Integrating security evaluations into virtual commissioning

Alexander Giehl[1], Norbert Wiedermann[1], Makan Tayebi Gholamzadeh[1] and Claudia Eckert[1]

***Abstract*— Virtual commissioning is an important part of modern plant and factory organization. Research in this area focuses on safety, reliability, liveness, and repeatability. Security evaluations are currently not considered in virtual commissioning research and applications. Vulnerabilities in controller software and in the implementation of industrial equipment are receiving increased attention from attackers and cyber criminals. This is due to the rapidly advancing interconnection in modern, digital factories. This increase of the possible attack surface needs to be addressed as a part of comprehensive risk analysis within the domain of Industrie 4.0. Virtual commissioning, as an established process, is well-suited to address this lack of security evaluation. In this work, we propose a conceptual architecture for a simulation testbed that can be integrated in the virtual commissioning toolchain and show how to model and evaluate industrial equipment.**

I. Introduction

Industrie 4.0 is the ongoing transformation of the manufacturing landscape in Germany that is also observed in other countries [31]. It is characterized by the increasing integration of cyber-physical systems (CPSs) into the manufacturing process [24]. A CPS can be defined as a technology that manages the digital interconnection of physical assets and computing devices in an embedded component [1]. These CPSs are characterized through increasing complexity and provide more computational power than traditional manufacturing equipment. This enables a shop floor with a large amounts of CPSs to shift intelligent control tasks towards these systems. This development results, thus, in a trend towards a more decentralized operational technology (OT) architecture. Interaction between groups of CPSs requires precise timing and early testing procedures in order to minimize the financial risk of delayed commissioning. To address this need in the context of increased factory digitalization, the application of simulation models for virtual testing and virtual commissioning (VC) is a logical development.

VC concerns itself with the verification of manufacturing systems by using a simulation of the plant [2]. The plant simulation interacts with a controller that is used to control an automated manufacturing system or component. The controller can be a physical device, meaning a hardware-in-the-loop (HIL) simulation is performed for VC. VC can also be conducted by employing a simulated model of the controller, thus, only using simulated or virtual components. This is referred to as *constructive commissioning*. These two types of commissioning are summarized in Table I at Line 1 and Line 2 respectively. In contrast to VC stands real or traditional commissioning. Here, a real plant is involved in the commissioning process. Here, either a virtual controller is used for *reality-in-the-loop commissioning* (see Line 3) or a real controller is used with the real plant, which is the actual or *real commissioning* process (see Line 4). Manufacturing components are usually controlled by programs executed on a programmable logical controller (PLC) controlling actuators and sensors [3], [4].

TABLE I: Types of commissioning [2].

Line	Type	Plant	Controller
1	Hardware-in-the-loop commissioning	Virtual	Real
2	Constructive commissioning	Virtual	Virtual
3	Reality-in-the-loop commissioning	Real	Virtual
4	Real commissioning	Real	Real

VC is an important part in the deployment of modern manufacturing equipment since it improves the quality of the equipment and its quality of operation. As it can be used throughout the entire engineering process, errors in planning, design, and programming can be detected and corrected early on [5]. This reduces the need for costly error correction in the later stages of engineering, especially in the prototyping phase. Furthermore, the increase of computational technology in CPSs allows for VC to be employed in large-scale manufacturing systems [4]. Here, VC is used to increase the productivity of manufacturing systems as well as their flexibility. These aspects are relevant in the context of Industrie 4.0 and draw attention to VC by manufacturing operators and researchers.

Current research of VC on the verification of PLC programs is focused on the evaluation of the concepts of repeatability, liveness, and safety [2]. Repeatability means that repeated experimentation or measurements, e.g., taken by sensors, under the same conditions delivers statistically consistent results. The concept of liveness states that the execution of concurrent processes in a system delivers a desired result [6] and is related to the concept of safety [7]. Safety means that undesired results do not occur during active operation of the system. Safety is an aspect in the design and construction of dependable systems [8].

However, safety properties and features alone do not guarantee the absence of undesired or even harmful results to manufacturing systems or humans. Safety does not take the

*This work was supported by the German Federal Ministry of Education and Research (BMBF) [grant numbers 16KIS0324, 16KIS0933K].

[1]Alexander Giehl, Norbert Wiedermann, Makan Tayebi Gholamzadeh, and Claudia Eckert are with Fraunhofer AISEC, Parkring 4, 85748 Garching b. München {alexander.giehl, norbert.wiedermann, makan.tayebi, claudia.eckert}@aisec.fraunhofer.de

intended manipulation of the equipment or its control logic into account. This was observed, for example, in the Stuxnet attack, which targeted and harmed the industrial process of a critical infrastructure [9]. Here, the PLC program was subtly altered over a prolonged period of time, thus, avoiding detection by human operators and automated processes. The intention of Stuxnet was to cause disruption of availability within the industrial process. This is not an isolated occurrence, as other attacks in the previous decades show [10]. Those kinds of attacks and the development of countermeasures are researched within the field of information technology (IT) security.

IT security concerns itself with the protection of computer systems by implementing protection goals such as confidentiality, data integrity, and availability [11]. Confidentiality means that information is disclosed only to authorized entities. Data Integrity describes the accuracy and consistency of stored or transmitted data and ensures that no manipulation or unauthorized alteration of the data occurs. Availability is a concept also related to the design of dependable systems [8]. It encompasses the availability for authorized and correct service and is often seen as the most relevant protection goal in manufacturing by OT operators [12].

In this work, we focus in the integration of security evaluations into manufacturing, specifically into VC. It is reasonable to perform security testing on virtual components as security tests can lead to the destruction of the real target of evaluation (TOE). Also, CPSs for Industrie 4.0 are often novel products and, therefore, testing on real equipment may not always be possible prior to prototyping. As the example of Stuxnet and other related attacks show, the consideration of IT security in manufacturing is becoming more relevant. Thus, we propose the integration of IT security evaluations in VC as it is an established process with a basis of available simulation tools. Those tools, however, require security extensions as discussed in the following section.

II. Related Work

VC is described as a process of four major, sequential steps [2]: (1) Process planing, (2) Physical device modeling, (3) Logical device modeling, and (4) System control modeling. Concurrent implementation of Steps 2-4 is possible [3].

In Step 1, the scope of the manufacturing system is defined and a sequence of operations is developed that allow the device to fulfill its scope. Step 2 and Step 3 are supported by VC. Research on physical device modeling (Step 2) concerns itself with improving the physical and behavioral accuracy of the model or simulation [13]. This is typically achieved by geometrical and kinematic modeling with the former having received more attention from researchers [2]. Logical device modeling can be distinguished between verification of theoretical properties (see Section I) and the generation of dependable PLC programs. This specific process can be integrated into modern product development cycles as described by [14].

Security and its evaluation have not received sufficient attention in VC research so far. The current state of security in manufacturing is given by [15]. The results of that survey show, that VC is currently not in the focus of security researchers. In [16], the risk of intellectual property (IP) loss resulting from reverse engineering simulation models is mentioned. This threat is related to the protection goal confidentiality in IT security as it allows in theory to reproduce a product by studying its design and composition. It can be relevant for both, physical and logical device models. A reverse engineering process for VC is given by [4]. Digital twinning approaches develop further towards new VC methods that can integrate security. In [17], an OPC/UA client is integrated that can offer and implement security functionalities for testing. Furthermore, in [18] the possibility of security validations are mentioned. However, security evaluation are not conducted explicitly within those two studies.

There exists ample research on the design and construction of testbeds to study security of industrial controls systems (ICSs). A detailed and recent overview on those testbeds is provided by [19]. In their study, the authors evaluated 30 ICS testbeds and identified that most ICS security testbeds are designed for vulnerability analysis, education purposes, or testing of defense mechanisms. The detailed study of individual components is typically not in the focus of current ICS security testbed design. This is, however, a relevant aspect of VC applications as VC is mostly concerned with verification of individual manufacturing components and equipment.

Proposed security evaluation methods initially concerned themselves with standard IT components, e.g., office equipment [20]. With increasing standardization efforts, the number and complexity of such security evaluation methods and frameworks increased [21]. Also, security considerations are conducted more for other domains, such as the industrial or manufacturing domain [8], [22]. Specific methods adopting to characteristic requirements for these domains have also been proposed recently [23], [24].

In this work, we propose a testbed capable of integrating security evaluation methods in VC. The focus is not on physical modeling or the emulation of PLC ladder logic. Rather, the aim is to show how to integrate security evaluation methods within a VC toolchain.

III. Conceptual Framework

This section provides an overview of the conceptional testbed architecture for the integration of security evaluations in VC.

Our proposed architecture is summarized in Figure 1. The the left-hand side and the middle of Figure 1 shows a typical VC pipeline [2]. It consists of two functional blocks: a virtual plant and a controller. The virtual plant itself contains a number of virtual components comprising the manufacturing system. The controller is responsible for the logical control of one or several virtual components. The components belonging to the TOE are controlled by it. In our proposed testbed, a virtual and a real controller are employed interchangeably. Thus, our testbed contains

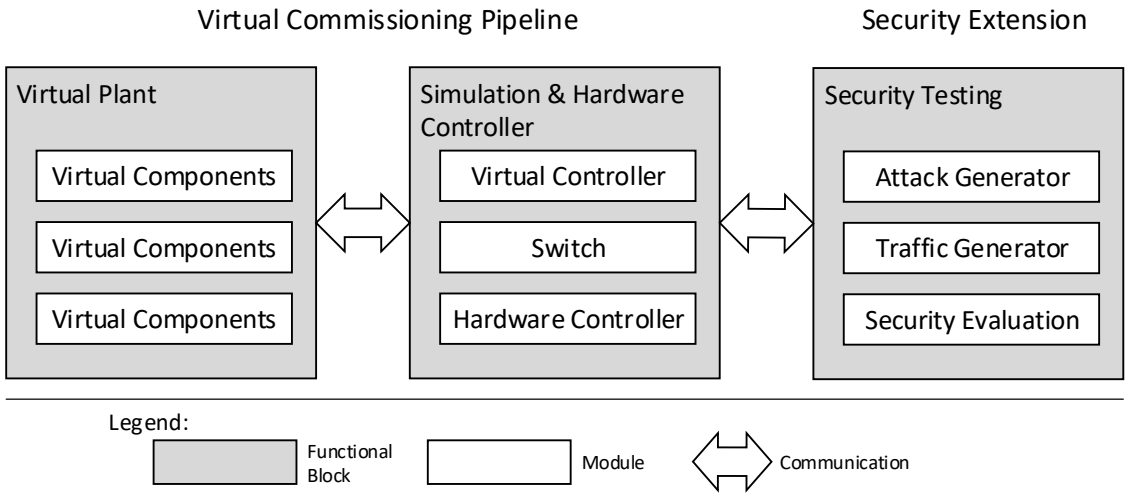

Fig. 1: Conceptional architecture.

the VC types of HIL and constructive commissioning (see Table I). For most tests, it is reasonable to first test the IT security with a virtual controller in order to avoid destruction of the TOE. Furthermore, the actual equipment may not be prototyped in some cases [5]. In the second step, the real controller, if available, can be used to verify the outcome of the conducted test. The alternation between the virtual and the real controller is performed by a switch module.

The conceptual descriptions in this article are illustrated by an ongoing example that is developed over the course of this article in several parts. It is then implemented and simulated in Section V.

Example pt. 1: Consider a shopfloor setup consisting of various robotic arms. Those are placed in a row to perform a combination of pick-and-place and welding operations. Applications for such a setup can be the assembly of vehicles or large machinery. One of the robotic arms is the TOE.

The functional block for security testing is located on the right-hand side of Figure 1. It consists of the attack and traffic generators as well as the IT security evaluation module. The attack generator module is responsible for providing traffic that mimics or inflicts attacks while the traffic generator module provides typical, non-malicious traffic. In this work, the attack scenarios are limited by the following assumptions.

Assumption 1: The attacker successfully penetrated the factory and has access to the internal OT network.

This means the attacker finished the initial step of gaining access to the internal networks of the factory. The attacker can achieve this via several attack vectors, e.g., a successful cyber attack via public networks such as the Internet or by positioning himself or a compromised device inside the factory [12].

Assumption 2: A viable attack vector is known to the attacker.

The attacker is modeled after the Dolev-Yao model for security evaluations in public key cryptography [25]. This means that the attacker can eavesdrop on transmitted messages and generate any message freely. Therefore, the attacker is in complete control of the messages transmitted.

Example pt. 2: The attacker can send arbitrary messages to one robotic arm in the shopfloor setup (cf. Example 1). Further, the attacker is able to capture and alter messages sent to the robot, thus, executing a man-in-the-middle (MITM) attack. This allows for a variety of possible threats to the robot and the assembly line. For example, the attacker can send a stop command resulting in an immediate halt of the robot; or the attacker can alter welding parameters resulting in welds of insufficient quality.

The assumptions above describe the attacker's capabilities and, therefore, constitute a definition of the attacker model, which is necessary for security evaluations. The security evaluation module itself contains the security model for the TOE and its associated process(es). For this work, we use a simple model consisting of three parts: security category, security criteria, and evaluation metric [20].

Security categories help to retain focus for the studies within the testbed. IT security is a broad area of research with a multitude of applications. To manage complexity, it is beneficial to focus on a certain area or domain of IT security. In the context of VC, existing categorizations for reasoning about security in ICS can be employed as VC is located in this domain. A recent taxonomy, i.e., a scheme of categorization, for security in ICS is provided by [26]. It allows to model the physical aspects of manufacturing systems. This is important to capture the impact of availability on production systems and to better encompass CPSs (see Section I).

Security criteria are used to evaluate the security of the TOE. Either by conforming to a specific security criteria present in the TOE or by comparing implemented security criteria against a predefined set of criteria. However, as

it is the case with security categories, it is important to manage the complexity for the security criteria. Therefore, we develop a more compact set of security criteria for this work.

Another term for security criteria is countermeasure, e.g., authentication measures or encryption. [27] provide an overview on security countermeasures for the domains of smart grid and smart home appliances. These are related domains to manufacturing as smart grid uses components similar to manufacturing. For example, PLCs and smart home appliances are similar to CPSs.

Finally, the evaluation metric provides quantifiable results from the testbed. Evaluating if a security criteria has been met or not is possible in principle. However, for some criteria such a binary distinction may not be applicable or suitable. For example, consider intrusion detection systems (IDSs). They are concerned with detecting malicious traffic, i.e., computer attacks, in an IT network [11]. They achieve this by inspecting packets or data records. It is not sufficient, for example, to measure the number of packets classified by the IDS as malicious. This is due to the fact that even sophisticated IDSs produce false positives.Thus, in the case of IDSs, more differentiated metrics are necessary, e.g., measuring the rate of false positives and true positives.

In this work, we focus on the impact of attacks on manufacturing systems. Thus, we employ metrics related to system and network performance as well as to the physical process of the manufacturing system. [29] provide a compilation of these metrics adopted to the use in industrial setups. These metrics can also be used to, e.g., construct more elaborate metrics for the evaluation of IDSs as illustrated above. The metrics listed in [29] are related to the quality of the product or the operation of the manufacturing equipment (the TOE). Thus, the metrics are suitable for VC as they aid in improving the quality of the TOE.

The overall goal of the security evaluation module is to provide comparability of the results among different testbed implementations. Example 3 provides an overview on modeling with the security evaluation framework described above.

Example pt. 3: An attacker can exploit weaknesses in the operating system (OS) of an industrial component and introduce malware in order to target a robotic arm. The attacker aims at maximizing the attack's impact and targets the integrity of the manufactured product (the security category according to [26]). He achieves this by attacking the material of the product. For welding, the strength of material is easily manipulated by changing the parameters of the resulting welding seam. Valid security criteria to counteract these attack are, e.g., authenticity checks of packets received with new welding parameters via keyed cryptographic hash functions [27]. The evaluation metric to measure the impact of the attack and the effectiveness of the implemented countermeasures can be achieved by dynamic performance metrics related to discrete processes. These can include, but are not limited to, product quality, defect rates, or defects per unit [29].

In Section IV, the ongoing example is further developed.

IV. Case Study: Discrete Assembly

In this section, a case study for our testbed described in Section III is given. As highlighted, the testbed can be used to model a variety of industrial setups. To demonstrate the viability, we focus on a discrete assembly process as sketched by Example 3.

Industrial assembly processes are categorized by the flow of material [29]. In a continuous process, material is constantly flowing through the manufacturing environment, e.g., in water treatment or other chemical processes. In discrete processes, the flow of material is quantifiable. Examples are automotive assembly or sorting operations performed by robots (see Example 3). Stop and wait states are typical to occur in discrete processes. A combination of both, continuous and discrete, processes, can be encountered in manufacturing as well. These are characterized by an interruption of the continuous material flow, where discrete operations need to be performed. This is the case for, e.g., pharmaceutical or metal-alloy assembly.

The choice for a discrete process was made to reduce complexity of the experimental testbed (see Section V). Also, actual manufacturing equipment for replicating parts of the assembly line are available for this work allowing verification of the simulation results with real-life equipment. Simulations are inherently built to offer conclusions under altering simulation parameters and simplified conditions [30]. Verification of a specified set of simulation parameters, i.e., those resulting from a security incident, with a physical representation of the simulation model can help to filter out those sets of parameters that operate under invalid conditions.

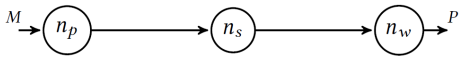

Fig. 2: Abstracted model of an assembly line.

Figure 2 shows an abstracted view on a discrete assembly line represented as graph [31]. The incoming raw material is shown on the left, denoted as M. By traversing the graph along its nodes n, M is transformed in the final product P. The assembly line is capable of performing three operations: pick-and-place (denoted as n_p), sorting (n_s), and welding (n_w). The arrows connecting the nodes provide a directed transition of M, thus, representing the material flow through the assembly line.

Example pt. 4: The assembly line shown in Figure 2 is used in the manufacturing of automotive parts (or vehicles). Pieces of metal are picked from incoming component logistics by one or several robotic arms located at n_p and placed on a conveyer belt (the directed arrows). From there, the raw material is transported to n_s, where the material is sorted, again by robotic arms. This sorting step is necessary to ensure proper handling by the welding robots at n_w. Here, pieces of metal are joined together producing the desired automotive component, e.g., a part of the chassis.

For the scenario described in Example 4, Assumption 3 is derived.

Assumption 3: Only one model of robotic assembly equipment is used.

The robotic arms used in Example 4 are assumed to be the same model produced by the same company. This assumption is reasonable as usage of identical manufacturing equipment offers benefits to the plant operator by reduced initial purchasing prices and ease of maintenance. Furthermore, modeling effort is reduced.

Two scenarios are discussed within the context of Example 4: sorting and welding. For this, we detail the architecture for testbed construction as illustrated in Figure 1. On the left-hand side, we specify the following virtual components: the production process of the assembly line, he TOE with its simulation model, and the physical counterpart of the TOE. The physical TOE requires its own controlling instance while the controller for the virtual devices, including the switch controller, can be executed on the same machine (cf. center of Figure 1). This machine represents a process control computer as found on the shopfloor. On the right-hand of Figure 1, the process supervision is located. It contains an engineering station for maintaining and adjusting programs for the assembly line actuators. Also, a generic traffic generator is located here. This component can be used, e.g., to generate network load or to replay real-life communication data captured from an OT network. This can be useful for certain scenarios and allows for verification under more realistic conditions. Finally, the attacker node is located here as well (cf. Assumption 1).

The attacker is responsible for executing attacks on the TOE. One or several attack vectors are known to him (cf. Assumption 2). Building up on Examples 3 and 4, the following attack vectors are the subject of the case study on discrete assembly.

Scenario 1: Threatening the availability of the assembly line via sorting parameter manipulation.
In this scenario, the attacker's goal is to threaten the availability of the assembly line. He aims to minimize his effort and maximize the impact. He attacks the integrity of the communication between the control devices and the devices interacting with the process . Specifically, the sorting portion is targeted by the attacker (cf. Figure 2, node n_s). This scenario is motivated from a real-life case study on the security of manufacturing parameters [31]. We model this scenario according to our choice of security evaluation framework (see Section III):

- **Security categories:** Availability attack on manufacturing equipment by damaging equipment.
- **Security criteria:** Cryptographic hash functions for ensuring integrity and authenticity.
- **Evaluation metrics:** Percentage of availability of the assembly line.

Example pt. 5: By manipulating the parameters for force and placement within the messages transmitted between the robotic arm and its controller, the attacker can cause the arm to damage the conveyer belt by forcing the arm into it. Cryptographic countermeasures are not in place, which is common in manufacturing [12] . The percentage of time the assembly line is not able to continue normal (or any) operation at all is the measure of success for the attacker and, also, for the countermeasures in place (if any).

Scenario 2: Advanced persistent threat (APT) on the assembly line via subtle welding parameter alteration.
As opposed to Scenario 1, the attacker aims at threatening the integrity of the product. In this scenario, the attacker puts more effort into the design of the attack vector. He aims to stay undetected for a prolonged period of time to maximize its impact, i.e., affecting the largest possible amount of products assembled. The principle attack vector is similar to Scenario 1 as the attacker mostly focuses on alteration of communication between Controller and Security Testing. Attack targets are the welding robots (cf. Figure 2, node n_w). This attack is inspired by Stuxnet and similar advanced attacks on actual industrial plants [10]. Again, the security model is provided in accordance to the chosen security evaluation method:

- **Security categories:** Integrity attack on the material of the product by altering the material strength.
- **Security criteria:** As Scenario 1. In addition, a sophisticated IDSs capable of detecting subtle alterations in manufacturing parameters.
- **Evaluation metrics:** Product quality, product defect rate.

Example pt. 6: By manipulating the parameter set transmitted to the welding robots, the attacker can weaken the welding points set for joining the metal pieces together. This can go undetected in quality assurance and lead to negative results in the context of automotive manufacturing.

As mentioned in Section III, the metrics chosen for evaluation of the security criteria relate to ensuring the quality of operation for the TOE, the main goal of VC. This is clear in the case of Scenario 2, where product quality and product defect rate directly correspond to the quality of operation. Scenario 1 shows the added benefit of integrating security evaluations into VC. By studying this scenario, security-relevant interactions can be considered in addition to quality evaluations. Thus, implementation of proper security controls can be supportive in increasing overall product quality.

This section showcased, how to apply security evaluations to discrete manufacturing processes. The study of continuous manufacturing processes is also possible by adding a model and simulation of the process in question and integrating it into an experimental framework (see Section V). The scenarios detailed are implemented according to the reference architecture and simulated in experimental setups in the following section.

V. Experimental Evaluation

In this section, the experimental evaluation of our modeling framework described in the previous sections is provided. First, we give a reference implementation of the framework in Section V-A. Next, we use our implementation in Section V-B to simulate the case study on discrete assembly with varying parameters.

A. Implementation

The implementation is based on the robotics framework Robot Operating System (ROS), a middleware for developing robotic applications [32]. It is open-source, accompanied by a comprehensive documentation, and actively developed as of writing of this work. It provides hardware abstraction and, therefore, aids in keeping the proposed testbed open to future extensions to other scenarios. ROS offers an extensive ecosystem, which is employed during development of the testbed.

The modules of the testbed related to the physical processes of the plant (see Figure 1) are implemented with tools from the ROS ecosystem. The simulation of the TOE is implemented in *rviz*. It is a 3D robotic modeling environment and allows for detailed observation of the TOE. Also, motion planing is conducted within *rviz*. The assembly line scenario is simulated in *gazebo*. It can be used within the ROS ecosystem to provide a simulation of a robot and its environment as it includes a world building editor and a physics engine. Thus, *gazebo* is a useful tool to simulate the impact the TOE has on the environment. To provide a more realistic model and simulation of manufacturing equipment, the ROS-Industrial (ROS-I) extension for ROS is used. ROS-I includes interfaces to common industrial networks, sensors, and actuators. The physical representation of the TOE, a robotic arm used commonly in manufacturing, is included in the testbed via the virtual controller.

The modules for Controller and Security Testing are implemented by using and modifying constructs from the ROS framework. In the context of ROS, a *node* is an executable that is executed by the ROS framework. *Nodes* communicate with other nodes inside the framework meaning they are able to send and receive messages. Thus, they are well-suited to represent attacker and traffic generators within the testbed. In addition, the nodes collect the data required for security evaluations. The virtual controller is also a separate node that sends and receives messages from the real and virtual TOE and other virtual components. The switch module is provided by two different launch configurations for ROS: one for security evaluations of the virtual TOE and one for the real TOE respectively.

Table II provides an overview about the implementation of the testbed. It shows the mapping from the conceptual framework (see Section III) over the discrete assembly scenario (see Section IV) towards the implementation described in this section. Each line illustrates how a conceptual module is translated to the scenario and the implementation. The following software components are used for implementation: Ubuntu 14.04, ROS Indigo, ROS-I 0.4.3, *rviz* r1.11.19, and *gazebo* 2.2.3.

The implementation is focused on providing support for the discrete assembly line scenario but can be extended to encompass continuous industrial processes as well. For this, a simulation environment for the continuous process needs to be integrated into the testbed. This can be achieved by employing the publish-subscribe architecture ROS is based on. Messages sent by the simulation environment can be processed by a dedicated ROS node and, thus, extracted and used within VC for more detailed equipment verification.

TABLE II: Implementation of testbed modules.

Line	(Testbed) Module	(Scenario) Module	Implementation
1	Virtual component	Simulation assembly line	*gazebo*
2	Virtual component	Simulation TOE	*rviz*, *gazebo*
3	Virtual controller	Virtual controller	ROS node
4	Switch	Switch	ROS launch files
5	Attack generator	Attacker node	ROS node
6	Traffic generator	Traffic node(s)	ROS node(s)

B. Simulation

For each of the two scenarios described in Section IV, a experiment is implemented. In Experiment 1, Scenario 1 is implemented and simulated as described by Example 5. For Experiment 2, the same is true for Scenario 2 and Example 6. The simulation of the scenarios is executed within *gazebo*. The world building capabilities of *gazebo* have been used to model the simple assembly line shown in Figure 2. Both experiments are first conducted within the simulation environment. When the expected results are met by the simulation, the switch module is used to conduct a real life verification of the simulation results. For this, an industrial grade robotic arm with an end effector is used. This robotic arm represents the TOE within the VC process as outlined in Section II. The simulations are executed on a standard office PC with a dedicated graphics card.

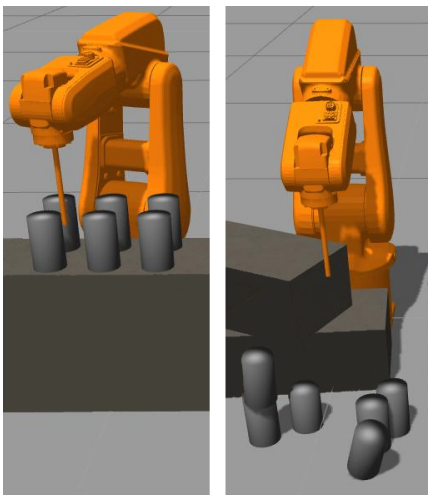

Fig. 3: Simulation of Experiment 1 during normal operation (left) and an attack scenario (right).

The result of simulation for Experiment 1 is seen by the images in Figure 3. They show one component, i.e., a robotic arm, of the assembly line positioned next to the conveyor belt. The images are edited to remove corporate branding,

which is part of the model we used. On the left-hand side, normal operation of the robot is shown, where the robot is moving between a couple of cylindrical objects. The movement is controlled by a ROS node executing a predefined program of movement instructions. On the right-hand side, the results of the successfully executed attack described in Scenario 1 are shown. A manipulated parameter file is sent to the simulation via the attacking ROS node, which causes the arm to alter its movement and crush into the conveyor belt. The effect for the attack is visualized by the destruction of the affected conveyer belt segments. Those segments are assumed to be non-operational after the impact caused by the arm making the belt (and also the arm) unavailable until repairs are finished. With sufficent countermeasures, e.g., cryptographic hash functions, this effect can be mitigated and the malicious messages are discarded [33].

The simulated scenario in Experiment 1 showed the results of an integrity attack on industrial euqipment with the goal of reducing the availability of the production line (see Section IV). In the case with no security criteria present, the availability of the production line dropped from 100% to 0%, whereas the availability stayed constant with sufficient countermeasures in place.

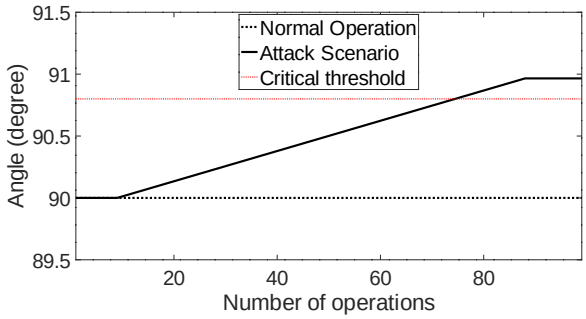

Fig. 4: Simulation results of Experiment 2 ($\alpha_C = 90.8$).

Experiment 2 is concerned with the simulation of welding applications. For this, the robotic arm is equipped with a welding rod. The arm applies the welding seam by conducting a parallel movement to the object receiving the seam. The movement is executed in a fixed distance from the object in order to apply a seam of sufficient density. If the distance is outside certain boundaries, the seam is likely to tear. To verify the quality of operation for the welding process, measurement values are retrieved from the simulation environment. The values are extracted from the ROS nodes responsible for controlling the process.

The retrieved measurements are given by Figure 4. It shows the movement of the robot over time, i.e., the number of consecutively executed welding operations. The robot's movement is represented by the angle between the object and the welding rod. Ideally, this angle is close to zero during normal operation as parallel movement of the arm is the expected behavior. Normal operation is shown by the dashed line in Figure 4. The arm is moving with high precision keeping the angle close to 0 degree. This is realistic, as the arm used in the experiments is designed for high precision operations. The solid line in Figure 4 shows the development of the angle during the presence of an APT, which is implemented as a separate ROS node. The APT initially continues normal operation but starts at operation $n = 10$ to slowly alter the movement of the arm leading to a steadily increasing angle. At $n = 75$, the APT reaches the threshold for a critical angle $\alpha_C = 90.8$ (red line), where the seam is assumed to be of insufficient quality. From $n = 90$ onwards, the angle is not increased further to avoid detection. The impact of the attack can be measured by the reduction in product quality. Affected products are those products manufactured after $n = 75$ until the detection of the APT. It is assumed that those products are more likely to tear resulting in a reduced product quality and, ultimately, in a higher defect rate of manufactured goods.

Both attacks are first simulated and then executed on the actual device via the switch module. This way, we confirmed the findings of the simulation for both experiments within a real-world setup.

VI. Discussion

The testbed we presented can be integrated into existing VC pipelines as modeling and simulation activities are already an essential part for VC. Our approach enhances existing VC pipelines and processes by the possibility to conduct security evaluations within a testbed. As we demonstrated, the technical requirements for this are available and can be utilized.

We used a method for security evaluation that is based on existing work. The catalogs we used can be extended by increasing the focus on security within manufacturing. For security categories this can be achieved by adopting established evaluation methods, e.g., from the automotive domain [34]. For security criteria, a dedicated catalog specifically focused on security in manufacturing can be used reasonably [26]. For the evaluation metrics, enhancements specifically for measuring security criteria can be added [24]. Also, a metric that counts the detection of false positives in IDS systems can aid in improving evaluation of this security criteria [28].

However, for security evaluations to be effective within VC, procedural adjustments in existing plant and commissioning operations are required. The enforcement of security evaluations in product commissioning needs to be integrated into polices that require the regular use of processes and testbeds as described by us in this study.

VII. Conclusion

In this work, we showed how to integrate security evaluations into a testbed that can be used within VC processes. For this, we give a conceptual architecture, which highlighted the approach on integrating attackers and security evaluations into a high-level VC setup. From there, we conducted a case study on discrete assembly. We modeled two scenarios from a security perspective highlighting attack vectors occurring in manufacturing. Finally, we implemented and simulated the scenarios using the ROS middleware and the simulator

gazebo demonstrating the applicability of our approach. Our simulation shows how to estimate the impact of an attack and we discussed possible countermeasures for the attack. Future extensions and improvements to the security evaluation framework are sketched.

REFERENCES

[1] J. Lee, B. Bagheri, and H.-A. Kao, "A cyber-physical systems architecture for industry 4.0-based manufacturing systems," *Manufacturing Letters*, vol. 3, pp. 18–23, 2015.

[2] C. G. Lee and S. C. Park, "Survey on the virtual commissioning of manufacturing systems," *Journal of Computational Design and Engineering*, vol. 1, no. 3, pp. 213–222, 2014.

[3] M. Ko, E. Ahn, and S. C. Park, "A concurrent design methodology of a production system for virtual commissioning," *Concurrent Engineering*, vol. 21, no. 2, pp. 129–140, 2013.

[4] S. C. Park, M. Ko, and M. Chang, "A reverse engineering approach to generate a virtual plant model for plc simulation," *The International Journal of Advanced Manufacturing Technology*, vol. 69, no. 9-12, pp. 2459–2469, 2013.

[5] P. Hoffmann, R. Schumann, T. M. Maksoud, and G. C. Premier, "Virtual commissioning of manufacturing systems a review and new approaches for simplification." in *ECMS*. Kuala Lumpur, Malaysia, 2010, pp. 175–181.

[6] S. Owicki and L. Lamport, "Proving liveness properties of concurrent programs," *ACM Transactions on Programming Languages and Systems (TOPLAS)*, vol. 4, no. 3, pp. 455–495, 1982.

[7] B. Alpern and F. B. Schneider, "Recognizing safety and liveness," *Distributed computing*, vol. 2, no. 3, pp. 117–126, 1987.

[8] A. Avizienis, J.-C. Laprie, B. Randell, and C. Landwehr, "Basic concepts and taxonomy of dependable and secure computing," *IEEE transactions on dependable and secure computing*, vol. 1, no. 1, pp. 11–33, 2004.

[9] M. Brunner, H. Hofinger, C. Krauß, C. Roblee, P. Schoo, and S. Todt, "Infiltrating critical infrastructures with next-generation attacks," *Fraunhofer Institute for Secure Information Technology (SIT), Munich*, 2010.

[10] S. D. Antón, D. Fraunholz, C. Lipps, F. Pohl, M. Zimmermann, and H. D. Schotten, "Two decades of scada exploitation: A brief history," in *Application, Information and Network Security (AINS), 2017 IEEE Conference on*. IEEE, 2017, pp. 98–104.

[11] C. Eckert, *IT-Sicherheit: Konzepte-Verfahren-Protokolle*. Walter de Gruyter, 2013.

[12] A.-R. Sadeghi, C. Wachsmann, and M. Waidner, "Security and privacy challenges in industrial internet of things," in *Design Automation Conference (DAC), 2015 52nd ACM/EDAC/IEEE*. IEEE, 2015, pp. 1–6.

[13] P. Puntel-Schmidt and A. Fay, "Levels of detail and appropriate model types for virtual commissioning in manufacturing engineering," *IFAC-PapersOnLine*, vol. 48, no. 1, pp. 922–927, 2015.

[14] B. Kehoe, S. Patil, P. Abbeel, and K. Goldberg, "A survey of research on cloud robotics and automation," *IEEE Transactions on automation science and engineering*, vol. 12, no. 2, pp. 398–409, 2015.

[15] M. Lezzi, M. Lazoi, and A. Corallo, "Cybersecurity for industry 4.0 in the current literature: A reference framework," *Computers in Industry*, vol. 103, pp. 97–110, 2018.

[16] S. Süß, A. Strahilov, and C. Diedrich, "Behaviour simulation for virtual commissioning using co-simulation," in *Emerging Technologies & Factory Automation (ETFA), 2015 IEEE 20th Conference on*. IEEE, 2015, pp. 1–8.

[17] A. Martins, H. Costelha, and C. Neves, "Supporting the design, commissioning and supervision of smart factory components through their digital twin," in *2020 IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC)*. IEEE, 2020, pp. 114–119.

[18] M. Dahl, A. Albo, J. Eriksson, J. Pettersson, and P. Falkman, "Virtual reality commissioning in production systems preparation," in *2017 22nd IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*. IEEE, 2017, pp. 1–7.

[19] H. Holm, M. Karresand, A. Vidström, and E. Westring, "A survey of industrial control system testbeds," in *Secure IT Systems*. Springer, 2015, pp. 11–26.

[20] R. von Solms, H. Van Der Haar, S. H. von Solms, and W. J. Caelli, "A framework for information security evaluation," *Information & Management*, vol. 26, no. 3, pp. 143–153, 1994.

[21] Y. Zhiwei and J. Zhongyuan, "A survey on the evolution of risk evaluation for information systems security," *Energy Procedia*, vol. 17, pp. 1288–1294, 2012.

[22] S. Chockalingam, D. Hadžiosmanović, W. Pieters, A. Teixeira, and P. van Gelder, "Integrated safety and security risk assessment methods: a survey of key characteristics and applications," in *International Conference on Critical Information Infrastructures Security*. Springer, 2016, pp. 50–62.

[23] Y. Wang, R. Fakhry, S. Rohr, and R. Anderl, "Combined secure process and data model for it-security in industrie 4.0," in *Proceedings of the International MultiConference of Engineers and Computer Scientists 2017*, vol. 2, 2017.

[24] A. Giehl, N. Wiedermann, and S. Plaga, "A framework to assess impacts of cyber attacks in manufacturing," in *2019 11th International Conference on Computer and Automation Engineering Proceedings*. New York, NY, USA: ACM, 2019.

[25] D. Dolev and A. Yao, "On the security of public key protocols," *IEEE Transactions on information theory*, vol. 29, no. 2, pp. 198–208, 1983.

[26] Y. Pan, J. White, D. C. Schmidt, A. Elhabashy, L. Sturm, J. Camelio, and C. Williams, "Taxonomies for reasoning about cyber-physical attacks in iot-based manufacturing systems." *International Journal of Interactive Multimedia & Artificial Intelligence*, vol. 4, no. 3, 2017.

[27] N. Komninos, E. Philippou, and A. Pitsillides, "Survey in smart grid and smart home security: Issues, challenges and countermeasures," *IEEE Communications Surveys & Tutorials*, vol. 16, no. 4, pp. 1933–1954, 2014.

[28] P. Schneider and A. Giehl, "Realistic data generation for anomaly detection in industrial settings using simulations," in *Computer Security*. Springer, 2018, pp. 119–134.

[29] R. Candell, K. Stouffer, and D. Anand, "A cybersecurity testbed for industrial control systems," in *Process Control and Safety Symposium, International Society of Automation, Houston, TX*, 2014.

[30] P. Grim, R. Rosenberger, A. Rosenfeld, B. Anderson, and R. E. Eason, "How simulations fail," *Synthese*, vol. 190, no. 12, pp. 2367–2390, 2013.

[31] A. Giehl and N. Wiedermann, "Security verification of third party design files in manufacturing," in *2018 10th International Conference on Computer and Automation Engineering Proceedings*. ACM, 2018.

[32] M. Quigley, B. Gerkey, and W. D. Smart, *Programming Robots with ROS: a practical introduction to the Robot Operating System*. " O'Reilly Media, Inc.", 2015.

[33] A. Giehl and S. Plaga, "Implementing a performant security control for industrial ethernet," in *2018 International Conference on Signal Processing and Information Security*. IEEE, 2018.

[34] D. Angermeier, K. Beilke, G. Hansch, and J. Eichler, "Modeling security risk assessments," in *17th Embedded Security in Cars (escar Europe)*, 2019, pp. 133–146.