Security Standard Compliance Verification in System of Systems

Ani Bicaku, Mario Zsilak, Peter Theiler, Markus Tauber, and Jerker Delsing

***Abstract*—Standard compliance in system of systems (SoS) means complying with standards, laws, and regulations that apply to services from several sources and different levels. Compliance is a major challenge in many organizations because any violation will lead to financial penalties, lawsuits fines, or revocation of licenses to operate within specific industrial market. To support the business lifecycle, organizations also need to monitor the actual processes during run time and not only in their design time. Standard compliance verification is important in the lifecycle for reasons, such as detection of noncompliance as well as operational decisions of running processes. With the promotion of connectivity of systems, existing and new security standards can be employed but there are important aspects, such as technically measurable indicators, in the standards and automation of compliance verification that need to be addressed. This article presents an automated and continuous standard compliance verification framework used to check devices, systems, and services for standard compliance during secure onboarding and run time. In addition, a case study for the Eclipse Arrowhead framework is used to demonstrate the functionality of the standard compliance verification in SoS.**

***Index Terms*—Automation, security, service-oriented architecture (SoA), standard compliance, standard verification, system of systems (SoS).**

I. Introduction

RECENT developments in technology and innovation are revolutionizing the industrial environment by preparing strong foundation for new opportunities and promoting greater efficiency toward the fourth Industrial Revolution (Industry 4.0), which is the ongoing automation of the traditional manufacturing using modern smart technologies. The ability of traditional systems to cope with new technologies will be a significant factor to success in the business [1]. Organizations have to decide if they should migrate into Industry 4.0 and which technology is appropriate by considering initial costs and benefits on productivity. Most of the industrial organizations are addressing this by adopting system of systems (SoS) technologies to ensure interoperability of products regardless of the manufacturer, location, operating system, or hosted services. SoS are large-scale integrated systems, which are independently operable on their own, but are networked together to achieve a higher goal [2]. Since they are a key technology in the digitalization of industrial environments and operate in legacy and new devices, security is a major challenge.

The migration of industrial control systems (ICSs) from isolated to remotely accessible is motivated by the need to enable remote maintenance, control, firmware update, data collection, and new sources of data to improve automation [1], [3]. However, the intermix of devices, systems, and services in industrial environments makes it more difficult to maintain security, including confidentiality, integrity, and availability [4]. Several requirements need to be fulfilled, such as integration (easy integration of devices), heterogeneity (different devices/services from different vendors), interoperability (interconnect services through several interfaces and platforms), and accessibility (easy access services everywhere at any time) to have automated systems [5]. These issues have been studied and addressed by the Eclipse Arrowhead framework [6] during Arrowhead[1] project, and extended with new features and systems in the Arrowhead Tools[2] project.

Securing access to information is crucial for any organization. It is important not only to secure the exchange data but also to identify who is providing and accessing the data without reducing the security level. This helps organizations to increase trust and avoid cost of security breaches by assuring the customer that they have identified and measured their security risks [7]. There are several security standards that organizations can choose to comply with (e.g., ISO 27000 series, NIST Special Publications, ISA/IEC 62443 series, etc.). Being compliant to these standards can support ICS to migrate from traditional control systems to Industry 4.0 applications, by providing standardized services and interfaces for legacy systems and devices. Moreover, ISA/IEC 62443 series are developed by both, ISA99 and IEC committees to improve the security of components and systems used in industrial automation and control systems (IACSs). The ISA/IEC 62443 series can be utilized across industrial systems, are approved by many countries, and are key standards in the

Manuscript received September 28, 2020; revised January 8, 2021; accepted February 27, 2021. Date of publication April 14, 2021; date of current version June 13, 2022. This work was supported in part by the European Commission through the European H2020 research and innovation program, in part by ECSEL Joint Undertaking, and in part by the National Funding Authorities from 18 involved countries under the research project Arrowhead Tools under Grant 826452. *(Corresponding author: Ani Bicaku.)*

Ani Bicaku is with the University of Applied Sciences Burgenland, 7000 Eisenstadt, Austria and also with the Embedded Intelligent Systems Lab., Luleå University of Technology, 971 87 Luleå, Sweden (e-mail: ani.bicaku@outlook.com).

Mario Zsilak, Peter Theiler, and Markus Tauber are with the University of Applied Sciences Burgenland, 7000 Eisenstadt, Austria (e-mail: mario.zsilak@forschung-burgenland.at; peter.theiler@fh-burgenland.at; markus.tauber@fh-burgenland.at).

Jerker Delsing is with the Embedded Intelligent Systems Lab., Luleå University of Technology, 971 87 Luleå, Sweden (e-mail: jerker.delsing@ltu.se).

Digital Object Identifier 10.1109/JSYST.2021.3064196

[1] www.arrowheadproject.eu

[2] www.arrowhead.eu/arrowheadtools

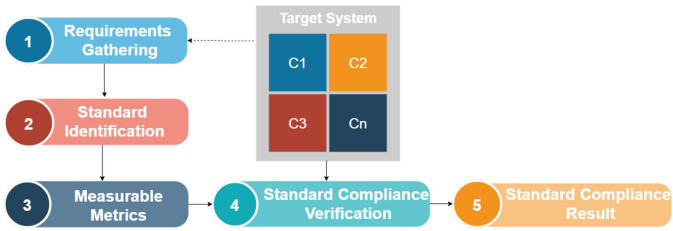


Fig. 1. High level view of the standard compliance verification process.

industry. Standard compliance verification helps to perform root-cause analysis and to understand the reason for specific noncompliance problems. It allows the organization to improve its responsiveness and level of compliance, but also verification of compliance against investigations of security-related events.

Standard compliance is even more critical for SoS due to their evolving nature. In order to take advantage of the SoS technology, applications should be adapted to several security standards. Verifying standard compliance includes checking the system during its onboarding and operation time against internal and external regulatory acts.

As mentioned earlier, several challenges need to be addressed to be compliant to security standards in SoS. If a single system does not provide the required security level, the entire SoS is vulnerable. This can be avoided if evidence of security standard compliance is shown. Since they evolve during time, standard compliance should be continuously verified in different time frames or when a component is added/removed from the SoS. This article addresses the integration of standard compliance verification in a dynamic and evolving SoS environment. An important component to show the standard compliance is by using a framework to support their verification. Complexity is increased in SoS, where applications from several vendors are intended for different levels of compliance. Although the development of SoS has gained popularity in recent years, more work to address security standard compliance is still needed, as in the current status, it is difficult to provide indicators for the level of compliance in organizations.

To address this, we extend our previous work [8]–[10] in continuous standard compliance verification and propose a service-oriented architecture (SoA) approach that can verify the standard compliance of a device, system, and service in the Eclipse Arrowhead local cloud. This framework named monitoring and standard compliance verification (MSCV) is designed as a measurement tool for assessing security compliance toward security standards, as shown in Fig. 1.

In order to show the fulfillment of standard constrains, the first step is to gather the requirements from the target system that can be a single component, a number of components, or distributed systems. After defining the requirements, specific machine readable standards will be identified and measurable metrics will be extracted. The MSCV will monitor and verify standard compliance from the components (C1–C*n*) of the target system in order to provide the compliance result.

To show this, we propose integrating the MSCV during the Eclipse Arrowhead secure onboarding procedure [11], to check for standard compliance verification when a new device interacts with the Arrowhead local cloud for the first time. Also, we use the MSCV to check for standard compliance verification during operation time of devices, systems, and services within the local cloud at any point of time.

The remainder of this article is structured as follows. In Section II, we provide an overview of similar approaches. Section III describes the MSCV approach and Section IV provides the integration in Arrowhead framework. In Sections V and VI, the application of MSCV in a realistic scenario is shown and Section VII concludes this article and discuss our future work.

II. Related Work

Advances in technology are revolutionizing the architecture of ICS by making them more global and connected. This leads to new security challenges and targeted attacks. Different standards and best practice guidelines have been proposed for securing them. However, their compliance to the standards should be verified to check if these systems are capable to securely interoperate with other components. In order to have a general overview of security standard compliance verification, we show an evaluation of existing literature regarding standard compliance verification.

Recently, many standard compliance approaches have been proposed that utilize a broad range of technologies and techniques to realize compliance monitoring solutions [12], [13]. We have evaluated compliance monitoring approaches, which mainly address the compliance during execution. Furthermore, the selected approaches are evaluated based on the concept details and documentation provided in the publication.

Ziegler *et al.* [14] discussed standard compliance as part of the ANASTACIA research project, which tries to develop a trustworthy, autonomic security framework that implements trust and security by design and optimizes security in all phases. Some components are designed to involve security and privacy standards and include real-time monitoring for assessing privacy risks and security aspects, but it lacks in technical applicability and representative metrics for monitoring.

Choi *et al.* [15] also did a study, where they proposed a system hardening and security monitoring scheme with the suitable technology in order to mitigate security vulnerabilities with implemented security features on Internet of Things (IoT) systems. This work also deals with a practical prototype, which checks for possible malware on the devices, but explores security according to a RedHat checklist and not relevant IoT standards.

Matheu *et al.* [16] presented an architectural framework, which includes concepts and processes related to security assessment and testing methodologies. The goal of his work was to show a certification approach for IoT security with risk assessment, testing, and label the security of an IoT system with a chosen configuration and context. However, this work does not propose any actions to monitor the environments for security compliance based on standards.

IoT Security Foundation [17] provided an IoT security compliance framework, which consisted of a checklist and questionnaire, with which different stakeholders in organizations can

assess and categorize the security processes directly regarding IoT or related to it, for example, the supply chain. Unfortunately, it lacks any guidance as how the checked security properties can be implemented and how the compliance aspects can be measured or monitored in real time.

Russell and Van Duren [18] had a wider approach to the topic and released a security guide called "practical IoT security" in 2016, where they talked about securely building and deploying systems in an IoT-connected environment. In it, they talk about IoT in general, security threats, countermeasures, and about relevant cybersecurity standards important to IoT compliance monitoring, but it has a very abstract view on the topic without relevant technical examples. Thus, the practicability and concrete measures for IoT security compliance monitoring are not covered in this article.

There are other article considering SoA standards and standard compliance, such as [19]–[22], but they mostly check the standards or describe the need for compliance to the standards, and none of them provides a possible way how to make these systems compliant to specific standards.

The focus of this article is to address automated and continuous standard compliance verification in a dynamic and evolving SoS. To show the results, a use case during the onboard and runtime of a device, system, and service in the Eclipse Arrowhead framework is used.

III. Monitoring and Standard Compliance Verification

The need for standard compliance verification in SoA should emerge during all the lifecylce phases (from specification to decommissioning). In the specification and design phase, the compliance with a set of requirements should be verified. At runtime, the activities of the process, including all other interacting components, should be checked if they fulfill the standard requirements because the runtime phase can last for many years and decades in case of ICSs. The decommissioning phase includes the activities of taking out the components from the process and should be based on a specific standard by fulfilling all the specific requirements for this phase. The requirements specified in each phase will be translated in measurable indicator points (MIPs) employed by the MSCV in order to provide an aggregated result based on the standard, requirements, and the monitored MIPs. If these MIPs are fulfilled and the standard compliance evaluation has an accepted level for the involved stakeholders, they can communicate and share data between each other. Based on the organizations and the specific standards, compliance verification is different within the organizations and even across the systems within the same organization (different systems have different purposes). The security standards are platform independent and the MIPs extracted from the standards should be platform independent, meaning that they can be reused from other technologies without affecting the underlying infrastructure, e.g., identification and authentication gives the capability to identify and authenticate all involved users, including devices, systems, services, and humans [23]. However, the MIPs listed in most standards are not technically implementable or in some cases are not available in all platforms. It is very important to define the right MIPs in order to fulfill the standard and these MIPs should be maintained. In order to have a comprehensive list of MIPs, they should be defined in clear and accurate way taking into account also the environmental changes. They should be aggregated by a framework in order to provide the compliance verification and summarize the details for each compliance check. In general, standard compliance verification deals with the problem of understanding whether services, systems, and devices represent behaviors that are aligned with the controls defined by compliance rules (such as best practice guidelines and national or international standards).

The MSCV as a service presented here is an extension of our previous work [8], [9], [11] used for an industrial IoT use case. The architecture of MSCV is a composition of different components combined in three core parts: (i) monitoring agents used to gather data from the target system, (ii) evidence gathering mechanism (EGM) used to acquire, store and analyze security, safety and legal related evidence, and (iii) compliance module, which receives data about the monitored metrics from EGM and assigns a weight value to each metric for the compliance calculation. The MSCV framework shows the standard compliance verification for a single standard or a number of standards. Also, it gives the possibility to classify the results (security, safety, organizational, etc.) based on the monitored standards. Another feature of the framework is compliance for a single component or the entire target system [10].

The aim of this article is to further extend the MSCV for SoS and implement it in the Eclipse Arrowhead framework (explained in Section IV) to show the standard compliance of devices, systems, and services during the onboarding procedure and operation time. Fig. 2 illustrates a simplified version of the MSCV divided in three phases (standards, monitoring, and results). Here, we introduce the standard verification process, whereas in Section VI, we show its usability in a use case.

A. Phase I: Standards

- *Requirements*: In its everyday business, an organization is subject to a variety of different regulations, laws, standardization bodies, etc. It is up to the organization to understand the compliance rules that directly affect its business, thus producing what we call requirements. They will be taken in consideration to choose the appropriate standard for extracting measurable metrics and check for compliance.
- *Standard Identification:* After gathering the requirements, the next step is to evaluate machine readable standards. The evaluation is based on the domain (security, safety, organizational, legal, etc.).
- *Measurable Metrics*: The chosen standard will be further evaluated to extract measurable indicators. These indicators would be used to check how compliant the components or the system is with the specific standard.

B. Phase II: Monitoring

1) *Measurable Metrics Monitoring:* The monitoring of the identified measurable metrics is possible via pluggable monitoring agents from different monitoring tool plugins

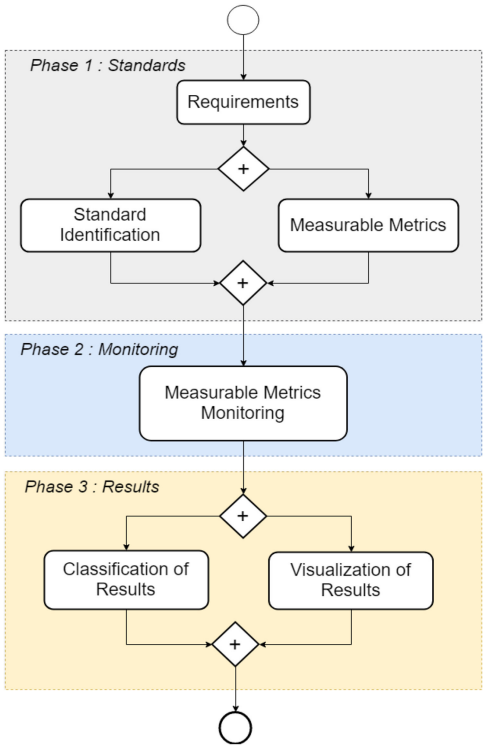


Fig. 2. Standard compliance verification process in three phases: standards, monitoring, and results.

(e.g., Nagios plugin, Ceilometer plugin, Zabbix plugin, etc.) and customized scripts.

C. Phase III: Results

- *Standard Compliance Results Classification*: The compliance verification will be calculated based on the algorithm presented in our previous work [9] and the results will be classified in security, safety, organizational, etc., standard compliance based on the evaluated standard.
- *Standard Compliance Result Visualization*: The MSCV provides the compliance results for one or several standards, also for a single component or the entire target system.

IV. Eclipse Arrowhead Framework

The Eclipse Arrowhead framework makes use of the SoA principles with the aim to create local automation clouds. The objective of the framework is to provide interoperability, real-time performance, scalability, and secure communication through multicloud interaction [6], [24], [25]. The architecture is build on the SoA fundamentals, such as the following:

- *Loose coupling*: (the property that makes possible to implement services in several devices and they are not supervised services);
- *Late binding*: (the property of connecting to the known resource at a specific time);
- *Lookup*: (the property that makes possible to publish and register services, but also to use already existing services).

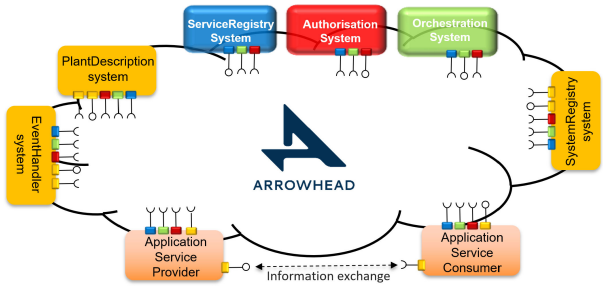


Fig. 3. Arrowhead framework architecture, where a service is what used to exchange information from a providing system to a consuming system.

The architecture addresses the move from large monolithic organizations toward multistakeholder cooperations, thus addressing the high-level requirements in today's society, such as sustainability, flexibility, efficiency, and competitiveness [6]. In order to develop and maintain Eclipse Arrowhead compliant devices, systems, and services, it is important to know their definition based on the Eclipse Arrowhead framework.

A *device* is a piece of hardware equipment with computational and communication capabilities. The device can dynamically host one or several systems, including their running services. All Eclipse Arrowhead compliant devices should be registered within the DeviceRegistry [11].

A *system* is the software system that is hosting and/or consuming one or multiple services. An Eclipse Arrowhead system can be the provider of one or several services and at the same time, it can consume one or several services. All systems should be registered within the SystemRegistry [11].

A *service* is the information exchanged with a providing system to a consuming system. It is produced by an Eclipse Arrowhead compliant system, can have metadata, and should support functional and nonfunctional requirements [6].

A *local-cloud* is a self-contained network containing the mandatory core systems and at least one application system deployed [6].

An *SoS* within Eclipse Arrowhead is a set of systems, which are administrated by the Eclipse Arrowhead mandatory core systems and exchange information by means of services [6]. Therefore, a local cloud is considered as SoS within the Eclipse Arrowhead framework.

A. Eclipse Arrowhead Framework Systems and Services

The Eclipse Arrowhead framework architecture is composed of a number of systems, classified in: mandatory core systems, automation support core systems, and application systems, as shown in Fig. 3.

ServiceRegistry System: color coded in blue, provides the database of all running and registered services with the Eclipse Arrowhead local cloud. It makes possible to register a service in the ServiceRegistry, and to lookup for existing services.

Authorization System: color coded in red, provides authentication, authorization, and optionally accountability. It defines and provides the rules for consumption of services. Based on these rules, a specific device, system, or service is allowed or not to consume other services registered in the local cloud.

Orchestration System: color coded in green, provides the necessary mechanisms for distributing orchestration rules and endpoints to dynamically allow the consumption of existing services and systems to create new functionalities.

Automation Support Core Systems: such as SystemRegistry, DeviceRegistry [11], PlantDescription [26], Event Handler [24], etc., are used to facilitate automation application design, engineering, and operation. These systems are used to provide support for automation, security, service exchange, interoperability, and many other features of the local cloud.

Application Systems: are the systems aiming to consume the registered services in the ServiceRegistry or providing new services for the local cloud. In order to build an Eclipse Arrowhead compliant local cloud, it is essential to consume the three mandatory core systems and at least consuming or producing one application system.

To assure that the Eclipse Arrowhead local cloud is not compromised when a new device is interacting, an onboarding procedure is needed. Therefore, we have designed and implemented the Eclipse Arrowhead secure onboarding procedure [11]. In order to verify that new devices, systems, and services interacting with the Arrowhead local cloud are compliant to specific standards in terms of security, safety, organizational, legal, etc., we provide a new system named MSCV explained in the next section. This system provides standard compliance verification for a single standard or a number of standards, and the result can be shown for a single components or several components.

B. MSCV System in Eclipse Arrowhead Framework

The MSCV system monitors and verifies if the new device, system, and service, which is interacting with Eclipse Arrowhead framework, fulfill the requirements of a specific standard. The standard compliance is defined based on given sets of requirements, which are selected specific standards and derived MIPs. Those reflect configurations of systems that help to demonstrate the state of compliance [10]. The MSCV system will perform compliance verification and this result will decide if the device, system, and service can continue with the onboarding procedure in order to register devices in DeviceRegistry, systems in SystemRegistry, and services in ServiceRegistry [11]. The MSCV system is one of the support core systems of the Eclipse Arrowhead framework.

The MSCV system produce the StandardComplianceVerification service and consumes the mandatory core services and the OnboardingProcedure service (see Fig. 4).

Once the MSCV has been implemented, devices, systems, and services are monitored for standard compliance verification and effectiveness. The MSCV will monitor the necessary configurations and logs regarding the security metrics defined by the standards (e.g., secure boot, strong password, multifactor authentication, etc.). The system should capture this data at all the levels, so that it can be used to provide results to other Eclipse Arrowhead systems (e.g., OnboardingProcedure, Orchestration, EventHandler, etc.) regarding standard compliance. The MSCV can be used during the Eclipse Arrowhead secure onboarding procedure and during operation time.

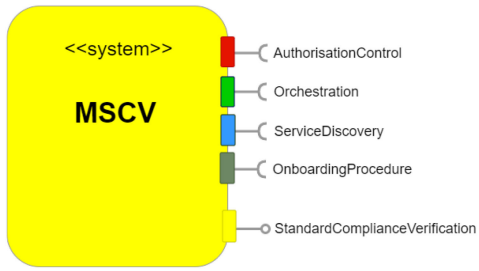


Fig. 4. MSCV system that produces the StandardComplianceVerification service and consumes AuthorizationControl, Orchestration, ServiceDiscovery, and OnboardingProcedure services.

During operation time, the execution of the standard compliance verification is triggered by the Orchestrator and the results from the MSCV will decide if the specific services will be allowed to interact with the requested service. Also, it can be triggered by the EventHandler to verify standard compliance whenever is needed by the local cloud.

V. Measurable Security Indicators (MSIs)

In order to show security standard compliance, security requirements for such systems should be addressed. In this article, we have addressed the security standard compliance verification of devices, systems, and services interacting with Eclipse Arrowhead local cloud. Therefore, several standards are evaluated based on the capability to provide technical and machine readable metrics for secure ICSs.

ISO/IEC 27001 is a standard for information security management that defines requirements for establishing, implementing, operating, monitoring, reviewing, maintaining, and improving a documented information security management system [27]. It describes different areas for security controls, such as asset management, access control, and compliance, but these controls also cover many different organization concerns, which are not of technical nature.

A more specific standard is *ISO/IEC 27032* with the goal to improve the state of cybersecurity, and its dependencies on other security domains [28]. The implementation of best practices in form of metrics would check the overall status of security, but these would cover rather generic security aspects. Moreover, it includes topics as assets or threats and roles of stakeholders, which are not in the scope of metrics construction for this article. For this reason, metrics could be easily derived, but these would be not specific enough to harden system security and many parts of this standard would not be useful for finding security compliance metrics.

Another security standard is *IEC 62351* addressing data and communication security for management of power systems and associated information exchange in smart grids and defines security for communication protocols from other standards [29]. The high-level guidance from the standard allows to derive several security metrics, but the information for these metrics would only consider smart grids and not control systems in general. These metrics could be interpreted in multiple ways and the controls would only partly cover the ICS.

Another option for security metrics derivation is *NIST Special Publication 800-82*. It addresses security guidance for ICS, such as "supervisory control and data acquisition systems, distributed control systems, and other control system configurations" [30]. The standard has a specific section for ICS security controls, which could be translated into metrics, but it also considers topics without technical nature, such as awareness and training of personnel. Additionally, more specific guidance for the controls is not contained in this publication and are only covered in other NIST publications. Security metrics fitting for the practical use case would be possible, but not all controls could be technically verified, and they could only be checked on a superficial level with the provided information.

ISA/IEC 62443 series are standards that provide guidance for implementing secure IACS. More specific two standards from the series can be used to extract MSIs, IEC 62443-3-3 "system security requirements and security levels," and IEC 62443-4-2 "technical security requirements for IACS components." In these standards, the MSIs are categorized in seven foundational requirements (FRs), which are as follows.

- IAC—Identification and authentication control (13 MSI).
- UC—Use control (12 MSI).
- SI—System integrity (9 MSI).
- DC—Data confidentiality (3 MSI).
- RDF—Restricted data flow (4 MSI).
- TRE—Timely response to events (2 MSI).
- RA—Resource availability (8 MSI).

They provide security requirements and security levels that can be implemented in multiple ways [31]. All metrics with the same topic are summarized in categories and are further enhanced with several guidelines. Also, ICS networks are mostly built upon CPS and IoT solutions, which makes this standard even more suitable.

In order to verify security standard compliance, we use a security metric model documented in [10]. The structure of these MSI starts with category names in form of headers, which are extracted one-to-one from the requirements and are also abbreviated for using them as identifiers for each MSI. The MSIs are listed in form of a table that begins with the identifier and metric name. The next two lines continue with the MSI definition, rephrased from the section of each respective category for better understanding, and the high-level explanation how the MSI can be monitored with generic approaches. Last, it is explained in technical terms how it can be monitored with concrete methods on the device, system, and service level. The extracted MSIs (51 MSIs) from ISA/IEC 62443 series are documented in Github.[3] The security standard from where the MSIs are extracted do not provide technical implementation details or monitoring possibilities for device, system, and service layer. A detailed MSI documentation is necessary to understand and monitor each metric. The IAC security metrics are presented in Table I, where for each measurable metric, we provide: (i) ID, (ii) metric name, (iii) definition, (iv) monitoring possibility, and (v) implementation in device, system and service level. Following

[3]https://github.com/mzsilak/core-java-spring/tree/mscv

TABLE I
LIST OF MSIS FOR IAC EXTRACTED FROM IEC 62443-3-3

IAC - Identification and Authentication Control	
ID	*Measurable Metric*
IAC -1	Human user identification and authentication
IAC -2	Software process and device identification
IAC -3	Account management
IAC -4	Identifier management
IAC -5	Authenticator management
IAC -6	Wireless access management
IAC -7	Strength of password-based authentication
IAC -8	Public key infrastructure certificates
IAC -9	Strength of public key authentication
IAC -10	Authenticator feedback
IAC -11	Unsuccessful login attempts
IAC -12	System user notification
IAC -13	Access via untrusted networks

```
# Test: CUST-0010
    # Description : Search for configured device multi-factor authenticator solution
    Register --test-no CUST-0010 --weight L --network NO --category security --description
     "Searching device multi-factor authenticator"

    if [ ${SKIPTEST} -eq 0 ]; then
        LogText "Test: Searching device multi-factor authentication"
          FIND=$(${GREPBINARY} -i "auth required pam_google_authenticator.so"
            /etc/pam.d/common-auth | ${GREPBINARY} -v ^#)

        if [ "${FIND}" != "" ]; then
            LogText "Result: device multi-factor authentication seems to be configured"
            Display --indent 2 --text "+ Verifying device multi-factor authentication"
              --result "${STATUS_OK}" --color GREEN
         AddHP 1 1

        else
            LogText "Result: device multi-factor authentication seems to be missing"
            Display --indent 2 --text "+ Verifying device multi-factor authentication"
              --result "${STATUS_NOT_FOUND}" --color YELLOW
         AddHP 0 1
        fi
    fi
```

Fig. 5. Customized script for IAC-1.

is shown the IAC-1 documentation. The same documentation is produced for all 51 MSIs extracted from this standard.

- [*ID*]: IAC-1.
- [*Name*]: Human user identification and authentication.
- [*Definition*]: This security metric supports IAC by assuring that human users can be identified and authenticated on all interfaces that allow human access.
- [*Monitoring*]: This metric be monitored by assuring that there are accounts with passwords or other authentication means configured for the component under consideration.
- [*Implementation*]:
  - *Device*: For monitoring this property, a script can be used to check if there is a configured multifactor authenticator solution with device means installed.
  - *System*: For monitoring this layer, scripts can be used to look for user accounts with passwords configured in configuration files, e.g., $/etc/passwd$ and $/etc/ssh/sshd\_config$.
  - *Service*: For monitoring this layer, scripts can be used to look for user accounts with passwords configured in configuration files, such as $/etc/passwd$ and $/etc/ssh/sshd\_config$ or custom configuration files.
- [*Script*]. Fig.5 shows a customized script.

Fig. 6 shows an example of the MSCV normalized database, which is the structural relationship organized in columns

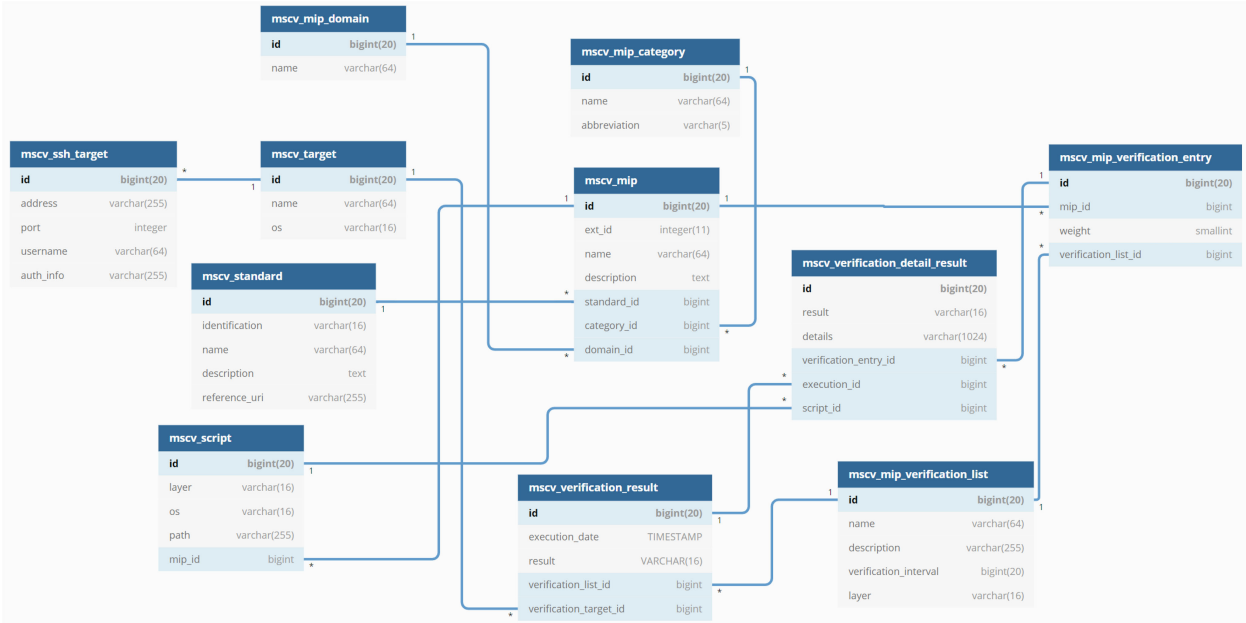


Fig. 6. MSCV normalized database.

(attributes) and tables (relations) to show their dependencies. The normalized database is part of the EGM used for the classification of the domains (security, safety, etc.) and the specific MSIs, measurable safety indicators based on the evaluated standard. The mscv_mip_verification_entry and mscv_mip_verification_list are more dynamic data. The mscv_mip_verification_entry allows us to configure a weight to an MIP, whereas the mscv_mip_verification_list is a list of entries to use for a verification run. This separation from an MIP itself allows to configure different subsets of metrics or different weights for MIPs for different (physical or logical) targets, different use cases, or even different organizations. It is possible to see the weight value for each MIP based on the requirements from the organization and the importance of each MIP. The mscv_verification_result is the execution of all script against a specific target and the mscv_mip_verification_entry would represent every single metric per layer. With 51 metrics in 3 layers, we would get 153 entries per run. The mscv_mip_category represent the seven categories (and categories of other standards), e.g., "IAC". That means, if we load the controls of IEC 62443-3-3, we would have only seven rows in this table. The mscv_mip_domain represent the domain (safety, security, organizational, etc.). Hence, the domain would be on every MIP and not on the standard itself. For Standard IEC 62443-3-3, we only have one row: Security. The MIP itself contains all metrics. Total of 51 rows for IEC 62443-3-3 standard.

VI. Security Standard Compliance Verification During Onboarding and Operation Time

This section presents the use case scenario to show the functionality and usage of the MSCV system. Fig. 7 shows a new device interacting with the Eclipse Arrowhead local cloud for the first time. As described in Section IV, the objective of

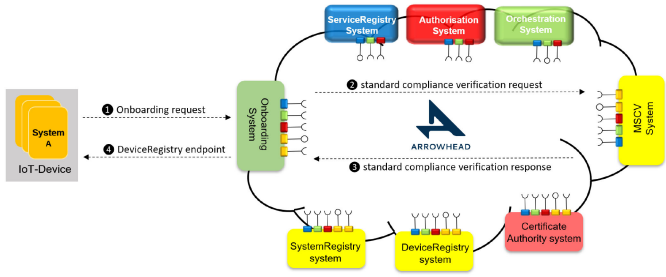


Fig. 7. Arrowhead framework onboarding use case: A new device is interacting for the first time with the local cloud and wants to register the device, systems, and services in the Arrowhead framework.

the framework is to efficiently support the development and operation of interconnected systems. It is based in SoA principle and its elements are systems that provide and consume services by cooperating as SoS. In order to consume/produce Arrowhead services, the following should be securely onboarded and registered: (i) the hardware device, (ii) the software systems running on the device, and (iii) the services running on top of the systems. The onboarding procedure is described step-by-step in [11], without the MSCV system.

The secure onboarding procedure already provides security features during the onboarding process. Without the manufacturer certificate, the device cannot be registered in the DeviceRegistry, without the device certificate, the system cannot be registered to SystemRegistry, and without the system certificate, the service cannot be registered to ServiceRegistry. This provides a chain of trust (device-system-running services) within the local cloud and the certificates are validated by the CertificateAuthority system. After the successful onboard, in order to consume other services, the device, system, or service should contact the Orchestrator system. The Orchestrator system

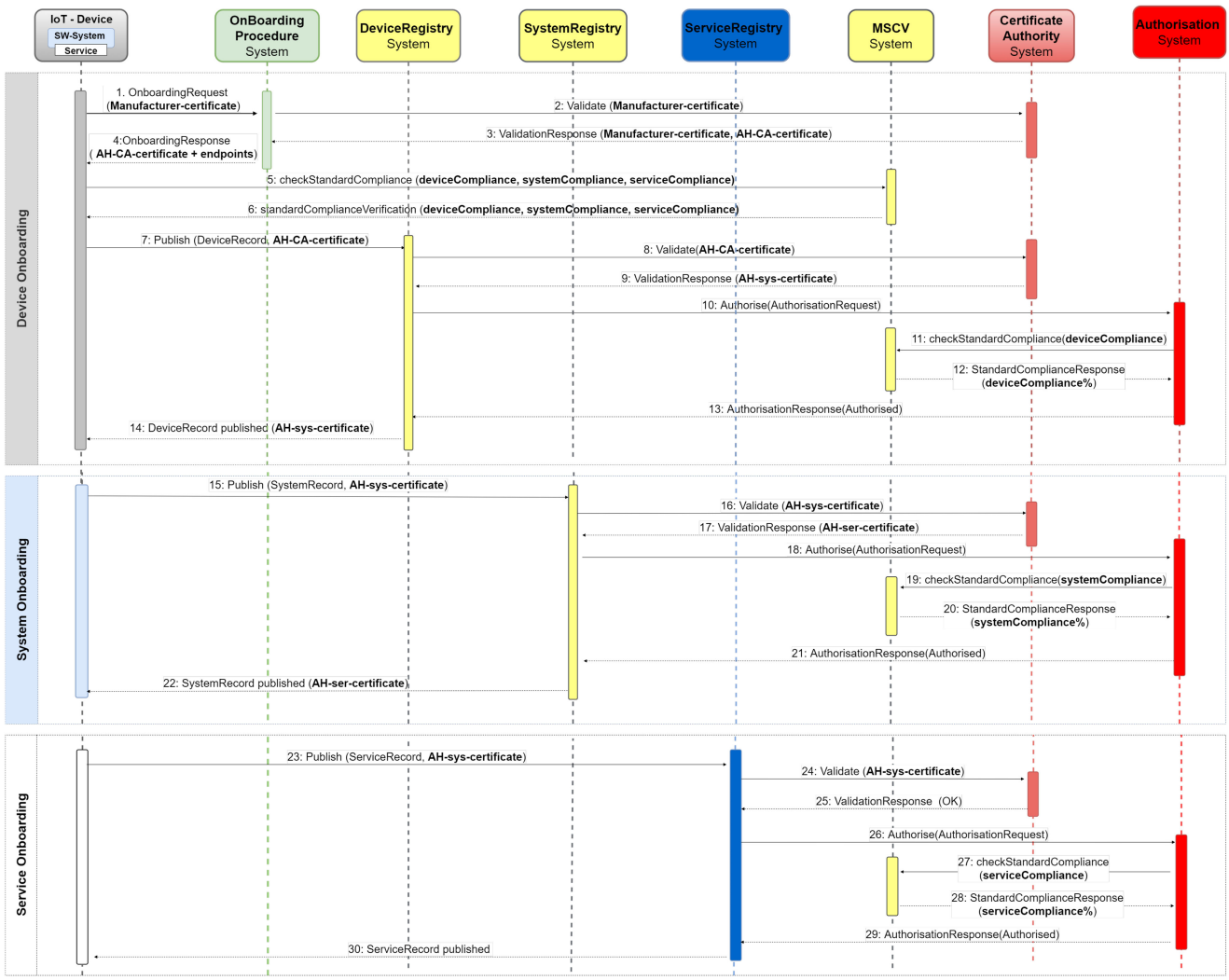


Fig. 8. Secure onboarding and MSCV sequence diagram.

will contact the Authorization system and if authorized will provide back the endpoint of the requested service [11].

MSCV will enhance the security of Eclipse Arrowhead framework during the onboarding procedure by checking if the device, system, and service that wants to register to the local cloud is compliant to a specific security standard (in this example to ISA/IEC 62443 series). The MSCV can be used to check for standard compliance for other domains, such as safety and organizational standards [8], [10] and any other domain if the metrics can be technically implemented.

Fig. 8 shows the step-by-step secure onboarding procedure, including the MSCV in a sequence diagram. The procedure is divided in three main phases: device (i) device onboarding, (ii) system onboarding, and (iii) service onboarding. Each phase includes the step-by-step interaction of every Arrowhead system. In order to provide standard compliance verification, the OnboardingProcedure provides the endpoint of the MSCV to the new device that shall checkStandardCompliance (step 5) after the manufacturer certificate is validated from the CertificateAuthority (steps 2 and 3). Before registering the new device to the DeviceRegistry, the DeviceRegistry shall check for authorization with Authorization system (step 10) and the Authorization system shall checkStandardCompliance (step 11). If the device is authorized and if the standard compliance is fulfilled, the device can be registered to the DeviceRegistry system. The same procedure is for registering the new system to SystemRegistry and the new services to ServiceRegistry.

In case the device, system, or service do not fulfill the onboarding conditions and the compliance to the standard the process stops, and the device, system, and service will not be onboarded. If every step of the secure onboarding procedure has succeeded and the MSIs from the standards are fulfilled, the device gets integrated into the local cloud and can consume the registered services within the Arrowhead framework.

During run time, the MSCV system includes two use cases. Run-time standard compliance verification using Orchestrator is shown in Fig. 9. After successfully onboarded to the Arrowhead framework, the new device, system, or service wants to interact with other services within the local cloud. To provide the endpoints of the requested service, the Orchestrator system should check with the Authorization system, which will request

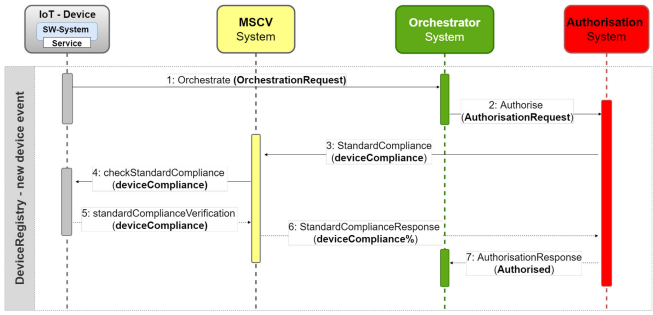


Fig. 9. MSCV and Orchestrator sequence diagram.

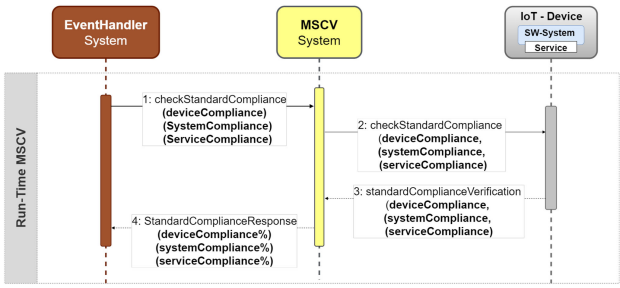


Fig. 10. MSCV and Eventhandler sequence diagram.

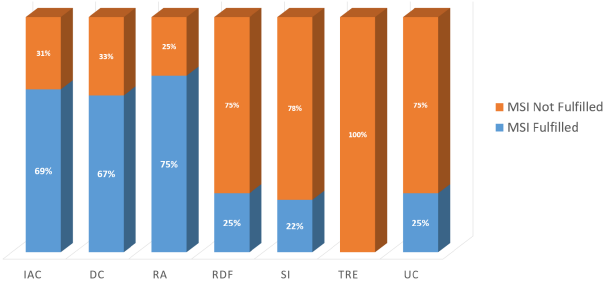


Fig. 11. Results of the standard compliance verification for the device layer based on ISA/IEC 62443 FRs.

the standard compliance verification from the MSCV system and if it is compliant will authorize the Orchestrator to provide the requested endpoints.

Run-time standard compliance verification using Event Handler. Fig. 10 shows the interaction with EventHandler. The EventHandler system sends notifications concerning events and monitors the events over the local cloud. Whenever, it needs to check for standard compliance verification, it will send a checkStandardCompliance request to the MSCV system regarding a device, system, or services and the MSCV will perform a standard compliance verification of the device, system, or service and will provide the result. Following are provided results of the MSCV for standard compliance verification considering ISA/IEC 62443 standards. The results are classified based on the seven FRs of the series to show the device, system, and service compliance. MSCV results for the new device registered in the Arrowhead local cloud during the onboarding procedure are shown in Fig. 11. MSCV results for the new system registered in the Arrowhead local cloud during the onboarding procedure are

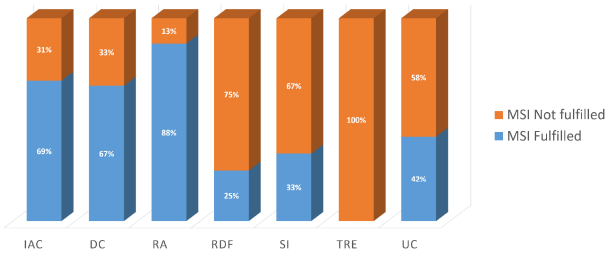


Fig. 12. Results of the standard compliance verification for the system layer based on ISA/IEC 62443 FRs.

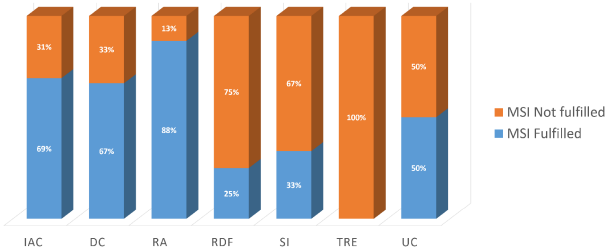


Fig. 13. Results of the standard compliance verification for the service layer based on ISA/IEC 62443 FRs.

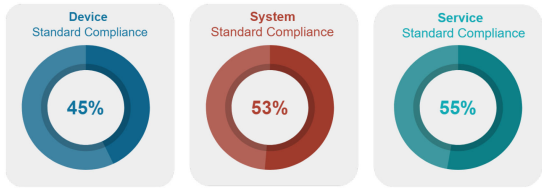


Fig. 14. Overall standard compliance for device, system, and service.

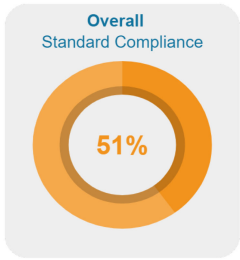


Fig. 15. Overall standard compliance.

shown in Fig. 12. MSCV results for the new service registered in the Arrowhead local cloud during the onboarding procedure are shown in Fig 13.

The results shown in the aforementioned figures (Figs. 11–13) can be extracted from the MSCV and they are shown for the scope of this article to provide more information, but these details are not provided during the secure onboarding procedure.

The MSCV in Arrowhead provides only the compliance percentage shown in Fig. 14 for device, system, and service.

Also, the MSCV has the possibility to calculate the overall compliance to the security standard for device, systems, and running services during the onboarding procedure or during run time, as shown in Fig. 15. This result can be used for SoS communication, where Arrowhead local clouds can communicate with each other only if they are standard compliant.

VII. Conclusion

Standard compliance verification in SoA is necessary to support interoperability between different services. If SoA services are compliant to security standards, they can provide trust to other stakeholders. This article presented the MSCV as a service for SoA implemented in the Eclipse Arrowhead framework. The MSCV system integrated in the Arrowhead local cloud produces one service, the StandardComplianceVerification service, and consumes the three mandatory core systems of the Arrowhead framework (ServiceRegistry, Orchestration, and Authorization system) and the OnboardingProcedure system.

The MSCV consists of three main phases: (i) standards (including requirements and measurable metrics), (ii) monitoring (including the monitoring agents for each metric), and (iii) results (including classification of results and visualization of the standard compliance). The MSCV can be used to check for compliance other domains (e.g., safety, organizational, etc.) if their standards can provide technically implementable MIPs. The MSCV shown in this article is implemented as part of the secure onboarding procedure in the Arrowhead local cloud and provides standard compliance verification for devices, systems, and services that want to interact with the local cloud for the first time. Also, MSCV can be used for run-time standard compliance by the Orchestrator system or the EventHandler system. The overall standard compliance result can be used in SoS communication to make possible that different Arrowhead local cloud communicate in a secure and standard compliant manner.

To provide continuous and automated standard compliance verification, a major challenge is to select relevant standards with technically implementable metrics. After the identification of the security standard, the next challenge is their implementation in device, system, and service layer. The existing standards provide requirements and metrics, but do not provide guidelines how to implement them. To address this, the metrics are transformed in MIPs and possible ways how to implement them are provided based on existing monitoring agents or customized scripts. Since one standard has a large number of metrics, an EGM is used to collect and classify the result of each metric.

This article provides a use case during the onboarding procedure and shows the standard compliance verification of a new device, system, and service based on the seven FRs documented in ISA/IEC 62443 series. The MSCV shows the result of the standard compliance verification for a specific layer or the overall compliance, by checking if the device, system, and service fulfill the MSIs.

As part of our future work, we will investigate further security standards that can be technically implemented and consider other dependable aspects, such as safety. In this article, we have evaluated Industry 4.0, which has the goal to interconnect devices, processes, and systems for better efficiency and performance optimization. As part of our future work, we will investigate possible standards for new technologies, such as Industry 5.0, which will take the existing technologies a step further, including the interaction of humans and devices.

References

[1] K. Stouffer, J. Falco, and K. Scarfone, "Guide to industrial control systems (ICS) security," *NIST SP*, vol. 800, no. 82, pp. 16–16, 2011.

[2] M. W. Maier, "Architecting principles for systems-of-systems," *Syst. Eng.: J. Int. Council Syst. Eng.*, vol. 1, no. 4, pp. 267–284, 1998.

[3] O. Carlsson, J. Delsing, F. Arrigucci, A. W. Colombo, T. Bangemann, and P. Nappey, "Migration of industrial process control systems to service-oriented architectures," *Int. J. Comput. Integr. Manuf.*, vol. 31, no. 2, pp. 175–198, 2018.

[4] E. Newcomer and G. Lomow, *Understanding SOA With Web Services*. Reading, MA, USA: Addison-Wesley, 2005.

[5] T. Cucinotta *et al.*, "A real-time service-oriented architecture for industrial automation," *IEEE Trans. Ind. Informat.*, vol. 5, no. 3, pp. 267–277, Aug. 2009.

[6] J. Delsing, *IoT Automation: Arrowhead Framework*. Boca Raton, FL, USA: CRC Press, 2017.

[7] T. Tsiakis and G. Stephanides, "The economic approach of information security," *Comput. Secur.*, vol. 24, no. 2, pp. 105–108, 2005.

[8] A. Bicaku, C. Schmittner, M. Tauber, and J. Delsing, "Monitoring Industry 4.0 applications for security and safety standard compliance," in *Proc. IEEE Ind. Cyber-Phys. Syst.*, 2018, pp. 749–754.

[9] A. Bicaku, C. Schmittner, P. Rottmann, M. Tauber, and J. Delsing, "Security safety and organizational standard compliance in cyber physical systems," *Infocommun. J.*, vol. XI, no. 1, pp. 2–9, 2019.

[10] A. Bicaku, M. Tauber, and J. Delsing, "Security standard compliance and continuous verification for Industrial Internet of Things," *Int. J. Distrib. Sensor Netw.*, vol. 16, no. 6, 2020, Art no. 1550147720922731.

[11] A. Bicaku, S. Maksuti, C. Hegedũs, M. Tauber, J. Delsing, and J. Eliasson, "Interacting with the arrowhead local cloud: On-boarding procedure," in *Proc. IEEE Ind. Cyber-Phys. Syst.*, 2018, pp. 743–748.

[12] N. S. Abdullah, S. Sadiq, and M. Indulska, "Emerging challenges in information systems research for regulatory compliance management," in *Proc. Int. Conf. Adv. Inf. Syst. Eng*., 2010, pp. 251–265.

[13] L. T. Ly, F. M. Maggi, M. Montali, S. Rinderle-Ma, and W. M. van der Aalst, "A framework for the systematic comparison and evaluation of compliance monitoring approaches," in *Proc. 17th IEEE Int. Enterprise Distrib. Object Comput. Conf*., 2013, pp. 7–16.

[14] S. Ziegler, A. Skarmeta, J. Bernal, E. E. Kim, and S. Bianchi, "ANASTACIA: Advanced networked agents for security and trust assessment in CPS IoT architectures," in *Proc. Global. Internet Things Summit*, 2017, pp. 1–6.

[15] S.-K. Choi, C.-H. Yang, and J. Kwak, "System hardening and security monitoring for IoT devices to mitigate IoT security vulnerabilities and threats," *KSII Trans. Internet Inf. Syst.*, vol. 12, no. 2, pp. 906–918, 2018.

[16] S. N. Matheu, J. L. Hernandez-Ramos, and A. F. Skarmeta, "Toward a cybersecurity certification framework for the Internet of Things," *IEEE Secur. Privacy*, vol. 17, no. 3, pp. 66–76, May/Jun. 2019.

[17] R. Atoui *et al.*, "IoT security compliance framework," *IoT Security Foundation*, Livingston, U.K., 2016.

[18] B. Russell and D. Van Duren, *Practical Internet of Things Security: Design a Security Framework for an Internet Connected Ecosystem*. Birmingham, U.K.: Packt Publishing Ltd., 2018.

[19] M. McRoberts and W. Martin, "SOA technical architecture governance—Evaluating maturity of standards," in *Proc. IEEE Int. Syst. Conf*., 2010, pp. 335–340.

[20] T. Espinha, C. Chen, A. Zaidman, and H.-G. Gross, "Maintenance research in SOA—Towards a standard case study," in *Proc. 16th Eur. Conf. Softw. Maintenance Reengineering*, 2012, pp. 391–396.

[21] D. M. Cannon and G. A. Growe, "SOA compliance: Will it sabotage your efforts?" *J. Corporate Accounting Finance*, vol. 15, no. 5, pp. 31–37, 2004.

[22] S. O'grady, "SOA meets compliance: Compliance oriented architecture," *White Paper, RedMonk*, 2004.

[23] J. Bass, "Final versions of the UOCAVA security best practices documents, NISTIR 7711 and NISTIR 7682," Nat. Inst. Standards Technol., Gaithersburg, MD, USA, 2011.

[24] P. Varga *et al.*, "Making system of systems interoperable—The core components of the arrowhead framework," *J. Netw. Comput. Appl.*, vol. 81, pp. 85–95, 2017.

[25] P. Varga and C. Hegedus, "Service interaction through gateways for inter-cloud collaboration within the arrowhead framework," in *Proc. 5th IEEE Wireless Vitae*, Hyderabad, India, 2015.

[26] O. Carlsson, D. Vera, J. Delsing, B. Ahmad, and R. Harrison, "Plant descriptions for engineering tool interoperability," in *Proc. IEEE 14th Int. Conf. Ind. Informat.*, 2016, pp. 730–735.

[27] R. Sheikhpour and N. Modiri, "An approach to map COBIT processes to ISO/IEC 27001 information security management controls," *Int. J. Secur. Its Appl.*, vol. 6, no. 2, pp. 13–28, 2012.
[28] I. O. for Standardization, *Information Technology—Security Techniques—Guidelines for Cybersecurity*, ISO/IEC 27032:2012, 2012.
[29] R. Schlegel, S. Obermeier, and J. Schneider, "Assessing the security of IEC 62351," in *Proc. 3rd Int. Symp. ICS SCADA Cyber Secur. Res. BCS Learn. Develop. Ltd.*, 2015, pp. 11–19.
[30] K. Stouffer, S. Lightman, V. Pillitteri, M. Abrams, and A. Hahn, "NIST special publication 800-82, revision 2: Guide to industrial control systems (ICS) security," *Nat. Inst. Standards Technol.*, Gaithersburg, MD, USA, 2014.
[31] I. E. C., "Industrial communication networks—Network and system security—Part 3-3: System security requirements and security levels," *Nat. Inst. Standards Technol.*, Gaithersburg, MD, USA, 2013.

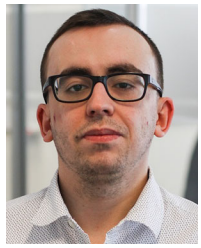
Ani Bicaku received the B.Sc. degree in telecommunication engineering from the Polytechnic University of Tirana, Tirana, Albania, in 2012, the Dipl.Ing. degree in communication engineering from the Carinthia University of Applied Sciences, Klagenfurt, Austria, in 2015, and the Ph.D. degree in cyber-physical systems from the Luleå University of Technology, Luleå, Sweden, in 2020.

From 2014 to 2016, he was with the Austrian Institute of Technology (AIT), in the AIT's ICT-Security Program and was responsible for evaluating data security, data privacy, and high assurance in cloud computing. In 2016, he joined the University of Applied Sciences Burgenland, Eisenstadt, Austria, as part of the Cloud and Cyber-Physical Systems Research Center.

Dr. Bicaku is a member of Austrian Electrotechnical Committee of the Austrian Electrotechnical Association (OVE) at IEC and CENELEC standardization bodies within TC65-WG10 "Security for Industrial Process Measurement and Control—Network and System Security." He has been part of several EU projects, e.g., SECCRIT, SEMI40, PRODUCTIVE4.0, ArrowheadTools, and Comp4Drones.

Mario Zsilak received the B.Sc. degree in information and communication systems and services from the University of Applied Sciences Technikum Wien, Vienna, Austria, in 2017. He is currently working toward the master's degree in M.Sc. Program Business Process Management and Engineering with the University of Applied Sciences Burgenland, Eisenstadt, Austria.

He is currently a Software Engineer with the Center for Cloud and CPS Security, Forschung Burgenland GmbH, Eisenstadt, Austria. He has contributed to Arrowhead in a number of projects.



Peter Theiler received the B.Sc. degree in IT infrastructure and IT management and the master's degree in cloud computing engineering from the University of Applied Sciences Burgenland, Eisenstadt, Austria, in 2017 and 2019, respectively.

Markus Tauber received the Ph.D. degree in computer science from the University of St Andrews, St Andrews, U.K., in 2010.

He is currently a FH-Professor with the University of Applied Sciences Burgenland, Eisenstadt, Austria, where he holds the position of the Director of the M.Sc. program Cloud Computing Engineering and leads the research center Cloud and Cyber-Physical Systems Security. Between 2012 and 2015, he coordinated the research topic "High Assurance Cloud" with the Austrian Institute of Technology (AIT) part of AIT's ICT-Security Program. Amongst other activities, he was the Coordinator of the FP7 Project "Secure Cloud Computing for CRitical Infrastructure IT" and involved in the ARTEMIS Project Arrowhead. From 2004 to 2012, he was with the University of St Andrews, where he was a Researcher on various topics in the area networks and distributed systems.

Jerker Delsing received the M.Sc. degree in engineering physics from the Lund Institute of Technology, Lund, Sweden, 1982, and the Ph.D. degree in electrical measurement from Lund University, Lund, Sweden, in 1988.

During 1985–1988, he worked part time with Alfa-Laval—SattControl (now ABB) with development of sensors and measurement technology. In 1994, he was promoted as an Associate Professor in heat and power engineering with Lund University. In early 1995, he was appointed as a Full Professor in industrial electronics with Luleå University of Technology, Luleå, Sweden, where he is currently the scientific head of EISLAB. His present research profile can be entitled IoT and SoS automation, with applications to automation in large and complex industry and society systems.

Prof. Delsing and his EISLAB group have been a partner of several large EU projects in the field, e.g., Socrades, IMC-AESOP, Arrowhead, FAR-EDGE, Productive4.0, and Arrowhead Tools. He is a Board Member of ARTEMIS, ProcessIT.EU, and ProcessIT Innovations.