Future Generation Computer Systems 91 (2019) 206–222

Contents lists available at ScienceDirect

Future Generation Computer Systems

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

Engineering security-aware control applications for data authentication in smart industrial cyber–physical systems

Béla Genge [a,*], Piroska Haller [a], Adrian-Vasile Duka [b]

[a] Department of Informatics, Petru Maior University of Tîrgu Mureş, N. Iorga, No. 1, Tîrgu Mureş, Mureş, 540088, Romania
[b] Department of Electrical Engineering and Computer Science, Petru Maior University of Tîrgu Mureş, N. Iorga, No. 1, Tîrgu Mureş, Mureş, 540088, Romania

ARTICLE INFO

Article history:
Received 20 February 2018
Accepted 2 September 2018
Available online xxxx

Keywords:
Industrial cyber–physical systems
Software architecture
Lightweight key generation
Data authenticity
Integrity
Hash chains

ABSTRACT

The massive proliferation of sophisticated technologies into the heart of traditional Industrial Control Systems has given birth to "smart Industrial Cyber–Physical Systems" (ICPS). While this industrial revolution has brought upon a wide range of advantages, it also raised new design challenges and exposed ICPS to a new breed of cyber–physical attacks. This paper aims to integrate security primitives (e.g., enforcing/verifying data authenticity) in control applications by formulating an innovative architectural paradigm shift. More specifically, our proposal is twofold. We elaborate a novel *security-aware* control application, which: (i) defines a new control application architecture embracing two security primitives that are called at the beginning and at the end of each program to verify and to enforce the required security properties; and (ii) runs the key management code as a separate program in order to isolate its implementation and to ensure its minimal interference with the rest of the programs. Then, we design a lightweight key distribution protocol exploiting the characteristics and computational advantages of symmetric key cryptography and hash functions. Extensive experimental results on a testbed replicating the precise hardware and software of a node from a Romanian gas transportation network, demonstrate the effectiveness of the proposed scheme and its applicability to resource-constrained ICPS.

1. Introduction

The massive proliferation of sophisticated technologies into the heart of traditional Industrial Control Systems (ICS) has given birth to a unique technological ecosystem. Nowadays, modern ICS encompass a variety of objects ranging from sensors and actuators, product tracking devices, industrial equipment, video surveillance cameras, to traditional computer systems (e.g., personal computers — PCs), and networking equipment. These "smart Industrial Cyber–Physical Systems" (ICPS) deliver advanced services and features, they improve operational benefits of control, reliability and safety, and facilitate the implementation of novel infrastructural paradigms (e.g., the Smart Grid).

While this industrial revolution has brought upon a wide range of advantages, it also raised new design challenges and exposed ICPS to a new breed of *cyber–physical* attacks where traditional cyber threats can result in the loss of vital services such as transportation, water, and gas supply. The recently reported cyber-attacks targeting the Ukrainian electricity grid [1,2] demonstrated the exceptional impact of cyber–physical attacks, where an ordinary malware infection may leave vast populated regions without electricity. Furthermore, they emphasized the need to develop effective protective strategies in existing critical industrial systems, where security measures need to be carefully designed such that the normal operation of industrial equipment is not affected.

In light of these issues, several international organizations including the National Institute of Standards and Technology (NIST) [3], the International Electrotechnical Commission (IEC) [4,5], and the American Gas Association (AGA) [6] have been involved in the definition of security standards. Recently, the Open Platform Communications (OPC) Foundation created the OPC Unified Architecture (OPC UA) [7]. Besides these, a plethora of solutions have been developed to enhance the architecture of ICPS [8–11], and to improve the security of industrial communication protocols [12,13].

Despite these efforts, we find that most previous approaches require major changes to the ICPS network's architecture, and/or significant changes to the underlying communication stack. On the other hand, as confirmed by related studies [14], the implementation of computation-intensive cryptographic algorithms in current control hardware does not constitute a practical solution. Furthermore, despite the technological progress and the roll-out of high-end devices with OPC UA support, most of the currently deployed high-end control hardware do not include security features (i.e., security protocols). As a result, we estimate that over the next 10 to 15 years a large number of industrial control devices with limited

[*] Corresponding author.
E-mail addresses: bela.genge@ing.upm.ro (B. Genge), phaller@upm.ro (P. Haller), adrian.duka@ing.upm.ro (A.-V. Duka).

https://doi.org/10.1016/j.future.2018.09.001
0167-739X/© 2018 Elsevier B.V. All rights reserved.

security features will still be available and operational in the field. Therefore, we believe that in order to enforce data authenticity in existing ICPS installations, a specially tailored software-oriented solution is needed.

This paper formulates a paradigm shift in terms of the positioning of security primitives (e.g., enforcing/verifying data authenticity). As such, it advocates that security constructions, if properly tailored, can be efficiently deployed in the application level of control hardware, and that existing applications can be enhanced to enforce fundamental security properties (e.g., authenticity and integrity) on the exchanged data. Furthermore, it demonstrates the feasible implementation of such a solution in a transparent and independent way from the underlying communication protocols, without demanding changes to the protocol stack, and to the ICPS infrastructure. More specifically, our proposal is twofold. First, based on a thorough analysis of existing control applications, we develop a novel *security-aware control application* (SACA). SACA entails two essential, yet practical, changes to the applications hosted by industrial controllers: (i) it defines two security primitives that are called at the beginning and at the end of each program to verify and to enforce the required security properties; and (ii) it defines the key management code as a separate program in order to isolate its implementation and to ensure its minimal interference with the rest of the programs. Second, we design a lightweight key distribution protocol exploiting the characteristics and computational advantages of symmetric key cryptography and hash functions.

The proposal is extensively evaluated in order to demonstrate its resilience to a wide variety of cyber attacks. Experimental results are conducted in the context of a real ICPS operating in a Romanian gas transportation network. To the best of our knowledge this is the first work that presents a comprehensive methodology for the integration of security solutions into control applications.

Throughout this paper we make the following major contributions:

1. We define a paradigm shift by designing a new security-aware control application architecture that embraces the traditional control application architecture and common industrial communication protocols.
2. We design and implement a lightweight key distribution protocol that addresses the limited computational resources of industrial controllers, while providing a lightweight structure that limits the impact of cryptographic computations on real-time applications.
3. We design a key masking scheme to ensure resilience against the compromise of long-term cryptographic keys.
4. We formulate intrinsic implementation details based on a case study involving an automation installation from the Romanian gas transportation network. These represent helpful insights that showcase the plausible integration of the proposal in industrial control systems.

The rest of this paper is organized as follows. Section 2 provides an introduction to the architecture of ICPS and a thorough analysis of related studies. The proposed architecture and key generation scheme are detailed in Section 3. Then, a security analysis showing the resilience of the proposal to various threats is presented in Section 4. The experimental results are provided in Section 5, and the conclusions are formulated in Section 6.

2. Background and related work

2.1. Overview of ICPS

The architecture of modern ICPS can be viewed as a unique technological ecosystem consisting of various devices ranging from sensors and actuators, industrial equipment, video surveillance cameras, to traditional PCs and networking devices. In terms of communications we find a broad range of technologies (traditional and industry-grade) including wired and wireless. At their core, ICPS include the Supervisory Control And Data Acquisition (SCADA) system, which embraces all the Information and Communication Technologies (ICT) hardware and software required to monitor the physical process and to implement control loops. Here, we find industrial controllers (e.g., Programmable Logic Controllers — PLCs), and Remote Terminal Units (RTUs) that read data from sensors and produce the local control strategy by issuing control signals to actuators. These controllers also handle the communication requirements of the industrial system, by exchanging data with other field devices, as well as, with the SCADA servers, and, eventually executing the received commands.

The communications between SCADA servers and PLCs is usually implemented in two ways: (i) through an OPC layer that helps map the PLC devices, program, and monitor the hardware controllers; and/or (ii) through a direct memory mapping notation, which makes use of communications protocols such as Modbus and DNP3.

The architecture of ICPS can be distributed across large geographical regions interconnected by state of the art Cloud-based solutions. To this end, there have been many recent developments and applications based on cloud-assisted industrial systems [15]. These can enable flexible processing of vast amounts of data, they can provide on-demand services and applications, with built-in support for redundancy and data availability. An example architecture of an ICPS is shown in Fig. 1.

2.2. Typical architecture of control applications

Considering that the main decisions pertaining to the design of our proposal are formulated by taking into account the particularities of control applications, we briefly revisit their software architecture. The main unit of execution within a controller (e.g., PLC) is the task. We mainly distinguish between two classes of tasks: system tasks and user tasks. System tasks are the ones that handle communications (e.g., the task running the communication with the OPC server/client), self-diagnosis operations, etc. User tasks are scheduled to run periodically or on an event base and trigger the execution of blocks of code, named “programs”. Each user task can be assigned a priority level and can run several programs. Data exchanges between end-points are based on *variables*. Typical data types include scalar variables (one, up to four bytes of data), alongside arrays, strings, structures, and user-defined data types. User-defined data structures are a common way to group data and once the data structure is defined, the variable instances can be selected for reading/writing via protocols such as the OPC.

At this point we need to make an important observation on the data exchanges performed through the OPC protocol. While this protocol is widely used in ICPS, it is aimed at facilitating rapid data sharing and access among the system’s nodes. As a result, once a variable is selected for reading/writing via OPC, the update and data sharing of this variable is handled independently by the underlying communication drivers and without any interactions with the programs running on the controller. Therefore, programs that share variables via OPC do not govern the actual data exchange and they do not have knowledge on the entity that performed the changes or the time at which the variable was modified. In case a different protocol is used for data transfer, such as Modbus/TCP, the variable-view is transformed to a memory register-map. In this case the copying of variables from/to Modbus packets is actually performed from within a dedicated program.