/or control legacy systems remotely. SCADA systems typically operate on large-scale processes across multiple sites, and can be used over long distances. A SCADA system consists of one supervisory system and many slave systems. Slave systems are divided into two subcategories – *Remote Terminal Units* (RTUs), and *Programmable Logic Controllers* (PLCs).

Originally, RTU devices only collected telemetry data, and only PLC devices possessed process control capabilities. However, current incarnations of the two products have gained enough features that the distinction is no longer meaningful. When SCADA systems were first developed, they communicated solely using the Modbus serial standard, RS485. Recently, many SCADA systems have been configured to communicate via TCP/IP instead of Modbus. This allows easier integration with existing high-bandwidth infrastructure, but introduces an array of risks of exposing system vulnerabilities to a third party and the *Internet* at large.

SCADA-MCS systems were originally created to be deployed in non-networked environments. Therefore they lack of adequate security against Internet-based threats and cyber-related forensics. In recent years, SCADA systems have undergone significant technology advancement that potentially increase the risks to which they are exposed. Among these risks, one that is the most evident and crucial is its increased connectivity that could permit remote controls over the Internet, or the incorporation of general purpose tools, thus incorporating already known vulnerabilities of these.

Any cyber-attack against SCADA systems demands forensic investigation to understand the cause and effects of the intrusion or disruption on such systems. Therefore, SCADA system forensics is an essential process within the cyber-security lifecycle that not only helps to identify the cause of an incident and those responsible but to help develop and design more secure CPS and critical infrastructure systems of the future. However, a SCADA system has a critical requirement of being continuously operational and therefore a forensic investigator cannot turn off the SCADA system for data acquisition and analysis. Current live forensic tools are typically

Incident	Year	Vulnerability	Malware	Attack Vector
Ukrainian Power Outage	2015	Excel Macros	BlackEnergy	Spear Phishing
New York Dam	2013	Command & Control	N/A	Cellular Modem
European Energy Company	2016	Windws Antivirus	SFG malware	Security appliance bypass backdoor on Windows

Table I. MAJOR CYBERQATTACKS ON CONTROL SYSTEMS

used to acquire and examine memory from computers running either Windows or Unix. This makes them incompatible with embedded devices found on SCADA systems that have their own bespoke operating system. Additionally, only a limited number of forensics tools have been developed for SCADA systems, with no development of tools to acquire the program code from PLCs. With direct access to the memory addresses of the PLC, PLC debugging tools have promising functionalities as a forensic tool, such as the “Snapshot” function that allows users to directly take values from the memory addresses of the PLC, without vendor specific software.

A. Attacks on Control Systems – Scope of the Problem

B. Solution Novelty and Summary of Contributions

The proposed solution is designed conforming to the guidelines of OSA/OSE standards. Consequently, the solution framework provides several benefits over custom and proprietary solutions. Some of the the key benefits are delineated below –

- Empirically measures the performance of state-of-the-art forensics tools.
- Empirical validation is performed in realworld SCADA-MCS settings.
- Numerous tools are compared for their performance under the same SCADA-MCS operating conditions.
- Analyzes the potential for combining two or more existing tools efficiently to support live forensics on SCADA-MCS.
- Provides requisite capabilities for any of these tools – either independently or in combination to support comprehensive live forensics on SCADA-MCS.

C. Roadmap

The remainder of this paper is organized as follows. In section III, we review relevant background information and related works. Then in section ??, we present the target MCS model followed by a detailed discussion on the design and implementation of the proposed solution in section VI. We present results evaluating the proposed solution framework using the proposed metrics in section VII-D. Finally, we conclude this paper with concluding remarks with directions for future research in section VIII.

II. RELATED WORK

A large portion of research in the domain of control systems forensics has focused Programmable logic controller (PLC) firmware, which provides a software-driven interface between system inputs and physically manifested outputs, is readily open to modification at the user level. PLC is a special

Table II. LIST OF ACRONYMS AND NOTATIONS USED.

Notation	Description
AC	Access Control
ADU	Application Data Unit
API	Application Programming Interface
ARC	Architecture
ARM	Advanced Risk Machine
CPS	Cyber Physical System
DLL	Dynamically Link Library
HW	Hardware
IPC	Inter Process Call
IR	Incident Response
LKM	Loadable Kernel Module
MCS	Machinery Control System
MTU	Master Terminal Unit
OE	Operating Environment
OSA	Open Systems Architecture
OSE	Open Systems Environment
PDU	Protocol Data Unit
PLC	Programmable Logic Controller
PPC	Power PC
RPC	Remote Procedure Call
RAM	Random Access Memory
RTU	Remote Terminal Unit
SW	Software

form of microprocessor-based controller with proprietary operating system. Due to the unique architecture of PLC, traditional digital forensic tools are difficult to be applied.

In [8], authors propose a program called *Control Program Logic Change Detector* (CPLCD), which works with a set of *Detection Rules* (DRs) to detect and record undesired incidents on interfering normal operations of PLCs. In order to prove the feasibility of their CPLCD solution, authors set up two experiments for detecting two common PLC attacks and illustrate the advantages of using a network traffic analyzer such as Wireshark in tandem with CPLCD for performing digital forensic investigation on PLCs.

In [2] authors note that current efforts to protect PLCs against firmware attacks are hindered by a lack of prerequisite research regarding details of attack development and implementation. In their effort, authors address threats posed by PLC firmware counterfeiting and the feasibility of such attacks, this research explores the vulnerability of common controllers to intentional firmware modifications. In particular, this work derives the firmware update validation method used for the Allen-Bradley ControlLogix PLC. Through a proof-of-concept, authors demonstrate how to alter a legitimate firmware update and successfully upload it to a ControlLogix L61. Subsequently, they present possible mitigation strategies which includes digitally signed and encrypted firmware as well as preemptive and post-mortem analysis methods to provide protection. Authors conclude their work noting that their work will motivate future research in PLC firmware security through direct example of firmware counterfeiting.

In [5] authors create a high level software application capable of detecting critical situations like abnormal changes of sensor reads, illegal penetrations, failures, physical memory

content and abnormal traffic over the communication channel in a SCADA environment. A key challenges authors are faced is to minimize the impact of the tool on the SCADA resources, during the data acquisition process.

Authors in [3] provide an overall forensic taxonomy of the SCADA system incident response model. The proposed taxonomy model discusses the development of forensic readiness within SCADA system investigations, including the challenges faced by the SCADA forensic investigator and suggests ways in which the process may be improved.

In [7], authors explore the problem of acquiring program code from PLCs under two hypotheses – 1) *Program code is an important forensic artefact that can be used to determine an attacker's intentions*; and 2) *PLC debugging tools can be used for forensics to facilitate the acquisition and analysis of the program code from PLCs*. Based on the experiments conducted using tools from NIST CFTT, the results rejected the second hypothesis since their tool – PLC Logger – had failed half of the tests. With this, authors conclude that the PLC Logger in its current state has limited suitability as a forensic tool. To be accepted as a forensically sound tool, the shortcomings have to be addressed.

In [1],

In [4],

In [6],

III. BACKGROUND AND PRELIMINARIES

MCS Vulnerabilities can be broadly categorized into the following two groups – *Platform Specific Vulnerabilities* and *Network Specific Vulnerabilities*.

A. ProfiBus – A Review

Process Field Bus – PROFIBUS – is a standard for fieldbus communication in automation technology and used by Siemens. PROFIBUS is standardized in IEC 61158 and is the world's most successful fieldbus with 53.7 million devices installed by the end of 2015. Utilizing a single, standardized, application-independent communication protocol, PROFIBUS supports fieldbus solutions both in factory and process automation as well as in motion control and safety-related tasks.

B. MCS Operational Environment Challenges and Limitations

Below is a list of challenges and constraints in designing and developing tools and techniques to support live forensics in SCADA-MCS environment.

1) *Hardware Platform Limitations*: The three most popular HW ARCs for embedded components of MCS such as RTU and PLC are “ARM”, “PPC”, and “x86/x64”. However, x86/x64 is out numbered by ARM and PPC by a significant margin. While open-source memory forensics frameworks such as Volatility and Linux do provide support for ARM and PPC, they are primarily built around x86/x64 systems. This is due to the enormous market share of Desktops/PCs.

2) *Software Platform Limitations*: Majority of networked components for SCADA run on legacy platforms with extremely small quantities of physical memory. Consequently, they have little room for natively expanding their capabilities to support augmented logging for IR and forensics analysis.

3) *Lack of Standards and API Definitions*: There is very limited information about the SCADA MCS architecture and system APIs available for building enhanced plug-in tools to support out-of-band processing to support forensics.

4) *Proprietary Solutions*: Proprietary tools have limited capabilities and economically infeasible. Furthermore, proprietary tools often tend to be tailored for very specific tasks and platforms. Such lack of flexibility is very concerning given the evolving and volatile state of cyber-attacks. Existing proprietary and open-source tools with live forensics capabilities are prohibitive from a resource standpoint – massive memory footprint and significant IO bound operations topping the list of resource concerns.

5) *Ready Adoption Challenges*: Adoption of custom open-source tools “as-is” can present significant challenges with regards to legal and regulatory requirements. Porting necessary requirements for cyberforensics of SCADA system components into existing open-source tools warrants significant effort.

6) *High Degree of Proprietary Solutions and Heterogeneity*: Not only are the solutions proprietary in nature due to the unique domain requirement of each and every control systems environment, but also there degree of heterogeneity among field devices even within the same domain is significant to have a “one-size-fits-all” solution. Such proprietary solutions with such heterogeneity will make live forensics extremely challenging some of which are as follows – network constraints due to OEM restrictions, node/device constraints due to variations in HW/SW, communications security due to legacy and resource constrained devices and proprietary protocols, and the likes.

C. Static VS. Dynamic LKMs

When developing kernel modules, there are two fundamental choices – either a static module or a dynamic module. While both these kernel module types have a unique set of features, they also share some commonalities. Before the module is implemented, it is critical to know what does each type have to offer.

Static Module. A static module is very robust and is integrated into the OS kernel by binding it to the base kernel (BK). Therefore, a static module will remain in memory whenever the system is booted and running. To unload a static kernel module, the base kernel has to be recompiled. Furthermore, the rigid binding of static module to the base kernel necessitates recompiling the core kernel whenever a change is made or a new functionality is added. Memory overhead is permanent module remains in the memory along with BK. Higher risk of errors in BK. Buggy module can destabilize the BK and prevent the system from booting this can create false positives for DoS attacks. Counterintuitive as it makes the design rigid and monolithic hindering the objective of a modular approach for an (M)OSA. Binding with BK does not add any significant performance improvement.

Open Source Tools	Capabilities	Limitations
Rekall Memory Forensics Framework	<p>A computer forensics tool that scans a memory dump image or virtual machine snapshot and extracts useful information</p> <p>Includes a toolkit for acquiring memory dumps on Windows, Linux, and OSX systems</p> <p>Can be used either as a standalone application or as a scripting library to automate memory analysis</p>	<p>Requires a “profile” (aka config file) to analyze images of systems with address spaces that aren’t included in its releases</p> <p>Generating profiles requires compiling Rekall provided kernel module</p> <p>Analysis capabilities are primarily limited to x86/amd64 systems</p>
Volatility Memory Forensics Framework	<p>Functionality can be extended with plugins and address space profiles</p> <p>Forensics tool-suite that scans and extracts information from memory dump images of Win, OSX, Linux, & Android systems</p>	<p>Volatility is not capable of acquiring memory dumps – it is dependent on other tools like LiME</p> <p>Framework’s profile subsystem has compatibility limitations similar to that of Rekall</p> <p>Compatible with ARM systems, primarily Android, does not carry over to other embedded ARM systems running desktop Linux</p>
Volatilitux	<p>Released in 2010 to add Linux compatibility in Volatility</p> <p>Has compatibility with ARM-based desktop Linux distributions</p>	<p>Not actively updated since 2010, and has limited support for Linux Kernel beyond 2.6.x</p> <p>Compatible only with images captured using Volatilitux</p> <p>Very limited analysis capability – only 5 commands available</p>
LiME (Linux Memory Extractor)	<p>Linux LKM capable of acquiring memory dump in several different formats – “.lime”, “.raw”, and “padded”</p> <p>Compatible with desktop Linux distributions and Android</p>	Lacks memory dump processing and analysis capability
Dumpchk.exe	<p>Microsoft freeware for validating Windows memory dumps</p> <p>Limited capability of parsing certain types of metadata from memory dumps</p>	<p>Only supports Windows memory dumps</p> <p>Proprietary and non-extensible</p>
Autopsy/ The Sleuth Kit (TSK)	Disk forensics analysis tool capable of handling numerous file system types for disk analysis	Does not support memory acquisition or memory analysis
Bulk Extractor	<p>Disk forensics analysis tool</p> <p>Capable of analyzing disk images for certain types of forensically relevant information (phone, CC numbers, URLs, etc.)</p>	<p>Can only process acquired memory images for network connection, registry keys, sockets, etc.</p> <p>Does not support memory analysis at process level unlike Volatility or Rekall</p>

Table III. SUMMARY OF OPEN-SOURCE FORENSICS TOOLS THAT SUPPORT LIVE FORENSICS EITHER DIRECTLY OR INDIRECTLY.

Dynamic Module. Integrated into BK at run time when needed. Loaded dynamically when needed and then unloaded. Very flexible and can be easily modified to add additional functionalities. Memory overhead is temporary since they are loaded when actually being used. Eliminates risk of introducing errors into BK. Buggy modules do not have any bearing on the BK since they are not loaded until the system is up and running. Facilitates a modular design where in modules can be rapidly designed, developed and deployed with out of band testing. Dynamic modules are equally fast when loaded into memory since both modules resides in different parts of the same memory type.

IV. STATE-OF-THE-ART LIVE FORENSICS TOOLS – A REVIEW

State-of-the-art tools with live forensics capabilities primarily include *Volatility* and *Rekall*. Both this frameworks currently provide very limited support for ARM and PowerPC (PPC) memory architectures, since they have been primarily developed around the x86 systems. While it is possible to expand either of these frameworks to support live forensics on MCS systems, it is significant effort to write a plugin/LKM for ARM/PPC memory architectures taking into considerations the subtle variations implemented by vendors creating proprietary

memory architecture.

A. SCADA-MCS – Operational Environment Challenges

A majority of the MCS OE is comprised of legacy HW combined with proprietary protocols and OS. Further, there is very limited architecture specifications and API support and specifications for developing and integrating tools for supporting “in-band” native forensics analysis capabilities. The legacy nature of the MCS OE platform also makes it extremely hard for “out-of-band” forensics analysis.

- Proprietary tools are rigid due to customized solutions, have limited capabilities, and are prohibitively expensive.
- APIs available on the target SCADA system for integrating live forensics tools are limited.
- Support for expand existing open source forensics tools to support SCADA live forensics are minimum.
- MODBUS TCP/IP on registered port 502 and Serial in either ASCII Mode or Binary Mode (RTU)
- Kernel support for custom APIs such as DLLs (Windows) and shared libraries (Unix/Linux).

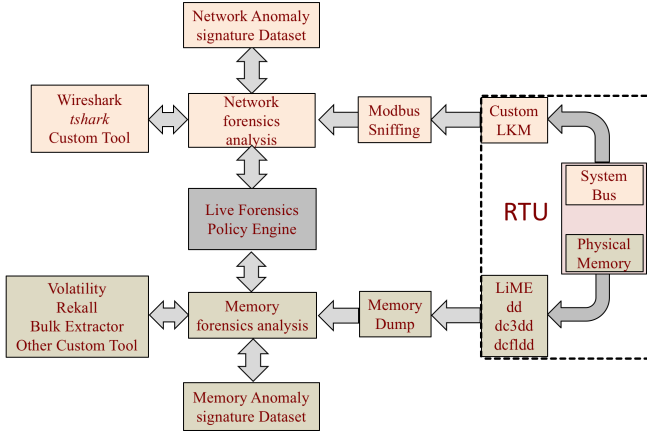


Figure 1. KRAFT Architecture with both Memory Dump and Modbus Traffic Analysis capabilities.

- Wrapper functions for system calls such as libc, glibc, etc.

V. KRAFT MODEL

KRAFT is a hybrid live forensics toolkit framework wherein we utilize two vastly different methods – Memory Analysis and Traffic Analysis. These two approaches have different purposes with regards to security and forensics analysis. However, the two methods can be coupled to work complementing each other. Such a hybrid tool platform will enable robust security and forensics analysis. The core of such analysis tools and techniques will be built around correlation. The open systems architecture toolkit framework KRAFT will use the following open source tools along with numerous custom tools.

- **LiME (Linux Memory Extractor).** LiME is an open source loadable kernel module that can acquire live memory from popular processor architectures – x86/x64, PPC, and ARM. Furthermore, LiME can extract the live memory in three different formats – .lime, .raw, . Compatible with desktop Linux distributions and Android. Does not include any tools to analyze the images it collects.
- **Bulk Extractor.** This is an open source file system forensics analysis tool suite. Unlike, LiME, bulk extractor is not capable of acquiring live memory. Furthermore, bulk extractor is also not capable of extracting process level information from a memory dump. However, the tool can scan memory dump images for certain types of forensically relevant information such as – IP address, Network sockets, Credit card numbers, URLs, Passwords, and Encryption keys. While the tool evidently has limitations in processing live memory, its limited capabilities are very useful in correlating live memory information and raw packet capture providing a unique purview during security and forensics analysis.

VI. SOLUTION MODEL

A. KRAFT Validation Metrics

We have also presented useful metrics for validation of the proposed solution for each of the two approaches – *Modbus Sniffing* and *Memory Dumping*. We have also identified potential live forensics artifacts relevant to MCS OE that can be used as indicators of threat/attack, and serve as the baseline for augmenting KRAFT’s live forensics capabilities.

1) *Metrics for Modbus Traffic Analysis Approach:* We propose using the following metrics for validating the proposed KRAFT toolkit’s performance under the modbus traffic analysis approach. While numerous metrics exist and provide robust validation, below are the metrics we specifically use for establishing baseline traffic pattern against which KRAFT’s performance in effective live forensics of the MCS field devices will be measured. These metrics are –

- *average number of packets per unit time*
- *average request/response packet size per unit time per device type*
- *average number of ingress/egress connections per device type*
- *most frequently used and least frequently used function codes accessed per device type*
- *average number of u-cast, m-cast, b-cast packets per unit time per device type*
- *average number of errors reported per unit time per field device type*
- *average number of request/response with non-standard protocols from unusual ports per unit time per device type*

2) *Metrics for Memory Dump Analysis Approach:* We shall use the following candidate metrics for validating the proposed KRAFT toolkit’s performance under the memory dump analysis approach. The below candidate metrics for validation of KRAFT will be used to first establish a baseline traffic pattern against which KRAFT’s ability to support effective live forensics of the MCS field devices will be measured. The suggested metrics are as follows –

- Average allocated memory per process
- Number of DLLs/Library functions in memory
- Number of AC violations by processes
- Network IO usage
- Central Processing Unit (CPU) usage
- Number of failed processes/operations logged
- Number of IPC/RPC calls made per unit of time

Some of the open source memory forensics tools we have explored in this research work include – *Rekall*, *Volatility*, *Bulk Extractor*, and *LiME*.

Deviation Impetus	Potential Target
Unusual access requests	<ul style="list-style-type: none"> – to devices that were never accessed before – from devices that have never requested before – to secure/restricted resources (system files)
Unusual commands	<ul style="list-style-type: none"> – authorized patterns not seen before – unauthorized patterns – unreasonable rates
Unusual packet format	<ul style="list-style-type: none"> – valid header but not seen before – unknown packet header – payload size unusual
Unusual account access	<ul style="list-style-type: none"> – too many requests – too many failed requests – requests to unknown accounts – Missing regular and expected requests

Table IV. INDICATORS OF ABNORMAL EVENTS – MALICIOUS AND SYSTEM MALFUNCTIONS.

VII. MODEL EVALUATION

A. Validation Framework

This week, we conducted some testing on Rekall's WinPMem tool to determine how different operational parameters affect the time and storage space it requires to capture memory dumps. Our tests included:

- 1) Capturing memory dumps of the same system with different amounts of system memory installed, and recording the time and storage overhead for each.
- 2) Capturing memory dumps using different file compression options: the snappy algorithm, the zlib algorithm, and no compression. Although Rekall's website claims that snappy compression is the fastest option, we found that storing the files in uncompressed form was often slightly faster on the machine we used.
- 3) Observing WinPMem's behavior when we try to append a new memory dump to an existing one. This is the default behavior for when the specified output file already exists.

B. Transmitting memory dumps via Modbus TCP

From number of bytes (2) in the message length field, we conclude that each Modbus TCP frame can hold a payload up to 65535 bytes in size. Since very few embedded systems with system memory of 64 KB or less are still in use, any implementation of memory dump transmission via Modbus TCP should be designed to accommodate file splitting on the server side and file splicing on the client side. Table 1 shows how different sizes of memory dump would be split for transmission:

1) *Packet Crafting Module:* We have implemented a custom kernel module that is a packet crafter such that the memory dump can be packaged into independent data packets and/or embedded into unused bits of data packets exchanged between the field device and the MTU over the course of several request/response messages.

C. Observations

In item-3 above, we expected winpmem to either append a complete memory dump to the existing one and double its size,

or to append a subset of the address space that had changed since the previous dump was captured. Instead, the program returned an error. We intend to investigate this further.

Rekall has a small but powerful suite of built-in plugins to analyze Windows registry entries. In this example, we will use them to find and display registry key pairs contained in our memory dump. With our image file open in Rekall, we begin by calling the hives plugin.

D. Results

E. Secure Memory Dump Transfer to MTU

Protecting the data against corruption and truncation would require additional measures. Two possibilities exist to alert the client to any errors in data transmission or splicing:

- 1) Transmit a single checksum for the entire memory dump, and verify the integrity of the reassembled dump-file at the MTU. This would require very little overhead. The comparison in Table ?? shows that for several common RAM configurations, packaging a checksum from the Linux command sha256sum with the memory dump requires no extra packets to be sent. Only one checksum of the memory dump file would be necessary because TCP already performs checksum-based verification of each packet. The only concerning aspect of this method is its potential to become too complex, in terms of computation and disk access, to be feasible on an embedded system.
- 2) Overlap the transmissions such that each fragment is transmitted with a small amount of data from the previous fragment. This method would decrease the complexity of disk and computational operations, but transmission overhead would increase proportionally to the size of the overlap.

VIII. CONCLUSION AND FUTURE WORK

IX. DEFINITIONS

Definition 1. *Cyber-Physical Systems are co-engineered interacting networks of physical and computational components that are built from, and depend upon, the seamless integration of computational algorithms and physical components.*

Definition 2. *Control Systems manage, command, direct or regulate the behaviour of other devices or systems ranging from a home thermostat controlling a domestic boiler to large industrial control systems which are used for controlling processes or machines.*

Definition 3. *Industrial control system (ICS) is a general term that encompasses several types of control systems and associated instrumentation used in industrial production, including supervisory control and data acquisition (SCADA) systems, distributed control systems (DCS), and other smaller control system configurations such as programmable logic controllers (PLC) often found in the industrial sectors and critical infrastructures.*

Definition 4. *A programmable logic controller (PLC), or programmable controller is an industrial digital computer which has been ruggedised and adapted for the control of manufacturing processes, such as assembly lines, or robotic*

Artifact	Evidentiary Information
Timestamps	Process initialization; Process termination
Usage pattern	Resident memory usage; CPU and IO activity; Disk capacity and usage; Network activity
Security/Audit logs	Unauthorized login attempts; Unauthorized access attempts
System logs	Crash, Halt; Shutdown, Reboot
Activity logs	Processes and Process IDs; Executable-Process ID maps; Resource utilization; Process automation
Kernel logs (e.g., Unix sys call printk())	Modules loaded and unloaded; Emergency messages; Critical errors

devices, or any activity that requires high reliability control and ease of programming and process fault diagnosis.

Definition 5. Supervisory control and data acquisition (SCADA) is a control system architecture that uses computers, networked data communications and graphical user interfaces for high-level process supervisory management, but uses other peripheral devices such as programmable logic controllers and discrete PID controllers to interface to the process plant or machinery.

Definition 6. A remote terminal unit (RTU) is a microprocessor-controlled electronic device that interfaces objects in the physical world to a distributed control system or SCADA (supervisory control and data acquisition) system by transmitting telemetry data to a master system, and by using messages from the master supervisory system to control connected objects.

Definition 7. A Distributed Control System (DCS) is a computerised control system for a process or plant, in which autonomous controllers are distributed throughout the system, but there is central operator supervisory control.

Definition 8. A loadable kernel module (LKM) is an object file that contains code to extend the running kernel (aka base kerne) of an operating system typically used to add support for new hardware (as device drivers) and/or filesystems, or for adding system calls to provide additional functionality.

Definition 9. Advanced RISC Machine (ARM) is a family of reduced instruction set computing (RISC) architectures for

computer processors, configured for various environments. ARM has reduces costs, heat and power use making it highly desirable for embedded systems in SCADA-MCS environment.

REFERENCES

- [1] Irfan Ahmed, Sebastian Obermeier, Martin Naedele, and Golden G Richard III. Scada systems: Challenges for forensic investigators. *Computer*, 45(12):44–51, 2012.
- [2] Zachry H Basnight. Firmware counterfeiting and modification attacks on programmable logic controllers. Technical report, DTIC Document, 2013.
- [3] Peter Eden, Andrew Blyth, Pete Burnap, Yulia Cherdantseva, Kevin Jones, Hugh Soulsby, and Kristan Stoddart. A forensic taxonomy of scada systems and approach to incident response. In *Proceedings of the 3rd International Symposium for ICS & SCADA Cyber Security Research*, pages 42–51. British Computer Society, 2015.
- [4] Joe Stirland, Kevin Jones, Helge Janicke, and Tina Wu. Developing cyber forensics for scada industrial control systems. In *The International Conference on Information Security and Cyber Forensics (InfoSec2014)*, pages 98–111. The Society of Digital Information and Wireless Communication, 2014.
- [5] Pedro Taveras. Scada live forensics: real time data acquisition process to detect, prevent or evaluate critical situations. *European Scientific Journal*, ESJ, 9(21), 2013.
- [6] Ronald M van der Knijff. Control systems/scada forensics, what’s the difference? *Digital Investigation*, 11(3):160–174, 2014.
- [7] Tina Wu and Jason RC Nurse. Exploring the use of plc debugging tools for digital forensic investigations on scada systems. *The Journal of Digital Forensics, Security and Law: JDFSL*, 10(4):79, 2015.
- [8] Ken Yau and Kam-Pui Chow. Plc forensics based on control program logic change detection works. *The Journal of Digital Forensics, Security and Law: JDFSL*, 10(4):59, 2015.