Computer Science and Information Systems 20(1):359–380 https://doi.org/10.2298/CSIS210923060M

# Blockchain-based model for tracking compliance with security requirements*

Jelena Marjanović, Nikola Dalčeković, Goran Sladić

Faculty of Technical Sciences, Trg D. Obradovića 6,
21000 Novi Sad, Serbia
{jelena.stankovski, nikola.dalcekovic, sladicg}@uns.ac.rs

**Abstract.** The increasing threat landscape in Industrial Control Systems (ICS) brings different risk profiles with comprehensive impacts on society and safety. The complexity of cybersecurity risk assessment increases with a variety of third-party software components that comprise a modern ICS supply chain. A central issue in software supply chain security is the evaluation whether the secure development lifecycle process (SDL) is being methodologically and continuously practiced by all vendors. In this paper, we investigate the possibility of using a decentralized, tamper-proof system that will provide trustworthy visibility of the SDL metrics over a certain period, to any authorized auditing party. Results of the research provide a model for creating a blockchain-based approach that allows inclusion of auditors through a consortium decision while responding to SDL use cases defined by this paper. The resulting blockchain architecture successfully responded to requirements mandated by the security management practice as defined by IEC 62443-4-1 standard.

**Keywords:** industrial control systems, secure development lifecycle, blockchain.

## 1. Introduction

With technological improvements that are permeated through various aspects, software engineers and everyone involved in product development have become more aware of the impact a potential bug can produce on everyday life. While bugs in the production can lead to disrupted availability of the service or a product, a vulnerability could also lead to loss of confidentiality and integrity of the system. Those vulnerabilities have a greater impact if they were to occur in an ICS, as they perform data acquisition and real-time control [1] and the root cause of those vulnerabilities must be addressed [2]. A prime example of ICS is the Supervisory Control and Data Acquisition (SCADA), and if a vulnerability was to enter such a system, it may cause blackouts, thus leaving cities without power. A flaw in the energy management system led to a blackout in the northeastern U.S. in 2004, which could have been prevented if the code audit had been done in the implementation phase [3]. Another worrying evidence is that critical infrastructures have become a target for various cyber-attacks, where threat actors vary from competitors, hacktivists, cyber-criminals, nation-states. The impact of cyberattacks

* This is an extended version of the ECBS 2021 conference paper ”Improving Critical Infrastructure protection by Enhancing Software Acquisition Process Through Blockchain”.

that have happened to ICS, starting with the first publicly known SCADA cyberattack [4] to the latest Colonial pipeline malware attack [5] is ever-increasing, which has also been noted by several authors [6], [7], [8]. Cyber-attacks in the global oil supply chain were previously recognized as a potential issue, so authors of [9] have analyzed cyber threats and have provided immediate countermeasures. A model and an algorithm to optimize the survivability of a mission-critical system under attacks for a certain time duration by maintaining redundancy of components can be used when designing such a system [10]. Vulnerabilities in industrial control systems have shown that cybersecurity posture must improve, and the root cause of ICS vulnerabilities should be addressed [11].

Performing the root cause analysis for issues that have occurred is the way to minimize future mistakes and learn in the process, but a more beneficial approach is the shift left approach, i.e., implementing security checks from the early development. Such an approach can be complemented by implementing the industry best practices from a secure development lifecycle process. The standard IEC 62443-4-1 [12], named Secure product development lifecycle requirements, SDL for short, helps industrial automation and control systems (IACS) increase their security posture, by implementing security best practices in every aspect of the product development lifecycle. The IEC 62443-4-1 standard is divided into eight practices, addressing security requirements definition, secure design, secure implementation (including coding guidelines), verification and validation, defect management, patch management, and product end-of-life. [12]. As the IEC 62443-4-1 standard is written in the form of 47 requirements, the process of requirement engineering is of great importance. This process has been utilized by various industries, as it allows requirements to go through several stages and can be tracked through their phases. Requirement engineering practices has been analyzed and improved [13], [14], [15], [16], evaluated for startups [17] and adjusted for cyber-physical systems [18], [19].

Requirement engineering is a process that follows the lifecycle of a requirement. An approach to keep information dated and versioned, while at the same time have a tamper-proof resolution that guarantees that information that was written has not been altered, is to utilize features that blockchain technology provides. While some industries require that information stored on the blockchain is made publicly available and is required that the information is publicly verified, most of the industries have decided to keep some or all the information available only to interested parties. Private blockchain networks are suitable for corporations that need to utilize blockchain technology but the information that is stored on the blockchain cannot be publicly available. Hyperledger Fabric is a distributed ledger, used for creating blockchain solutions that require a private permissioned blockchain network, a network that is created and maintained by a pre-authorized set of members. An overview of blockchain classification was done by Golosova et al. [20], where the difference between public, private, permissioned, and permissionless blockchain was provided. Hyperledger Fabric has smart contracts, transactions, peers, consortiums as other blockchain implementations, but it has also introduced terms such as organization, ordering service, and channel [21] [22].

A step forward was made in requirement tracking, as the author proposed requirement tracing utilizing blockchain technology [23]. Blockchain technology provides immutability of information being stored on the distributed ledger. Demi et al. [24] claim that blockchain has the potential to enhance the immutability, trust, visibility, and

traceability of requirements throughout the software development lifecycle (SDLC). What we see as an appropriate extension to Demi's hypothesis [23], that would be beneficial to ICS wanting to improve its security posture, is to have a private permissioned, blockchain-based model, so that organizations can track and manage security requirements throughout the secure development lifecycle, which would allow ICS to promote cooperation and trust among different parties. The extension we made is to utilize only private-permissioned blockchain networks, as ICS will not set its requirements and their compliance on a public blockchain, that would be accessible to anyone with appropriate tools, and to focus on security requirements for ICS. We propose a blockchain-based model for tracking compliance with security requirements, as blockchain technology provides timestamped and tamper-proof information that is stored on the ledger, which is beneficial to parties auditing the process. Additionally, a universal blockchain architecture that can be used within the proposed model is offered. We chose to evaluate and provide details of this model, on Security Management practice, as a governing practice that ensures all other practices in the IEC 62443-4-1 standard are executed appropriately.

The remainder of this paper is organized into five sections. Section 2 provides review of related work. Section 3 introduces a proposed blockchain-based model and architecture. Section 4 evaluates the proposed blockchain model for security management requirements, combined with supply-chain management architecture. Section 5 concludes with a final discussion on the solution and future steps.

## 2. Related Work

The issue of addressing security practices from the beginning of a product lifecycle has been discussed by several authors [25], [26] relying on various standards and guidelines that provide knowledge on incorporating security into the product. Secure development lifecycle (SDL), whether product or software is the intended area of applicability, is a process of building secure products or software, by encompassing security and privacy considerations throughout all phases of the development process, helping developers to build highly secure software while addressing security compliance requirements, and reducing development costs [27]. Security standards and guidelines change over time, as seen in the case of Comprehensive, Lightweight Application Security Process-CLASP, which has been put to archive, but its segments are incorporated into IEC 62443-4-1 standard [12], [28]. One of the segments that were not incorporated directly into the IEC 62243-4-1 standard is roles and their responsibilities. Instead, only a requirement for defining roles and responsibilities is added (SM-2 Identification of responsibilities). Apart from CLASP, NIST Special Publication 800-64 Rev. 2 [29], named Security Considerations in the System Development Life Cycle, has also been withdrawn, guiding readers to refer to NIST SP 800-160 Volume 1 [30]. Microsoft's SDL has been guiding software developers over the last two decades [31], [32], [33]. Differences between CLASP and Microsoft's SDL have been addressed several times [34], [35], [36] which is beneficial to those gaining broader knowledge. Another standard that takes into consideration security from the very begging of the product development lifecycle is IEC 62443-4-1 standard. The IEC 62443-4-1 standard is one of 13 standards included

in the IEC 62443 series, developed by the ISA99 committee. Standards within series are grouped in general, policies and procedures, system and component groups [12] covering a broad area of security for industrial automation and control systems, from Terminology, concepts, and model (IEC 62443-1-1) to Technical security requirements for IACS components (IEC 62443-4-2). IEC 62443 is a source of common understanding of cybersecurity-related issues for industrial and automation control system (IACS) owners, component developers, and service providers [12]. This paper focuses on the IEC 62443-4-1 standard and Security Management practice, presented as a first and crown practice, containing 13 requirements, ranging from the development process to continuous improvement.

The demand for secure development lifecycle practices has been identified by academia and industry, but the applicability and justification of resources, both human and financial, remained a debate. This issue with additional cost that security practices are believed to be adding to the development was discussed in [37], which concluded that, at the time, few cost-estimation models that take security into account have been proposed and that the existing models were not properly validated. While the argument that security practices introduce excessive overhead in terms of time and money, authors [32] showed that Microsoft's Secure Development Lifecycle can be used even on a small team, that consists of one developer, but argue that the proper cost-benefit analysis of implementing a robust framework on a small team, should be conducted. Another group of authors [38] assumes that security will introduce overhead in terms of time and additional human resources. There are some challenges when secure practices are left out of software development and they need to be introduced, particularly in agile web development relying on SCRUM methodology. Such development is based on fast feature production, which usually lasts less than 30 days. The authors argue that such a short period does not leave time for security practices to be implemented at full scale and propose a secure SCRUM process, allowing "agile" security activities to be introduced to the process. The process was evaluated by a team of developers, describing the process as "medium" agile and "medium" cost-effective. As authors assumed, such a process introduced overhead in terms of time, but the analysis of not applying secure practices, which could lead to security issues such as breach and DDoS, has not been discussed. An alternative perspective is that companies take security and secure practices differently, depending on the industry, size, and organizational structure. Also, security experts in companies have various roles, from security engineers, consultants, or auditors. The work done by security auditors was presented by authors [39] in form of interviews, providing insight into security practices, such as static code analysis, and penetration tests. As the authors have concluded, a combination of organizational processes, developer training, and tools is needed for improving application security. The traceability for the processes has not been discussed and that is gained by utilizing our proposed blockchain-based model, since the possibility of tracking all information that has been put on the decentralized ledger, digitally signed and tamper-proof information, is an out-of-the-box feature of the blockchain.

Further research shows that one direction in securing products is incorporating security tools, such as AttackSurface Host Analyzer (AHA), allowing continual monitoring and improving ICS, but such activity is not sufficient for an overall increase in product security posture [40]. An argument for poor SDL implementation in the industry may lay in the interviewees' perspective of lacking security experts [41], there is

a mechanism that can contribute to additional security practitioners in the early stages of developers' working life. Walden et al. [42] have recognized the need for secure development lifecycle courses at the undergraduate level, as adopting base principles of SDL at the begging of higher education, provides future engineers concepts and the importance of securing development lifecycle. The course introduced 10 modules and a web project for demonstration, allowing students both conceptual and practical knowledge of SDL, but the research lacked results from this introduction. Also, the importance of introducing quality materials and advanced techniques in teaching the value of secure development lifecycle was recognized by authors [43] where teaching security design analysis in a hybrid flipped classroom was introduced in the class of 2015/2016. The proposed framework showed that the newer generation had better learning outcomes, reflected in system understanding and dataflow diagram quality. Since the framework showed an increase in students' understanding, it could lead to creating additional security courses using the hybrid flipped classroom method, which would further increase overall understanding of the importance of a secure development lifecycle.

With our approach of enabling compliance for securing the product development lifecycle, other ICS, such as smart grids, that are highly regulated, can append their blockchain implementations, for their compliances. Smart grids in the USA are regulated by NERC CIP standards, that every utility is obligated to follow. Authors from [44] have recognized the potential that blockchain technology has in terms of making information available, secure and tamper-proof, which can be considered a prerequisite for standard compliance. Using blockchain proof of authority as a mechanism for providing widely witnessed evidence on what can be considered the truth, while not relying on a single party, authors in [44] have proposed a supply chain blockchain solution, which is per NERC Critical Infrastructure Protection 13 standard. Blockchain security controls that enable compliance with NERC CIP 13 standard, also enable Customers, Manufacturers, Hardware, and Software Suppliers, as identified actors, to exchange information in a secure, transparent, traceable, and tamper-proof manner. A similar group of authors [45] has also analyzed blockchain utilization for other NERC CIP standards: CIP 007-5, CIP 008-5, CIP 009-5, CIP 010-1, and CIP 011-1. Realization of compliance is suggested through using keyless signature blockchain infrastructure, while NERC CIP requirements were fulfilled through 18 critical controls. An in-depth analysis on facilitating compliance with NERC CIP 010 standard, which is aimed at configuration management and vulnerability assessment, is provided by authors in [46]. Requirements and their measures from NERC CIP 010 standard can be fulfilled by seven identified blockchain controls. While the authors from [44], [45], [46] describe how NERC CIP compliance can be achieved utilizing various blockchain implementations, they also point out that such an implementation should be thoroughly analyzed as it is still in the nascent stage [44].

The most comprehensive analysis of the IEC 62443-4-1 standard presented in several papers, from a similar group of authors [47], [48], [49]. The standard was analyzed in terms of applicability, deliverables, CMMI and authors have also addressed the topic of integrating IEC 62443-4-1 standard with agile software engineering. This analysis was done through several aspects, from creating a BPMN that includes SDL and agile process to interviewing key stakeholders of those processes. In the paper [49], authors made a self-assessment tool that can provide to development teams an insight to current

compliance with the IEC 62443-4-1 standard, without the need for additional, costly, involvement of external auditors. The tool consists of eight assessment sheets, each corresponding to practice in the IEC 62443-4-1 standard. Through three research questions, this tool was evaluated by Siemens employees, with different backgrounds in IEC 62443-4-1 standard and expertise in security compliance assessment. While utilizing this tool, the results can be explicitly tracked to the 4-1 requirements delivering a common ground for auditors and project participants, we argue that such an approach lacks traceability and is not tamper-proof.

Compared to other solutions, our proposed blockchain-based model differs in that it takes into consideration that ICS, which will follow certain SDL practices, wish to keep their information private, tamper-proof, and easily auditable. As the private-permissioned blockchain provides a tamper-proof solution, enabling only a pre-authorized set of users to participate in the process, our proposed model enables those ICS to fully commit to implementing security requirements that are part of the SDL process, while at the same time, have a way of incorporating auditing opportunities.

## 3. Requirement's Compliance Tracking Model and Architecture

Hyperledger Fabric is a private-permissioned blockchain network that enables only preauthorized users to participate in the network. This feature, combined with blockchain out-of-the-box traceability and tamper-proof features, can be utilized for tracking compliance with various security standards. While other solutions provide private permissioned blockchains, Hyperledger Fabric is focused on enterprise-level solutions, covering a wide range of industries that can utilize their solution. Given that the proposed model is oriented towards ICS that wish to comply with security requirements from Secure Development Lifecycle, such blockchain implementation seems suitable.

The proposed blockchain-based model is shown as an activity diagram in Figure 1, illustrating which steps the Project Manager, Security Advisor, and Blockchain Administrator should do to create a platform that would enable other participants to contribute to compliance with chosen standard requirements. In the beginning, the Project Manager chooses the standard to be implemented and Security Advisor checks for the applicability of that security standard, as the security expertise is within that role. If the standard is not applicable, this activity diagram is finished. Next, the Security Advisor should define use case diagrams for the requirements that are in the scope. Security Advisor can consult the Project Manager if any assistance is needed. All other steps are for the Blockchain Administrator, and that is to determine the number of channels, define organizations and consortiums based on uses case, and determine whether Inter Planetary File System (IPFS) should be utilized. Certain solutions, that involve blockchain network, require files to be uploaded. As the blockchain network is not created for storing files, an Interplanetary File System (IPFS) can be used. The IPFS is a peer-to-peer hypermedia protocol designed to make the internet faster, safer, and more open [50]. In a peer-to-peer network such as IPFS, if one node is down, other nodes in the network can serve needed files. Utilizing IPFS for storing files, the blockchain network remains solely for storing transactions and maintaining the world

state. Versioned documents will be available for download from IPFS, while the information about the version number, last update, and last modifier can be stored within smart contracts. Finally, the architecture, which depicts all organizations, consortiums, applications, and channels is created and deployed on the Hyperledger Fabric.

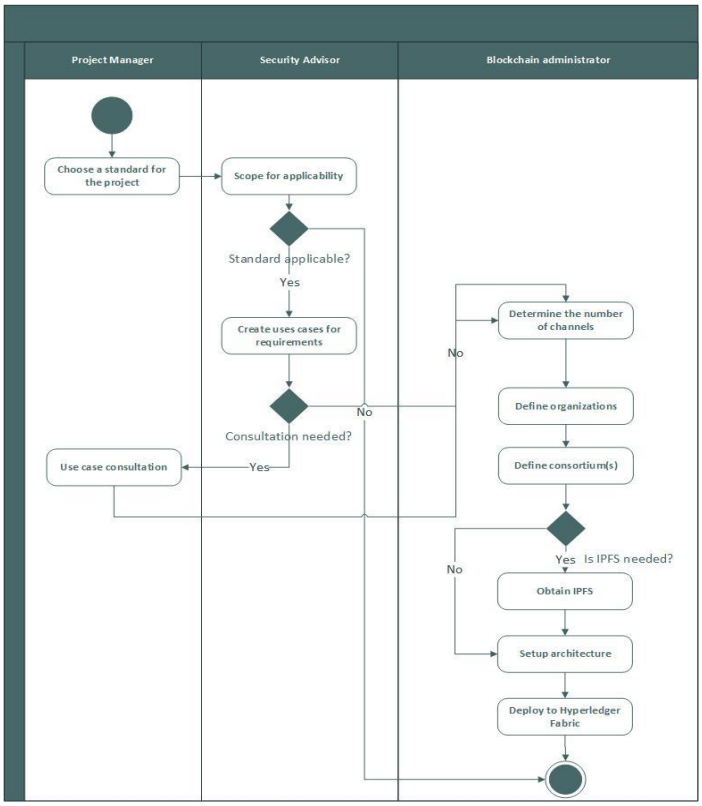


**Fig. 1.** Blockchain-based model for tracking compliance with security requirements

In addition to the proposed blockchain-based model for tracking compliance with security requirements, a customizable architecture is also proposed. This customizable architecture, that combines Hyperledger Fabric architecture and IPFS, enables the Blockchain Administrator to easily adjust use cases that have been created by the Security Advisor into a deployable architecture. The number of channels is the number of presented use cases, the organizations within Hyperledger Fabric are presented as actors in use cases and consortiums are presented as a use case that certain actors can update. Figure 2 shows a customizable architecture that can be adjusted to the required number of channels, organizations, and consortiums.

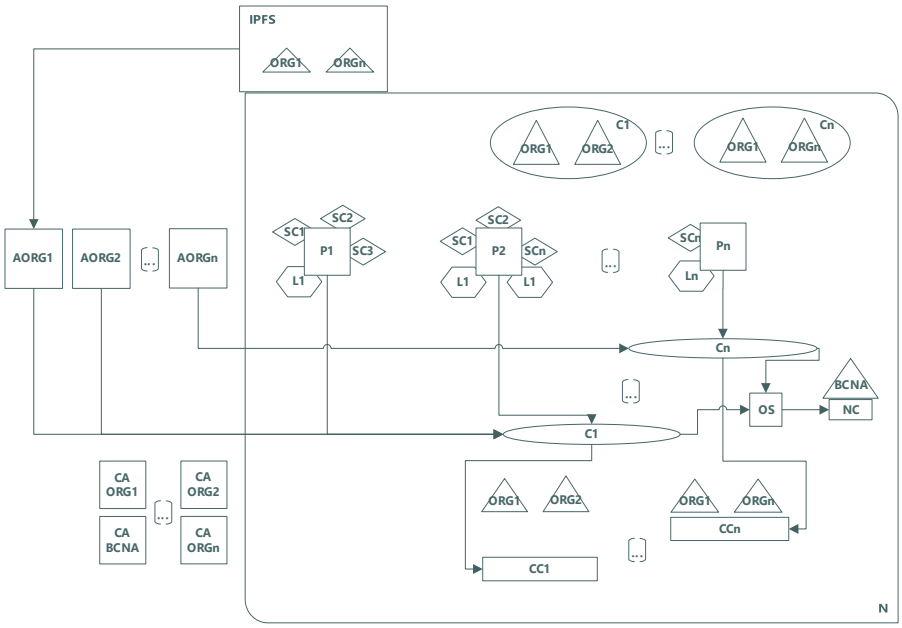


**Fig. 2.** Customizable architecture for tracking requirement compliance combining Hyperledger Fabric architecture and IPFS

The customizable architecture shown in Figure 2 represents the blockchain network N, with related Certificate Authorities, applications for organizations, and IPFS. The blockchain network N contains one Ordering Service (OS), one Network Configuration, and Blockchain administrator organization (BCNA). The number of consortiums (Cn), channels (CHn), channel configurations (CCn), organizations (ORGn), peers (Pn), smart contracts (SCn), ledgers (Ln), certificate authorities (CAn), and applications (AORGn) is defined through use cases and their actors. If IPFS is identified in the use cases, it can be added externally to the blockchain network and connected to the corresponding applications. Following this universal architecture, an architecture that is appropriate for the supply-chain management use case described in the next section will be discussed.

## 4. Model evaluation through Security Management practice

The blockchain-based model that has been proposed in the previous chapter will be evaluated on a security standard, guiding how to interpret the discussed steps. We have chosen IEC 62443-4-1 standard for Secure Product Development Lifecycle to evaluate how this model can be used for demonstrating compliance, as every information on the blockchain network is timestamped and digitally signed. Particularly, Security Management practice has been chosen for this paper, as the overall practice in IEC 62443-4-1 standard, whose implementation is a prerequisite for other practices. We also believe that the proposed model can be applied to other security standards that would require similar evidence for compliance.

The first step from the proposed model was for the Project Manager to choose a standard to be compliant with. That step was done by choosing all 13 requirements from Security Management practice, from IEC 62443-4-1 standard. The requirements are named with the prefix SM (Security Management) and have an assigned incremental number, as well as the name of the requirement, e.g., the SM-1 Development process is the first requirement from Security Management practice, named Development process. In the following sections, activities presented in the proposed model, from choosing the scoping applicability to setting up the architecture, are discussed.

### 4.1. Security Advisor activities

The first step for the Security Advisor, presented in Figure 1, is to perform the scoping and applicability decisions. The applicability of the standard requirements can be tracked on the blockchain, and it will be discussed later as part of the Security Management practice. The following step is to create use cases for the requirements and throughout this activity, Security Advisor can consult with the Project Manager. Although the requirements from IEC 62443-4-1 standard are grouped into practices, for Security Management practices, we have divided requirements into four divisions, as it offers better organization of channels and consortiums that need to be defined on a blockchain network. The divisions are:

1. Team and project management: this division covers requirements SM-2 through SM-5, as they require teams to have defined roles and responsibilities, team members should complete assigned trainings, and applicability of this standard to the product in scope should be done.
2. Development environment: these requirements are aimed at securing the development environment (SM-7), considering the development process (SM-1), while at the same time ensure the controls for private keys (SM-8) and ensure file integrity (SM-6).
3. Supply chain management: this division is for requirements SM-9 and SM-10, that are directed at vendor's supply chain and engaged 3rd party company, that provides custom-developed components.
4. Quality assurance: this division is created to gather requirements SM-11, SM-12, and SM-13, as they are focused on tracking security bugs to closure (SM-11). This proposed framework is essentially how the SM-12 requirement, named Process verification can be fulfilled. The requirement that focuses on increasing the SDL process maturity ultimately increasing software quality and security is Continuous improvement (SM-13).

These four divisions are the use cases that would satisfy the Security Management requirements, described below.

#### Team and project management use case

For this use case, which is named *Team&project management* and presented in Figure 3, actors Team and Auditors have been defined, where the Actor Team is a generalization for actors Project Manager, Security Advisor, and Software Component Manager. Every

actor must authenticate to the network, while the Project Manager and Security Advisor create and maintain a consortium.

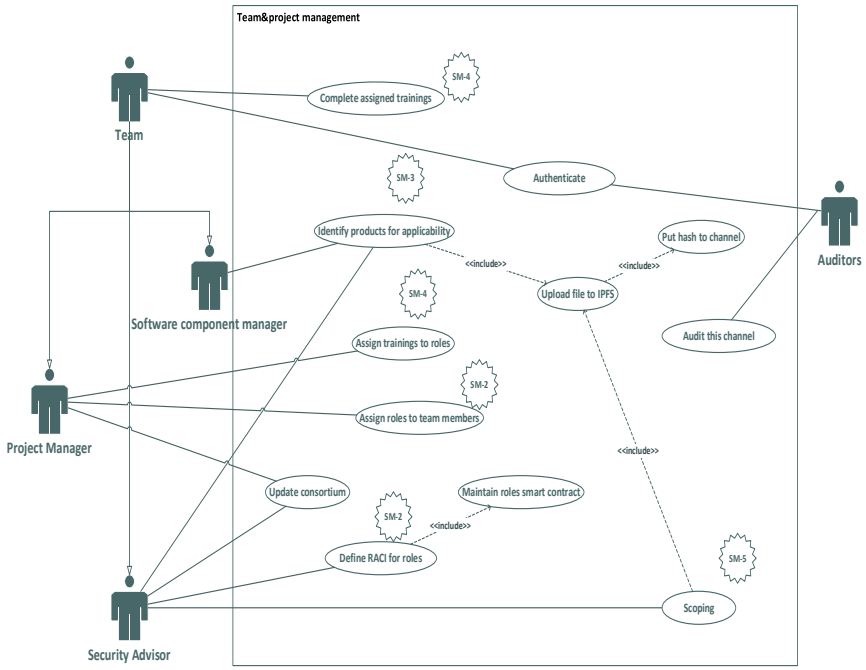


**Fig. 3.** Team&project management

The actors contribute to this use case through the following scenarios:

1. The actor Team: the generalization of the Project Manager, Security Advisor, and Software Component Manager is presented as the actor Team, as everyone involved in the SDL must complete assigned trainings. The completion of assigned trainings is required by the requirement SM-4 Security expertise. Through this requirement, Project Managers are expected to assign trainings to identified roles. The identification of roles, i.e., assigning roles to team members is expected from the Project Manager and is needed by requirement SM-2 Identification of responsibilities. This requirement includes defining RACI (responsible, accountable, consulted, informed) matrix, and that is done by Security Advisor, as that role possesses the knowledge. The Security Advisor is responsible for performing Scoping, which is directed by the requirement SM-5 Process scoping. As shown on the activity diagram in Figure 2, the Security Advisor will scope a standard and select requirements that the product should be made compliant with. This is also applicable for the requirement SM-3, Identification of applicability, where the Software Component Manager will provide information to which components SDL should be applied.
2. The actor Auditors: presented here as the role of auditing this channel, i.e., seeing that roles have been delegated, every role has completed the assigned

trainings, the processes artifacts for identifying applicability and scoping are created.

**Development environment use case**

The Development environment use case is created for grouping Software Component Manager's, Security Advisor's, and Project Manager's responsibilities, respectively, which are in line with the requirements SM-1 and SM-6 through SM-8. All actors that are presented in this use case, must authenticate to the blockchain network. The project Manager and Security Advisor will form a consortium, that will enable them to add or remove new organizations to the channel, as well as to limit their rights. The actor Auditors can be present at any channel, to inspect the compliance with those requirements.

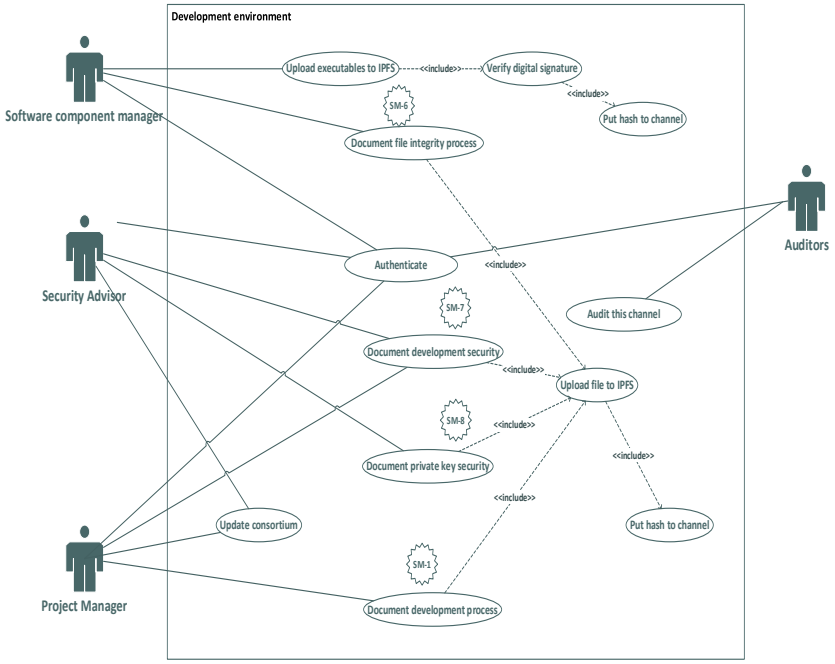


**Fig. 4.** Development environment channel

Following actors and scenarios contribute to this use case:

1. The actor Software Component Manager: responsible for uploading executable files to the IPFS, as the requirement SM-6 File integrity, requires the product to have a mechanism that shows that files have not been altered. Although the blockchain network is used for verifying that information has not been altered, this requirement is aimed at files and executables, which should not be placed on a blockchain network, rather they should be placed on IPFS. The upload of executable files should include verification of the supplier's digital signature, and

such information should be included in the channel, through a designated smart contract. This process can be described through a document and uploaded to the IPFS. Once the file is uploaded to the IPFS, the obtained hash should be stored on the smart contract.

2. The actor Security Advisor: the role of Security Advisor in this use case is seen through requirements SM-7 Development environment security and SM-8 Controls for private keys. Both requirements expect both technical and procedural controls to be put in place, but since technical controls, such as Hardware Security Modules for private keys should not be used by the blockchain network itself, both these requirements are focused on implementing the processes.
3. The actor Project Manager: this role is similar to the one Security Advisor has, as the requirement SM-1 Development process is aimed at documenting and enforcing product development processes, for configuration management, requirement engineering, implementation practices, etc. The process can be seen as a document that can be uploaded to the IPFS and that hash should be placed to the channel, through a smart contract.
4. The actor Auditors: the role of actor Auditors is to verify that requirements SM-1, SM-6, SM-7, and SM-8 have been met and that the traceable documentation is created, which can be done through inspecting the Development environment channel on the blockchain network. All information that is put on the blockchain network has timestamped and signed changes, which eases the auditing of channels and requirements.

**Supply chain management use case**

Participants in the process of managing the supply chain, seen through the integration of software components procured from different vendors, are defined as actors in the use case diagram shown in Figure 5. An actor called Software Component Vendor represents companies that provide software that can be custom-made for a specific customer or can be commercial-off-the-shelf (COTS) components. The actor Purchaser is the company that will be using the software that Software Component Vendor provides. The Purchaser actor is a generalization for actors named Software Component Manager, Security Advisor, and Project Manager. Also, the actor Auditors is presented in the figure which will be able to inspect the whole process of tracking software components' supply chain.

Every actor in the use case must authenticate to the blockchain network to be able to participate in any activity. Following actors contribute to this use case:

1. The actor Software Component Vendor: the actor Software Component Vendor is also responsible for filling the security questionnaire, which includes uploading the file to Interplanetary File System (IPFS). The security questionnaire is a document used for assessing the security posture of companies whose components will be integrated into the system. In case the Software Component Vendor is producing a tailor-made software component, besides filling the security questionnaire, the code should be deployed in a predefined repository, which is managed by a Software Component Manager of the

Purchaser. Through this security questionnaire, the requirement SM-9 Security requirements for externally provided components can be fulfilled. The security questionnaire will provide enough information to the Security Advisor, which is responsible for analyzing the questionnaire, to determine whether Software Component Vendor security posture is adequate.

2. The actor Purchaser: the actor Purchaser, as a generalization for actors Software Component Manager, Security Advisor, and Project Manager, can update a consortium for adding and deleting organizations. The actor Software Component Manager can manage the software component repository which is utilized by the Software Component Provider. The role of the Security Advisor is to perform questionnaire analysis, which includes obtaining the file from the IPFS. Through this generalization, requirements SM-9 and SM-10, can be fulfilled. The Security Advisor will inspect the security questionnaire and the Software Component Manager will manage the software component repository, which is in line with the SM-9 requirement, while the Project Manager will audit Software Component Vendor, which is defined by the requirement SM-10 Custom developed components from third-party suppliers.
3. The actor Auditors: if an external auditor wants to inspect the process of managing the software components that are either COTS or tailor-made for a Purchaser, that can be done by updating the consortium between Software Component Vendor and Purchaser. For Auditors to be able to inspect any of the ledgers, authentication must be performed against their CA. After successful authentication, Auditors can proceed with inspecting the security questionnaire. Once the audit is finished, Software Component Vendor and Purchaser can update the consortium so that Auditors are deleted from the channel, thus losing the capability to inspect the ledger.

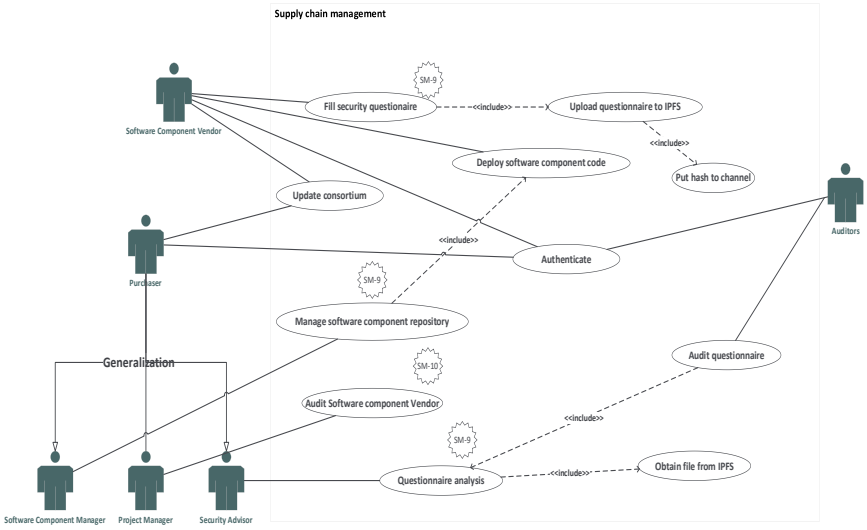


**Fig. 5.** Supply-chain management channel

### Quality assurance use case

The actors Test analyst, Project Manager, Security Advisor contribute to the quality by following requirements SM-11 through SM-13. While the quality itself is permeated through several requirements in multiple practices in SDL, these requirements from the Security Management practice are focused on increasing the quality through management. As in previous use cases, all actors must authenticate to the channel. Future organizations, that are shown as Project Manager and Security Advisor actors in this use case, form a consortium, which enables them to manage the channel configuration and add or remove other organizations, such as Auditors, to the channel.

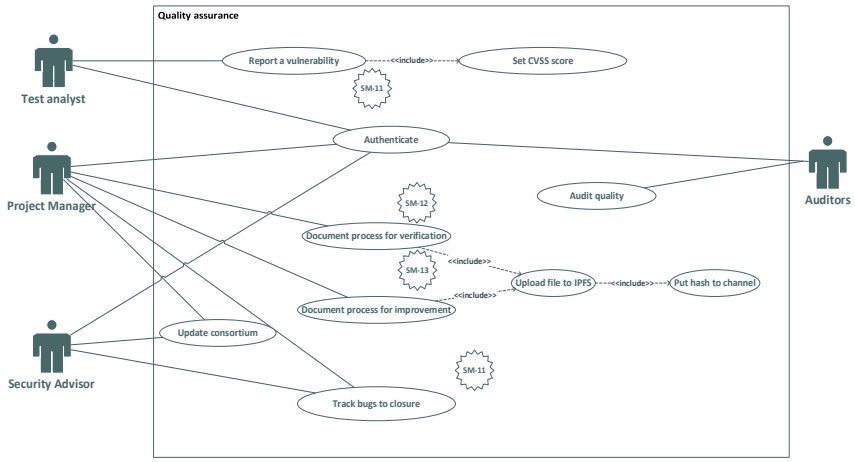


**Fig. 6.** Quality assurance channel

The following actors and scenarios are part of this use case:

1. The actor Test Analyst: the requirement SM-11 Assessing and addressing security-related issues is aimed at verifying that the product has not been released with security-related issues, but the role of the Test analyst is to report such vulnerabilities, which includes setting the correct CVSS score. Setting proper CVSS score will allow other participants in the channel to be aware of the critically and make prioritizations if such action is needed.
2. The actor Project Manager: this actor is responsible for tracking security-related issues to closure, which is defined by the requirement SM-11. Also, the Project Manager should contribute to the documents that should be made for compliance with SM-12 Process verification and SM-13 Continuous improvement. While the requirement SM-12 itself is incorporated into every use case diagram and enabled by the later proposed architecture, the document that describes the process itself can be uploaded to the IPFS, and the document hash should then be put to the appropriate smart contract on the channel. Similarly, the requirement SM-13 is incorporated into every use case diagram, as it allows the actor Auditors to inspect the processes, which enables them to suggest further improvements. The document that describes the improvement process can be uploaded to the IPFS and the hash can be put to the smart contract.

3. The actor Security Advisor: the role of Security Advisor is presented as a support for tracking security-related issues to closure, as this role contains the needed knowledge about security-related issues and can advise on prioritization if such is needed.
4. The actor Auditors: as in previous use cases, the actor Auditors can inspect the channel, once the consortium enables them read rights to the channel. This audit includes verification that all security bugs are tracked to closure, that Test Analysts have reported vulnerabilities with correct CVSS scores, and that proper documentation, is uploaded to the IPFS and can be tracked.

### 4.2. Blockchain Administrator activities

Due to paper limitations, only the blockchain architecture for the supply-chain management use case will be presented. The blockchain network architecture for tailor-made or COTS components is shown in Figure 7. This architecture is the proposed solution of how Purchaser can track software components that are developed specifically for that system, i.e., tailor-made or COTS components which are incorporated as-is in the system. The architecture is created as a specification of the customizable architecture, applied to the Supply-chain management use case. With such architecture, compliance with requirements SM-9 and SM-10 can be proven.

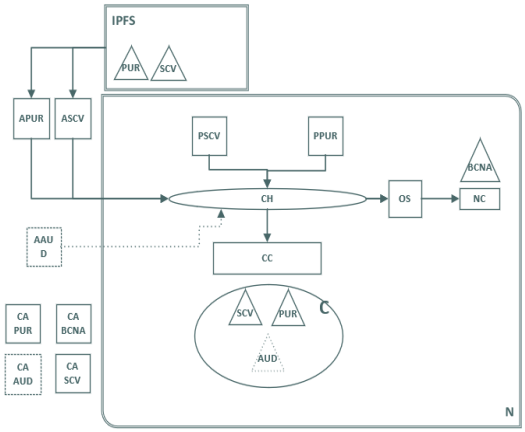


**Fig. 7.** Supply-chain management network architecture

Upon blockchain network creation, Network Configuration (NC), Organization Blockchain Network Administrators (BCNA), and Ordering Service (OS) are added. NC is the set of rules and policies allowing organization BCNA to maintain the network, add consortiums, new organizations, create channels, add new ordering services and peers. Consortium (C) is created between organization Purchaser (PUR) and Organization Software Component Vendor (SCV), which have formed a Channel (CH) through Channel Configuration (CC). This consortium is formed for organizations to be able to create their channel, with the necessary smart contracts to carry out the tracking of

software components through security questionnaire. Also, by leveraging organizations, only preauthorized members can write and read from the channel, following the need-to-know basis. Every organization in this solution has its own Certificate Authority (CA), which is presented as CA BCNA, CA PUR, and CA SCV. Also, organizations SCV and PUR have their peers who host the ledger, named PSCV and PPUR, respectively. Since organizations SCV and PUR have formed a consortium, they can add additional organizations to the channel, but only if both organizations agree on such activity. This is necessary in case an external auditor must examine the process. By changing the channel configuration, organizations PUR and SCV can add organization Auditors (AUD). For organization AUD to be able to participate in the channel, an application AAUD is added. As the use case diagram shows before any changes are made on the network, participants must authenticate, while the CA for Auditors is created (CA AUD). The AUD, AAUD, and CA AUD are represented by dotted lines as it is added by the organizations SCV and PUR when necessary and is removed once the audit has been finished. Creation of organizations that will be part of the blockchain network, reduces the risk of tampering with software components that are incorporated into the system as COTS or tailor-made components, as all actors are known in advance and are authenticated, while the blockchain technology provides a tamper-proof solution that guarantees that no information is changed once written on the ledger. All these features combined, allow compliance with requirements SM-9 and SM-10.

Applications APUR and ASCV contribute to the security questionnaire by uploading the document on the IPFS, after which they will receive a file hash-code which is then stored on Channel CH. Utilizing IPFS for storing files, the blockchain network remains solely for storing transactions and maintaining the world state. Versioned security questionnaire documents will be available for download from IPFS, while the information about the version number, last update, and last modifier will be securely stored on the Channel CH, within the smart contract Security Questionnaire. Both organizations SCV and PUR can contribute to security questionnaire through applications ASCV and APUR, which have interfaces for communication with IPFS and blockchain network.

### Application and Smart Contracts

Following the Supply-chain management network architecture shown in the previous chapter, a small ExpressJS application was created. ExpressJS is a NodeJS framework, used for creating server-side web applications. The Ethereum network is a public blockchain network that provides vast number of tools available for creating and verifying blockchain smart contracts. One of the most popular test networks for Ethereum is the Ropsten network, allowing users to test their smart contracts without the need to invest in any platform or online service. For this prototype implementation, QuickNode as an Ethereum node was used, connected to the Ropsten test network. Utilizing Ropsten test network for prototyping, provides users an easy and unrestricted option of interacting with the blockchain. The architecture presented in Figure 7 includes IPFS, as a method of uploading documents, so for the prototype, a local IPFS node was used.

The smart contract named `Document` is a representation of the document that will be stored on the IPFS. On the ledger, the uploaded hash, as well as the document name and owner will be stored. `Document` constructor is created for instantiating a new `Document` with the given `documentName_` and `uploadedIPFSHash_`, while the `owner` of that created `Document` will be extracted from the global attribute `msg.sender`. The contract `SecurityQuestionnaires` is created for keeping the collection of documents, implemented as a `mapping` of `string` to a `Document`. Function `upload_questionnaire_hash` is called once the document hash is obtained from the IPFS and the `queryDocumentByName` function is called for retrieving document details. Following code snippet is written in Solidity programming language and provides insight on how smart contracts for `SecurityQuestionnaires` and `Document` are implemented.

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.13;

contract Document {
    address owner;
    string documentName;
    string uploadedIPFSHash;

    constructor(string memory documentName_, string memory
uploadedIPFSHash_){
        owner = msg.sender;
        documentName = documentName_;
        uploadedIPFSHash = uploadedIPFSHash_;
    }
}

contract SecurityQuestionnaires {
    mapping (string => Document) documents;

    function upload_questionnaire_hash(string memory documentHash_,
string memory documentName_) public {
        documents[documentName_] = new Document(documentHash_,
documentName_);
    }

    function queryDocumentByName(string memory documentName_) public
view returns (Document) {
        return documents[documentName_];
    }
}
```

The following snippet is written in ExpressJS and displays libraries for interaction with IPFS, Ethereum network and file system, which are introduced in the beginning of snippet and initialized with appropriate configurations. Provided `filePath` is used for creating a buffer which is uploaded to IPFS. The `url` is the QuickNode's endpoint to the Ropsten test network, which is used to create a `customHttpProvider` which is needed for creating a `signer`. The `signer`, together with the `address` of the smart contract and `abi`, is used for creating an object `contract`, which is used for calling the methods from the smart contract previously explained. Once the upload of the document is complete, IPFS returns the hash of that document, which is then uploaded to the `security_questionnaires` smart contract.

```
  var ipfsClient = require('ipfs-http-client');
  var ethers = require('ethers');
  var fs = require('fs');
  var ipfs = ipfsClient.create('http://localhost:5001')
  var url = QUICK_NODE_ENDPOINT;
  var customHttpProvider = new ethers.providers.JsonRpcProvider(url);
  var address = 'SMART_CONTRACT_ADDRESS';
  var abi = ABI_JSON;
  var signer = new ethers.Wallet(ethers.Wallet.fromMnemonic("PRIVATE_KEY,
customHttpProvider);
  var contract = new ethers.Contract(address, abi, signer);

  const uploadDocument = async function (filePath) {
      var testFile = fs.readFileSync(filePath);
      var testBuffer = Buffer.from(testFile);
      var uploadResult = await ipfs.add(testBuffer);
      contract.upload_questionnaire_hash(filePath, uploadResult);
  }
```

The snippets provided for the smart contracts and the ExpressJs display the most important part of the code which is needed for supporting the basic use case of uploading the document to the IPFS and storing the obtained hash on the ledger.

### 4.3. Discussion

The proposed model for tracking security requirements proposes utilization of Hyperledger Fabric, as a private permissioned blockchain network that enables only preauthorized users to contribute to the network. Though several papers [47], [48], [49] propose how requirements from the IEC 62443-4-1 standard can be tracked, our proposed blockchain architecture enables tamper-proof solution which is supported by blockchain basis. The authors [23] explore the idea of utilizing blockchain technology for tracking security requirements, but the paper doesn't address security requirements from IEC 62443-4-1 standard. The value of our work lies on the idea of utilizing private permissioned blockchain network for tracking security requirements that are critical when implementing secure lifecycle development for ICS. The ICS, such as smart grids, can benefit from this approach as it provides a solution which cannot be tampered with, leading to improved security posture, as well as plainer certifications. The certification process can be assisted by our solution, giving the auditors a timestamped and verifiable information about compliance with security requirements that have been stored on the blockchain. Private permissioned blockchain network should be utilized as the information stored on the blockchain should remain available only to preauthorized set of users. Though this paper presented a prototype implementation on the Ropsten network, i.e., one of Ethereum's test networks, the same principles can be applied to the Hyperledger Fabric. The public availability of smart contracts that have been added to the Ropsten test network enables easier verification of the prototype and simpler collaboration in this prototype phase. As this prototype implementation lacks the configuration that is needed for creating consortiums, which is only available in the Hyperledger Fabric, such improvements are planned in future work.

## 5. Conclusion

The ever-increasing cybersecurity threats that are emerging from various inputs, can be addressed by utilizing security practices, which are incorporated into the security development lifecycle (SDL) requirements. Implementation of those requirements involves providing evidence that the compliance has been met. The tamper-proof traceability that blockchain technology provides out-of-the-box, enables various interested parties, such as Auditors, to verify that the compliance with requirements, has been met. In this paper, we have presented a private permissioned, blockchain-based model, so that organizations can track and manage security requirements throughout a secure development lifecycle, which would allow ICS to promote cooperation and trust among different parties. Apart from the proposed blockchain-based model and universal architecture, an evaluation of the model was done against 13 requirements from the Security Management practice, from IEC 62443-4-1 Secure product development lifecycle. That evaluation involved discussion of the activities that have been set by the proposed model for the Project Manager, Security Advisor, and Blockchain Administrator. As the Project Manager has chosen the Security Management practice, the Security Advisor created four use cases that correspond to the grouped requirements from the SM practice. Those use cases were the inputs for the Blockchain Administrator to create an architecture for the supply-chain management use case. Within each use case, a possibility of allowing the Auditors to inspect the compliance with the requirements was given through modifying the consortiums, allowing Auditors to read tamper-proof information stored on the ledger. By utilizing private-permissioned blockchain technology, participants in this network are pre-authorized and have predefined rights. The future work shall include the design of guidelines for smart contracts that would enable further development of the architecture among all parties that are part of the supply chain process verification. Also, the usability of this framework for other security related standards, such as ISO 27001, shall be discussed.

## References

1. Zhivich, Michael, and Robert K. Cunningham. "The real cost of software errors." IEEE Security & Privacy 7.2 (2009): 87-90.
2. Graham, J., Hieb, J., & Naber, J. (2016, June). Improving cybersecurity for industrial control systems. In 2016 ieee 25th international symposium on industrial electronics (isie) (pp. 618-623). IEEE.
3. Neumann, Peter G. "Risks to the public in computers and related systems." ACM SIGSOFT Software Engineering Notes 29.2 (2004): 8-16.
4. McLaughlin, Stephen, et al. "The cybersecurity landscape in industrial control systems." Proceedings of the IEEE 104.5 (2016): 1039-1057.
5. Smith, Don C. "Cybersecurity in the energy sector: are we really prepared?." (2021): 265-270.
6. Morris, Thomas H., and Wei Gao. "Industrial control system cyber attacks." In 1st International Symposium for ICS & SCADA Cyber Security Research 2013 (ICS-CSR 2013) 1, pp. 22-29. 2013.

7. Drias, Zakarya, Ahmed Serhrouchni, and Olivier Vogel. "Analysis of cyber security for industrial control systems." In 2015 international conference on cyber security of smart cities, industrial control system and communications (ssic), pp. 1-8. IEEE, 2015.
8. Maglaras, Leandros A., et al. "Cyber security of critical infrastructures." Ict Express 4.1 (2018): 42-45.
9. Nasir, Muhammad Ali, Shizra Sultan, Samia Nefti-Meziani, and Umar Manzoor. "Potential cyber-attacks against global oil supply chain." In 2015 International Conference on Cyber Situational Awareness, Data Analytics and Assessment (CyberSA), pp. 1-7. IEEE, 2015.
10. Al-Haija, Qasem Abu, and Swastik Brahma. "Optimization of Cyber System Survivability Under Attacks Using Redundancy of Components." In 2019 53rd Annual Conference on Information Sciences and Systems (CISS), pp. 1-6. IEEE, 2019.
11. Graham, James, Jeffrey Hieb, and John Naber. "Improving cybersecurity for industrial control systems." In 2016 ieee 25th international symposium on industrial electronics (isie), pp. 618-623. IEEE, 2016.
12. IEC: 62443-4-1. Security for industrial automation and control systems Part 4-1 Product security development life-cycle requirements (2018)
13. Haley, Charles B., Jonathan D. Moffett, Robin Laney, and Bashar Nuseibeh. "A framework for security requirements engineering." In Proceedings of the 2006 international workshop on Software engineering for secure systems, pp. 35-42. 2006.
14. Pandey, Dhirendra, Ugrasen Suman, and A. Kumar Ramani. "An effective requirement engineering process model for software development and requirements management." In 2010 International Conference on Advances in Recent Technologies in Communication and Computing, pp. 287-291. IEEE, 2010.
15. Mishra, Deepti, Alok Mishra, and Ali Yazici. "Successful requirement elicitation by combining requirement engineering techniques." In 2008 First International Conference on the Applications of Digital Information and Web Technologies (ICADIWT), pp. 258-263. IEEE, 2008.
16. Fiorineschi, Lorenzo, et al. "Testing a new structured tool for supporting requirements' formulation and decomposition." Applied Sciences 10.9 (2020): 3259.
17. Gupta, Varun, et al. "Requirements engineering in software startups: A systematic mapping study." Applied Sciences 10.17 (2020): 6125.
18. Mengist, Alachew, Lena Buffoni, and Adrian Pop. "An Integrated Framework for Traceability and Impact Analysis in Requirements Verification of Cyber–Physical Systems." Electronics 10.8 (2021): 983.
19. Rehman, Shafiq Ur, and Volker Gruhn. "An effective security requirements engineering framework for cyber-physical systems." Technologies 6.3 (2018): 65.
20. Golosova, Julija, and Andrejs Romanovs. "The advantages and disadvantages of the blockchain technology." In 2018 IEEE 6th workshop on advances in information, electronic and electrical engineering (AIEEE), pp. 1-6. IEEE, 2018.
21. https://hyperledger-fabric.readthedocs.io/en/release-2.3/glossary.html, accessed August 2021
22. https://developer.ibm.com/technologies/blockchain/articles/blockchain-basics-hyperledger-fabric/, accessed August 2021.
23. Demi, Selina. "Blockchain-oriented requirements engineering: A framework." In 2020 IEEE 28th International Requirements Engineering Conference (RE), pp. 428-433. IEEE, 2020.
24. Demi, Selina, Ricardo Colomo-Palacios, and Mary Sánchez-Gordón. "Software Engineering Applications Enabled by Blockchain Technology: A Systematic Mapping Study." Applied Sciences 11.7 (2021): 2960
25. Woon, Irene MY, and Atreyi Kankanhalli. "Investigation of IS professionals' intention to practise secure development of applications." International Journal of Human-Computer Studies 65.1 (2007): 29-41.

26. Weider, D. Yu, and Kyle Le. "Towards a secure software development lifecycle with square+ r." In 2012 IEEE 36th Annual Computer Software and Applications Conference Workshops, pp. 565-570. IEEE, 2012.
27. https://www.microsoft.com/en-us/securityengineering/sdl, accessed August 2021.
28. https://us-cert.cisa.gov/bsi/articles/best-practices/requirements-engineering/introduction-to-the-clasp-process, accessed August 2021.
29. https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-64r2.pdf, accessed August 2021.
30. https://csrc.nist.gov/publications/detail/sp/800-160/vol-1/final, accessed August 2021.
31. Lipner, Steve. "The trustworthy computing security development lifecycle." In 20th Annual Computer Security Applications Conference, pp. 2-13. IEEE, 2004.
32. Kainerstorfer, Michael, Johannes Sametinger, and Andreas Wiesauer. "Software security for small development teams: a case study." In Proceedings of the 13th International Conference on Information Integration and Web-based Applications and Services, pp. 305-310. 2011.
33. Rindell, Kalle, Sami Hyrynsalmi, and Ville Leppänen. "Aligning security objectives with agile software development." In Proceedings of the 19th International Conference on Agile Software Development: Companion, pp. 1-9. 2018.
34. Gregoire, Johan, Koen Buyens, Bart De Win, Riccardo Scandariato, and Wouter Joosen. "On the secure software development process: CLASP and SDL compared." In Third International Workshop on Software Engineering for Secure Systems (SESS'07: ICSE Workshops 2007), pp. 1-1. IEEE, 2007.
35. Rindell, Kalle, Sami Hyrynsalmi, and Ville Leppänen. "Aligning security objectives with agile software development." In Proceedings of the 19th International Conference on Agile Software Development: Companion, pp. 1-9. 2018.
36. Roudiès, Ounsa. "Benchmarking SDL and CLASP lifecycle." In 2014 9th International Conference on Intelligent Systems: Theories and Applications (SITA-14), pp. 1-6. IEEE, 2014.
37. Venson, Elaine, Xiaomeng Guo, Zidi Yan, and Barry Boehm. "Costing secure software development: A systematic mapping study." In Proceedings of the 14th International Conference on Availability, Reliability and Security, pp. 1-11. 2019.
38. Maier, Patrik, Zhendong Ma, and Roderick Bloem. "Towards a secure scrum process for agile web application development." In Proceedings of the 12th International Conference on Availability, Reliability and Security, pp. 1-8. 2017.
39. Thomas, Tyler W., Madiha Tabassum, Bill Chu, and Heather Lipford. "Security during application development: An application security expert perspective." In Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems, pp. 1-12. 2018.
40. Hahn, Adam, Ali Tamimi, and Dave Anderson. "Securing your ics software with the attacksurface host analyzer (aha)." In Proceedings of the 4th Annual Industrial Control System Security Workshop, pp. 33-39. 2018.
41. Moyón, Fabiola, Daniel Méndez, Kristian Beckers, and Sebastian Klepper. "How to integrate security compliance requirements with agile software engineering at scale?." In International Conference on Product-Focused Software Process Improvement, pp. 69-87. Springer, Cham, 2020.
42. Walden, James, and Charles E. Frank. "Secure software engineering teaching modules." In Proceedings of the 3rd annual conference on Information security curriculum development, pp. 19-23. 2006.
43. Luburić, Nikola, et al. "A framework for teaching security design analysis using case studies and the hybrid flipped classroom." ACM Transactions on Computing Education (TOCE) 19.3 (2019): 1-19.
44. Mylrea, Michael, and Sri Nikhil Gupta Gourisetti. "Blockchain: Next generation supply chain security for energy infrastructure and nerc critical infrastructure protection (cip) compliance." Resilience Week 16 (2018).

45. Mylrea, Michael, Sri Nikhil Gupta Gourisetti, Randy Bishop, and Matt Johnson. "Keyless signature blockchain infrastructure: Facilitating nerc cip compliance and responding to evolving cyber threats and vulnerabilities to energy infrastructure." In 2018 IEEE/PES Transmission and Distribution Conference and Exposition (T&D), pp. 1-9. IEEE, 2018.
46. Mylrea, Michael, and Sri Nikhil Gupta Gourisetti. "Blockchain for supply chain cybersecurity, optimization and compliance." In 2018 Resilience Week (RWS), pp. 70-76. IEEE, 2018.
47. Moyon, Fabiola, Kristian Beckers, Sebastian Klepper, Philipp Lachberger, and Bernd Bruegge. "Towards continuous security compliance in agile software development at scale." In 2018 IEEE/ACM 4th International Workshop on Rapid Continuous Software Engineering (RCoSE), pp. 31-34. IEEE, 2018.
48. Dännart, Sebastian, Fabiola Moyón Constante, and Kristian Beckers. "An assessment model for continuous security compliance in large scale agile environments." In International Conference on Advanced Information Systems Engineering, pp. 529-544. Springer, Cham, 2019.
49. Moyón, Fabiola, Christoph Bayr, Daniel Mendez, Sebastian Dännart, and Kristian Beckers. "A light-weight tool for the self-assessment of security compliance in software development–an industry case." In International Conference on Current Trends in Theory and Practice of Informatics, pp. 403-416. Springer, Cham, 2020.
50. Nyaletey, Emmanuel, et al. "BlockIPFS-blockchain-enabled interplanetary file system for forensic and trusted data traceability." 2019 IEEE International Conference on Blockchain (Blockchain). IEEE, 2019.

**Jelena Marjanović (née Stankovski)** received her B.Sc. in 2015. and her M.Sc. in 2016 from the Faculty of Technical Sciences, University of Novi Sad. From 2016, she is a PhD student on Faculty of Technical Sciences, University of Novi Sad. Currently, she is a Teaching Assistant on the Faculty of Technical Sciences, University of Novi Sad, and a Senior Cybersecurity Analyst in the Energy Management Software Solutions Industry. Her research interests include blockchain, software engineering and cybersecurity.

**Nikola Dalčeković** received the B.Sc., M.Sc., and Ph.D. degrees in computer science from the Faculty of Technical Sciences, University of Novi Sad, in 2012, 2013, and 2019, respectively. Currently, he is a Product Security Officer in the Energy Management Software Solutions Industry. He worked as a Teaching Assistant in computer science with the Faculty of Technical Sciences, University of Novi Sad, from 2014 to 2020. At the same Faculty, he continued his career as an Assistant Professor from 2020 until 2022 when he decided to move to California, United States to help with industry innovation in the cybersecurity area. His research interests include cybersecurity, software engineering, and distributed computing.

**Goran Sladić** received the Ph.D. degree from the University of Novi Sad, in 2011. He is currently a Professor of computer science with the Faculty of Technical Sciences, University of Novi Sad. He has published over 80 articles and participated or lead in more than 20 projects. His research interests include cyber security, blockchain, software engineering, software architectures, and context-aware computing.

*Received: September 23, 2021; Accepted: August 04, 2022.*