Improving Network Security Monitoring for Industrial Control Systems

Tiago Cruz[1], Jorge Barrigas[1], Jorge Proença[1], Antonio Graziano[2], Stefano Panzieri[3], Leonid Lev[4], Paulo Simões[1]

[1] DEI-CISUC, University of Coimbra, Coimbra, Portugal
[2] Selex ES, Rome, Italy
[3] Dip. Informatica e Automazione, Università degli Studi 'Roma Tre', Rome, Italy
[4] Israel Electric Corporation, Haifa, Israel

***Abstract*— Programmable Logic Controller (PLC) technology plays an important role in the automation architectures of several critical infrastructures such as Industrial Control Systems (ICS), controlling equipment in contexts such as chemical processes, factory lines, power production plants or power distribution grids, just to mention a few examples.**

Despite their importance, PLCs constitute one of the weakest links in ICS security, frequently due to reasons such as the absence of secure communication mechanisms, authenticated access or system integrity checks. While events such as the Stuxnet worm have raised awareness for this problem, industry has slowly reacted, either due to reliability or cost concerns.

This paper introduces the Shadow Security Unit, a low-cost device deployed in parallel with a PLC or Remote Terminal Unit (RTU), being capable of transparently intercepting its communications control channels and physical process I/O lines to continuously assess its security and operational status. The proposed device does not require significant changes to the existing control network, being able to work in standalone or integrated within an ICS protection framework.

***Keywords*— Industrial Control Systems, Critical Infrastructure Protection, SCADA, Programmable Logic Controllers**

I. INTRODUCTION

In recent years there has been a series of attacks on the Industrial Control Systems (ICS) of Critical Infrastructures (CIs), such as power stations, power distribution grids, wastewater treatment units or even nuclear fuel processing plants, with consequences that affect an increasing larger number of persons. In most cases, forensic analysis has frequently attributed the success of such attacks to the fact that most of the existing ICS technologies were designed with reliability in mind, relegating security to a secondary role.

Until recently, the ICS industry traditionally relied on isolation (the "airgap principle"), together with a "black box" approach, based on obscurity and use of proprietary and/or poorly documented technologies to ensure security. With convenience and cost pushing for the introduction of commodity technologies from the IT world in ICS, such as TCP/IP networking, the airgap started to close. Besides that, the use of clear-text communications, the lack of Authentication, Authorization and Accounting (AAA) capabilities or the absence of integrity checking mechanisms nullified the benefits of the security-by-obscurity approach, exposing ICS to reverse engineering attempts. As a consequence, ICS have become vulnerable to a range of threats that were unconceivable before on this ecosystem, constituting an important target for several reasons, from ego-driven, frivolous or revenge attacks, to stealthy Advanced Persistent Threats (APTs) running over a long time frame as part of a cyber-warfare strategy to gather intelligence and later cripple public and/or military CIs.

As part of a wide scale-effort to address the problems of CI interdependency and security, addressed by European FP7 projects such as MICIE [1] and, later, CockpitCI [2], the authors were involved in the design of a Critical Infrastructure Protection (CIP) framework. The Shadow Security Unit (SSU), introduced in this paper, was developed as part of that framework. It consists on a device deployed in parallel with a PLC or Remote Terminal Unit (RTU), which transparently intercepts communications and physical I/O control to provide continuous security and status monitoring. The SSU is able to detect a series of attacks, such as unauthorized accesses; program tampering; Denial-of-Service or flooding, among others, while also adding remote diagnostic features.

The rest of this paper is organized as follows: Section 2 discusses ICS security, with a focus on SCADA (Supervisory Control and Data Acquisition) systems; Section 3 covers the SSU architecture and Section 4 concludes the paper.

II. A REVIEW OF ICS/SCADA SECURITY ISSUES

SCADA is a common designation for several technologies, protocols and platforms used in ICS for control and automation of production lines, power plants (nuclear, thermoelectric, wind farms), management of distribution grids (electricity, gas, oil, water) and other applications. SCADA systems include several types of components [3], namely:

- **Master stations** (deployed on the process network) supervise processes, controlling and monitoring Slaves and often providing support for HMI (Human-Machine Interface) consoles. They are also frequently connected to other applications, such as databases, to log process data.
- **Slave devices** (Programmable Logic Controllers – PLCs and Remote Terminal Units – RTUs, deployed on the control network) are embedded systems connected to one or more Master stations, as well as to sensors and actuators, being responsible for monitoring and control.

- **Field devices** (deployed on the field network) constitute the physical interface with the process, extracting information about it (sensors) and enabling the execution of actions affecting its behaviour (actuators).

These components are interconnected among them using technologies, such as Ethernet, RS-485 [4], CAN [5], EtherCAT [6] or Profinet [7], among others. The control flow between the Master station and the PLC/RTU allows exchanging process-related data or executing actions by modifying process control parameters mapped on device registers. This process involves SCADA protocols, such as Modbus [8], IEC 60870-5-104 (IEC 104) [9] or DNP3 [10].

Having become a cornerstone of many ICS and CIs, SCADA systems constitute a potential target for malicious activity. As they were originally restricted to isolated environments, SCADA systems were considered relatively safe from external intrusion. However, as architectures evolved, these systems started to assimilate technologies from the Information and Communication Technologies (ICT) world, such as TCP/IP and Ethernet networking, encouraging the interconnection of the ICS with organizational ICT network infrastructures and even with the exterior (e.g., for remote management). This trend, together with the increasing adoption of open, documented protocols, exposed serious weaknesses in SCADA architectures and brought a new wave of security problems that were not conceivable when such systems were first designed [11][12], prompting a significant increase in the number of externally initiated attacks on ICS systems, especially when compared with internal attacks [13].

Despite general knowledge among manufacturers and end-users alike, the problem of security in SCADA systems remained ignored for several years, with the disclosure of related security issues having seemingly little or no effect in discouraging the usage of unsafe components or prompting the adoption of measures to protect them. This is partly due to the fact that, for ICS, availability is the main concern, meaning that technological maturity is frequently regarded as an implicit recognition of value and reliability – for instance, protocols such as Modbus [8], which was originally developed in 1979, are still very popular in production systems, despite suffering from security problems such as the lack of encryption or any other protection measures [12]. This situation has become the root cause of many ICS security issues, such as the Stuxnet [14] worm, which was famous for raising awareness for the problem of ICS security.

When it comes to their fundamental governing principles, ICS and ICT infrastructures have differences that are deeply rooted in their own specific characteristics, as noted by ISA-99 [15]; namely, an inverted set of priorities which is one of the main causes of SCADA security problems. For ICS, availability comes first, even if at the cost of integrity and confidentiality – just the opposite of the ICT philosophy.

The differences between the ICT and ICS contexts mean that there is no "one size fits all" solution when it comes to choose and deploy security mechanisms. This calls for the development of domain-specific cyber-security mechanisms for ICS. This is one of the main objectives of the CockpitCI project, which focuses on improving the resilience and dependability of CIs, by detecting cyber-threats and sharing security information among CI operators. Among the domain-specific cyber-security mechanisms that are being researched in the scope of CockpitCI, the Shadow Security Unit, which is the main subject of this paper, constitutes a novelty in terms of ICS security that will be presented in the next chapter.

III. The Shadow Security Unit

During the development of the CockpitCI cyber-security detection framework, the authors were faced with the problem of effectively protecting PLCs and RTUs. While playing an important role in the ICS context, such devices often lack adequate security mechanisms, having been in the past successfully targeted by a wide array of attacks, such as flooding, buffer overflow exploits or man-in-the-middle, just to mention a few. To address this problem, several authors have proposed the use of techniques such as bump-in-the wire VPNs, tight access control-list mechanisms or introducing mutual authentication – countermeasures whose deployment is not feasible in all scenarios, for reasons such as latency overhead [12], reliability or the need for introducing profound changes on well-established protocols and architectures.

This situation has led to the development of a device for minimally intrusive, continuous PLC/RTU security monitoring: the Shadow Security Unit (SSU – see Figure 1).

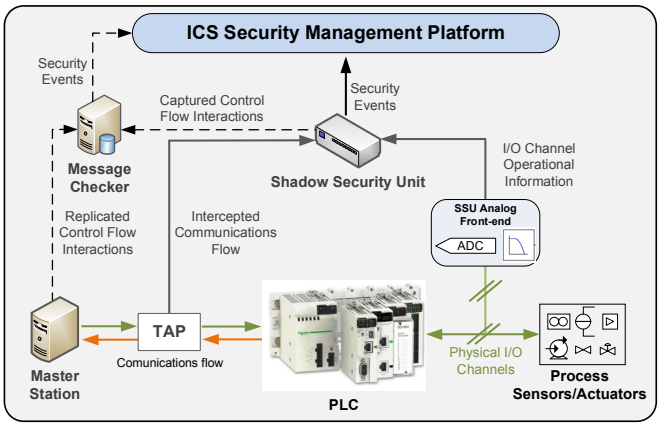

Figure 1: Shadow Security Unit deployment (dashed lines connect optional components)

The SSU is attached in parallel to RTU/PLCs, being able to capture and decode the SCADA protocol information flow, correlating this information with the status of the physical I/O modules that interface with sensors and actuators on the field. This enables the possibility of implementing a redundant security-checking mechanism that follows a "black box" approach regarding the analysis of the monitored device behaviour. It is transparent to the production system, requiring minimum changes to the existing architecture and, since it is out of the critical control path, it cannot interfere with the operation of the system, further minimizing any other potential impact such as latency overhead or an eventual malfunction on the monitored device. Moreover, these behaviour analysis capabilities can be effective for detection of a Stuxnet-like

scenario, making it possible to detect the abnormal behaviour of a reprogrammed PLC, even if it was reprogrammed by a legitimate, yet compromised, Master station or HMI.

While a conventional NIDS may share some of the SSU capabilities, there are limitations: first, when deployed in inline mode a NIDS constitutes an undesirable point-of-failure that might also degrade latency (a sensitive issue, especially in real-time control scenarios); when deployed in passive mode, its effectiveness may be hampered by contention, especially in large scale scenarios – when the monitoring interface capacity of a switch is insufficient to handle the aggregated traffic from the source ports, overflow packets are dropped. Eventually, using a NIDS might not be feasible at all if the PLC/RTU is deployed on a remote location, served by a low bandwidth connection (such as VHF links used for telemetry).

Figure 2 presents the SSU architecture implemented for Ethernet with Modbus/TCP protocol scenarios, next discussed.

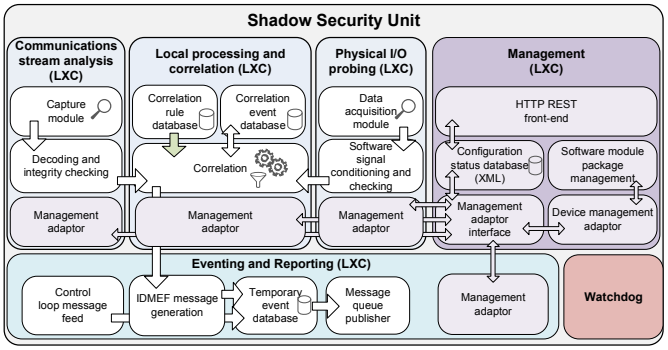

Figure 2: Shadow Security Unit architecture

A. *Communications stream analysis*

This module is responsible for transparently capturing and processing the command flow between the Master station and the PLC/RTU. For Ethernet-based systems, a low-cost active aggregation switch TAP is deployed between the monitored device and the upstream link of the communications infrastructure, forwarding a copy of all network traffic to the SSU, while requiring little change to the existing setup.

The SSU network interface connected to the TAP is devoid of IP configuration and does not reply to ARP requests (a requisite for transparency), working in promiscuous mode to capture all the network traffic and pre-process it using a rule-based filtering mechanism. Once captured, the protocol stream will be decoded and analysed, looking for errors or inconsistencies in Modbus/TCP protocol data units. The SSU is able to decode the semantics of each command and track the exchanged messages between Master stations and the monitored device, making it possible to perform integrity checks, detect unauthorized access or scouting attempts.

This module is also capable of extracting information about network flow traffic, such as packet rates or inter-message arrival times. This can be used to diagnose and uncover problems such as malware infections, Distributed Denial-of-Service (DDoS), flooding or brute-force attacks, arbitrary equipment failures, and even zero-day threats.

B. *Physical I/O probing*

Physical I/O probing modules provide information about the physical (discrete and/or continuous) inputs and outputs of the PLC/RTU that constitute the process cyber-physical interface. For such purpose, the SSU bundles (soft) real-time data acquisition capabilities – a signal pre-conditioning stage coupled to a precision differential voltage probe using operational amplifier technology feeds an 8 channel, 10-bit, successive approximation (SAR) ADC (Analog to Digital Converter) [16] with a 200K sample per second conversion rate (5ms cycle), adequate for discrete I/O scenarios – for analog I/O, a 12 or 16-bit ADC may be required

Later on, a software module processes the captured data stream at a fixed sampling rate, using timestamping to label the captured data for later correlation with the information from the communications stream analysis modules. In order to be able to properly interpret the captured data, the SSU must be configured (via the management API) with the characteristics of each physical interface such as the direction (input/output), type (discrete/analog) and voltage ranges. Moreover, the physical probing module also offers the possibility of remotely diagnosing the health of the physical process interface components.

C. *Local processing and correlation*

The information from the communications and I/O probing modules is fed to an embedded correlator component [17]. This correlator is in charge of the assessment of the monitored device status, checking if commands arrive from a legitimate source, if they are coherent with the expected control interface flow or if the physical I/O data is in-line with expected values (range checking) – also, for discrete control scenarios, the correlator incorporates a series of logical state maps (extracted from the PLC/RTU programming) that are used to check the behaviour of the monitored unit. The correlator incorporates pre-processing capabilities, for event reduction and aggregation within predefined time windows. Correlation rules are managed using the management API.

As the SSU is implemented using a low-cost Single Board Computer (SBC) system [18], there is a limitation in terms of processing capabilities that restrict the ability to perform a deeper analysis of the information that is captured. For this reason, the SSU is able to operate in a flexible way, either in standalone or as part of a security assessment loop.

D. *Eventing and reporting*

The SSU correlation module generates events that are sent to a Security Management Platform, such as a Security Information and Event Management (SIEM). Generated events are encoded using a general XML data format, the Intrusion Detection Message Exchange Format (IDMEF) [19].

Detection capabilities can be reinforced through the use of a (optional) Message Checker system (see Figure 1), creating a closed loop for command flow control. In this case, the eventing and reporting module is able to replicate copies of communication control flows and physical I/O data for consumption by a message checker system or high-level analysis mechanism, such as an anomaly detection system.

E. Management

The SSU architecture supports an out-of-band HTTP REST [20] API for device, service/component and software module management, accessible through a separated network interface (which is also used for eventing). Each internal SSU module is responsible for maintaining its configuration parameters (i.e., state and semantics) mapped into a common data model using an internal management adaptor. This adaptor performs the attribute mapping for the specific configurations of each SSU module, via an XML that instantiates a data model structure (described in an XSD file), creating a uniform management interface. REST API operations manipulate the data model instances for the SSU, with each property being identified using XML Path Language (XPath) expressions. Once a specific property is changed, the management adaptor for the associated SSU component translates the change into an action.

F. Real-time, secure execution environment

(Soft) real-time processing capabilities for the SSU are supported using the *Xenomai* [21] Linux kernel extensions. Also, the software modules of the SSU are deployed within Linux Containers LXC [22], providing filesystem, network, root privilege and system resource isolation. This approach delivers three main benefits: added security, thanks to component isolation from the remaining modules of the SSU; increased reliability and resilience in case of an individual component failure, since it is isolated from the remaining system; and ease of management, as all components correspond to software packages that can be individually updated, sparing the need for complete firmware updates. All modules communicate with each other using local UNIX domain sockets. Component management is ensured through the management API, which requires component images to be packed and signed using certificates, for increased security.

G. Watchdog

A watchdog module provides in-device monitoring of component and service operation. This component works at two levels: first, it implements a series of software routines that periodically check component operation and attempt recovery or restart in case of stalled operation; second, it implements system-level checks through a kernel module working together with a watchdog service that keeps a regular heartbeat to the hardware watchdog [23] timer of the SBC. Using the hardware watchdog allows the possibility of rebooting the entire SSU platform in case of a critical failure, after a predefined number of missed timer events.

IV. Conclusions and Next Steps

The presented Shadow Security Unit concept is a low-cost solution for securing SCADA systems, being complementary to existing SIEM architectures and, to the best of the authors' knowledge, constituting an new approach to the problem of security monitoring in ICS.

Future developments of the SSU include upgrading the behaviour detection mechanisms, by adding anomaly detection capabilities through the use of One-Class Support Vector Machines [24][25] for rogue threat detection, and also by using formal verification models fed with control and I/O status information, to track the PLC operation during runtime. Finally, the integration of the SSU with mechanisms for automatic deployment of countermeasures against cyber-attacks is underway, as part of the CockpitCI project.

Acknowledgements

This work was partially funded by the CockpitCI (FP7-SEC-2011-1 Project 285647) Project and by Project QREN ICIS (Intelligent Computing in the Internet of Services – CENTRO-07-0224-FEDER-002003).

References

[1] MICIE, FP7-ICT-SEC-2007-1 Project 225353, http://www.micie.eu.
[2] CockpitCI, FP7-SEC-2011-1 Project 285647, http://CockpitCI.eu.
[3] D. Bailey, E Wright, "Practical SCADA for Industry (IDC Technology)", Elsevier Press, 2003, ISBN 978-0750658058.
[4] Electronic Industries Alliance, "Electrical Characteristics of Generators and Receivers for Use in Balanced Multipoint Systems", EIA Std. RS-485, 1983.
[5] International Standards Organization, ISO 11898-1:2003, "Road vehicles - Controller area network (CAN) - Part 1", 2003.
[6] International Electrotechnical Commission, P-IEC/PAS 62407, 1.0, "Real-time Ethernet control automation technology (EtherCAT)", 2005.
[7] PROFIBUS & PROFINET International (PI), www.profibus.com.
[8] Modbus-IDA, "Modbus Application Protocol Specification 1.1b", 2006.
[9] International Electrotechnical Commission, "Telecontrol equipment and systems - Part 5-104: Transmission protocols - Network access for IEC 60870-5-101 using standard transport profiles", 2006
[10] IEEE, std 1815-2010, "IEEE Standard for Electric Power Systems Communications -- Distributed Network Protocol (DNP3)", doi: 10.1109/IEEESTD.2010.5518537, July 2010.
[11] V. Igure, S. Laughter, R. Williams, "Security issues in SCADA networks", Elsevier Computers & Security Journal, Volume 25, Issue 7, Pages 498-506, doi:10.1016/j.cose.2006.03.001, 2006.
[12] B. Zhu, A. Joseph, and S. Sastry, "A taxonomy of cyber attacks on SCADA systems," in Proc. of the 2011 International Conference on Internet of Things and 4th International Conference on Cyber, Physical and Social Computing, 2011, pp. 380 - 388.
[13] R. Evans, "Process Control System Cyber Security Standards - An Overview", in Proc. of 52nd International Instrumentation Symposium, 2006, Cleveland, USA.
[14] Ralf Langner, "To Kill a Centrifugue: A Technical Analysis of What Stuxnet's Creators Tried to Achieve", The Langner Group, 2013.
[15] International Society of Automation, "ISA-99.00.01: Security for Industrial Automation and Control Systems - Part 1: Terminology, Concepts, and Models", October 2007.
[16] Microchip Inc., "MCP3004/3008: 2.7V 4-Channel/8-Channel 10-Bit A/D Converters with SPI Serial Interface", Data Sheet, 2008
[17] Simple Event Correlator Project, http://simple-evcorr.sourceforge.net/.
[18] Raspberry Pi Foundation, http://www.raspberrypi.org.
[19] H. Debar, D. Curry, B. Feinstein "RFC 4765: The intrusion detection message exchange format (IDMEF)", March 2007.
[20] R. Fielding, "Architectural Styles and the Design of Network-Based Software Architectures", Ph.D. Dissertation, Univ. of California, , 2000.
[21] J. Brown, B. Martin, "How fast is fast enough? Choosing between Xenomai and Linux for real-time applications", in Proc. of the 12th OSADL Real-Time Linux Workshop, 25-27, Nairobi, Kenya.
[22] LXC Project, "Linux Containers: Userspace tools for the Linux kernel containers", https://linuxcontainers.org.
[23] H. Philip, "Who Watches the Watcher?", August 2012, http://pi.gadgetoid.com/article/who-watches-the-watcher
[24] J. Ma, S. Perkins, "Time-series novelty detection using one-class support vector machines", in Proc. of the International Joint Conference on Neural Networks, pp. 1741-1745, July 2003.
[25] L. Maglaras, J. Jiang, T. Cruz, "Integrated OCSVM mechanism for intrusion detection in SCADA systems", IET Electronics Letters, vol. 50, pp. 1935-1936, 2014, doi: 10.1049/el.2014.2897.