QualSec: An Automated Quality-Driven Approach for Security Risk Identification in Cyber-Physical Production Systems

Matthias Eckhart, Andreas Ekelhart, Stefan Biffl, *Member, IEEE*,
Arndt Lüder, *Senior Member, IEEE*, and Edgar Weippl, *Senior Member, IEEE*

***Abstract*— As the threat landscape in the industrial domain continually advances, security-by-design is an ever-growing concern in the engineering of cyber-physical production systems (CPPSs). Often, quality aspects are not considered when securing CPPSs, which creates attack vectors that could lead to malicious activity affecting the products' quality. Since quality control systems generally provide inadequate protection against intentionally introduced defects, and can be susceptible to attacks, quality considerations must be integrated into security-aware CPPS engineering. For this purpose, we propose the QualSec method that automatically identifies security risks pertaining to CPPSs, building on the quality characteristics associated with manufacturing operations to determine cascading effects. QualSec is based on a semantic representation of engineering knowledge, allowing to efficiently reuse engineering models from AutomationML artifacts. Moreover, QualSec utilizes Petri nets to facilitate the analysis of security risks and cascading effects. In this way, QualSec informs users about possible attack paths for compromising quality characteristics, how attackers may disguise their malicious actions, and the possible consequences of attacks with respect to product quality. We demonstrate the benefits of QualSec in a case study and analyze its scalability through a rigorous performance evaluation.**

***Index Terms*— Cyber-physical production systems, information security, industrial control systems, AutomationML, Petri net, production systems engineering.**

The COMET center SBA Research (SBA-K1) is funded within the framework of COMET — Competence Centers for Excellent Technologies by BMVIT, BMDW, and the federal state of Vienna, managed by the Austrian Research Promotion Agency (FFG). This work has been partially supported and funded by the FFG via the Austrian Competence Center for Digital Production (CDP) under the contract number 881843 and via the Bridge 1 program under the grant number 880609. Moreover, the financial support by the Christian Doppler Research Association, the Austrian Federal Ministry for Digital and Economic Affairs and the National Foundation for Research, Technology and Development is gratefully acknowledged.

M. Eckhart, A. Ekelhart, and E. Weippl are with SBA Research, 1040 Vienna, Austria, and also with the Christian Doppler Laboratory for Security and Quality Improvement in the Production System Lifecycle, University of Vienna, 1090 Vienna, Austria (e-mail: meckhart@sba-research.org; aekelhart@sba-research.org; edgar.weippl@univie.ac.at).

S. Biffl is with Inst. of Information Systems Engineering, TU Wien, Austria, and also with CDP, Austria (e-mail: stefan.biffl@tuwien.ac.at).

A. Lüder is with Institute of Ergonomics, Manufacturing Systems and Automation, Otto-von-Guericke U., Germany, and also with CDP, Austria (e-mail: arndt.lueder@ovgu.de).

I. Introduction

SINCE new threats that can compromise the secure and safe operation of cyber-physical production systems (CPPSs) are continuously emerging, managing security risks at the beginning of the systems' lifecycle is paramount. This requires, on the one hand, that the individual engineering activities (e.g., software development and testing [1]), including the exchanged artifacts, are sufficiently protected against adversaries [2]. On the other hand, security must be established as a 'first-class citizen' in the engineering process to achieve CPPSs that are secure by design [3]. In the latter case, knowledge from diverse domain experts is essential, given that the engineering of CPPSs is by itself a highly multi-disciplinary endeavor. The cyber-physical nature of attacks launched against CPPSs further underlines this need: Attacks executed from cyberspace can lead to physical harm and may endanger human life. Thus, both security and safety concerns need to be considered jointly. In this context, it is worth pointing out that quality is likewise interdependent with security but often not perceived as such. For instance, security risks may manifest themselves as symptoms of a quality control (QC) issue (e.g., data integrity breach due to poor handling of QC logbooks), meaning that addressing this underlying problem could also improve the overall security. Conversely, strengthening the security of a CPPS may also lead to higher quality (e.g., additional sensors put in place to prevent covert product modifications may at the same time unveil defects). Recognizing this interdependence may not only help to promote the fact that information security adds value to an organization (in this case, realized via quality improvements) but also increases the awareness of cyberattacks that focus on the quality of the manufactured products.

The potential severity and multi-dimensional characteristic of sabotage attacks targeting product quality necessitate a holistic security risk assessment approach that also incorporates quality considerations. However, current risk assessment workflows defined in leading industrial security standards and guidelines (e.g., IEC 62443-3-2 [4] or VDI/VDE 2182-1 [5]) adopt a rather resource-centric view, neglecting the product and process components. This leads to an incomplete understanding of security risks that is also inconsistent with the Product, Process, and Resource (PPR) concept [6], which

plays a predominant role in the engineering of CPPSs. In other words, engineers currently consider quality concerns without assuming intentional wrongdoing (i.e., in isolation from security concerns). This isolated view weakens both quality controls and security controls.

Moreover, given the vast complexity of designing secure CPPSs, systems integrators need a highly efficient security risk identification method that leverages the data and models that emerge during the engineering process.

The article at hand aims to remedy these pressing issues. Building upon prior work [7], the QualSec method presented in this paper interprets the interlinking of PPR engineering information to automatically identify (i) critical quality characteristics of products, (ii) attack steps to compromise them, and (iii) the resulting consequences on the production process. Since the method can be seamlessly embedded into existing toolchains and makes direct use of already available engineering knowledge contained in AutomationML artifacts, the effectiveness and efficiency of the security risk identification step can be raised significantly. Further, the method automatically generates Petri nets that model the sequence of manufacturing processes in a quality-oriented way, allowing to employ reachability analysis that supports risk identification.

The main contributions of this work are as follows:

- We propose QualSec, that is, a quality-driven method for the automated identification of security risks sourced from engineering models of CPPSs. QualSec draws upon PPR information, including the sequences of manufacturing steps, to thoroughly inform about security risk sources and consequences.
- We present a quality ontology that contains the QC domain knowledge available in production systems engineering (PSE). This ontology enriches semantics-based security risk assessments and can be interlinked with other ontologies to build knowledge graphs (KGs) for security applications.
- We introduce the notion of a quality-oriented Petri net (QOPN) to represent the relationships between manufacturing operations, quality control steps, and cyberattacks.
- We provide an open-source implementation of QualSec, test its practicality by conducting a case study, and analyze its scalability via a rigorous performance evaluation.

To the best of our knowledge, this is the first work that considers the relationship between quality and security in a risk identification context with an emphasis on the PPR concept, making it highly relevant to the industrial informatics and information security communities.

The rest of this paper is organized as follows. Section II provides background information and discusses related work. In Section III, we motivate the need for quality-driven security risk identification and define the scope of QualSec. Then, in Section IV, we explain the details of our novel method. Section V demonstrates the benefits and practicality of the introduced method by means of a case study. After that, in Section VI, we discuss the results of our performance evaluation. Finally, in Section VII, we conclude our work and give an outlook on future research.

II. Background and Related Work

In this section, we briefly review background information on AutomationML and QC in the context of cyber-physical systems (CPSs) and discuss related work on supporting security-aware CPPS engineering.

A. AutomationML

The Automation Markup Language (AutomationML, hereafter abbreviated as AML) is an XML-based data format that aims to improve the data exchange among heterogeneous engineering tools [8]. This format harmonizes and unifies data models of different engineering disciplines by integrating the Computer Aided Engineering Exchange (CAEX) data format, COLLADA, and PLCopen XML to enable modeling of the topology, geometry and kinematics, and behavior and sequencing of the CPPS [9]. The reason for utilizing AML artifacts to implement the automated identification of quality-driven security risks in CPPSs is threefold: First, AML has been standardized in the IEC 62714 series and gained wide acceptance within the CP(P)S engineering community, many of whom have joined the AutomationML association[1] to develop the format further. Second, the scope of AML far exceeds the mere exchange of information by enabling a model-based engineering approach [10]. Third, the Product, Process, and Resource (PPR) concept fits naturally into the AML architecture as a way of structuring plant models [6]. Thus, the interlinking of information regarding products (e.g., features, quality requirements), processes (e.g., sequencing of manufacturing steps), and resources (e.g., physical and logical objects, networks) can be directly harnessed for risk assessment purposes.

B. The Role of Quality Control in Cyber-Physical Systems Security

Surprisingly, little scholarly work has focused on quality control in the context of CPS security thus far. However, of the few works published in this area, we consider the papers by Elhabashy *et al.* [11], [12] to be most relevant to the article at hand. In [11], the authors proposed a taxonomy of cyber-physical attacks involving QC systems, which is composed of (i) attack objectives, (ii) targeted components, (iii) attack methods, and (iv) attack locations. Their subsequent work [12] reveals that QC systems may have numerous potential vulnerabilities and shortcomings that attackers can passively exploit (i.e., without changing the QC systems themselves). The findings presented in [11], [12] highlight the importance of adopting a QC perspective when assessing security risks and, therefore, strongly motivate the proposed method.

An interesting observation reported by Wells *et al.* [13] is that there is a significant need to raise awareness about cyber-attacks that have an adverse effect on product quality. Their finding suggests that security needs to be firmly established in the engineering and quality improvement process to become a natural part of the engineer's work. For this reason, our method

[1] https://www.automationml.org

is designed to allow tight integration into the engineering environment.

Other works, such as [14], [15], analyze sabotage attacks in additive manufacturing (AM) processes. Sturm *et al.* [14] explore different attack vectors in AM that cybercriminals may use to trick systems into producing faulty products. In particular, they conducted a case study to investigate how `STL` files can be manipulated such that voids inside the produced parts are created. The authors of [14] accentuate that void attacks in AM are typically difficult to detect and may cause a loss of structural integrity. Belikovetsky *et al.* [15] demonstrate a complete attack scenario involving an AM process, targeting the 3D-printed propellers of a quadcopter. This attack is particularly devious, as the introduced defects remain unnoticed by basic quality checks and cause a critical failure after a certain amount of operating time. Both publications simulate realistic threat scenarios that challenge the state of how product quality issues can be mitigated in the event of an attack, thereby motivating a quality-driven consideration of security risks in CPPSs.

C. Model-based Security Risk Identification in Cyber-Physical Systems

Several model-driven, risk-based approaches have been proposed in the past years that aim to support the engineering of secure CPPSs. In the following, we briefly summarize the most relevant works.

In [16], [17], Apvrille and Roudier present an extension for the Systems Modeling Language (SysML) named *SysML-Sec*, which facilitates the model-driven design of safe and secure (sub-)systems (e.g., embedded systems). This extension enables users to incorporate security and safety properties into SysML models, which can then be validated by means of formal verification and simulation. Another security extension for SysML was introduced in [18], which focuses primarily on the architectural aspects of industrial control systems (ICSs), such as CPPSs, rather than the design of the systems' individual components (e.g., a controller). Lemaire *et al.* [19], [20] have further improved the security-aware, model-based engineering of ICSs by utilizing a formal reasoning framework to automate the identification of security risks in SysML models.

Besides SysML, researchers have also investigated AML for the purpose of extracting relevant information from CPPS blueprints to automate security risk assessments. In [21]–[23], a knowledge-based approach was introduced that applies security rules to AML artifacts in order to discover vulnerabilities in engineering models. These rules were created based on security domain knowledge [24] and modeled with the Web Ontology Language (OWL) and the Semantic Web Rule Language (SWRL). In this context, it is worth noting that their approach directly accesses the engineering data in AML without converting it to OWL.

Recently, Eckhart *et al.* [7] proposed a new method that further advanced this research area. Their method employs an AML-to-OWL transformation mechanism, enabling semantic interlinking and the use of semantic technologies (e.g., applying semantic reasoning to infer new knowledge). In this way, the method is able to identify threats, vulnerabilities, and consequences automatically by executing a set of queries and rules, which were written in the SPARQL Protocol and RDF Query Language (SPARQL) and the Shapes Constraint Language (SHACL), respectively. The results of the risk identification then serve as an input for the automated generation of attack graphs, which visualize the most critical paths adversaries may take when launching cyberattacks against CPPSs. In the paper at hand, we build upon the approach described in [7] to automate the identification of quality-driven security risks in CPPS engineering models.

Finally, it is worth noting that researchers have also applied Petri nets (PNs) for security analysis purposes [25], [26]. Henry *et al.* [27], [28] employ PNs for attack analysis in the context of ICSs. The authors of [27], [28] then use coverability analysis to determine the extent to which an adversary can gain unauthorized access to resources. Ten *et al.* [29] use generalized stochastic Petri nets (GSPNs) as part of a framework that aims to quantify the vulnerability of power systems. In comparison to [27]–[29], our proposed method has a clear focus on the quality aspects of the produced parts and provides a significant level of automation in terms of risk identification.

III. Considered Attack Scenario and Scope of QualSec

The attack model considered in the article at hand assumes resourceful adversaries capable of remaining under the radar until defective products caused by intentional sabotage slip through QC and are shipped to customers. Based on a casual review of past cyberattacks against CPPSs, we sketch a realistic scenario in which threat actors either gain their initial foothold within the business network and then pivot to the control system network or directly gain unauthorized access to control devices via unprotected remote maintenance services. Furthermore, we assume that adversaries attack the CPPS at the weakest point they can find, which commonly coincides with exploiting publicly-known vulnerabilities. The objective of attackers is to compromise manufacturing systems during operation in order to cause product quality issues deliberately. From an attacker's perspective, overcoming QC that functions as a defense against such attacks can be achieved in two ways: Either by manipulating the products' quality characteristics selectively, affecting only those which are not subject to quality inspection, or by exploiting QC vulnerabilities [12] to avoid detection of malicious product alterations.

Fig. 1 illustrates an example scenario in which an adversary attacks a vulnerable programmable logic controller (PLC) ❶ in a car manufacturing process. Since the compromised PLC controls a spot welding robot, the adversary can induce subtle changes in the welds, resulting in loss of product integrity (e.g., poor durability of the produced car body) and eventual failure of the vehicle. The consequence of this cyber-physical attack remains undetected throughout the manufacturing process as subsequent inspection for the purpose of QC can be evaded. The reason for this is that QC systems are typically not designed to uncover issues that have been created with

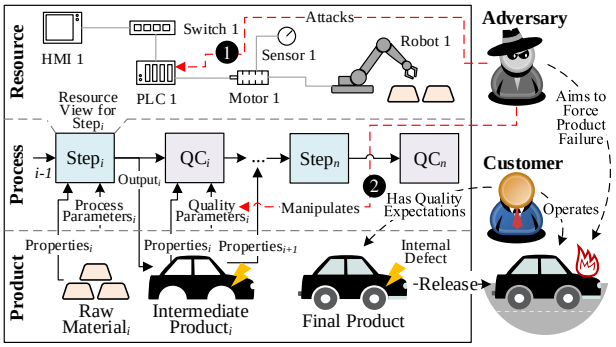


Fig. 1: Attack scenario targeting the products' quality.

malicious intent [13]. Even if the implemented QC checks would detect malicious product changes, an adversary may also exploit the QC systems to manipulate quality parameters (e.g., inspection locations, thresholds) ❷, ensuring that any modifications go unnoticed [12]. Furthermore, increasing the defect rate and disrupting production processes constitute additional attack objectives that adversaries may pursue [11].

Our novel method, named QualSec, aims to automate tasks of the risk identification step that are carried out as part of security risk assessments during the engineering of CPPSs. One of its core features is to incorporate the semantics, structure, and sequence of the manufacturing process to identify (i) product quality characteristics that attackers may compromise, and (ii) possible propagation effects thereof. To illustrate the scope and purpose of our contribution, we define the following set of questions:

Q1 *What are the security vulnerabilities in assets of CPPSs that threats may exploit?*
The first question aims to uncover architectural security weaknesses and vulnerabilities in systems that are intended to be integrated into the plant topology. Answers to this question build upon public sources, such as Common Vulnerabilities and Exposures (CVE), security advisories, and industrial security standards and guidelines. We repurpose the method presented in [7] to enable a quality-driven consideration of cyber-physical risk that is realized by answering the next questions.

Q2 *Given a set of vulnerable assets, which quality characteristics of the workpiece or product can attackers deliberately alter, and would these defects remain undetected due to insufficient QC?*
Based on the answer given to Q1, this question aims to inform engineers about potential consequences on product quality that may be caused by an adversary who exploits vulnerable assets to execute such sabotage attacks. Answers to this question provide engineers guidance on how to prioritize risks.

Q3 *What are the consequences of an attack that targets a certain quality characteristic in terms of cascading effects relating to product quality?*
Similar to the previous question, Q3 focuses on the quality characteristics that adversaries may be able to influence in the course of an attack. However, as the sequence of manufacturing steps can create dependencies among quality attributes (e.g., diameter and location of drilled pilot holes must be correct for subsequent joining), this question places special emphasis on the indirect effects of sabotage attacks. As a result, engineers can quickly spot critical quality characteristics whose malicious alteration would lead to a chain reaction.

Q4 *How can attackers disguise their malicious actions to evade QC?*
Finally, the last question addresses the case where an adversary might attempt to attack those QC systems that would catch product defects caused by prior manipulations of quality characteristics. Informing engineers about the minimal set of assets needed to be hacked to bypass the QC in place may provide guidance on prioritizing the systems to be hardened.

IV. Method

An overview of our proposed method and its steps is shown in Fig. 2. In the course of engineering CPPSs, professionals from various disciplines design and model systems using specialized tools. The created engineering artifacts are managed in the AML format to facilitate data exchange. In step ❶, engineers annotate the plant topology contained in the AML document with security- and quality-relevant information using the AML extension libraries (AMLsec and AMLqual). Step ❷ transforms both the plant topology and the description of the manufacturing process to OWL. Step ❸ builds the Knowledge Base (KB) by connecting the semantic representation of the plant topology and production process with additional know-how from the security ontology [30], the ICS security ontology [7], the quality ontology, and linked open security data. Based on the process description contained in the KB, step ❹ generates the QOPN. Finally, step ❺ automatically performs the quality-driven security risk identification by executing rules and queries against the KB and analyzing the QOPN.

Before we explain each element of QualSec in detail, we state the assumptions that the QualSec method relies on:

- *Risk Identification at Design Time:* As the purpose of QualSec is to reveal security risks in the CPPS during the engineering process, we only consider what the QC system can check at design time.
- *Model of the Manufacturing Process:* It is assumed that the manufacturing process is modeled in the sequential function chart (SFC) language in line with the PLCopen XML specification. To construct the QOPN, we only consider the structure of the SFC network, which can be represented graphically. Other elements of the SFC language, as standardized in the IEC 61131-3, are not relevant to QualSec.
- *State of a System is Binary:* If an attack against a production system succeeds, it is assumed that the adversary gains full control and can manipulate all quality characteristics that the compromised system can influence during the respective manufacturing step. Similar reason-

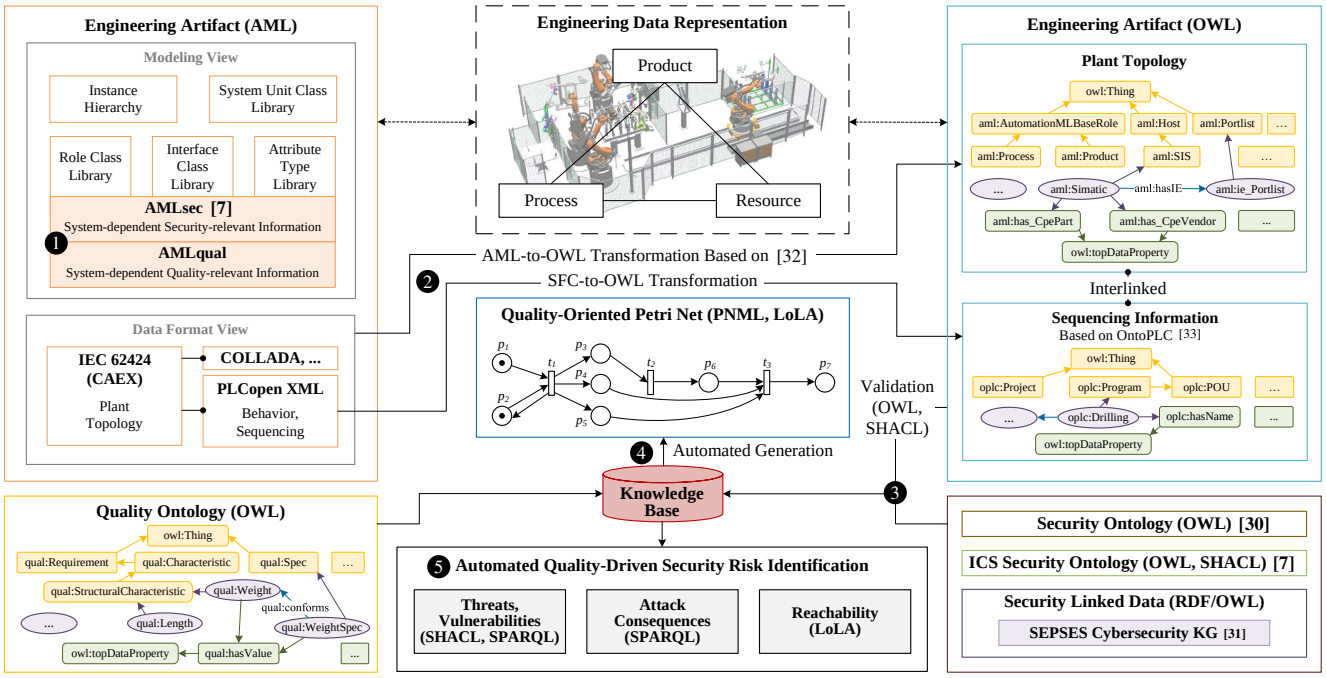


Fig. 2: Overview of QualSec, the quality-driven security risk identification method (based on [7]; robot cell illustration in [34]).

ing applies to QC systems and the outcome of quality checks.

- *Quality Measurements are Performed In-Line:* Since QualSec incorporates the description of the manufacturing process, we only consider QC efforts that are undertaken along the production line and are modeled as such. Off-line quality checks could be accommodated by manually extending the semantic representation of the manufacturing process.

A. Engineering Data Representation

To lift the engineering models contained in AML artifacts to ontologies, we rely on the semantics expressed via AML's libraries of role classes (`RoleClassLib`), interface classes (`InterfaceClassLib`), and attribute types (`AttributeTypeLib`). More precisely, we link the semantics of components modeled in AML to an equivalent representation maintained in our ontologies. The normative libraries specified as part of AML are primarily used for this purpose, thereby reducing the additional modeling effort required to use QualSec. However, certain security-relevant modeling constructs that would significantly enhance QualSec's analysis capabilities are missing in those standard libraries. To overcome this limitation, we reuse AMLsec [7], which comprises libraries that engineers can apply to model security-relevant information (e.g., zones, network protocols, security devices). We carry the idea of realizing semantic matching one step further and introduce a set of libraries named AMLqual that engineers can use to augment their model with quality-relevant information. For example, `AMLqualRoleClassLib` includes, *inter alia*, role classes for QC methods (e.g., ultrasonic testing), to enrich the semantics of `InternalElements` that model the QC system.

Another vital aspect of QualSec is the interlinking of engineering information according to the PPR concept, which can be fully accommodated within the AML format [6]. According to the AML standard, links between modeled products, processes, and resources are established by using an `ExternalInterface` named `PPRConnector`, which is part of the `AutomationMLInterfaceClassLib`. Furthermore, objects within the logic model (i.e., the SFC program), which contains the sequencing information of the manufacturing process, are referenced from CAEX in the usual AML-way by using `LogicElementInterfaces`.

B. Ontological Modeling

As shown in Fig. 2, the KB is composed of the semantically lifted engineering model (i.e., plant topology and sequencing information), the (ICS) security ontology, the quality ontology, and the security-related linked data.

The CAEX-based plant topology within the AML artifact is transformed to OWL using the translation procedure of Hua and Hein [32]. To incorporate the PLCopen XML data into our KB, we have implemented an SFC-to-OWL transformation that instantiates an ontological model from OntoPLC [33]. After lifting the AML artifact to a semantic representation, we perform validation checks using SHACL and then automatically augment the engineering knowledge with security- and quality-specific know-how.

The structure of the security knowledge follows a layered approach, where the middle ontology layer is realized by the security ontology [30] that models rather abstract concepts within the information security domain. The ICS security

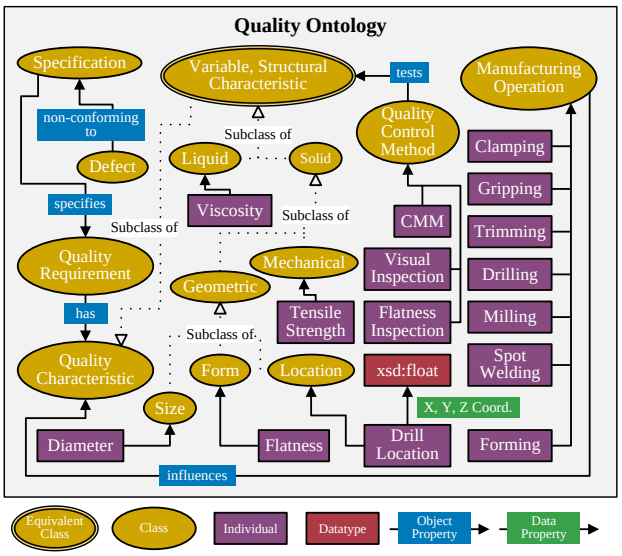


Fig. 3: Visualization of the quality ontology (excerpt).

ontology expands this basic knowledge with information obtained from system-independent (e.g., security standards and guidelines) and system-dependent (e.g., technical requirements of CPPSs) sources. Furthermore, the semantic data model within the KB is interlinked with the *SEPSES Cybersecurity KG* [31] in order to include the latest information on publicly disclosed security issues.

Another vital component of QualSec is the quality ontology. We have designed a comprehensive ontology for the QC domain to capture the knowledge of quality characteristics, methods to check them, and manufacturing processes that influence them (cf. Fig. 3). The rationale behind the quality ontology is to create semantic relations between the PPR information from the engineering model and QC domain knowledge. In this way, we can derive the information that is required to construct the QOPNs that enable quality-driven security risk identification.

To answer Q1, we apply a set of SHACL rules and SPARQL queries that are executed against the KB, yielding risk sources (i.e., threats and vulnerabilities) and attack consequences (i.e., violation of security or safety goals).[2] The employed vulnerability detection rules can be categorized into two classes: First, node and property shapes are used to implement a validation procedure that checks for security weaknesses in the modeled elements of the plant topology (e.g., insecure network protocols and cryptographic algorithms, configuration vulnerabilities). Second, SPARQL-based constraints are employed to detect violations of zone and conduit requirements (ZCR-3.2–3.6) as per the IEC 62443-3-2 [4]. Additionally, we perform a CVE check by using the SEPSES Cybersecurity KG [31] to determine if the systems intended to be integrated into the plant are affected by known (public) vulnerabilities.

[2]For a more detailed description of this approach, we refer readers to [7].

C. Quality-Oriented Petri Nets

The identification of risks to product quality and consequential events is based on the results of constructing and analyzing Petri nets (PNs) that model manufacturing processes. The PN [35] is a well-established formalism with decades of research behind it and represents a convenient tool to model discrete event systems (DESs). In the following, we introduce the notion of QOPNs, specify a generation method for QOPNs, and explain how QOPNs can be analyzed to support the identification of security risks.

1) Preliminaries: Following the definitions given in [36], a marked PN is defined as a 5-tuple (P, T, A, w, x), where (P, T, A, w) is a weighted bipartite graph comprising a finite set of places P, a finite set of transitions T, a set of arcs $A \subseteq (P \times T) \cup (T \times P)$, and a weight function on the arcs $w : A \to \{1, 2, 3, \dots\}$. Further, x is a marking of the set of places that is associated with a row vector $\mathbf{x} = [x(p_1), x(p_2), \dots, x(p_n)] \in \mathbb{N}^n$. The marking row vector $\mathbf{x}$ defines the state of the PN and a transition $t_j \in T$ is enabled, if and only if, $x(p_i) \geq w(p_i, t_j)\ \forall p_i \in I(t_j)$, where $I(t_j) = \{p_i \in P : (p_i, t_j) \in A\}$.

Recall that QualSec incorporates a formal representation of the manufacturing process that is first translated from SFC to OWL and then processed further to construct a QOPN. The beauty of QOPNs is that they capture the dependencies among process steps, quality characteristics, and attacks against them, leading to an enhanced understanding of propagation effects.

In general, a manufacturing process consists of n production steps $o_1, \dots, o_n$ that are executed by m production systems to fulfill l jobs. Each production step o influences h characteristics of the machined part or product, which are then checked by k quality control steps to determine whether they meet their stipulated quality specifications. Since the quality-driven security risk identification is performed from a process-centric point of view, the QOPN is based on the process-oriented Petri net (POPN) [37]. In a POPN, a place represents the status of a resource or job order, or an operation, while a transition denotes either the start or end of an operation [37]. The QOPN is a classical PN (P, T, A, w, x), as defined above, that extends the notion of the POPN. In Table I, we assign meaning to P and T to ensure proper interpretation of QOPNs.

It is worth reiterating that we do *not* aim to fully translate SFC programs in their complete form to PNs or one of the PN dialects. Instead, we utilize the sequencing information expressed via the SFC structure, which encodes the description of the manufacturing process, to construct QOPNs that aid security risk identification.

2) Modeling and Construction: A QOPN is composed of one or multiple QOPN templates that are assembled according to the formal process description at hand. To achieve a valid QOPN, the SFC model to be transformed must at least contain the sequence *Initial Step* → *Production Step* → *Terminal Step*, which leads to the template shown in Fig. 4.

The minimal QOPN depicted in Fig. 4 contains only one quality characteristic, q_1, and is shown in its initial state, where $x(u_1) = 1$ and $x(\bar{u}_1) = 0$ (the complement of $x(u_1)$) were specified arbitrarily for demonstration purposes. Note that the initial state of the generated QOPN depends on

TABLE I: Notation and semantics of QOPNs.

Places

P	$= \bigcup_{i=1}^{13} S_i$, $S_i \cap S_j = \varnothing$ for all $i, j \in \{1, \ldots, 13\}, i \neq j$, where
S_1	$= \{o_1, \ldots, o_n\}$ is a set of places denoting production steps,
S_2	$= \{r_1, \ldots, r_v\}$ is a set of places denoting the status of resources (i.e., production system or QC system ready), $v = m + k$,
S_3	$= \{u_1, \ldots, u_v\}$ is a set of places denoting that resources are vulnerable,
S_4	$= \{\bar{u}_1, \ldots, \bar{u}_v\}$ is a set of places used as a complement to S_3 (i.e., resources are *not* vulnerable),
S_5	$= \{y_1, \ldots, y_m\}$ is a set of places denoting that manipulating one or multiple quality characteristics through a compromised production system has been completed,
Q_o	$= \{q_1, \ldots, q_h\} \in S_6$ is a set of places denoting quality characteristics influenced by production step o,
$\bar{Q}_o$	$= \{\bar{q}_1, \ldots, \bar{q}_h\} \in S_7$ is a set of places denoting that quality characteristics, which are influenced by production step o, have been compromised,
S_8	$= \{c_1, \ldots, c_k\}$ is a set of places denoting QC steps,
S_9	$= \{a_1, \ldots, a_{k*2}\}$ is a set of places denoting whether a defect has been detected by a QC system,
S_{10}	$= \{z_1, \ldots, z_k\}$ is a set of places whose user-defined markings predefine that the corresponding (benign) QC system would detect any maliciously introduced defects,
S_{11}	$= \{\bar{z}_1, \ldots, \bar{z}_k\}$ is a set of places used as a complement to S_{10},
S_{12}	is a set of auxiliary places to model various structures (e.g., XOR-joins), and
S_{13}	$= \{s, f, d\}$, where s is a place denoting the job order status, f is a place denoting the finished product, and d is a place denoting the defects.

Transitions

T	$= \bigcup_{i=1}^{7} G_i$, $G_i \cap G_j = \varnothing$ for all $i, j \in \{1, \ldots, 7\}, i \neq j$, where
G_1	$= \{\alpha_1, \ldots, \alpha_{n*2}\}$ is a set of transitions denoting the start or end of a production step,
G_2	$= \{\beta_1, \ldots, \beta_{k*3}\}$ is a set of transitions denoting the start or end of a QC step (includes two variants of the end step to cover defect and no defect conditions),
G_3	$= \{\gamma_1, \ldots, \gamma_m\}$ is a set of transitions denoting attacks against production systems,
G_4	$= \{\delta_1, \ldots, \delta_{k*2}\}$ is a set of transitions denoting whether a QC system successfully detected a defect ($\delta^{\dagger}$) or failed to detect it ($\delta^{\ddagger}$) assuming that neither the QC system nor any quality characteristic under test was compromised beforehand,
G_5	$= \{\epsilon_1, \ldots, \epsilon_{k*2}\}$ is a set of transitions denoting whether a QC system detected a defect ($\epsilon^{\dagger}$) or did not detect it ($\epsilon^{\ddagger}$) after a quality characteristic was compromised (yet, the QC system itself remained intact),
G_6	$= \{\zeta_1, \ldots, \zeta_{k*2}\}$ is a set of transitions denoting whether a compromised QC system was manipulated in a way to suppress the detection of a maliciously introduced defect ($\zeta^{\dagger}$) or to detect a non-existent defect with the objective to waste material ($\zeta^{\ddagger}$), and
G_7	is a set of auxiliary transitions (similarly to S_{12}).

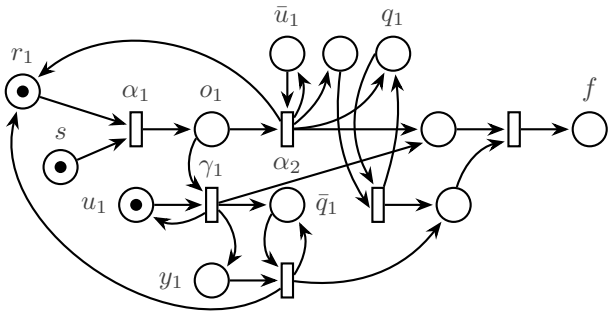


Fig. 4: Minimal QOPN (unlabeled nodes $\in S_{12} \cup G_7$).

prior results of the vulnerability analysis (in particular, to denote vulnerable resources) and optionally on user input (e.g., to predefine the outcome of a QC step). Furthermore, a sequence of manufacturing operations may be followed by one or multiple QC steps to check whether the involved quality characteristics meet the specified requirements. This case is covered by the QOPN template shown in Fig. 5. Owing to the sets G_4, G_5, G_6, and the PN structure given in Fig. 5, various attack scenarios involving QC systems can be modeled. Again, the QOPN depicted in Fig. 5 includes only one quality characteristic, and $x(u_1) = x(z_1) = x(\bar{q}_1) = 1$, as well as $x(\bar{u}_1) = x(q_1) = x(\bar{z}_1) = 0$, were specified arbitrarily for the purpose of illustrating the PN structure.

The templates were designed to ensure boundedness of the constructed QOPN, that is, $\forall \mathbf{x} \in \mathsf{Reach}(QOPN), \forall p \in P : x(p) \leq \kappa$, where $\mathsf{Reach}(QOPN)$ is the reachable state set of the QOPN and κ is a positive number. This property is an essential requirement for applying reachability-based analysis techniques due to the fact that the PN's reachability graph must be finite.

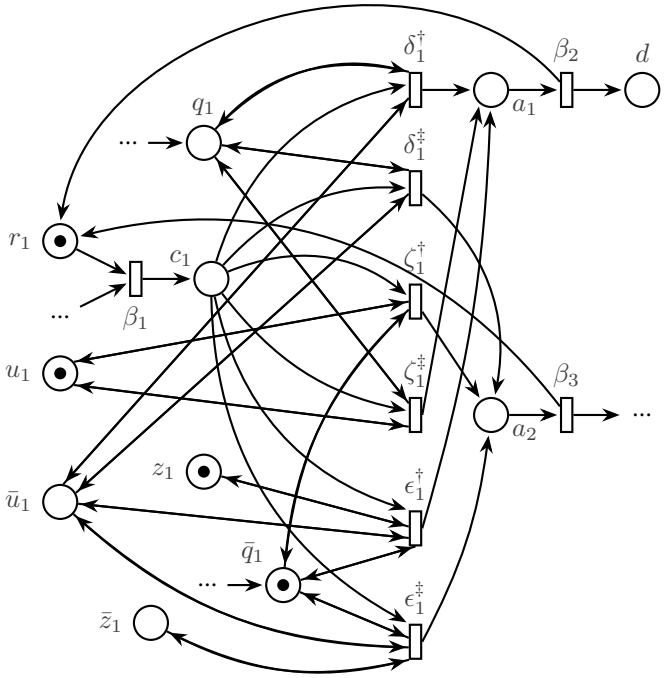


Fig. 5: QOPN template for a quality control step with a single quality characteristic under test.

3) Analysis: To answer Q2 and Q3, we reformulate these questions as reachability queries on QOPNs in Computation Tree Logic (CTL). The formulae for checking the desired reachability properties are expressed as EFϕ, where the state predicate ϕ takes the following forms:

Q2 $\big((\exists s \in \mathcal{S} : \lambda(s) > 0) \wedge (f > 0)\big)$, where $\mathcal{S} = \{u \in S_3 \mid x(u) = 1\}$ and λ is a relation from $\mathcal{S}$ to S_7. Informally, we describe this reachability problem as follows: Is it possible that the manufacturing process finishes without detected defects, even though some quality characteristics were compromised by exploiting vulnerable assets? After checking reachability, we analyze and filter the witness states to obtain a subset of S_7 that provides an answer to this question.

Q3 $\big((\exists s \in \mathcal{S} : \lambda(s) > 0) \wedge (f > 0)\big)$, where $\mathcal{S} = \{\bar{u} \in S_4 \mid x(\bar{u}) = 1\}$ and λ is a relation from $\mathcal{S}$ to S_7. This reachability problem can be understood as checking if the manufacturing process may finish without detected

defects, while some quality characteristics were indirectly compromised by exploiting vulnerable assets in preceding manufacturing steps. Similarly to Q2, we process the witness states after reachability checking to answer this question.

Q4 cannot be answered with a single reachability query and requires an iterative procedure, as shown in Algorithm 1. This algorithm takes a generated QOPN as input and produces a set U', which is a proper subset of S_3 containing places that correspond to resources of QC systems that need to be vulnerable and successfully compromised to evade quality checks. After initializing the result set U' and the set $\mathcal{T}$ that will contain transitions demonstrating the execution path starting from the initial marking, the state predicate ϕ is defined. Since we want to check if there is an execution path where a product defect is found during a QC inspection, we define the state predicate such that the number of tokens on the place d denoting the detected defects is greater than zero. Based on this, the reachability query is expressed in CTL as the following formula: EF$(d > 0)$. As long as there is a reachable state satisfying ϕ, the body of the loop is executed. In line 5, $\mathcal{T}$ is filled with the witness path, which is then processed in reverse: In each iteration, it is checked if the current element in the loop is a member of ${G^\dagger}_5$ (i.e., the transition denotes the detection of a defect). In the body of the `if`-statement, we retrieve the place denoting that the resource of the QC system that detected the defect is vulnerable, add it to the result set, retrieve the complementary place (i.e., resource not vulnerable), and adapt the marking such that the QC system is now indicated as vulnerable. Note that the procedure outlined in Algorithm 1 presupposes that at least one quality characteristic can be compromised through the exploitation of a vulnerable asset employed for a production step, since an answer to Q4 should reveal which QC system(s) an adversary would need to manipulate in order to conceal introduced product defects.

Algorithm 1: Reachability Analysis for Q4

Input: A QOPN $N \leftarrow (P, T, A, w, x)$
Result: A subset of places of N corresponding to resources that need to be compromised in order to disguise an attack on product quality $U' \subset S_3$

```
1  U' ← ∅                                   // result set
2  T ← ∅                                    // witness path set
3  φ ← (d > 0)                              // state predicate
4  while N satisfies EFφ do
5      T ← GetWitnessPath()
6      for i ← |T| to 1 do
7          if T(i) ∈ G†5 then
8              u_i ← GetResourcePlace(T(i))
                                            // vulnerable resource
9              U' ← U' ∪ {u_i}
10             ū_i ← GetComplementaryPlace(u_i)
11             x(u_i) ← 1; x(ū_i) ← 0
12             break
```

D. Implementation

We created the AMLqual libraries with the AutomationML Editor[3]. The quality ontology was modeled with Protégé[4] [38]. Since we build upon the results of Eckhart *et al.* [7], we have extended their prototype to incorporate our quality-driven risk identification method. In particular, we have implemented the SFC-to-OWL translation, the QOPN construction, and the export to Petri Net Markup Language (PNML) and LoLA file formats in Scala. To conduct reachability analyses, which is an integral part of QualSec, we utilize LoLA 2 [39], [40].

AMLqual, the source code of the implemented prototype, and the AML files used for the case study are publicly available on GitHub[5].

V. Case Study

This section presents the results of a case study that was conducted to showcase QualSec. The engineering data used in the case at hand is based on the official AML example of a robot cell [34], which aims to demonstrate how AML can be used to model the topology, behavior, and geometry of a robotic spot welding cell. To obtain a more comprehensive model, we extended these artifacts in the following ways: i) A description of a stamping process was integrated into the existing SFC (which only models the sequence of joining activities). ii) The plant topology was supplemented with PPR relations and communication-related information. iii) IT/OT assets were populated with system-dependent, security- and quality-relevant information using AMLsec and AMLqual.

The process considered in the case study comprises activities of vehicle manufacturing. More precisely, we focus on the stamping and joining processes for the inner front door panel, which represent a crucial part of the body in white (BiW) production line. It is evident that the structural characteristics of closures strongly influence the quality of the complete BiW; hence, conducting a quality-driven security analysis already during the engineering of the involved CPPS is prudent.

Fig. 6 illustrates the manufacturing steps from a PPR-centric perspective, where the process view is modeled in the SFC language. Due to space limitations, we cannot present an illustration of the plant topology considered in the case study. We, therefore, refer readers to the web version of the figure[6].

A. Results

In the following, we describe the most important results that we obtained by executing the QualSec prototype with the described input of the case study:

Q1 The results of the threat, vulnerability, and consequence identification indicate that 47 of the 370 assets of the plant topology (67 of which have the class OTComponent) have 193 vulnerabilities that may be exploited by 9 distinct threats, possibly leading to 80 consequences.

[3] https://www.automationml.org/download-archive
[4] https://protege.stanford.edu
[5] https://github.com/sbaresearch/amlsec
[6] https://github.com/sbaresearch/amlsec/blob/master/appendix/qualsec/plant-topology.pdf

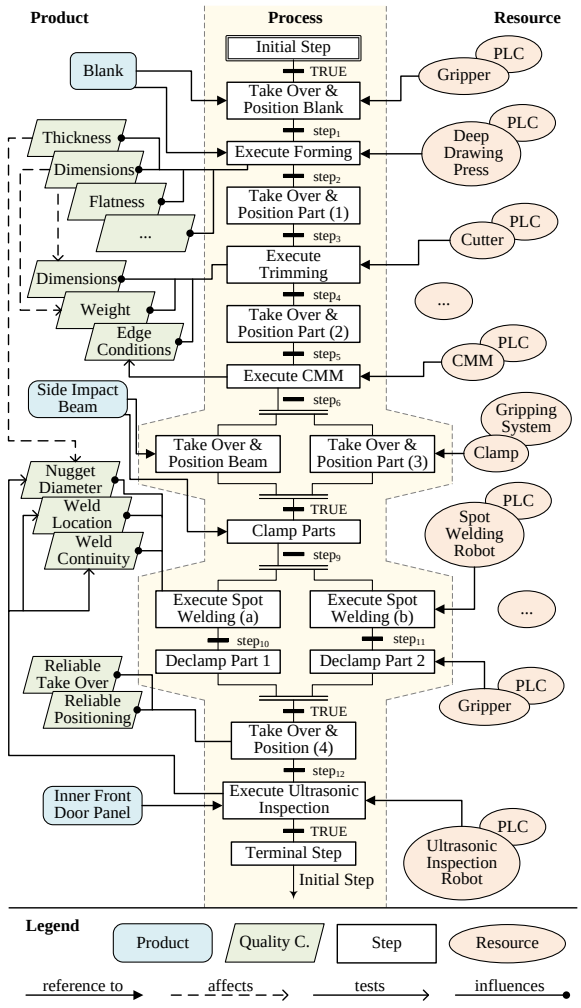


Fig. 6: Product-Process-Resource-centric view of the manufacturing process considered in the case study (gripping and positioning operations are combined into a single step for the sake of brevity).

Q2 The CVE check revealed that the PLC `S71516F_7`, which controls the deep drawing press, has a known vulnerability. If this vulnerable asset is compromised, the sheet metal forming step could be influenced to manipulate several quality characteristics of the stamped pieces, including their thickness, dimensions, and edge conditions. Possible defects resulting from this attack would remain undetected because the employed coordinate-measuring machine (CMM) only tests the edge conditions of the trimmed pieces.

Q3 Since the subsequent manufacturing operation relies on the correct dimensions of the formed blanks, an attack launched against the deep drawing press could also affect the dimensions and weight of the trimmed parts. Furthermore, an incorrect blank thickness would require a different size of the weld nugget formed as part of the spot welding step. Although the nugget diameter is checked through ultrasonic testing, the PLC `S71516F_11` controlling the spot welding quality inspection robot is vulnerable and can therefore be circumvented if successfully attacked.

An excerpt of these results is displayed in Table II. In a second iteration, the plant topology has been adapted based on the answers to Q1–Q3 given above to make the CPPS more resilient. More specifically, the vulnerability in `S71516F_11` has been mitigated and the CMM now also tests the dimensions of the stamped and trimmed parts.

Q4 To validate if the performed adaptations yield a security improvement, we execute the reachability analysis outlined in Algorithm 1, intending to identify those QC assets that potentially detect malicious product changes. The results showed that an attack against the deep drawing press could only be disguised by compromising the PLCs `S71518_2` and `S71516F_11`, which control the CMM and ultrasonic testing robot, respectively. Ideally, these devices are hardened to detect attacks that target the product quality.

TABLE II: Excerpt of the QualSec analysis results.

Legend:
vulnerable asset
■ pertains operation or QC
▲ compromised
⊛ affected by compromised qual.
covered by QC
QC evasion

Step	Asset		Quality Characteristics: Edge Conditions	Dimensions	Weight	Thickness	Nugget Diameter	Weld Continuity
Forming	`S71516F_7`	●	▲	▲	▲	▲		
Trimming	`S71518_1`	○	■	⊛	⊛			
CMM	`S71518_2`	○	■					
Spot Welding (a)	`KRC4_1`	○						
Spot Welding (b)	`KRC4_2`	○						
Ultrasonic Insp.	`S71516F_11`	●				⊛	■	■
							■	■
							■	■

B. Discussion

In the following, we reflect on the results of the case study and critically evaluate the usefulness of QualSec. To this end, we briefly reiterate the gaps in the literature and analyze how well QualSec achieves its goals to address them:

- *Efficient Security Risk Identification.* Systems integrators are in need of a method that assists engineers in addressing security issues during the integration phase [2]. Our work is based on [7], which represents a first step toward a fully automated identification of security risks using engineering data. We improved the method proposed in [7] by incorporating the model of the manufacturing process (i.e., sequencing information) into our KB to enrich its results. In this way, the security vulnerabilities identified for answering Q1 can be associated via PPR links to individual steps of the manufacturing process, which may support risk analysis and risk evaluation. However, note that the vulnerability analysis operates at the plant topology level. This limits the scope of analysis to the plant model and public sources (e.g., industrial security standards, advisories, CVEs). Furthermore, we only consider the structure of SFC programs to construct PNs (more specifically, QOPNs), whereas

other transformation techniques (e.g., [41]) provide more comprehensive coverage.

- *Quality Control and Security.* One of the first serious discussions of the relationship between QC and CPS security appeared in 2018 when Elhabashy *et al.* [11] proposed a cyber-physical attack taxonomy featuring a QC perspective. In a later work [12], they identified weaknesses in QC systems that adversaries might exploit to conceal the physical effects of attacks. Both works [11], [12] emphasize the necessity of taking QC aspects into account when designing CPPSs in order to make them more resilient to such attacks. QualSec aims to address this need by providing a risk-based approach that helps engineers better understand the impact of potential cyber-physical attacks in terms of product quality. The answers to Q2 and Q3 obtained through QualSec allow users to pinpoint compromised quality characteristics of workpieces in attack scenarios and analyze the dependencies among them. The method's results also indicate under which conditions the QC systems included in the plant topology could potentially detect malicious product changes. Since QualSec is intended to be used as a risk identification tool by systems integrators, its assessment scope is limited to the hierarchical structure of the plant, and it assumes the reasonable worst case. In other words, the presented method was not specifically designed to identify security issues in fine-grained system models (e.g., described in SysML) that would allow for a meaningful representation of vulnerability preconditions and postconditions. Thus, QualSec neglects the product supplier perspective entirely.
- *What-If Scenarios.* Engineers can use QualSec as a planning tool to perform what-if analyses that allow a safe simulation of attack scenarios involving malicious quality loss. QualSec's results for Q4 help defenders to determine potential chokepoints in the designed QC program that would allow adversaries to bypass QC systems if they are not adequately secured.

VI. Performance Evaluation

The performance and scalability of the prototypical implementation were measured through multiple tests that were carried out using different-sized engineering models (cf. Table III). The smallest dataset (A) corresponds to the engineering model that was used for the case study, which contains the plant topology for one site[7] and the corresponding logic model depicted in Fig. 6. For datasets B–F, we expanded the base model by increasing the number of sites (Vienna `InternalElement`) and the process description (SFC) in steps of two.

We measured the execution time of 60 experiments that were conducted by performing five runs per dataset with two cluster configurations. The first cluster consisted of the following three nodes: Node 1 hosted the triple store (Apache Jena Fuseki), a database for storing events (Apache Cassandra), and actors to provide a front-end and manage work items. Nodes 2 and 3 were used to run the work executor actors that perform the actual QualSec method. The second cluster consisted of two additional work executor nodes (i.e., five nodes in total). All nodes of both cluster configurations were cloud-hosted virtual machines running Fedora 35 x64 with 16 vCPUs and 32 GB RAM.

[7]See footnote 6.

TABLE III: Overview of the datasets used for the evaluation.

	A	B	C	D	E	F
Engineering Data						
`InternalElements` (in K)	0.87	1.74	3.49	5.23	6.97	8.71
AML Size (in MB)	1.00	2.00	4.00	6.00	8.10	10.10
Steps in SFC	23	44	86	128	170	212
After AML & SFC Trans.						
Triples (in K)	18.96	34.01	64.09	94.17	124.26	154.34
Knowledge Base Size (in MB)	2.20	4.00	7.60	11.20	14.80	18.40
After Method Execution						
Triples (in MM)	0.06	0.12	0.32	0.60	0.97	1.42
Knowledge Base Size (in MB)	5.50	11.90	30.70	57.40	92.00	134.60
QOPN Places (in K)	0.23	0.46	0.91	1.36	1.81	2.26
QOPN Transitions (in K)	0.12	0.24	0.47	0.71	0.95	1.18
QOPN Arcs (in K)	0.75	1.50	3.00	4.50	6.00	7.50
Assets (in K)	0.37	0.74	1.48	2.22	2.95	3.69

A. Results

Fig. 7 summarizes the performance evaluation. In Fig. 7a, we show the average execution time of the main steps of the setup phase (viz., AML-to-OWL transformation, SFC-to-OWL transformation, and model augmentation), the generation of the QOPN, and the reachability analyses for answering Q2–Q4. Note that these reported measurements were made with both cluster configurations (i.e., 10 runs per dataset) since the respective tasks were not processed in parallel by multiple work executor actors. The average execution time for the risk identification logic and the QualSec method in total are plotted per cluster setup in Figs. 7b and 7c, respectively.

In the following, we provide a breakdown of the time measurements (mean and standard deviation) collected with the smallest and largest datasets (A and F). The execution time of the AML-to-OWL transformation averaged 3.71 ± 0.16 seconds for dataset A and 252.71 ± 15.97 seconds for dataset F. Transforming the logic model from SFC to OWL averaged 0.80 ± 0.05 seconds for the smallest dataset and 1.36 ± 0.10 seconds for the largest dataset. The model augmentation step averaged 25.13 ± 1.72 seconds and 513.96 ± 36.17 seconds for datasets A and F, respectively. Generating the QOPN from the logic model with 23 steps averaged 0.72 ± 0.20 seconds, while for 212 steps, this task averaged 9.79 ± 0.55 seconds. The reachability analyses averaged 0.51 ± 0.03 seconds for dataset A and 10.74 ± 0.49 seconds for dataset F. Validating the model and identifying security risks using two work executor nodes averaged 25.26 ± 5.08 seconds for dataset A, while the average execution time for dataset F was 1559.75 ± 238.05 seconds. The measurements made for this step with four work executor nodes are 20.63 ± 1.42 seconds for dataset A and 1031.50 ± 48.34 seconds for dataset F. In total, the average execution time of QualSec (from the setup phase until obtaining the results of the case study) with the three-node cluster

was 59.61 ± 5.63 seconds for dataset A and 2400.70 ± 272.81 seconds for dataset F, while with the five-node cluster, the average execution time was reduced to 56.98 ± 1.09 seconds (A) and 1832.83 ± 70.27 seconds (F).

B. Discussion

Building upon earlier work [7], we answer Q1 by executing a set of SPARQL queries and SHACL rules. Consequently, the performance of the threat, vulnerability, and attack consequence identification depends on the following factors: (i) the implementation of the SPARQL, SHACL, and inference engines, (ii) the executed queries and rules, and (iii) the size and structure of the semantic data. As can be seen from Fig. 7b, scaling out the QualSec application with additional work executor nodes in a cluster can yield considerable performance improvements, especially for larger datasets.

Due to the fact that answering Q2–Q4 necessitates the construction of reachability graphs, the presented method suffers from the well-known *state explosion problem* [42]. Thus, albeit the reachability graphs are finite given the boundedness of QOPNs, the size of the state space can be unmanageable. Increasing the practicality of reachability analysis of PNs is a long line of research that has spawned various techniques to reduce the state space (e.g., stubborn sets [43]). LoLA [40] implements, inter alia, partial order reduction (the stubborn set method) and symmetry reduction, which can also be applied in combination [44]. We observe that the state space reduction techniques implemented in LoLA [40] alleviate state explosion, at least to the extent that Q2–Q4 can be answered within reasonable time (avg. 0.51 ± 0.03 seconds for dataset A). In fact, as can be seen from Figs. 7a and 7b, the execution time of the QOPN generation mechanism and reachability analysis to answer Q2–Q4 is negligible compared to the security risk identification phase that answers Q1.

VII. Conclusion and Future Work

In this paper, we have presented a method named QualSec that automates the identification of security risks pertaining to CPPSs based on engineering data. The novelty of QualSec is that it stimulates a quality-driven perspective on security that places special emphasis on the quality characteristics of the manufactured products. Our proposed method can reveal security issues in the plant topology and expose weaknesses in QC that adversaries may exploit to introduce defects during manufacturing deliberately. QualSec utilizes PPR knowledge modeled in CAEX and SFC as part of AML to create a semantic KB. Threats, vulnerabilities, and attack consequences are then automatically identified by executing several SHACL rules and SPARQL queries against the KB. Furthermore, the structure of the modeled manufacturing process is used to construct a QOPN automatically. This QOPN serves as a basis for reachability analysis to answer risk-related questions. Systems integrators can apply QualSec to initiate proper mitigation of security risks during the engineering phase. The resulting CPPSs may be more secure by design and thereby inhibit attackers from compromising the quality of manufactured goods, possibly contributing to a decline in the number of faulty products entering the market.

Further research should be undertaken to improve QualSec in the following ways: The current version of our method is intended to be used during the engineering of CPPSs and, therefore, heavily relies on the engineering data exchange format AML. However, since a QOPN is constructed based on a semantic representation of the production process, the input format does not necessarily have to be PLCopen XML. Incorporating additional sources into QualSec would extend the method's scope to cover the operation phase.

Another possible improvement of QualSec would be to increase the degree of detail of the systems' state. In this paper, we make the (relatively strong) assumption that the successful exploitation of a vulnerability results in full control of the system and allows an adversary to manipulate all quality characteristics that the compromised system can influence. The rationale behind this assumption is twofold: (i) The abstraction level of the plant model available at the engineering phase may hinder the definition of postconditions of exploiting vulnerabilities. (ii) Users might be primarily interested in worst-case scenarios. Nevertheless, enriching the KB may enable a finer-grained analysis of how quality characteristics can be influenced based on the privileges gained by an adversary.

There is also room for improvement with respect to the engineering data sources used for risk identification. In its current version, QualSec processes the plant topology in CAEX and the sequencing information in PLCopen XML, which are both part of AML. Utilizing COLLADA interfaces to incorporate geometry and kinematics information into QualSec appears to be an appealing extension of our work. In this way, the attack consequence identification component could be enhanced to address safety aspects more thoroughly.

Finally, we want to suggest some ideas to advance the PN-based analysis further. Probabilistic PNs may be applied to better reflect various quality inspection strategies (e.g., random sampling). Additionally, attaining a more rigorous translation from SFC to PN, also including timing information (time PN), would be worthwhile.

Acknowledgment

The authors would like to thank Walid Fdhila for informative discussions on the submitted manuscript and Yameng An for providing the initial version of OntoPLC [33].

References

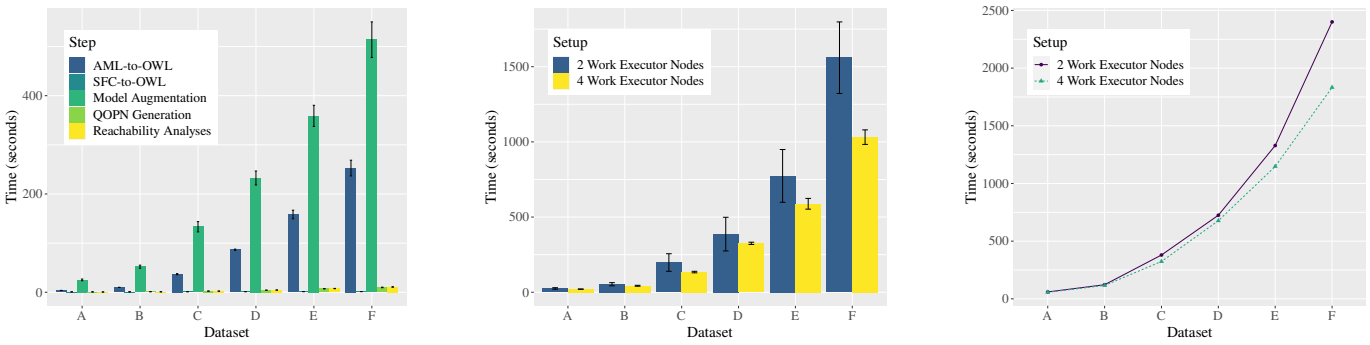
[1] M. Eckhart, K. Meixner, D. Winkler, and A. Ekelhart, "Securing the testing process for industrial automation software," *Comput. Secur.*, vol. 85, pp. 156–180, 2019.

[2] P. Kieseberg and E. Weippl, "Security challenges in cyber-physical production systems," in *Software Quality: Methods and Tools for Better Software and Systems*, 2018, pp. 3–16.

[3] M. Eckhart, A. Ekelhart, A. Lüder, S. Biffl, and E. Weippl, "Security development lifecycle for cyber-physical production systems," in *Proc. 45th Annu. Conf. IEEE Ind. Electron. Soc.*, 2019, pp. 3004–3011.

[4] IEC 62443-3-2:2020, "Security for industrial automation and control systems – Part 3-2: Security risk assessment and system design," International Organization for Standardization, Standard, 2020.

[5] VDI/VDE 2182-1, "Sheet 1: IT-security for industrial automation - general model," 2011.



(a) Setup, QOPN generation, case study (b) Validation and risk identification (c) Total

Fig. 7: Performance assessment results of our implemented prototype (error bars indicate standard deviations).

[6] M. Schleipen and R. Drath, "Three-view-concept for modeling process or manufacturing plants with AutomationML," in *Proc. IEEE Conf. on Emerg. Technol. Factory Autom.*, 2009, pp. 1–4.

[7] M. Eckhart, A. Ekelhart, and E. Weippl, "Automated security risk identification using AutomationML-based engineering data," *IEEE Trans. Dependable Secure Comput.*, vol. 19, no. 3, pp. 1655–1672, 2022.

[8] R. Drath, A. Lüder, J. Peschke, and L. Hundt, "AutomationML - the glue for seamless automation engineering," in *Proc. IEEE Conf. on Emerg. Technol. Factory Autom.*, 2008, pp. 616–623.

[9] N. Schmidt and A. Lüder, "AutomationML in a nutshell," AutomationML e.V., Tech. Rep., Nov. 2015.

[10] S. Faltinski, O. Niggemann, N. Moriz, and A. Mankowski, "AutomationML: From data exchange to system planning and simulation," in *Proc. IEEE Int. Conf. Ind. Technol.*, 2012, pp. 378–383.

[11] A. E. Elhabashy, L. J. Wells, J. A. Camelio, and W. H. Woodall, "A cyber-physical attack taxonomy for production systems: A quality control perspective," *J. Intell. Manuf.*, vol. 30, no. 6, 2018.

[12] A. E. Elhabashy, L. J. Wells, and J. A. Camelio, "Cyber-physical attack vulnerabilities in manufacturing quality control tools," *Quality Eng.*, vol. 32, no. 4, pp. 676–692, 2020.

[13] L. J. Wells, J. A. Camelio, C. B. Williams, and J. White, "Cyber-physical security challenges in manufacturing systems," *Manuf. Letters*, vol. 2, no. 2, pp. 74–77, 2014.

[14] L. D. Sturm, C. B. Williams, J. A. Camelio, J. White, and R. Parker, "Cyber-physical vunerabilities in additive manufacturing systems," *Int. Solid Freeform Fabr. Symp.*, vol. 7, pp. 951–963, 2014.

[15] S. Belikovetsky, M. Yampolskiy, J. Toh, J. Gatlin, and Y. Elovici, "dr0wned – cyber-physical attack with additive manufacturing," in *Proc. 11th USENIX Workshop Offensive Technol.*, 2017.

[16] L. Apvrille and Y. Roudier, "SysML-Sec: A SysML environment for the design and development of secure embedded systems," in *Proc. Int. Conf. Asia-Pacific Council Syst. Eng.*, 2013, pp. 1–16.

[17] Y. Roudier and L. Apvrille, "SysML-Sec: A model driven approach for designing safe and secure systems," in *Proc. 3rd Int. Conf. on Model-Driven Eng. and Softw. Dev.*, 2015, pp. 655–664.

[18] R. Oates, F. Thom, and G. Herries, "Security-aware, model-based systems engineering with SysML," in *Proc. 1st Int. Symp. ICS SCADA Cyber Secur. Res.*, 2013, pp. 78–87.

[19] L. Lemaire, J. Lapon, B. De Decker, and V. Naessens, "A SysML extension for security analysis of industrial control systems," in *Proc. 2Nd Int. Symp. ICS SCADA Cyber Secur. Res.*, 2014.

[20] L. Lemaire, J. Vossaert, J. Jansen, and V. Naessens, "Extracting vulnerabilities in industrial control systems using a knowledge-based system," in *Proc. 3rd Int. Symp. ICS SCADA Cyber Secur. Res.*, 2015.

[21] M. Glawe, C. Tebbe, A. Fay, and K.-H. Niemann, "Knowledge-based engineering of automation systems using ontologies and engineering data," in *Proc. Int. Joint Conf. Knowl. Discov., Knowl. Eng. Knowl. Manage.*, 2015, pp. 291–300.

[22] C. Tebbe, M. Glawe, A. Scholz, K.-H. Niemann, A. Fay, and J. Dittgen, "Wissensbasierte Sicherheitsanalyse in der Automation," *atp magazin*, vol. 57, no. 04, pp. 56–66, 2015.

[23] M. Glawe and A. Fay, "Wissensbasiertes Engineering automatisierter Anlagen unter Verwendung von AutomationML und OWL," *at-Automatisierungstechnik*, vol. 64, no. 3, pp. 186–198, 2016.

[24] C. Tebbe, M. Glawe, K.-H. Niemann, and A. Fay, "Informationsbedarf für automatische IT-Sicherheitsanalysen automatisierungstechnischer Anlagen," *at-Automatisierungstechnik*, vol. 65, no. 1, pp. 87–97, 2017.

[25] Y. Cherdantseva, P. Burnap, A. Blyth, P. Eden, K. Jones, H. Soulsby, and K. Stoddart, "A review of cyber security risk assessment methods for SCADA systems," *Comput. Secur.*, vol. 56, pp. 1–27, 2016.

[26] W. Knowles, D. Prince, D. Hutchison, J. F. P. Disso, and K. Jones, "A survey of cyber security management in industrial control systems," *Int. J. Crit. Infrastruct. Prot.*, vol. 9, pp. 52–80, 2015.

[27] M. H. Henry, R. M. Layer, K. Z. Snow, and D. R. Zaret, "Evaluating the risk of cyber attacks on SCADA systems via Petri net analysis with application to hazardous liquid loading operations," in *Proc. IEEE Conf. Technol. Homeland Secur.*, 2009, pp. 607–614.

[28] M. H. Henry, R. M. Layer, and D. R. Zaret, "Coupled Petri nets for computer network risk analysis," *Int. J. Crit. Infrastruct. Prot.*, vol. 3, no. 2, pp. 67–75, 2010.

[29] C.-W. Ten, C.-C. Liu, and G. Manimaran, "Vulnerability assessment of cybersecurity for SCADA systems," *IEEE Trans. Power Syst.*, vol. 23, no. 4, pp. 1836–1846, 2008.

[30] S. Fenz and A. Ekelhart, "Formalizing information security knowledge," in *Proc. 4th Int. Symp. Inf., Comput., Commun. Secur.*, 2009, pp. 183–194.

[31] E. Kiesling, A. Ekelhart, K. Kurniawan, and F. Ekaputra, "The SEPSES knowledge graph: An integrated resource for cybersecurity," in *Proc. Int. Conf. Semantic Web*, 2019, pp. 198–214.

[32] Y. Hua and B. Hein, "Interpreting OWL complex classes in AutomationML based on bidirectional translation," in *Proc. 24th IEEE Int. Conf. Emerg. Technol. Factory Autom.*, 2019, pp. 79–86.

[33] Y. An, F. Qin, B. Chen, R. Simon, and H. Wu, "OntoPLC: Semantic model of PLC programs for code exchange and software reuse," *IEEE Trans. Ind. Informat.*, vol. 17, no. 3, pp. 1702–1711, 2021.

[34] AutomationML, "AutomationML example: Robot cell," AutomationML, Tech. Rep., Mar. 2017. [Online]. Available: https://www.automationml.org/wp-content/uploads/2021/06/AML_RobotCell_en_public.zip

[35] C. A. Petri, "Kommunikation mit Automaten," Ph.D. dissertation, Universität Hamburg, 1962.

[36] C. G. Cassandras and S. Lafortune, "Petri Nets," in *Introduction to Discrete Event Systems*, 2nd ed., 2008, pp. 223–267.

[37] M. Zhou and N. Wu, "Process-oriented Petri net modeling," in *System Modeling and Control with Resource-Oriented Petri Nets*, 1st ed., 2010, pp. 43–55.

[38] N. F. Noy, M. Sintek, S. Decker, M. Crubezy, R. W. Fergerson, and M. A. Musen, "Creating semantic web contents with Protégé-2000," *IEEE Intell. Syst.*, vol. 16, no. 2, pp. 60–71, 2001.

[39] K. Schmidt, "LoLA: A low level analyser," in *Proc. 21st Int. Conf. Appl. Theory Petri Nets*, 2000, pp. 465–474.

[40] K. Wolf, "Petri net model checking with LoLA 2," in *Proc. 39th Int. Conf. Appl. Theory Petri Nets Concurr.*, 2018, pp. 351–362.

[41] N. Wightkin, U. Buy, and H. Darabi, "Formal modeling of sequential function charts with time Petri nets," *IEEE Trans. Control Syst. Technol.*, vol. 19, no. 2, pp. 455–464, 2011.

[42] A. Valmari, "The state explosion problem," in *Lectures on Petri Nets I: Basic Models: Advances in Petri Nets*, 1998, pp. 429–528.

[43] ——, "Stubborn sets for reduced state space generation," in *Proc. 10th Int. Conf. Appl. Theory Petri Nets*, 1991, pp. 491–515.

[44] K. Wolf, "Generating Petri net state spaces," in *Proc. 28th Int. Conf. Appl. Theory Petri Nets Models Concurr.*, 2007, pp. 29–42.