Secure Remote Maintenance via Workflow-Driven Security Framework

Prabhakaran Kasinathan
Cybersecurity Technology
Siemens AG
Munich, Germany
prabhakaran.kasinathan@siemens.com

Davide Martintoni
Applied Research and Technology
Collins Aerospace
Trento, Italy
davide.martintoni@collins.com

Benedikt Hofmann
Cybersecurity Technology
Siemens AG
Munich, Germany
hofmann.benedikt@siemens.com

Valerio Senni
Applied Research and Technology
Collins Aerospace
Rome, Italy
valerio.senni@collins.com

Martin Wimmer
Cybersecurity Technology
Siemens AG
Munich, Germany
martin.r.wimmer@siemens.com

Abstract

Remote Maintenance in collaborative manufacturing provides a lot of benefits such as reduced downtime in manufacturing operations. But at the same time, it increases the attack-surface by opening new attack paths to strictly controlled network zones. In this paper, we analyse a real-world cross-organizational remote maintenance scenario by collecting security requirements (e.g., authentication, authorization, and auditability), and present a workflow-based approach to model and formally enforce access control for that scenario. The proposed approach leverages the Workflow-Driven Security Framework (WDSF) to enforce the least privilege principle; to ensure workflow integrity and separation of duties, (i.e., business process enforcement and compliance); to protect the confidentiality and integrity of sensitive information; and to provide traceability and non-repudiation in case of root-cause analysis. The WDSF uses Petri Nets (PN) to model and enforce the workflow, and blockchain and smart contracts to guarantee accountability and traceability of workflow events. The Petri Nets workflows are modelled and validated using the WoPeD tool.

Index Terms—Blockchain, Security, Workflows, Petri Nets, Remote Maintenance, Workflow-Driven Security Framework

I. Introduction

Manufacturing plants employ Operational Technology (OT) technology and protocols that have been specified decades ago. Their security model is primarily based on perimeter security/protection, meaning that they are operated in a closed and strictly controlled environment.

However, remote connections from external networks to the internal manufacturing network are necessary to enable flexibility and fast reaction times to adapt production capabilities, and to leverage expertise from different entities across collaborative manufacturing and organizational boundaries. But, when internal networks and manufacturing devices are exposed to the Internet, then it increases the attack surface of the entire plant. Hence, this enables the attacker a) to compromise vulnerable services or devices, b) to escalate the attack by moving laterally to other parts of the plant's network topology. Moreover, when the manufacturing process involves producing critical resources and products (e.g., safety-critical parts in automotive and civil aviation sections) then the attack could lead to devastating effects. Some example consequences are financial impact e.g., business continuity; people's safety; and loss of reputation (i.e., brand damage). In particular, misconfigured and vulnerable Industrial Internet-of-Things (IIoT) devices are listed in the top 10 OWASP vulnerabilities [1], and a survey of such IoT enabled attacks is presented in [2]. Further challenges concerning cybersecurity in Industry 4.0 are presented in the ENISA report [3].

Usually, a plant consists of hardware and software components from different vendors and the plant owner has full control of the infrastructure i.e., centralized control & trust. In case of a successful attack, all involved entities must trust the plant owner to act correctly. Currently, it is easier for the plant owner to blame any involved vendor. So, the vendor must accept the consequences without having the possibility to prove otherwise. For instance, a plant employee could be responsible for a misconfigured service or device. Because of centralized control, many vendors/suppliers do not take the responsibilities or liabilities for operating devices or supplying software components. To address these issues, we need a framework that supports: (a) distributed trust and control, (b) enforcing use case specific security requirements defined by the plant owner; (c) ensures the auditability and traceability of entity actions.

The contributions of this paper are as follows: 1. To

provide a concrete remote maintenance use case and its security requirements in a typical industrial manufacturing plant and to analyse the associated threats and risks; 2. To demonstrate how the Petri Nets-based Workflow-Driven Security Framework (WDSF) ([4], [5], [6]) can be used to enforce the remote maintenance workflow and security requirements, and to protect edge IIoT devices and services; 3. To show how the modelled Petri Nets (PN) workflows can be semantically analysed to satisfy workflow-net properties like structural and soundness properties using the WoPeD tool [7]; 4. To evaluate the performance of the smart contracts that are derived from the modelled PN workflows (see [8]) and show that the proposed approach is suitable for the remote maintenance use case scenario. Thus, this approach eliminates the need for a trusted third party (TTP) to maintain and control the audit logs and in case of a dispute, authorized actors can trace the workflow actions to find the root-cause of the problem in a distributed and collaborative environment; 5. Finally, to demonstrate the prototypical system architecture and how different WDSF components are deployed in a proof-of-concept (PoC) demonstrator.

The paper is organized as follows: Sec. II introduces the remote maintenance use case and its requirements; Sec. III presents WDSF's approach; Sec. IV explains how the remote maintenance use case's workflow is modelled via Petri Nets and what properties are formally verified, and how they are enforced via WDSF. In addition, this Sec. presents the system architecture; Sec. V discusses the related work; Finally, Sec. VI concludes with future work.

II. Remote Maintenance Use Case

One of the fundamental requirements of a manufacturing plant is to control and restrict access to the machines and devices operating inside the factory as limited as possible according to the least privilege principle. The devices used in manufacturing scenarios need regular updates to operate securely and optimally e.g., installation of software updates, security patches, or updating configuration enable predictive maintenance analysis. Often this task is outsourced to specialists who have domain knowledge about the devices and the operating environment. Typically if a machinery stops operating in a plant, then to understand what kind of maintenance is necessary, a skilled person (usually from the original equipment manufacturer (OEM)) is required. This could take days to weeks depending on the OEM technician's availability, the location and access process of the plant. Finally, the specialists are given physical access to the manufacturing plant. This process is time-consuming and expensive: production may be stopped, and the risk of intentional and unintentional installation of a malware or theft of data and Intellectual Property (IP) is increased. There are several advantages in employing remote access such as quicker fix of the problem, business continuity, lower costs, and more importantly, the possibility to isolate access granted to the maintenance person i.e., enforcing need to know and least privilege principles. Thus, physical access can be limited to the cases where it is required and on-site support has to be provided e.g., with the assistance of the shop-floor plant employees.

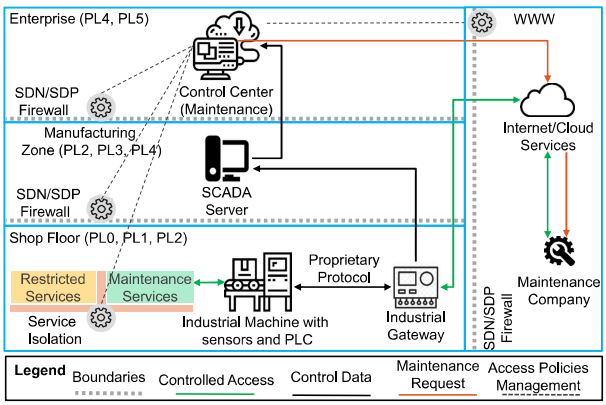

Fig. 1. Remote maintenance reference architecture in Industry 4.0 is mapped to the Purdue enterprise architecture levels

A remote maintenance use case includes the following steps: 1. An (incident) event occurs because of an equipment or device (e.g., programmable logic controller (PLC)) failure or malfunction; 2. A service request is sent to the maintenance company; 3. Access for a Service Engineer of the maintenance company needs to be approved and temporary access to the device must be granted; 4. Depending on the plant's infrastructure, the access permissions may involve traversing different layers of the plant's networks e.g., by means of changing the firewalls/routers configuration; 5. The Service Engineer performs the remote maintenance operation; 6. Finally, the access path is closed again by resetting the network to its default configuration, i.e., by closing the temporarily opened access path.

This paper considers a high-level architecture of digital infrastructure for collaborative manufacturing inspired from the use case that Advanced Laboratory on Embed-

TABLE I
Excerpts of prioritized threats and associated risks collected via STRIDE Methodology which focuses on the following threats: Spoofing (S), Tampering (T), Repudiation (R), Information disclosure (I), Denial of service (D), and Elevation of privilege (E).

STRIDE Threats	Risk Rating	Description
S-01	High	Access credentials are spoofed
I-01, E-01	Medium	Confidential data (information leakage) is accessed e.g., via privilege escalation resulting in a privacy breach
T-01, E-02	Medium	Access credentials are issued via tampering or privilege escalation e.g., without proper approvals
R-01	Medium	External entity potentially denies data transmission or reception
D-01	Low	Potential denial-of-service (DoS)

TABLE II
Protection requirements (RQ) identified as STRIDE threat countermeasures

Requirements	Description
RQ-01	Access credentials are provided with the least privilege principle e.g., short-lived, resource specific access.
RQ-02	All users are managed via an identity and access management system (IAM) e.g., authenticated and authorized before granting access.
RQ-03	System integrity protection e.g., only validated & authorized applications can be executed.
RQ-04	Protection against denial-of-service e.g., reducing downtime of operations.
RQ-05	Logging of actions of entities ensuring non-repudiation and immutability e.g., enables auditability and traceability.
RQ-06	Ability to model, design, and enforce business processes (e.g., for managing access-control) in a distributed environment.

ded Systems (ALES), a Raytheon Technologies (RTX) Company, brings into the European Union (EU) project COmprehensive cyber-intelligence framework for resilient coLLABorative manufacturing Systems (COLLABS) [9]. In Europe, RTX has multiple engineering and manufacturing sites, and employs innovative manufacturing lines to deliver products more quickly and with superior quality. In Fig. 1 we illustrate an abstract view of a RTX digital manufacturing infrastructure which is mapped to the Purdue Enterprise Reference Architecture[10]. The architecture represents the high-level data flows between different Purdue Levels (PL) of the plant. The shop floor includes PL0, PL1, PL2, respective industrial equipment, and operational technology (OT) components. The Manufacturing floor includes PL2, PL3, PL4 and it contains the supervisory control and data acquisition (SCADA) control systems. The Enterprise layer represents PL4, PL5 and include systems and functions controlling the entire production at a higher level of abstraction. The WWW represents the internet, it is outside of the enterprise boundaries and supports connectivity with other manufacturing sites as well as partners involved such as the remote maintenance. Notice that each zone is logically isolated from the other ones by gateways and firewalls that are configured to avoid unauthorized data flows. Currently, network access rules enforce coarse-grained access control and is not capable out-of-the-box to implement fine-grained access as required by the use case.

A. Security Requirements Identification

In a typical factory set-up, the following protection goals apply: Availability: minimizing the downtime of the production plant; Integrity: software and hardware integrity must be guaranteed by applying only approved modifications; Confidentiality: intellectual property must always be protected from unauthorized access; Non-repudiation and Traceability: in case of an incident, it must be possible to trace back what happened with non-repudiation guarantees. We used the STRIDE methodology to identify the threats presented in Tab. I (see [11], [12]). Note: These are not covering all the remote maintenance use case scenario specific threats but focus only on prioritized threats. For a broader threat landscape, we refer to ENISA's report [3]. Similarly, a risk rating is provided for each threat based on inputs provided by domain experts (both from industry and academy). Next, we investigated the risk reduction/mitigation strategies and presented them as protection requirements in the Tab. II.

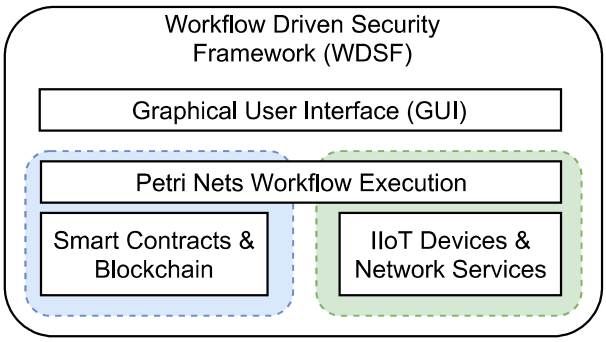

Fig. 2. WDSF Layered Architecture

III. Proposed Framework

In this paper, we adopt a Petri Nets (PN) based workflow framework previously developed in [4], [5], [6] - here referred as Workflow Driven Security Framework (WDSF) - to mitigate the identified threats and risks from a remote maintenance scenario as shown in Tab. I. WDSF is used to enforce the business logic (represented as workflows) as described by the plant owner and the requirements shown in Tab. II. The validated workflows are developed as smart contracts and executed via blockchain to enforce the business logic and provide accountability. In addition, the modelled PN workflows are used to generate smart contract templates, thus helping the developers to minimize business logic errors (see [8]). Besides, we present a blueprint deployment architecture of WDSF and its integration into a typical Industry 4.0 manufacturing plant. This paper shows how WDSF can be applied in a real-life use case such as remote maintenance by analysing use case requirements requirements, presenting the WDSF architecture, and deploying and evaluating its performance in a real-world (demonstrator) deployment.

A. WDSF high-level Architecture

The high-level architecture of WDSF is presented in Fig. 2 which consists of the following layers: 1) Graphical User Interface (GUI); 2) Petri Nets Workflow Execution 3.1) Smart Contracts & Blockchain; and 3.2) IIoT Devices & Network Services. These layers are combined to model, specify, validate, and enforce any workflow in our remote

maintenance use case. A brief description of each layer and their advantages are described below.

User Interface Layer: In existing manufacturing systems, many control systems and other technologies provide their own Graphical User Interfaces (GUI) and Application Programming Interfaces (APIs). These GUIs and APIs can be used to access and control technology specific features in a typical manufacturing setting. A workflow in a manufacturing environment involves using these features and controls. Therefore, in our approach, we provide mechanisms that can be integrated with the underlying WDSF that enforces business logic modelled as workflow and produces the audit trail of the workflow events. We rely on standard protocols such as REST interfaces to communicate with underlying WDSF layers. Note: The existing GUIs and APIs are designed to be extensible, i.e., to invoke WDSF components. For our demonstrator purposes, we developed a custom GUI that enables the workflow participant to execute and interact with the workflow.

Workflow Execution Layer: The WDSF introduces a Petri Nets abstraction layer which is used to model, specify, and execute workflows. To introduce practitioner-friendly business process modelling, WDSF in [6] introduces the use of Activity Diagrams from Systems Modelling Language (SysML) [13]. The PN workflows can be generated from the SysML activity diagrams. Additional components (e.g., IAM and Public Key Infrastructure (PKI) integrated with blockchain infrastructure) are required to achieve non-repudiation and immutable audit trails in distributed environments. State changes are triggered by entities that can control the PN execution layer. Further, we integrated smart contracts and a blockchain layer in the WDSF architecture to enable distributed validation of workflow events by all involved parties and to stop one entity from controlling the underlying workflow execution without the need for a trusted third party.

Smart Contracts and Blockchain Layer: In [8], an introduction to smart contracts and blockchain is given and the authors present an approach for translating PN workflows into smart contract templates that can be deployed on a blockchain. Briefly, a smart contract is a programmable code deployed on blockchain which is then executed and validated by blockchain nodes or peers. When the workflow participant triggers a state change in the PN, the workflow execution layer - on behalf of the workflow participant - triggers the appropriate smart contract method. The smart contracts are visible to relevant participants of the blockchain. Also, the participants must validate and endorse a transaction which invokes a method inside a smart contract - usually, the peers or special nodes with higher privileges perform transaction endorsement and validation. The WDSF approach when combined with a blockchain to record workflow related events, provides accountability and compliance i.e., business process enforcement, non-repudiation, traceability of events and related properties which are important to resolve disputes. In this work, we use Hyperledger Fabric (HLF) as the underlying blockchain infrastructure.

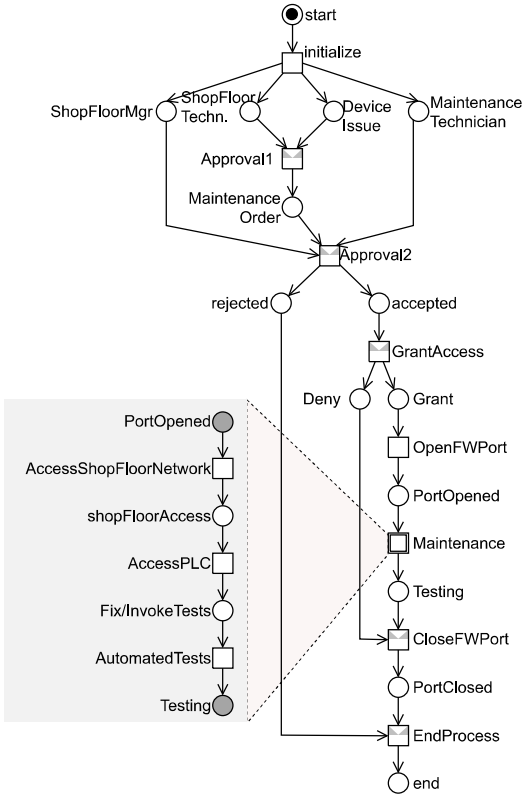

Fig. 3. Left: a sub-workflow of the use case showing that Petri Nets and the WoPeD tool supporting hierarchical modelling; Right: Petri Nets model of the remote maintenance use case modelled with WoPeD tool

B. Workflow Modelling and Validation

To satisfy functional and security requirements for critical industrial functionality, usually, a strict process is designed and defined by the owner of the production plant in collaboration with the maintenance company, and its goal is to securely provide external access to an internal machine. The process designed can then be translated and formalized as a Petri Nets (PN) workflow that can be enforced with the WDSF framework and can thus be integrated with other industrial components such as the network configuration system. The PN workflow corresponding to the use case is shown in Fig.3, and it represents on the right (main-workflow) the steps needed for the remote-maintenance use case. The main-workflow involves approvals from the shop floor technician, shop floor manager and a maintenance technician to start the maintenance process. The maintenance process is described as a sub-workflow on the left. The example PN workflows show the process of the workflow from a start place to the end-place. For the sake of brevity only a compressed version of workflow is depicted, however in

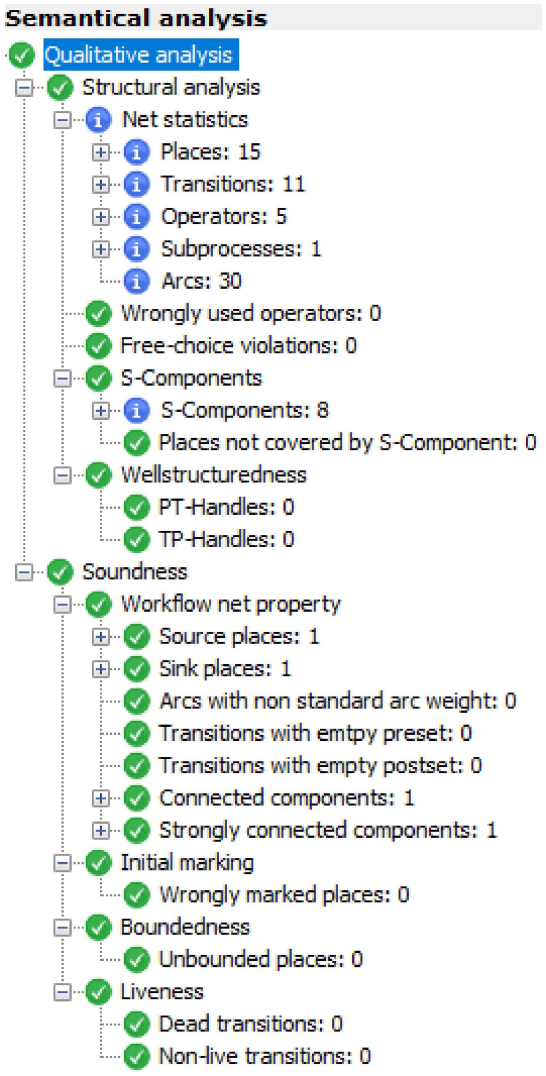

Fig. 4. The results of the structural analysis of the modelled Petri Nets workflow shows the satisfied workflow properties.

the complete version and implementation the workflow contains errors and error-recovery cases. A detailed explanation of the depicted workflow steps is provided in Sec. IV.

Cross-organizational workflows can quickly become complex to interpret and validate manually due to their high-level of abstractions, hierarchical structure, and semantics. PN workflows can be efficiently modelled, simulated, and validated using the open source WoPeD (Workflow Petri Net Designer)[1] tool [7]. Workflow-net properties (see [14]) such as deadlock-freedom, boundness, and liveness provide strong guarantees that the designed workflows work as expected without workflow errors. The qualitative analysis of the modelled workflow (see Fig.4) shows that the workflow is sound, well-structured, and satisfies the workflow-net properties; and does not contain

[1]https://sourceforge.net/projects/woped

TABLE III
WDSF addressing the remote maintenance requirements

Requirements	WDSF Requirements Compliance / Mitigation Strategy
RQ-01, RQ-02	The modular approach of WDSF allows integrating existing IAM, authorization server (AS) and Certificate Authority (CA) solutions. All participants are authenticated with their unique identities and authorized before executing a workflow action.
RQ-01	If a confidential/sensitive resource is protected by the WDSF framework, then a workflow participant may access that resource only after the successful execution of previous workflow steps. Thus, by default WDSF reduces the attack surface of the protected resources.
RQ-03	If the maintenance technician needs to execute a program, this can be submitted to security components such as malware analysis tools - integrated via WDSF - such that after successful analysis, an approved program can be executed.
RQ-04	The peer-to-peer nature of blockchain provides Crash-Fault-Tolerance (CFT) and the consensus mechanism may provide Byzantine-Fault-Tolerance (BFT) depending on the selected blockchain technology.
RQ-05	All workflow actions and access to protected resources are logged in an append-only immutable database for accountability and traceability. In addition, HLF can restrict via endorsement policies which participants may see channel specific data.
RQ-06, RQ-01, RQ-02	WDSF's modelling capabilities allow owners to model their business process as workflows and restrict users based on history-based access control (e.g., separation of duties, access to object B only after completing task A or access object A) in a distributed environment. Granted access is usually short-lived (timed access) and users' execution rights for workflow/business processes can be revoked any time by the owner of the workflows.

wrong operations, free-choice violations, and any dead transitions and non-live transitions. Table III presents how the requirements identified in Tab. II are satisfied by integrating WDSF framework with a workflow designed particularly to solve the remote-maintenance use case. Note: introducing WDSF does not protect against all security threats e.g., privilege escalation on the underlying operating system (OS) of protected resources cannot be secured by WDSF. Additional measures like hardening the OS are required. However, integrating WDSF imposes additional restrictions on users on how and on which conditions they could access a protected resource, thus reducing the overall threat landscape.

IV. Enforcing Secure Remote Maintenance via WDSF

The following sequence of steps occur in the modelled remote maintenance use case: 1. A piece of equipment or device (e.g., a Programmable Logic Controller (PLC)) is malfunctioning or requires maintenance. Thus, a shop floor engineer creates a maintenance request via the workflow execution app, the Approval1 transition - as shown in Fig. 3 - creates a maintenance order; 2. The request is processed by the Workflow Execution Application (WF-App) and then triggers the relevant smart contract method. The

request information includes meta-data such as equipment identifier, nature of the diagnosed problem. If confidential information is involved, then it is never stored in the blockchain but in a private database off-chain. Only the hash of the confidential info is stored on the blockchain as a proof. Thus, the metadata of the request is recorded on the blockchain and serves as a starting point of the audit log; 3. The maintenance company receives the order and a maintenance technician accepts it (via transition Approval2) using a separate workflow execution application; 4. Next, the maintenance request must be approved by the responsible shop floor manager and enterprise manager via his/her instance of the WF-App; 5. Once the approvals are completed, the system programmatically interacts with the Software Defined Network (SDN) / Software Defined Perimeter (SDP) enabled firewall to configure rules which enable access for the maintainer to reach the target network (see [15]). In addition, a timeout that triggers the closure of the firewall port can also be modelled within the workflow; 6. The maintainer is able to authenticate against the IAM which provides the appropriate OAuth tokens to access the desired and protected WDSF resource (i.e., the industrial edge device requiring maintenance). In addition to the approval process and opening and closing of ports, the maintenance process includes the technician performing some automated tests - which show that the part or equipment is fixed, and it is ready for production as shown in Fig. 3. The test results are automatically recorded and become part of the workflow events; 7. Finally, if the described sequence of workflow events concludes successfully, then the workflow ends.

A. WDSF - Architecture and Deployment

In Fig. 5, a blueprint deployment architecture of WDSF is presented which includes different layers protected by SDN/SDP firewalls, IAM integrated with the workflow authorization server (AS), edge devices and industrial PLCs. The specified workflow is registered with the workflow authorization server and the permissions required to execute each workflow step are configured by the workflow managers. The arrows indicate the first interaction between different components and later mutual authentication can be established. At least three organizations participate in this use case scenario: a) the plant owner; b) the maintenance company; and c) in case of a dispute, a compliance monitoring company is involved. Dedicated instances of a workflow execution application (WF-APP) are deployed for the individual organizations. WF-APP is designed to support multiple concurrent instances so that no central point of control is needed to support a fully decentralized setup. Our proposed architecture's complexity to run different micro-services can be reduced by using state-of-the art software testing and deployment patterns such as Continuous Integration (CI) / Continuous Deployment (CD).

The WF-Apps are configured following the standard OAuth client recommendations and specifications. In the presented deployment, WF-App instances 1 and 2 are trusted, therefore, they are configured and deployed as confidential clients. WF-App instance 3 is configured as a public client. The WDSF relies on a customized OAuth authorization server (Workflow Registration/Authorizations) to issue restricted OAuth authorization tokens (e.g., Java Web Tokens (JWT) or CBOR Web Tokens (CWT)). In addition, WF-App triggers the smart contracts that are deployed on a blockchain. The smart contracts validate whether a user invoking the method is allowed to perform the workflow step, validate the conditions, and record the workflow action's metadata in the immutable ledger. The WF-App may receive a push notification if there is a workflow relevant state change because of committed transaction. Finally, the blockchain infrastructure based on Hyperledger Fabric (HLF) provides non-repudiation and traceability to the workflow execution. WDSF uses restricted/scoped OAuth tokens to configure network devices and to access edge/industrial devices. If SDN/SDP enabled devices are used, then this can be achieved programmatically e.g., via Port Knocking mechanisms enabled via Single Packet Authorization (SPA) (see [16], [15]).

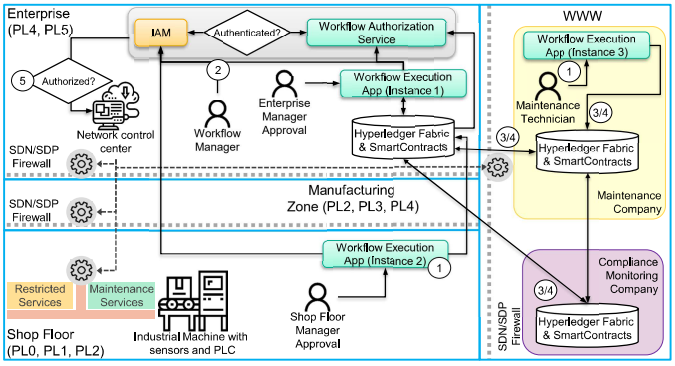

Fig. 5. WDSF Deployment Architecture

The SDP firewall protects the company network by preventing access or network scan from external IPs. The firewall is configured to accept SPA packets that allow only the authenticated users to establish the connection to necessary services based on pre-defined policies. The workflow execution application provides the necessary access credentials and therefore, the user can trigger Port Knocking via SPA packets. WDSF can be seen as a configuration management system with history-based access control (i.e., in this case proper approvals outside the networking environment are required to reach this step). Therefore, WDSF enforces the specified remote maintenance workflow and helps to integrate with industry components to programmatically reconfigure the network access rules in the SDP firewall, providing automatic access control updates to the firewall removing the need

for manual policy updates. Similarly, the industrial edge devices are configured to accept OAuth tokens issued by the plant's authorization server (AS). An AS can issue OAuth tokens with restricted scopes to enforce the least privilege principle in remote maintenance scenarios. The main reason to choose OAuth is for its interoperable, extendable and fine-grained scopes, and flexible security features such as grant types.

The use of the permissioned blockchain such as HLF provides confidentiality features (channels and private-data-collections (PDC)), pluggable consensus algorithm (e.g., Raft) and capability of writing and updating smart contracts via generic programming languages. Thus, we can achieve good performance in terms of transactions per second (TPS) and suitable latency for most of the generic cross-organizational industrial use cases.

B. WDSF Prototype

The WDSF prototype consists of micro-services including the workflow authorization service, Hyperledger Fabric network and smart contracts as represented in Fig. 5. WDSF micro-services were developed and practically tested as part of this ongoing research work. This section explains the core features/aspects of the WDSF. In particular, it focuses on how users are able to interact with the Petri Nets-based workflow execution application (WF-App), and how other components such as the smart contracts and SDN firewall / IIoT can be integrated with the WF-App.

The WF-App can be deployed as a smartphone application e.g., Android-App in distributed scenarios or as a standard Web-Application. The users must authenticate and get proper authorization tokens from the authorization service before interacting with the workflow. WF-App supports executing multiple different workflows at the same time.

In Fig. 6, a screenshot of WF-App's prototype web-interface where an excerpt of the Remote Maintenance workflow is shown. The graphical interface is not tailor made for particular use case but supports any generic workflow use cases showcasing features of WDSF. The transitions section includes different Petri Nets transitions modelled in the workflow. The tokens section includes different Petri Nets places holding one or more tokens as part of the workflow in addition it offers users to input tokens. Petri Nets section presents the dynamic view of workflow execution in terms of Petri Nets view, this allows to understand the state of the workflow. The Approval1 transaction (i.e., Maintenance Request approval from shop floor) is enabled because the shop floor user has entered valid tokens (i.e., issue token and user's userShopFloorTechnicianSignature token). Once a transition is enabled, it can be fired by an authorized user. Notice that the arc going out of the Approval1 transition invokes SCAPI (smart contract API) method exposed by the smart contracts deployed in HLF - which takes two input arguments (issue, userShopFloorTechnicianSignature). The result of the smart contract execution is placed in the MaintenanceOrder output place. At the end of this transaction, two actions will occur: a) validation of the shop floor technician's signature by the Petri Nets execution layer; b) the transaction metadata of Approval1 is recorded in an immutable HLF blockchain via the SCAPI.

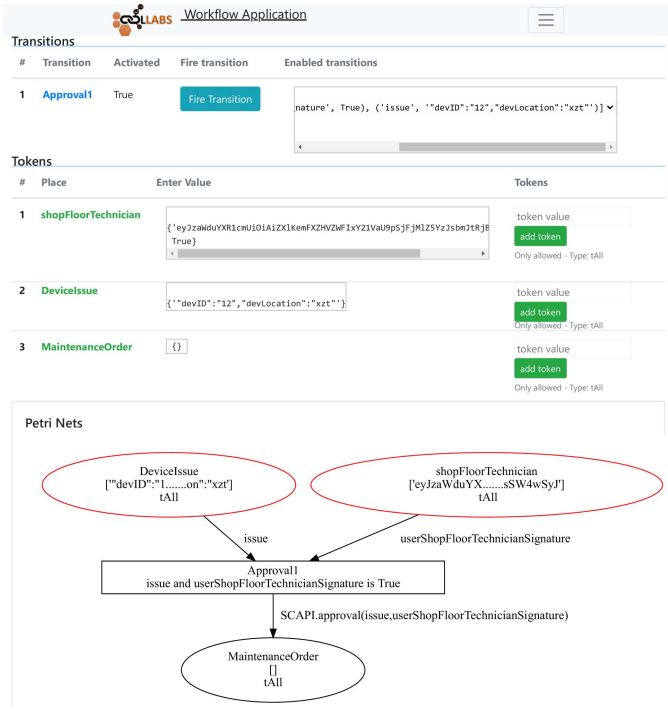

Fig. 6. WF-App prototype showing an enabled transition "Approval1" with matching pairs of input tokens.

Though in this paper we present only an excerpt, the remote maintenance use case is modelled completely and implemented i.e., including the IIoT device interaction, an IIoT API exposing the configurations options of the SDN firewall can be invoked to perform port opening and closing operations. The user authorization includes primarily two parts. The first part includes user authentication against the IAM, validation of workflow authorization and finally receiving the oAuth tokens to interact with IIoT components. The second part includes HLF user authentication and presenting appropriate credentials to interact with smart contracts. The tokens and credentials are short lived, valid for specific resources thus enforcing least privilege principle. The tokens are granted just-in-time and in a sequence i.e., only after completing step1 of a workflow the token required to complete step2 is granted, thus enforcing workflow-aware or history-based access control to resources.

C. Performance Evaluation

Blockchain as an immutable ledger may not be suitable for all applications, for instance bitcoin suffer from trans-

SC Methods	Avg TPS (Min / Max)	Avg LAT (Min / Max)
Writes	39.73 (35.9 / 44.1)	0.6s (0.27s / 2.88s)
Reads	111.72 (107.3 / 117.7)	0.17s (0.1s / 0.54s)

TABLE IV
Performance Evaluation results of HLF showing that it is suitable for the proposed remote maintenance use case

action latencies of 10 minutes or longer (see [17]) and this is definitely not suitable for many industrial applications. To make sure that the use of Hyperledger Fabric (HLF) blockchain is suitable for the remote maintenance use case, we evaluated the performance of WDSF including the HLF and present our results in this section. A study from [18] concluded that the maximum waiting times that users are willing to tolerate for web applications is about two seconds. We therefore aim for a latency of less than 2 seconds for the remote maintenance use case. Benchmark Setup: The experiments were conducted on HLF long-term stable (LTS) version 2.2 with the help of Hyperledger Caliper. In order to evaluate real world latency performance we hosted two nodes in different geographical locations i.e., one in Ohio (us-east AWS) and one in Frankfurt (eu-central AWS). Hardware configuration of each node is as follows: 4 GiB of RAM, 2x CPUs with max 3.3 GHz, and 25 GiB of storage that ran on Ubuntu 20.04. The test setup included 3 peer organizations i.e., maintenance company, manufacturing company, a compliance monitoring company and 1 ordering organization. Importantly, each peer organization ran one peer each, the ordering organization used one ordering service node (OSN) which used the default Raft implementation and LevelDB as the underlying database was used. Furthermore, the default configuration was used for the ordering service which means that after 2 seconds or after 10 new transactions a new block is published. Methodology: There were 3 write, 2 read and 2 delete functions written as part of the remote maintenance smart contract (SC). In addition, we integrated access control that validates which organization can invoke those write/read/delete functions. Each read/write function of the SC was invoked 1000 times at a fixed rate of 50 TPS and the benchmark was repeated 5 times.

Results and Conclusion: The average (Avg), minimum (Min), and maximum (Max) values were considered from the 5 different simulations for results. All transactions were accepted i.e., 1000 success / 0 failure. Table IV shows the results of write methods and reads in terms of Transactions per Second (TPS) and latency (LAT). In brief, several invocations of the smart contract are needed for creating, approving and completing an order. The read queries are fast because they are not submitted to the ordering service, whereas write queries are comparatively slow because they are submitted and recorded in the blockchain after a Raft consensus mechanism. Even after considering the pessimistic scenario of write transaction rate of 35.9 TPS and an optimistic average latency of 0.69s, we can have more than sufficient remote maintenance orders/completion with the proposed technology. Note: in some rare cases the max latency of 2.8s was recorded which is not below the acceptable 2 sec according to previous study, but considering the average latency of 0.69s for write calls we consider this as acceptable deviation.

V. Related Work

In this Sec. we present the related work concerning technology used by our approach a) Petri Nets for modelling and enforcing workflow; b) blockchain for immutability and accountability; c) finally, commercially available Privileged Access Management (PAM). In terms of modelling, validating and verifying workflow the most prominent approaches are the following: Petri Nets, automata, process algebra, business process modelling notation (BPMN). The advantages and the reasons for adopting Petri Nets are presented in [6], therefore we adoped to use Petri Nets. A review of the application of blockchain in next generation cybersecurity applications in the context of Industry 4.0 is presented in [19], and this review article shows that blockchain can enhance industrial technologies by adding decentralized security, trust, immutability, with a higher degree of automation through smart contracts. A generic blockchain enabled cyber-physical-system (CPS) architecture for Industry 4.0 manufacturing systems is presented by Lee et al., in [20]. Secure industrial remote maintenance by using software defined networking (SDN) and attributed-based access control (ABAC) is presented in [21]. Several commercial Privileged Access Management (PAM) solutions exist that enable secure remote access, and a report on their critical capabilities is presented in [22]. Most of these solutions consider centralized trust & control and focus on security aspects such as multi-factor authentication, just-in-time (JIT) access and zero standing privileges (ZSP) realizing least privilege principle. However, in this research paper we focus on having distributed trust while the resource providers still have complete control over their resources while enforcing, compliance, traceability, and auditability of business processes/workflows.

So far, no existing work or commercially available PAM solutions focuses on secure remote maintenance by applying distributed trust without having a centralized/trusted third party.

VI. Conclusion and Future Work

This paper first identified, analysed, and presented the security protection requirements of an industrial remote maintenance use case. Next, this paper presented how those protection requirements are addressed by using a generic workflow-driven security framework (WDSF) to enforce secure remote maintenance use case in an industrial scenario. The WDSF is combined with state-of-the-art technology such as SDN/SDP firewalls, blockchain and smart contracts to satisfy the requirements, to mitigate

the threats, and to enforce workflows/processes defined by the owner. This work also proved that the proposed WDSF framework - in terms of blockchain performance parameters such as transactions per second (TPS) and latency - is well suited or sufficient for the remote maintenance use case, as well as, applicable to any general industrial workflow scenarios.

This research work presented a blueprint architecture for deploying WDSF in a generic Industry 4.0 factory environment and showed how WDSF can be practically applied in collaborative industrial scenarios to achieve workflow integrity, transparency, and compliance. Thus, bringing WDSF research one step closer to production. The proposed approach is better in comparison to centralized approaches because of the shared responsibility approach and distributed trust. Hence, the stakeholders do not need to rely on a trusted third party (TTP) to achieve the following in a collaborative and distributive environment: to enforce their business processes (modelled and executed via Petri Nets and smart contracts), to protect their resources with least privilege, and to store tamper-proof audit logs for auditability and compliance purposes. In addition, Petri Nets based workflow modeling and execution enhances security by formalizing and verifying the workflow specification and execution. The remote maintenance workflow was designed, analysed and validated to satisfy workflow properties and to reduce errors using the WoPeD Petri Nets tool. Smart contracts were derived from the Petri Nets are deployed in the blockchain, which record workflow events in an immutable way, thus providing accountability and traceability in a distributed blockchain. Our observations point out that setting-up of WDSF can take more time than conventional centralized services setup. However, most of the integration and deployment can be automated with the help of CI/CD tools.

So far we studied the feasibility of applying WDSF in industrial scenario, as next steps we envision to provide end-to-end security and integrity of data coming from the edge industrial device to the WDSF platform. We plan to integrate shop floor machines that has hardware security elements such as trusted execution environment (TEE) and trusted platform module (TPM) with WDSF. In addition, to the guarantees provided by Petri Nets workflow validation, we plan to use formal verification tools to prove that the defined security properties hold in the modelled workflow. Such that from a security perspective, verification of a security property in a given system specification (e.g., access is granted only after proper approvals) is guaranteed.

Acknowledgements

This research has been funded by the European Union's Horizon 2020 Research and Innovation program under grant agreements No. 871518 and No. 830929.

References

[1] OWASP Foundation, "Owasp top ten," 2020.
[2] I. Stellios, P. Kotzanikolaou, M. Psarakis, C. Alcaraz, and J. Lopez, "A survey of iot-enabled cyberattacks: Assessing attack paths to critical infrastructures and services," IEEE Communications Surveys Tutorials, vol. 20, no. 4, pp. 3453–3495, 2018.
[3] ENISA, "Industry 4.0 cybersecurity: Challenges and recommendations," tech. rep., European Union Agency For Network And Information Security (ENISA), 2019.
[4] P. Kasinathan and J. Cuellar, "Securing the integrity of workflows in iot," in Proceedings of the 2018 International Conference on Embedded Wireless Systems and Networks, EWSN 2018. Madrid, Spain, February 14-16, 2018, pp. 252–257, 2018.
[5] P. Kasinathan and J. Cuellar, "Workflow-aware security of integrated mobility services," in ESORICS 2018, Barcelona,Spain, pp. 3–19, 2018.
[6] P. Kasinathan and J. Cuellar, "Securing emergent iot applications," in Engineering Trustworthy Software Systems: 4th International School, SETSS 2018, Chongqing, China, April 7–12, 2018, Tutorial Lectures, (Cham), pp. 99–147, Springer International Publishing, 2019.
[7] T. Freytag and M. Sänger, "Woped - an educational tool for workflow nets," in Proceedings of the BPM Demo Sessions 2014 Co-located with the 12th International Conference on Business Process Management (BPM 2014) (L. Limonad and B. Weber, eds.), vol. 1295 of CEUR Workshop Proceedings, p. 31, CEUR-WS.org, 2014.
[8] N. Zupan, P. Kasinathan, J. Cuellar, and M. Sauer, Secure Smart Contract Generation Based on Petri Nets, pp. 73–98. Singapore: Springer Singapore, 2020.
[9] COLLABS Consortium, "Deliverable 1.3 - system architecture definition," 2020.
[10] T. J. Williams, "The purdue enterprise reference architecture," Computers in industry, vol. 24, no. 2-3, pp. 141–158, 1994.
[11] Microsoft Corporation, "Stride chart," 2007.
[12] Microsoft Corporation, "Microsoft threat modeling," 2019.
[13] The Official OMG SysML site, "What Is OMG SysML?," 2020.
[14] W. M. P. van der Aalst, K. M. van Hee, A. H. M. ter Hofstede, N. Sidorova, H. M. W. Verbeek, M. Voorhoeve, and M. T. Wynn, "Soundness of workflow nets: classification, decidability, and analysis," Formal Aspects of Computing, vol. 23, pp. 333–363, May 2011.
[15] A. Sallam, A. Refaey, and A. Shami, "On the security of SDN: A completed secure and scalable framework using the software-defined perimeter," IEEE Access, 2019.
[16] W. Han, H. Hu, Z. Zhao, A. Doupé, G.-J. Ahn, K.-C. Wang, and J. Deng, "State-aware network access management for software-defined networks," in Proceedings of the 21st ACM on Symposium on Access Control Models and Technologies, SACMAT '16, p. 1–11, 2016.
[17] K. Croman, C. Decker, I. Eyal, A. E. Gencer, A. Juels, A. E. Kosba, A. Miller, P. Saxena, E. Shi, E. G. Sirer, D. Song, and R. Wattenhofer, "On scaling decentralized blockchains - (A position paper)," in Financial Cryptography and Data Security - FC 2016, vol. 9604 of Lecture Notes in Computer Science, pp. 106–125, Springer, 2016.
[18] F. F. Nah, "A study on tolerable waiting time: how long are web users willing to wait?," Behav. Inf. Technol., vol. 23, no. 3, pp. 153–163, 2004.
[19] T. M. Fernández-Caramés and P. Fraga-Lamas, "A review on the application of blockchain to the next generation of cybersecure industry 4.0 smart factories," IEEE Access, vol. 7, pp. 45201–45218, 2019.
[20] J. Lee, M. Azamfar, and J. Singh, "A blockchain enabled cyber-physical system architecture for industry 4.0 manufacturing systems," Manufacturing Letters, vol. 20, pp. 34 – 39, 2019.
[21] A. Kern and R. Anderl, "Securing industrial remote maintenance sessions using software-defined networking," in 6th International Conference on Software Defined Systems, SDS 2019, Rome, Italy, June 10-13, 2019, pp. 72–79, IEEE, 2019.
[22] M. Kelley, F. Gaehtgens, and A. Data, "Critical capabilities for privileged access management," Gartner Reports, 2020.