future internet

MDPI

Article

Toward Secure Software-Defined Industrial Networks Through Asset Administration Shell Digital Twins

Riccardo Bacca [1], Andrea Melis [2], Lorenzo Rinieri [2,*], Roberto Girau [2], Marco Prandini [2] and Franco Callegati [2]

[1] Centro Interdipartimentale di Ricerca Industriale—CIRI ICT, Alma Mater Studiorum—Università di Bologna, 40126 Bologna, Italy; riccardo.bacca@unibo.it

[2] Department of Computer Science and Engineering (DISI), Alma Mater Studiorum—Università di Bologna, 40126 Bologna, Italy; a.melis@unibo.it (A.M.); roberto.girau@unibo.it (R.G.); marco.prandini@unibo.it (M.P.); franco.callegati@unibo.it (F.C.)

* Correspondence: lorenzo.rinieri@unibo.it

Abstract

Industrial digitalization is moving from Industry 4.0 toward Industry 5.0's emphasis on resilience, human-centric operation, and sustainability. This shift is enabled by the convergence of Operational Technology and Information Technology, but this integration also broadens the exposure of industrial infrastructures to cyber threats targeting communication integrity and process continuity. Mitigating these risks requires network control that is both programmable and aware of each asset's operational context. However, there is still a lack of operational interfaces that translate the semantics of industrial assets into programmable, runtime-enforceable network behavior. In this paper, following a Design Science Research methodology, we introduce an asset-aware, closed-loop network control abstraction in which the industrial network itself is modeled as a managed asset through Asset Administration Shells. Asset state, lifecycle phase, and operational intent are translated into network policies enforced at runtime on programmable data planes, while in-network telemetry is exposed at the asset level and correlated with operational metrics. We validate the abstraction on a hybrid testbed that combines virtualized components with industrial-grade hardware and virtualized 5G connectivity, through three security-oriented use cases: (i) asset-driven customization of forwarding policies; (ii) human-centric secure maintenance with controlled remote access over 5G; and (iii) anomaly detection and isolation based on cross-layer telemetry correlation. The results show that asset-level operations can drive programmable network enforcement and make network telemetry available at the asset layer. Finally, the work outlines a first step toward standardizing network-oriented asset submodels by separating control-plane operations from data-plane state and telemetry.

Keywords: industrial security; P4; software-defined networking; asset administration shell; digital twin; Industry 4.0; Industry 5.0; design science research

Academic Editors: Filipe Pereira and Paulo Leitao

Received: 13 May 2026
Revised: 22 June 2026
Accepted: 23 June 2026
Published: 30 June 2026

1. Introduction

Operational Technology (OT) networks interconnect industrial control systems, machines, sensors, and actuators, whereas Information Technology (IT) systems provide enterprise services, data analytics, supervision, and remote access capabilities [1,2]. Historically, OT networks were organized as isolated or weakly interconnected production cells. This organization reduced cyber exposure, but it also limited unified asset visibility, dynamic network management, and semantic interoperability across industrial systems.

Industry 4.0 has changed this model by requiring continuous data exchange within OT environments and across the OT/IT boundary, enabling predictive maintenance, flexible production, and data-driven decision-making [3].

This convergence is essential, but it introduces two distinct challenges. The first concerns OT network management: industrial networks still suffer from limited unified visibility over assets, proprietary or non-standardized endpoint discovery, static configuration practices, and poor telemetry integration. These limitations make it difficult to adapt network behavior to the actual state and requirements of the production environment. The second concerns IT/OT interconnection: as OT systems become reachable from IT services, edge platforms, and cloud services, they are increasingly exposed to cyber threats, unauthorized access, and disruptions that can compromise production continuity [4]. Addressing these challenges requires network-control mechanisms that are both programmable and aware of the operational context of industrial assets.

A promising basis for asset awareness is the Asset Administration Shell (AAS), introduced in the Reference Architecture Model for Industrie 4.0 (RAMI 4.0) as the digital representation of an asset that becomes an Industrie 4.0 component [5]. The AAS exposes the data, functions, and relationships that characterize an asset and its interactions with other assets in a system or process [6,7]. In parallel, Software-Defined Networking (SDN) and programmable data planes provide runtime control over forwarding behavior, enabling network policies to be installed, modified, and monitored through software interfaces [8,9].

Despite these complementary capabilities, a gap remains between industrial asset semantics and programmable network enforcement. Existing AAS-based approaches generally represent machines, services, or lifecycle information, but they do not usually model the communication network itself as an AAS-managed industrial asset. Conversely, SDN and P4 provide powerful enforcement mechanisms, but their policies are typically expressed through network-level identifiers such as addresses, ports, tables, and forwarding rules, rather than through standardized industrial asset semantics. As a result, there is no commonly adopted AAS model for programmable network control that separates control-plane operations from data-plane state and telemetry.

The research question addressed in this paper is therefore: how can standardized asset semantics be coupled with programmable networking to support closed-loop, asset-aware security control in converged OT/IT environments? We address this question using a Design Science Research methodology, since the contribution is an engineered artifact that is designed, implemented, and evaluated in a representative technical environment [10,11].

The artifact presented in this work is an AAS-driven network control abstraction in which the OT network is represented as an industrial asset with dedicated AASs for control-plane and data-plane functions. AAS operations are translated into SDN/P4 enforcement actions, while network telemetry is exposed back through the AAS layer and correlated with asset-side operational metrics. This creates a bidirectional coupling between industrial asset semantics and programmable network behavior.

The contribution is positioned as an Industry 4.5 step: it validates the semantic and programmable mechanisms required by Industry 4.0 while providing a technical basis for selected Industry 5.0 requirements, including resilience, human-centric operation, and sustainability-aware network control. The paper makes four specific contributions:

1. It defines a network AAS abstraction that binds asset-level information to programmable network policies;
2. It implements the abstraction using Eclipse BaSyx, P4Runtime, Kathará, and virtualized 5G components;
3. It validates the framework through three security-oriented use cases, including hardware-in-the-loop validation with an industrial Revolution Pi device;

4. It identifies a first step toward standardizing network-oriented AAS submodels by separating control-plane operations from data-plane state and telemetry.

The remainder of this paper is organized as follows. Section 2 reviews the related work and highlights the research gaps addressed by the study. Section 3 presents the problem statement, methodology, and threat model. Section 4 describes the architecture. Section 5 presents the testbed, and Section 6 reports the validation use cases and results. Section 7 discusses limitations, maturity, Industry 5.0 implications, and standardization opportunities. Section 8 concludes the paper.

2. Related Work

The related work is organized around the functional chain required by the proposed artifact. First, industrial assets must be represented through a standardized semantic interface. Second, the network must be programmable at runtime. Third, monitoring information must be correlated across asset and network layers. Fourth, remote maintenance and 5G connectivity introduce new access-control requirements. Finally, Industry 5.0 extends the evaluation criteria from automation alone toward resilience, human-centricity, and sustainability. This structure allows us to identify the specific gap addressed in this paper: the lack of an operational binding between AAS-based asset semantics and programmable network enforcement.

In the Industry 4.0 stack, the Asset Administration Shell is the standardized digital counterpart of a physical or logical asset, exposing semantically structured information and services across the asset lifecycle [12,13]. AAS research has addressed interoperability, lifecycle data, safety, and security descriptors [14,15], and the AAS structure has been standardized in IEC 63278-1 [16]. These works establish the AAS as an industrial digital-twin interface, but they do not generally use AAS operations as runtime inputs for network policy enforcement.

Programmable networking provides the enforcement substrate needed to act on such asset-level information. SDN separates control logic from forwarding behavior, while P4 enables programmable data-plane functions such as filtering, counters, telemetry, and protocol-independent packet processing [8,9,17,18]. These capabilities have been explored in IIoT and industrial-network contexts [19,20]. However, most programmable-network approaches remain network-centric: policies are expressed in terms of flows, addresses, ports, or slices, rather than through standardized industrial asset semantics.

Industrial cybersecurity research has also investigated digital-twin-based anomaly detection, situational awareness, and cross-layer monitoring. Examples include digital-twin intrusion detection in industrial control systems [21,22], wide-area situational awareness for critical infrastructures [23], and bi-anomaly detection for Industry 4.0 environments [24]. Machine-learning approaches, including reconstruction-based anomaly detection [25] and malware classification [26], can improve detection coverage. Nevertheless, these approaches often remain monitoring-oriented: they detect or classify abnormal behavior, but they do not necessarily close the loop by translating asset-level observations into programmable network actions.

5G connectivity further increases the relevance of asset-aware network control. Industrial 5G supports remote maintenance, quality-of-service management, and network slicing [27,28]. Related work on semantic slicing, edge orchestration, and multi-layer resource management shows how application requirements can guide network and compute-resource allocation [29–33]. These directions are aligned with application-aware and deterministic networking efforts such as IETF APN and DetNet [34–36]. However, the intent is typically represented through network-native or application-native abstractions, not through AAS-based industrial asset models.

Industry 5.0 broadens the design criteria for industrial digitalization. Networked industrial systems should not only support automation and connectivity, but also resilience, human-centric operation, and sustainability. Human digital twin and human-centered production studies show how digital representations can include operators, work contexts, and assisted decision-making [37,38]. Sustainability-oriented digital-twin work and ITU-T recommendations on network energy and carbon assessment provide a foundation for carbon-aware digital infrastructures [39–42]. These directions motivate network-control mechanisms in which asset state, operator intent, and operational constraints can influence connectivity decisions.

Table 1 summarizes the main areas of related work and the gap addressed by the proposed architecture.

Table 1. Related-work areas and identified gaps.

Related-Work Area	Current Limitation	Role of This Work
AAS interoperability	AASs standardize asset semantics, but rarely expose network-control operations.	Models network control and data planes as AAS-managed assets.
Programmable networking	SDN/P4 enables runtime enforcement, but remains mostly network-centric.	Translates AAS-level operations into SDN/P4 actions.
Industrial security monitoring	Digital-twin security often stops at monitoring or alerting.	Couples AAS telemetry with in-network counters for mitigation.
5G-enabled maintenance	Remote maintenance increases OT exposure and requires coordinated access control.	Coordinates 5G access, OT isolation, and rollback through AAS operations.
Industry 5.0 operation	Resilience, human-centricity, and sustainability require network decisions to consider operational continuity, operator control, and environmental constraints.	Shows how AAS-driven network control can support Industry 5.0 pillars.

The closest related approaches can be compared along five dimensions: whether they rely on standardized AAS semantics, whether they support SDN/P4-based enforcement, whether monitoring and enforcement form a closed loop, whether 5G connectivity is part of the architecture, and whether validation includes industrial hardware-in-the-loop. Table 2 reports this comparison.

Table 2. Positioning of this work with respect to closely related approaches. ● indicates full support; ◐ indicates partial support or conceptual alignment without full end-to-end implementation; ○ indicates absence of the capability.

Approach	AAS-Driven	SDN/P4 Enforcement	Closed Loop	5G Integration	HIL Validation
Digital-twin IDS for ICS [21]	○	○	◐	○	○
Digital-twin-based ICS security monitoring [22]	○	○	◐	○	○
AAS security descriptors [15]	●	○	○	○	○
AAS-based IAM and access control [43]	●	○	○	○	○
P4-based IIoT enforcement [17]	○	●	◐	○	○
Programmable networking for ICS [19]	○	●	◐	◐	○
Zero-trust architecture for IIoT [44]	○	◐	○	○	○
Semantic 5G slicing [29]	○	◐	◐	●	○
Edge resource marketplace [31]	○	◐	◐	●	○
This work	●	●	●	●	●

The resulting gap is the absence of an end-to-end mechanism that binds standardized industrial asset semantics to programmable network behavior. Existing AAS work provides asset representation, programmable-network work provides enforcement mechanisms, and industrial-security work provides monitoring and detection methods. The contribution of this paper lies in coupling these layers operationally: AAS operations become triggers for SDN/P4 enforcement, while in-network counters are exposed back as AAS properties and correlated with asset-level telemetry. This bidirectional coupling is the distinguishing element of the proposed architecture.

3. Problem Statement

Modern OT environments are increasingly connected to IT services, edge platforms, and remote maintenance infrastructures. This connectivity improves visibility and operational flexibility, but it also exposes industrial assets that often rely on legacy protocols, static segmentation, and limited endpoint protection. In this setting, network security cannot be based only on host-level mechanisms or manually configured forwarding rules. The central problem addressed in this paper is the lack of an operational interface able to translate industrial asset semantics into programmable network behavior.

Existing AAS-based approaches provide a standardized representation of industrial assets, including their properties, services, and lifecycle information. However, they do not generally expose the communication network itself as an AAS-managed asset, nor do they define reusable mechanisms through which asset state, maintenance intent, or telemetry can directly influence network enforcement. Conversely, SDN and programmable data planes provide runtime control over forwarding behavior, but their policies are usually expressed through network-level abstractions such as addresses, ports, tables, and rules. The research problem, therefore, lies at the boundary between these two layers: how to bind standardized asset semantics to programmable network enforcement in a way that remains compatible with industrial interoperability requirements.

The design objective of this work is to make the OT network both programmable and asset-aware. Programmability is provided by SDN/P4 mechanisms capable of installing, removing, and updating forwarding rules at runtime. Asset awareness is provided by AASs, which expose the operational context required to decide when and how the network should be adapted. The proposed artifact is an AAS-driven network control abstraction in which network control-plane and data-plane functions are represented through dedicated AASs, while AAS operations are translated by a Network Manager into concrete SDN/P4 actions.

The methodological framing is Design Science Research (DSR), since the contribution is an engineered artifact that is designed, implemented, and evaluated in a representative technical environment [10,11]. Following the DSR process proposed by Peffers et al. [11], the study is structured as follows:

1. Problem identification: the absence of a semantic interface between AAS-based asset representations and programmable network enforcement.
2. Objective definition: the design of a network-control abstraction able to coordinate asset state, communication requirements, and security policies through AAS operations.
3. Design and development: the implementation of network-related AASs, operation handlers, and SDN/P4 integration mechanisms using existing open-source components.
4. Demonstration: the execution of representative use cases covering runtime traffic control, 5G-assisted maintenance, and cross-layer anomaly mitigation.
5. Evaluation: the assessment of the artifact on a hybrid virtualized and hardware-in-the-loop testbed, including latency measurements and functional validation of the control loop.

6. Communication: the presentation of the artifact, its validation results, and its limitations in this manuscript.

DSR is appropriate for this study because the phenomenon under investigation is created by the artifact itself: AAS-driven network enforcement does not exist as an observed organizational process, but as a designed mechanism that must be instantiated and evaluated. Alternative methodological frames would be less suitable. Action research would require iterative intervention in a specific industrial organization, while a case-study methodology would primarily explain an existing real-world setting. In contrast, DSR directly supports the design–build–evaluate cycle required to validate the proposed AAS-to-network control abstraction.

The resulting artifact addresses the problem by introducing network-related AASs that operate alongside the AASs of OT assets. These AASs expose network-control operations and telemetry properties through standardized asset-oriented interfaces. The Network Manager interprets these operations, translates them into SDN/P4 policies, and installs the corresponding rules in the programmable data plane. In the opposite direction, network measurements such as counters and forwarding-state information are exposed back through the AAS layer and can be correlated with asset-side metrics.

The novelty is therefore not the isolated use of AAS, SDN, P4, or 5G, but their operational coupling. Asset state, maintenance intent, and cross-layer telemetry become inputs to network-control decisions, while network behavior becomes observable and controllable through the same semantic layer used to represent industrial assets. This coupling provides the basis for the threat model and validation use cases developed in the following sections.

3.1. System & Threat Model

We describe the system and threat model considered in this work through the layered ICS structure of the Purdue Enterprise Reference Architecture [45], as shown in Figure 1. This model organizes industrial networks into hierarchical levels, ranging from physical processes (Level 0) to enterprise systems (Levels 4 and 5). We focus on the OT layers, including sensors and actuators (Level 0), basic control systems such as Programmable Logic Controllers (Level 1), supervisory systems such as SCADA and HMIs (Level 2), and site operations (Level 3).

In our system model, we assume that the AAS instances used for monitoring are securely instantiated with integrity protections in the IT network (Level 4/5). To bridge the gap with the OT world, each AAS maintains a dedicated, secure channel to its specific physical asset, enabling monitoring while remaining logically separated from the industrial control loop. This assumption is intentionally explicit. The prototype evaluates whether trusted AAS and controller components can translate asset semantics into network enforcement; it does not claim that the AAS infrastructure is intrinsically non-compromisable. In a production deployment, the AAS servers, registry, SDN controllers, and policy adapters must themselves be protected with authentication, authorization, logging, network segmentation, and integrity monitoring consistent with zero-trust principles [44].

3.2. Adversary Model

With respect to the attacker model, we consider an internal or adjacent adversary who has obtained access to the OT network and possesses partial Dolev–Yao [46] capabilities. A typical Dolev–Yao attacker has full control over the communication medium, allowing them to access, modify, inject, delete, or delay any message passing through the network. While this adversary possesses almost unlimited network capabilities, he is inherently constrained by the perfect cryptography assumption. This implies that the attacker cannot break cryptographic primitives and can only decrypt a ciphertext or forge a signature if

he possesses the correct keys. Consequently, computational attacks such as brute-force private-key attacks or dictionary attacks cannot be performed.

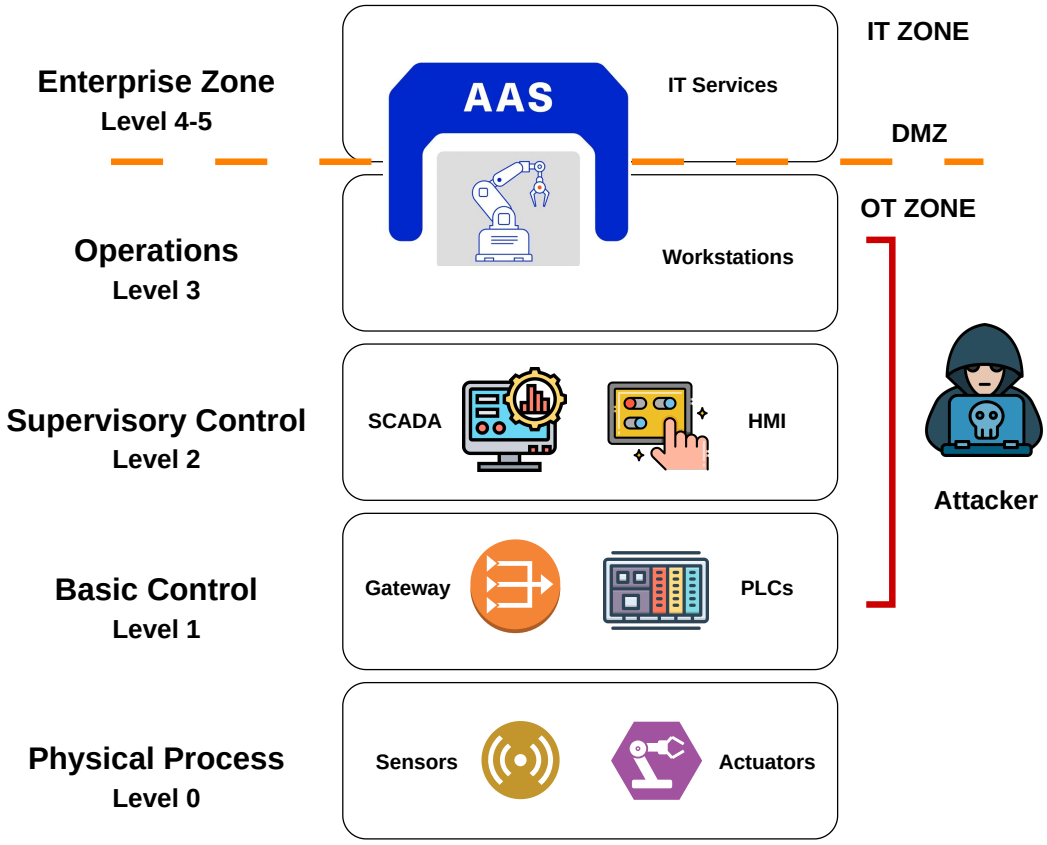

Figure 1. Threat model context based on the Purdue Reference Architecture. The attacker operates only within Levels 1–3. The Asset Administration Shell is trusted outside the attack surface.

Additionally, we introduce two specific constraints relevant to our ICS scenario:

- Aligning with previous works in ICS security [47–50], we assume that the attacker is able to operate only at or above Level 1 of the Purdue Enterprise Reference Architecture.
- We assume that the communication channel between the AAS and its physical asset is secured [15,43]. Since the adversary is inherently bound by the perfect cryptography assumption, he cannot compromise the encryption or integrity of this channel and therefore cannot tamper with the AAS monitoring data stream.

Finally, given the adversary's nature as a network-based Dolev–Yao intruder constrained by perfect cryptography and operating strictly within the OT network boundaries (Levels 1–3), he is inherently prevented from performing the following actions:

- The adversary cannot modify the AAS internal logic, APIs, or data schemas. Since the AAS resides in the protected IT domain and the Dolev–Yao model restricts the attacker to manipulating network traffic rather than executing arbitrary code on remote endpoints, the AAS's internal state remains inaccessible. So, the AAS is not part of the attack surface.
- Due to the perfect cryptography assumption, the adversary cannot decrypt traffic flowing through the secure tunnel between the AAS and the asset, nor can he inject valid messages into this channel without the necessary session keys.
- Since the adversary operates strictly above Level 1 and lacks physical access to the facility, he cannot manually tamper with hardware (e.g., cutting wires) or directly flash malicious firmware onto field devices.

3.3. Attack Objectives

The attacker's primary goal is to manipulate or degrade process integrity by exploiting weak authentication, insecure protocols, or poor segmentation between industrial levels. We identify three main attack classes:

1. Traffic Tampering: Includes MITM attacks on industrial protocols (e.g., Modbus, MQTT) to alter control logic, telemetry, or state updates between PLCs and sensors.
2. Sensor Data Spoofing: Involves falsifying telemetry readings (e.g., spoofed temperature, vibration) to mislead control systems or bypass safety thresholds.
3. Actuator Command Injection: Involves the unauthorized issuance of actuation commands (e.g., open valve, stop conveyor), leading to unsafe or unintended process changes.

Figure 2 shows an abstract attack tree modeling how an attacker at Level 1–3 might affect system integrity. The root node represents the goal (disruption), and leaf nodes describe specific tactics.

Finally, Table 3 summarizes the attacker's potential capabilities at each level, and whether the AAS can observe such activity given its architectural placement.

Table 3. Adversary capabilities by ICS level and AAS observability.

ICS Level	Attacker Capability	Observable by AAS
Level 3	Network traffic injection/replay	Yes (via network interface monitoring)
Level 2	SCADA/HMI traffic tampering	Partially (if logs or traffic mirrored to AAS)
Level 1	Sensor spoofing/ actuator injection	Indirectly (via inconsistent telemetry or command timing)
AAS/controller infrastructure	Trusted in this model	Security events only if assumption is relaxed

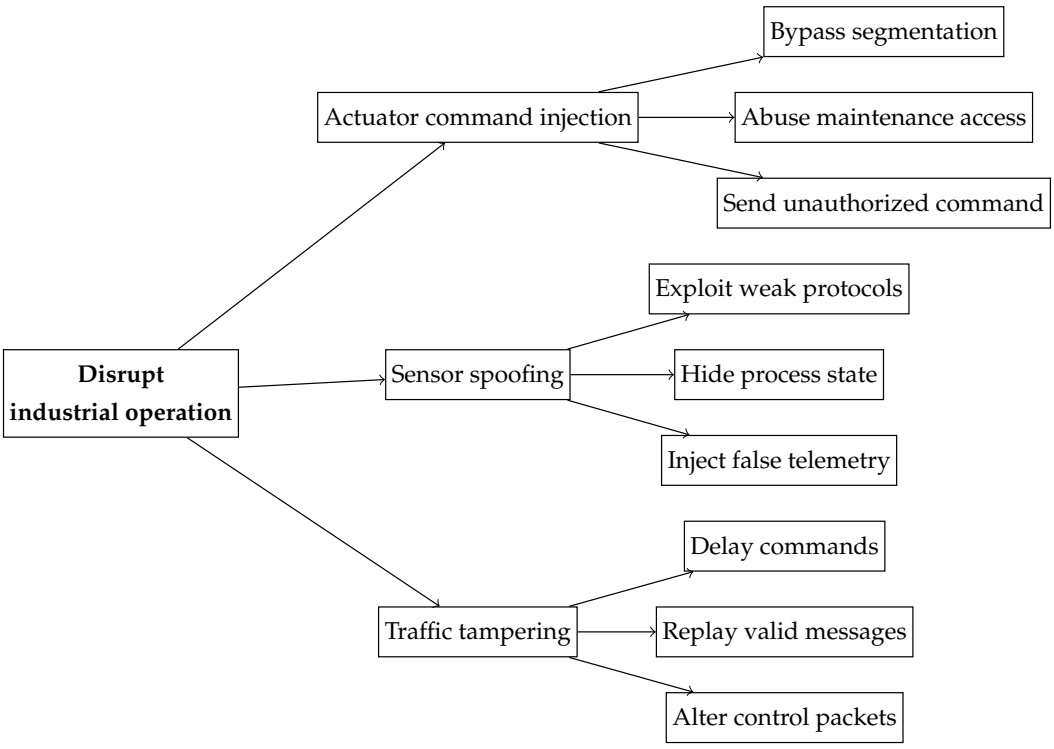

Figure 2. Attack tree for a Level 1–3 adversary attempting to disrupt industrial operation.

4. Architecture

This section presents the architecture that realizes the proposed asset-aware closed-loop network control abstraction for industrial environments. Rather than treating the communication infrastructure as a passive substrate, the architecture elevates the OT network to an industrial digital asset by equipping both its control and data planes with dedicated Asset Administration Shells. Through this design, asset semantics are mapped at runtime to concrete network enforcement actions implemented via programmable data planes. In parallel, in-network telemetry is continuously collected and exposed through the network AASs, where it is correlated with asset-level operational metrics. This bidirectional semantic coupling closes the control loop between assets and the network, as shown in Figure 3.

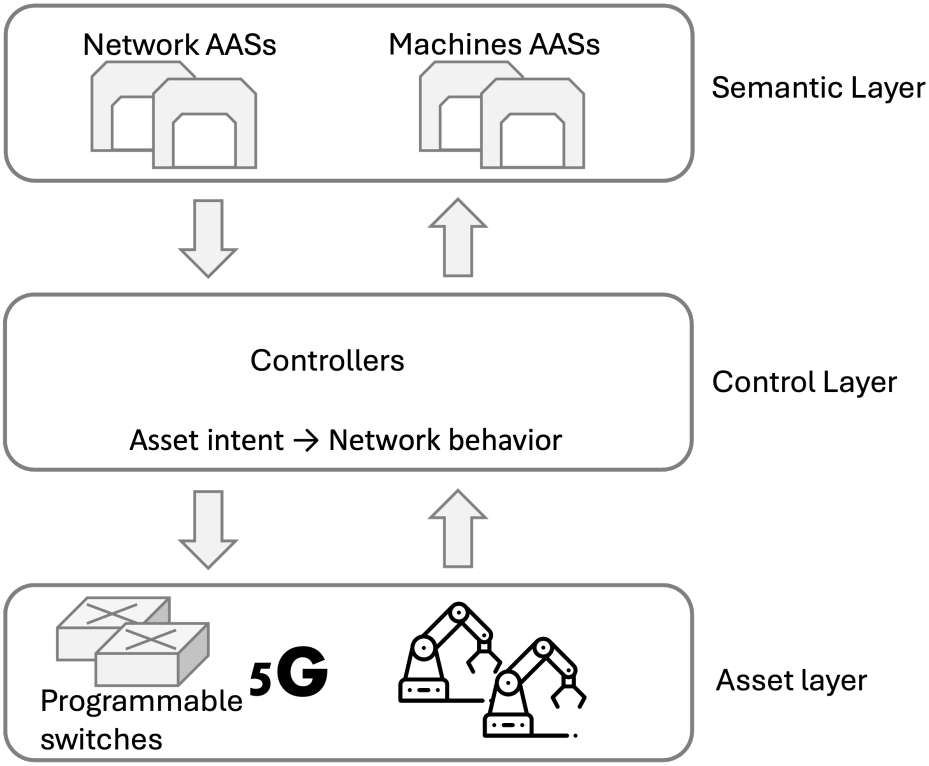

Figure 3. Illustration of the proposed control abstraction in which the OT network becomes an I4.0 asset through its own Asset Administration Shell (AAS). Asset-level operational state and intent are exposed via standardized AAS submodels and translated into network policies enforced at runtime through programmable data planes (SDN/P4).

4.1. Architectural Components

The architecture is shown in Figure 4, and its building blocks are:

- Three Factory Machines, i.e., the OT environment where actual production takes place.
- Two switches implementing the OT network data plane, thus allowing communication between the machines.
- A 5G network infrastructure, which we assume to be public and therefore usable to access the OT environment from the outside of the factory, i.e., for remote maintenance purposes.
- Two network controllers that can be used to program the network data plane, i.e., the switches' forwarding capabilities, as well as the interfaces towards the 5G network.
- The AAS infrastructure, which includes the various AASs needed, as explained later, and the AAS management infrastructure, typically hosted outside the OT in the factory IT network.

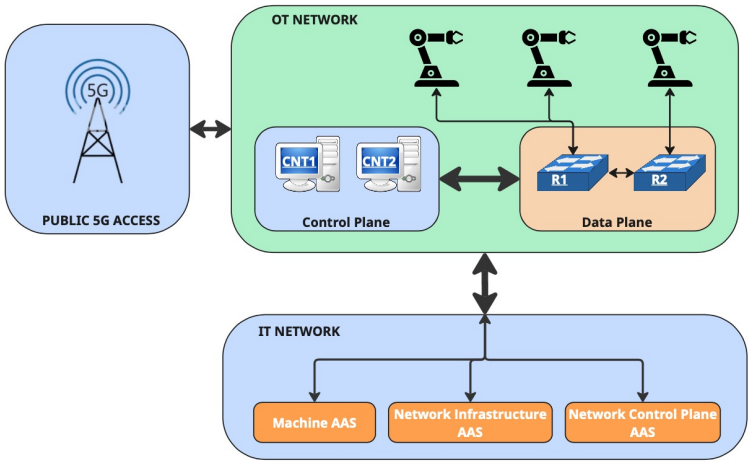

Figure 4. The proposed architecture that includes both the AAS Project and the testbed use cases.

In our testbed, the AAS infrastructure was implemented using Eclipse BaSyx [51], a set of open-source software components to support developers in implementing Industrie 4.0 concepts. Besides the AAS software tools, BaSyx provides the management infrastructure required to operate the AASs at runtime. This includes the AAS registry, a directory-like service through which AASs are announced and discovered, and the BaSyx web-based user interface. Each AAS is interconnected with the others and can be inspected or invoked by the administrator from the BaSyx web interface: the prototype interface views are reported in Appendix A (Figures A1 and A2).

As previously explained, the network has its own AAS: following modern architectural principles, we decided to separate the control plane from the data plane. This separation is known as Control–User Plane Separation (CUPS) [52], a principle that emerged in LTE [53] and 5G [54] and is nowadays a best practice in network design, because it improves scalability and flexibility. Therefore, we designed two distinct AASs to model the network:

1. The control plane AAS, which pairs with the network control plane and therefore with the logic governing network behavior;
2. The network infrastructure AAS, which pairs with the network data plane, i.e., the network nodes, and therefore determines their behavior according to the logic dictated by the control plane.

The control-plane AAS exposes controller-selection and forwarding-rule operations, while the data-plane AAS exposes switch-level functions such as counter retrieval. The 5G capability is integrated as a submodel of the machine AAS because connectivity is asset-related in the maintenance scenario.

Finally, we highlight that in practice, neither of these two AASs directly manipulates network devices. Instead, they act as a semantic and operational abstraction layer, exposing high-level intent (e.g., isolate an asset, enable maintenance mode, retrieve traffic counter) through standardized submodels and operations.

The AAS submodels exploit SDN controllers to convert AAS-level intent into network-level forwarding rules and telemetry queries. These latter actions can nowadays be implemented with two different technological approaches:

1. OpenFlow: we implemented an AAS submodel to interact with the Ryu SDN controller (https://ryu-sdn.org, accessed on 22 June 2026) REST APIs;
2. P4 Runtime and P4: we implemented a custom controller capable of installing into the switches specific P4 programs exploiting the P4 Runtime API (https://github.com/p4lang/p4runtime, accessed on 22 June 2026).

In our testbed, we tested both approaches and obtained results that were quite similar in terms of interaction efficiency and response time. Nonetheless, P4 aligns better with current network programmability trends and provides more effective telemetry features and forwarding control. Discussing the specific differences between these two SDN approaches is beyond the scope of this manuscript; an extensive survey on the topic can be found, for instance, in [55].

Finally, we also implemented the AAS for the 5G mobile terminal (the so-called User Equipment, or UE), with the capability to control 5G connections, as discussed in detail in Section 6.2.

4.2. AAS–SDN Implementation, Integration and Control Loop

Figure 5 illustrates the sequence of interactions triggered by an AAS operation such as `EnableMaintenanceMode`.

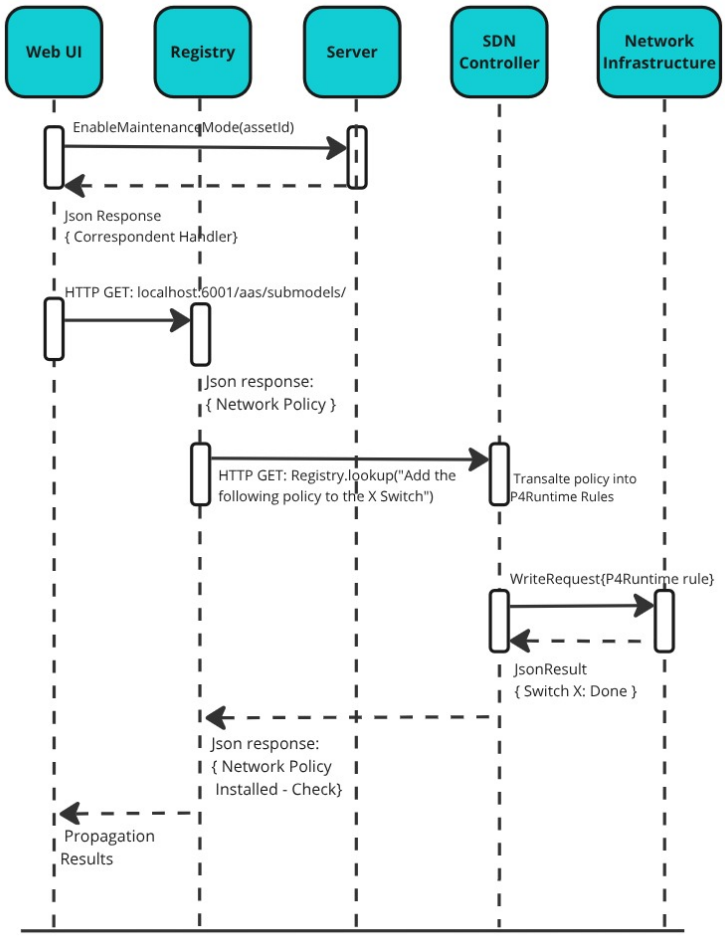

Figure 5. AAS-to-SDN operation workflow through P4Runtime.

The execution flow is as follows:

1. A network administrator invokes an AAS operation (e.g., `EnableMaintenanceMode(assetId)`) via the BaSyx Web UI.
2. The BaSyx Server dispatches the request to the corresponding AAS operation handler.
3. The handler invokes a policy adapter module, which formats the request as a network policy and forwards it to the SDN controller through a northbound API (REST or gRPC).
4. The SDN controller translates the policy into a set of P4Runtime `WriteRequests`, installing or updating table entries in the target P4 switch.

5. The switch enforces the new forwarding behavior immediately, while the AAS updates its exposed state properties (e.g., maintenance mode status and last update timestamp).

In the maintenance mode use case, this translation blocks intra-factory (east–west) traffic involving the selected asset, while allowing only explicitly authorized north–south traffic via the 5G interface. Security and connectivity policies are represented in the AAS through a dedicated Connectivity and Security submodel that encodes high-level intent independently of the underlying SDN technology. Each policy specifies scope and identifiers, traffic selectors (source/destination IP, protocol, port), the intended action (e.g., `ALLOW`, `DENY`, `MAINTENANCE_MODE`), the enforcement point, and lifecycle metadata. An excerpt of the submodel template used in the implementation is shown in Listing 1.

Listing 1. Example AAS connectivity security submodel (simplified).

```
{
  "idShort": "ConnectivitySecurity",
  "submodelElements": [
    {
      "idShort": "Policy_Block_M1_to_M2",
      "modelType": "SubmodelElementCollection",
      "value": [
        {"idShort": "policyId", "value": "blk-m1-m2"},
        {"idShort": "action", "value": "DENY"},
        {"idShort": "srcIp", "value": "10.0.1.10"},
        {"idShort": "dstIp", "value": "10.0.1.11"},
        {"idShort": "protocol", "value": "ANY"},
        {"idShort": "enforcementPoint", "value": "R1"},
        {"idShort": "status", "value": "INSTALLED"}
      ]
    }
  ]
}
```

In the prototype, policy updates are serialized by the Network Manager operation handler before they reach the SDN controller. This avoids simultaneous writes from multiple AAS instances in the evaluated use cases.

At runtime, the SDN controller maps this representation to concrete data-plane rules, e.g., by installing P4 table entries that match the specified fields and execute either forwarding or drop actions.

To support anomaly detection and situational awareness, the architecture integrates in-network telemetry with AAS-based observability. Traffic counters are maintained directly within the P4 data plane, either per-port or per-flow. The `GetCounters` AAS operation retrieves this data indirectly through the SDN controller:

1. The AAS invokes `GetCounters()` on the Network Infrastructure AAS.
2. The operation handler requests telemetry data from the SDN controller.
3. The controller issues a P4Runtime `ReadRequest` to retrieve the relevant counter values from the switch.
4. Retrieved statistics (packet and byte counts with timestamps) are returned to the AAS and exposed as read-only properties.

These properties are visualized in the Web UI and correlated with asset-level metrics (e.g., CPU usage from machinery AASs), enabling the distinction between transient traffic fluctuations and sustained anomalous behavior, as discussed in Section 6.

The main graphical interface for interaction is accessible through the IT Network and provides network administrators with all AASs, each accompanied by its respective submodel and operations. Each of them comprises multiple submodels needed to support the functions of the machinery or network component, focusing on network forwarding

rules and properties to achieve reliable, efficient, and secure communications within and outside the OT Network.

5. Testbed Setup

The testbed used to analyze the feasibility, effectiveness, and performance of the proposed artifact has a hybrid structure. The baseline environment is containerized and virtualized, while the hardware-in-the-loop extension replaces selected emulated elements with industrial-grade Revolution Pi equipment. It comprises three main components: the AAS Infrastructure and coordination system, the network configuration, and the 5G Infrastructure for external connectivity.

As already said, the AAS Infrastructure has been developed using Eclipse BaSyx. We used the BaSyx Registry (v1.5.0), BaSyx Server (v1.5.0), and AAS Web UI (v230703) to construct the main AAS distributed infrastructure, and the BaSyx Java Library to implement and deploy each AAS.

The AASs are hosted on the IT network using an Apache HTTP Server (v2.4.57) and registered within the BaSyx Registry and BaSyx Server. Subsequently, the Web UI retrieves information about submodels, names, and addresses of all AAS instances registered in the Server and Registry components, enabling direct interaction via the user interface. This approach abstracts from low-level complexity irrelevant to the user, who, in the given use case, is only concerned with monitoring the security of Factory Machinery.

- Each Asset Administration Shell consists of one or more submodels.
- Each submodel provides one or more Operations or Properties: Operations allow the user to modify system parameters or specific factory components, whereas Properties display relevant data about a given component.
- Each Asset Administration Shell can connect with others based on the provided network configuration, thereby contributing to overall factory coordination and security.

At the other end, the primary tool used to deploy all components and configure the Network in a single virtualized environment is Kathará [56] (v3.8.3), an open-source container-based network emulation system designed for demonstrating, developing, and testing production networks in a controlled sandbox. Kathará employs Collision Domains as fundamental network nodes. More specifically:

- Each component requiring network access is linked through one or more collision domains;
- Every component in this setup is connected to specific collision domains, ensuring that each P4 Switch (R1 and R2) can exchange information with Network Controllers, the AAS Infrastructure, and individual Machinery.
- Furthermore, each component connected to a collision domain must be assigned a unique IP address and a distinct network route configuration;
- Finally, each machinery unit is equipped with two different IP interfaces to its corresponding SDN switch. This setup simulates a realistic factory environment where machinery maintains multiple connections to the factory network and other components.

5.1. Hardware-in-the-Loop

To move beyond a purely software simulation, we extended the architecture with Hardware-In-The-Loop (HIL). As shown in Figure 6, the reference topology contains three central machines attached to the factory network and two P4 switches with external controllers, CNT1 and CNT2. In the HIL configuration, one machine and one SDN switch are replaced by Revolution Pi equipment (https://revolutionpi.com/en, accessed on 22 June 2026), as shown in Figure 7. Revolution Pi combines a Linux-based software environment with industrial-grade deployment characteristics. The overall architecture remains unchanged, as does the network configuration, thanks to Kathará's ability to connect to

external devices through Linux Bridges. The main RevPi is still connected to its corresponding collision domain as before. Still, it can now provide and gather information from sensors and actuators attached to it in a more standardized, industrial-friendly manner, thereby enabling the operability of Factory Machines.

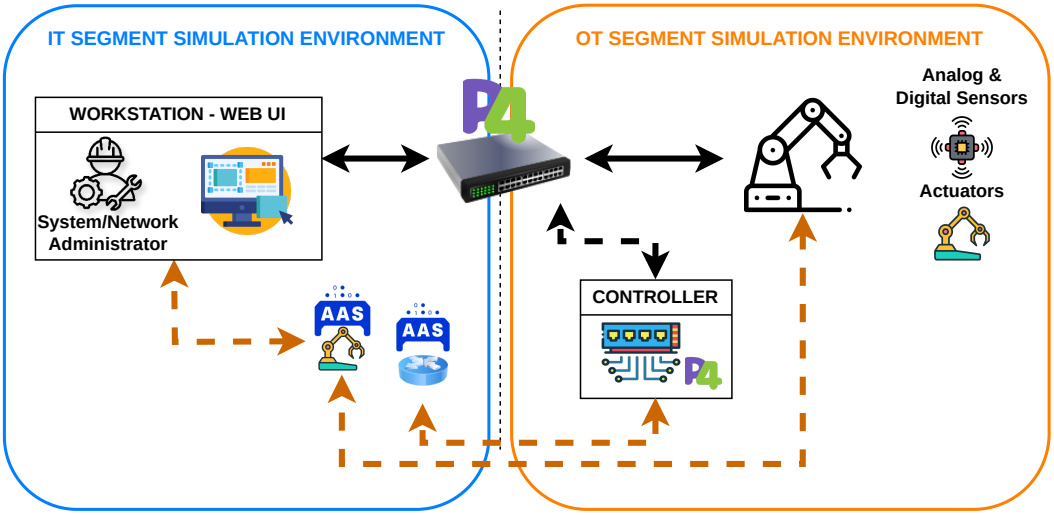

Figure 6. Containerized testbed infrastructure and collision domains.

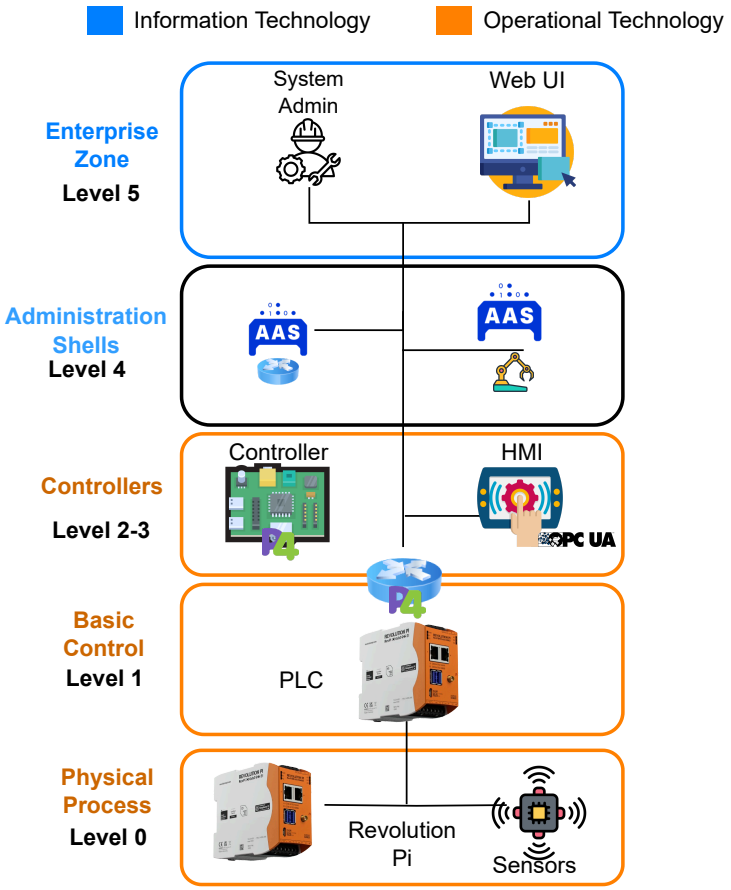

Figure 7. Hardware-in-the-Loop extension with Revolution Pi equipment.

5.2. 5G Implementation

The last step was to test the proposed architecture with external connectivity capabilities. Because access to an industrial private 5G deployment was outside the reproducible software testbed, we implemented a virtualized 5G infrastructure by means of:

- *UERANSIM* (https://github.com/aligungr/UERANSIM, accessed on 22 June 2026) which provides an open source 5G UE and RAN (gNodeB) implementation;
- *Open5GS* (https://open5gs.org/, accessed on 22 June 2026) which provides an open-source 5G core network implementation, including all the various services and supports a complete separation between the control and data plane.

Then we implemented an AAS to control 5G connectivity, including the capability to open 5G PDU sessions to specific destinations and to control QoS (PDU session minimum available bit rate). With these components, it is possible to enable 5G Communication from and to all network hosts in the Kathará virtualized structure that have a virtualized UE (User Equipment). This allows for different communication media, representing a more varied and standardized way to control, perform maintenance, or remotely access factory infrastructure.

6. Results

This section presents a set of use cases designed to validate the proposed architecture. The goal is to demonstrate the capability of the AAS to automate network control and enforce dynamic security policies. Specifically, we focus on the system's ability to customize forwarding rules based on operational needs, manage secure remote maintenance directly through the standard AAS interface, and isolate compromised assets.

The experimental validation relies on the reference architecture depicted in Figure 8, which comprises three manufacturing machines, two programmable switches implementing the P4 Runtime specification [57], and two custom network controllers developed in Go (https://www.go.dev). The infrastructure is logically partitioned into a control plane, hosting the decision-making logic, and a data plane, consisting of the forwarding devices.

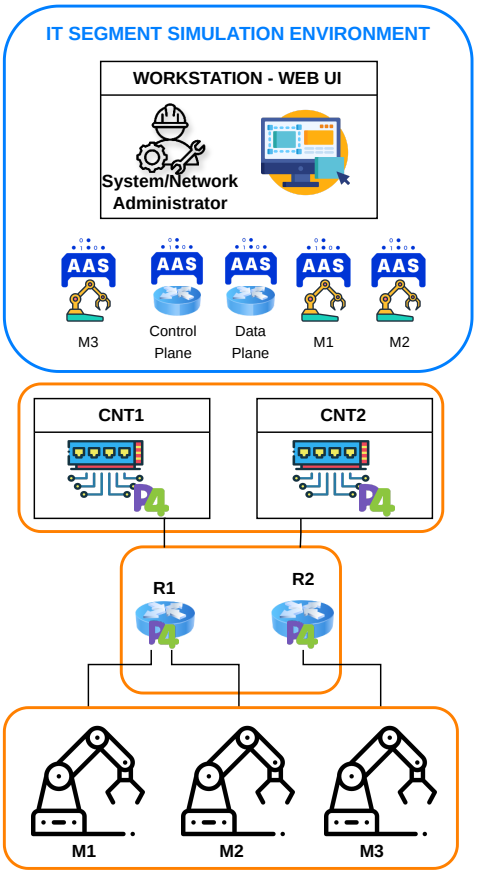

Figure 8. Reference topology employed in the validation use cases.

6.1. Real-Time Customization of Traffic Forwarding Rules

The first outcome of this workflow is illustrated in Figure 9. It shows how traffic within the OT network can be managed and customized in real time through interactions with the Control Plane and Data Plane AASs. Initially, a continuous, high-speed traffic stream is generated, with 1512-byte packets. Subsequently, the network manager activates and configures the SDN controller managing the P4 Switch to enforce a policy that blocks communication between the involved machines. This action is initiated via the AAS and is represented in the figure by the first spike, corresponding to the initial exchange between the controller and the switch. As expected, traffic is successfully interrupted. Afterward, the network manager engages directly with the Network Infrastructure AASs to manually apply new rules: first to permit traffic, and then to block it again (the second and third spike, respectively). These two interactions reflect the communication between the BaSyx Web User Interface and the Control Plane's AAS inside the factory network infrastructure. At the final step, the controller is deactivated, restoring standard network functionality.

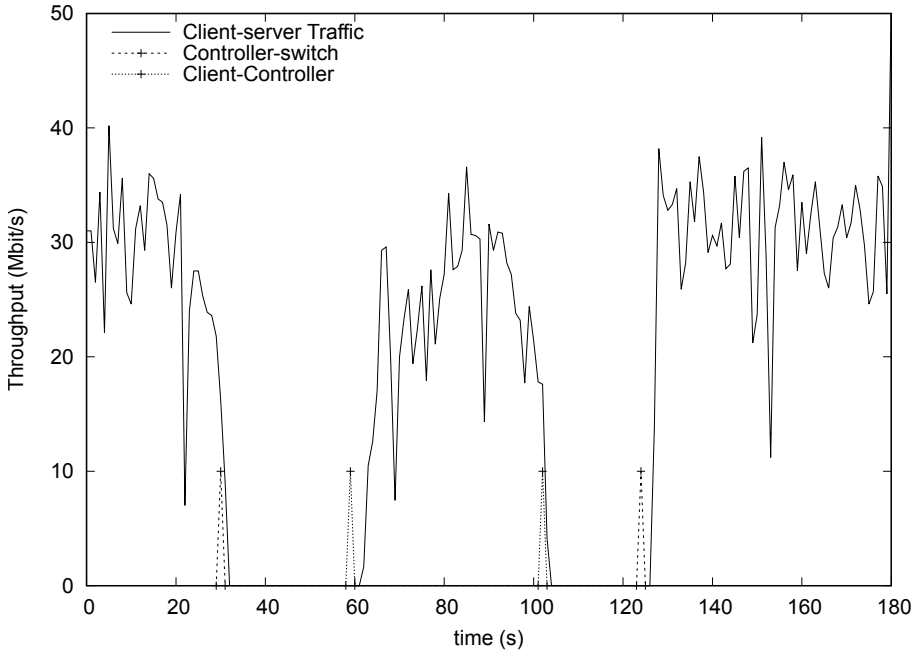

Figure 9. Example of the OT network traffic control achievable through the AAS. A packet flow between the involved hosts is blocked. The policy is then modified by interacting with the AAS submodels and restoring the traffic flow.

To provide a quantitative assessment, the experiment described in this first use case on real-time traffic control, depicted in Figure 9, was repeated 1000 times. Specifically, we evaluated the latencies associated with the rule-management operations (installation and removal) required to block and restore traffic flow. This large sample size provides a stable empirical estimate and allows for a precise decomposition of the total execution time. The results are summarized in Table 4.

Table 4. Statistical analysis of Control Plane latency metrics over 1000 iterations. Values are reported in milliseconds (ms).

Latency Component	Mean	Std Dev	Min	Max
End-to-End Request (T_{total})	4.2106	1.2731	2.7680	37.6330
Controller–Switch Interaction (T_{net})	1.9147	0.5260	1.2650	10.5500
AAS Overhead (T_{aas})	0.0397	0.0216	0.0170	0.5190

The End-to-End Request time, denoted as T_{total}, represents the aggregate latency perceived by the IT client (i.e., the AAS Web GUI), encompassing the entire command chain from issuance to acknowledgement. The Controller–Switch Interaction time (T_{net}) measures the latency required for the SDN controller to communicate with the P4 switch via P4Runtime to apply or remove the forwarding rules. This component (T_{net}) constitutes the baseline physical latency of the network infrastructure. Finally, the AAS Overhead (T_{aas}) isolates the computational cost introduced by the AAS logic, including request parsing and semantic adaptation.

The experimental data indicate that AAS integration introduces negligible overhead ($T_{aas} \approx 0.04$ ms), accounting for less than 1% of the total round-trip time. The system performance is dominated by the network baseline (T_{net}), which averages 1.91 ms. Regarding the operational impact, T_{net} represents the effective transition window during which the network configuration is updated. To place the observed values in context, the Industry 4.0 Digital Twin literature reports that real-time monitoring, sensor-driven synchronization, and control-loop stability require predictable round-trip times typically in the millisecond-to-tens-of-milliseconds range, depending on the closed loop being supervised [58–60]. The measured transition interval of approximately 2 ms is therefore compatible with supervisory security loops and with monitoring-grade synchronization between an asset and its AAS.

In the first use case, the primary objective is to demonstrate the feasibility of real-time monitoring of Network OT traffic and, as a direct consequence of network forwarding rules, subsequently modifying network behavior from a single interface within the IT Network. This highlights the connection between OT and IT Networks, a fundamental concept in Industry 4.0 that increases productivity, cybersecurity, and the overall efficiency of the entire plant. The primary user of this system, represented in Figure 8 as the Network Administrator, has the capability to modify the network setup to enable or disrupt communication between network nodes.

6.2. SDN-Based Maintenance Mode in a 5G-Connected Industrial Environment

This second use case models remote maintenance as a controlled human-in-the-loop operation. We assume the machines can be connected to an external 5G network, in order to be accessed remotely with a pre-defined quality of service. In this case, the interaction with the network infrastructure will be available only through the network terminal. We therefore implemented a 5G connectivity submodel in the AAS of each machine enabled for external access. The submodel controls the User Equipment and exposes operations to establish external connectivity through the public 5G network, in order to enable remote maintenance of factory machines.

Since external connections expose OT systems to attacks from the outside world, we consider it of paramount importance to control them in an integrated manner with internal connectivity to minimize the risk of cyberattacks. When a given asset is stopped and requires remote maintenance, we assume that the asset must first be disconnected from the OT network, in order to avoid potential backdoors into the production environment. It is then connected to the remote operator through a dedicated 5G PDU session.

The AAS 5G submodel interacts with the 5G User Equipment, i.e., the UERANSIM simulator in this case, and may request PDU session setup and tear-down. A screenshot of the related AAS and its Mobile Communication submodel is provided in Appendix A (Figure A3).

At the same time the P4 switch to which the asset is connected must block all traffic from/to the asset and the OT network (as shown in Figure 9), and the asset must then establish a PDU session towards the 5G. The transition is shown in Figure 10. The machine

first exchanges traffic on the internal OT network (purple line). The local OT traffic is then blocked, and the 5G traffic (green line) is started. After a configurable maintenance window, the original OT operations are restored. In the same figure, the purple plot is registered on the internal P4 switch and the green plot on the asset 5G interface, superimposed on the same time scale.

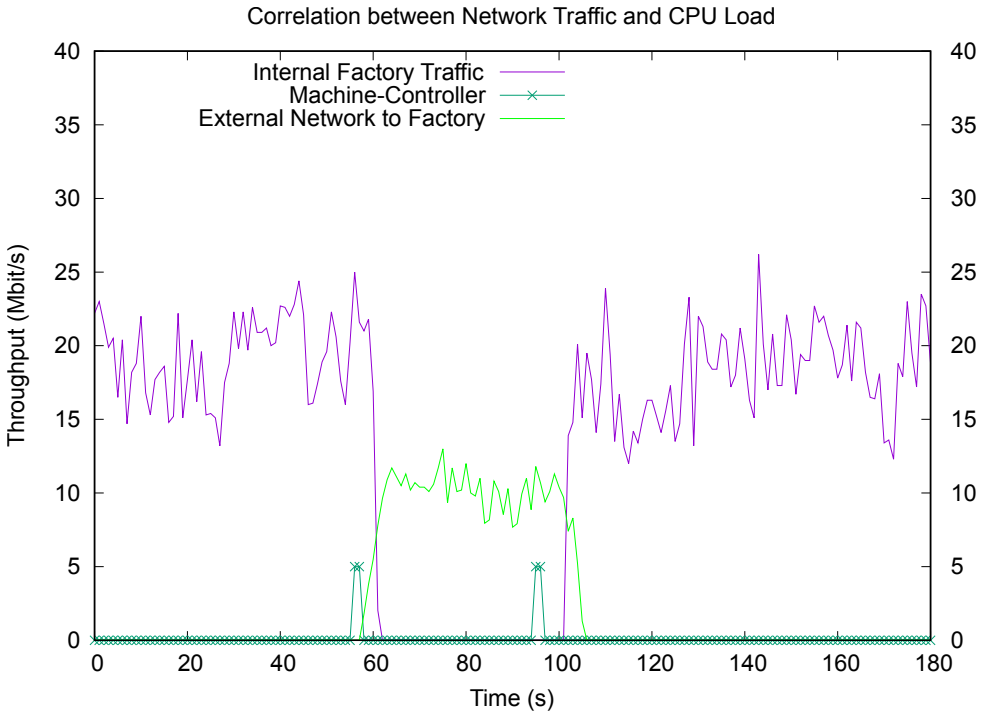

Figure 10. Maintenance mode transition between internal OT traffic and 5G access.

From an Industry 5.0 perspective, the maintenance mode is not merely an automated machine event. It is a safe interface for a remote expert or an on-site operator to interact with an asset while the AAS keeps the network state explicit, auditable, and reversible. The AAS therefore reduces cognitive and cybersecurity burden: the operator requests a semantic maintenance action, and the infrastructure coordinates isolation, external reachability, and rollback.

To provide some insight into this latter operation, we measured the time required to open and close the PDU session, as shown in Figure 11. These values are functional measurements from the virtualized 5G environment. They validate the orchestration chain, but should not be generalized to production radio latency without dedicated 5G hardware testing.

6.3. Anomaly Detection and Isolation via Cross-Layer Correlation

This third use case focuses on demonstrating cross-layer correlation in the context of remotely accessible OT networks, where exposure to external threat vectors increases the attack surface. Accordingly, our system implements a security mechanism based on cross-layer correlation, avoiding reliance on isolated network or machinery data. This combined analysis allows the system to distinguish between legitimate operational bursts and malicious activities.

First, the system monitors real-time network telemetry via packet–byte counters embedded in the P4 Switches. Specifically, the system utilizes the `GetCounters` operation, as detailed in Section 4, to periodically retrieve traffic statistics from the Network Infrastructure AAS. Then, this network data is not analyzed in isolation but is correlated with host-level metrics retrieved from the Asset Administration Shell of the target machinery.

In this specific use case, the chosen target machinery is the RevPi, integrated into the Hardware-in-the-Loop setup, as described in Section 5, while the system metric is the CPU usage, retrieved directly from the RevPi's AAS.

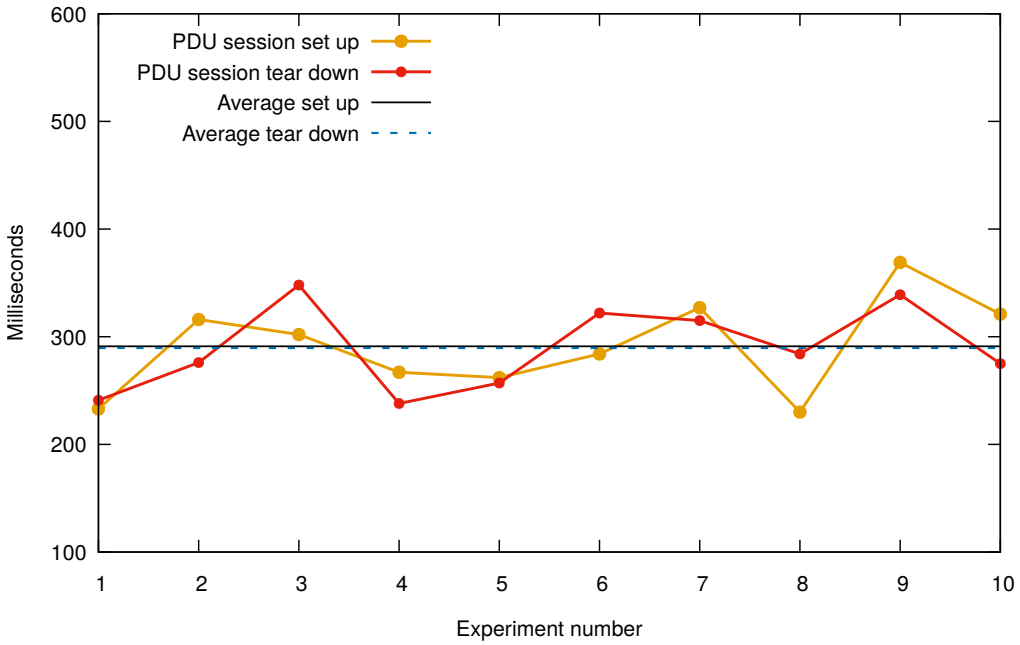

Figure 11. Time to set up and tear down the 5G PDU session. The plot shows 10 experiments and their respective averages.

6.3.1. Baseline Characterization and Threshold Definition

To ensure rigorous detection logic and minimize false positives, we conducted a 4 h preliminary profiling phase. During this period, we monitored the system under standard operational conditions to establish operational baselines and define specific safety thresholds for both network traffic and machinery performance. Telemetry data was retrieved from both AAS instances every second, and the observed metrics are summarized in Table 5.

Table 5. Operational baseline metrics for the target machinery (RevPi).

Metric	Mean Value	Max Recorded Value
CPU Usage	29.35%	51.68%
Network Throughput	3.89 Mbps	4.90 Mbps

Based on these observations, the following detection parameters were defined:

- RevPi Performance (CPU): while the RevPi exhibits an average CPU utilization of ≈29%, transient peaks can reach ≈51% during standard operations. To accommodate this normal load variance while identifying potential anomalies, the safety threshold was set at $\tau_{cpu} = 40\%$.
- Network Throughput: with a baseline traffic volume averaging 3.89 Mbps and legitimate spikes up to 4.90 Mbps, the network throughput threshold was conservatively set at $\tau_{net} = 10$ Mbps.

Crucially, the detection mechanism incorporates a temporal consistency check to mitigate false positives caused by transient spikes, such as legitimate load fluctuations observed in the RevPi's baseline. Given the sampling interval of one second, a momentary threshold

violation is insufficient to trigger an alarm. Instead, the system enforces a persistence constraint: an anomaly is formally declared only if the safety thresholds for both CPU usage and network throughput are simultaneously and continuously exceeded for a detection window of $W_{detect} = 30$ s. This logic filters operational noise in the evaluated scenarios. The parameters (τ and W_{detect}) were empirically calibrated for this testbed and should not be read as universal thresholds. More advanced models, including reconstruction-based anomaly detection and unsupervised learning, can be integrated at the AAS layer without changing the SDN enforcement path [25,61].

6.3.2. Attack Detection Scenarios

The detection workflow relies on the simultaneous violation of both τ_{cpu} and τ_{net}. The Network Manager submodel periodically polls the `GetCounters` property from the Network Infrastructure AAS and correlates it with the `CPU Usage` property from the Machine AAS. We illustrate two distinct scenarios to validate the robustness of this correlation mechanism against operational fluctuations that may be detected as false positives and its efficacy in mitigating actual threats.

First, we address the identification of a transient anomaly. Figure 12 illustrates a scenario where a temporary surge in traffic is detected, triggered by a legitimate system maintenance task: a software update initiated on the RevPi by the system administrator. This operation causes a simultaneous increase in network throughput (due to package downloading) and CPU usage (due to installation overhead). However, the violation of the thresholds is transient. Since the data values return to the nominal baseline ($<\tau_{net}$ and $<\tau_{cpu}$) within the predetermined observation window, the system correctly classifies this event as a non-malicious operational spike. Consequently, no blocking action is triggered, demonstrating the system's ability to maintain operational continuity and avoid unnecessary disruptions typical of overly aggressive intrusion detection systems.

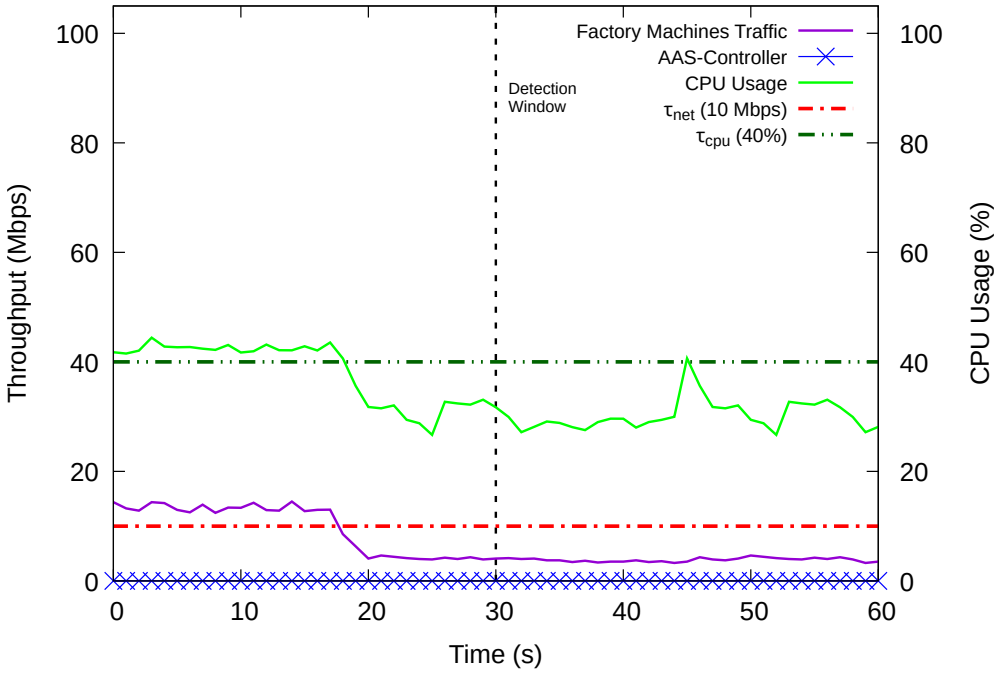

Figure 12. Transient anomaly scenario: the system detects a simultaneous spike in network throughput and CPU usage caused by a generic software update on the target machine. Since the values rapidly return to nominal levels, the correlation logic identifies this as a transient anomaly, and no isolation measures are taken.

Conversely, the second scenario demonstrates the detection and isolation of a sustained attack. Figure 13 depicts the system dynamics during a confirmed volumetric Denial-of-Service attack targeting the Modbus server on the RevPi, executed by an adversary operating from a compromised workstation located at Level 3 of the Purdue Enterprise Reference Architecture, as described in Section 3. In this context, the protection mechanism operates through a deterministic closed-loop sequence:

1. Monitoring and Correlation: the Network Manager observes a synchronized violation of both thresholds (τ_{net} and τ_{cpu}), indicating that the surge in network traffic is directly degrading the asset's computational performance.
2. Temporal Confirmation: unlike the transient update scenario, the violation persists continuously throughout the entire duration of the detection window (W_{detect}), satisfying the condition for anomaly confirmation.
3. Active Mitigation: upon confirming the threat, the Network Manager triggers an automated response via the SDN Controller AAS to enforce traffic isolation at the data plane level. Specifically, the controller sends a P4Runtime `WriteRequest` to update the P4 switch's IPv4 forwarding table (`ipv4_forward`). A new flow entry is installed that matches the malicious source IP address and executes a `drop` action. As evidenced in the graph, the throughput drops to operational values immediately after the detection window (approx. $t = 30$ s), confirming the successful isolation of the compromised host.

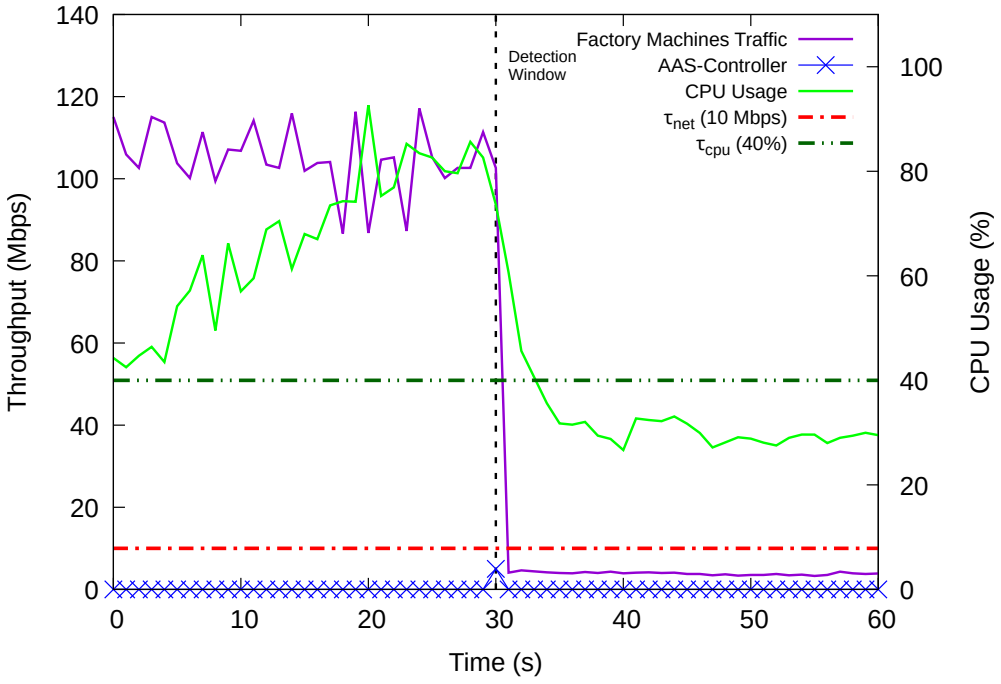

Figure 13. Anomaly scenario: a sustained violation caused by a volumetric DoS attack targeting the RevPi's Modbus server. The system correlates the high network load with abnormal CPU usage and triggers an automated response. The SDN controller pushes a blocking rule, effectively isolating the attacker.

Given the limited number of controlled scenarios, the results are interpreted as proof-of-concept evidence rather than as statistically grounded attack-success, false-positive, or false-negative rates. In the evaluated cases, the defense mitigated the sustained DoS scenario, while no isolation was triggered during the transient benign scenario. A broader quantitative assessment would require a labeled dataset covering multiple attack classes, benign operational and maintenance activities, and low-volume manipulations.

The implemented countermeasure relies on a binary drop action to validate the AAS-to-P4 enforcement loop with a minimal and deterministic mitigation policy. For Industry 5.0 resilience, the same mechanism can evolve toward graceful degradation: P4Runtime could install rate-limiting rules instead of drops, redirect suspicious traffic to an inspection path, preserve worker-safety telemetry, and maintain emergency human-override channels while isolating the malicious vector. The AAS layer is suitable for this extension because it can expose not only network addresses but also the asset role, maintenance state, and safety criticality needed for selective mitigation.

7. Discussion, Limitations, and Future Work

The experimental results validate the proposed AAS-driven control loop as a mechanism for translating asset-level information into programmable network actions. This section discusses the experimental boundaries of this result and examines how the same architectural principles can support more scalable, resilient, and human-centered industrial network control.

7.1. Technology Readiness and Evidence Boundaries

The implemented prototype is classified as TRL 4, corresponding to component and system validation in a laboratory environment. The hardware-in-the-loop integration with Revolution Pi equipment provides partial evidence toward TRL 5, since the AAS and SDN/P4 control loop interact with industrial-grade hardware in a relevant testbed. The 5G subsystem, however, is evaluated through UERANSIM and Open5GS rather than commercial 5G infrastructure, while the anomaly-detection logic is assessed through controlled scenarios. The current evidence, therefore, supports laboratory-scale architectural validation rather than field deployment readiness.

The measured 1000-iteration rule-management experiment shows that the AAS processing component is small compared with controller–switch interaction time in the implemented setup. This supports the claim that the semantic AAS layer is not the dominant latency source. It does not, by itself, prove negligible packet loss, jitter, or behavior under concurrent production traffic; those metrics require dedicated load experiments and are future validation targets.

To make the maturity assessment auditable rather than monolithic, Table 6 maps each architectural block to its current TRL and to the evidence that supports the assigned level. The composite system-level TRL is therefore reported as TRL 4 with partial movement toward TRL 5: the AAS layer, the SDN/P4 control loop, and the HIL asset are validated in a relevant testbed, while the 5G subsystem remains at TRL 3–4 because it is exercised only through virtualized RAN/Core components.

7.2. Detection Scope, Feedback Delay, and Control Stability

The anomaly-detection use case validates the cross-layer detection and enforcement architecture through a deterministic threshold-based logic. The detector correlates network-side indicators (traffic volume) with machine-side indicators (CPU load) and triggers mitigation only when the abnormal condition persists over a confirmation window. This design is suitable for sustained volumetric or resource-exhaustion events, where the cyber and physical symptoms are jointly observable. It is less suitable for stealthy manipulations, such as isolated malicious Modbus commands, low-rate process deviations, or false-data injection events that do not produce persistent changes in CPU load or throughput. The architecture does not depend on this specific threshold logic. The detector is placed above the SDN/P4 enforcement interface: it consumes AAS-exposed telemetry and, when an anomaly is detected, invokes the same mitigation procedure used in the prototype.

Therefore, the detection module can be replaced or extended without changing the AAS-to-SDN/P4 enforcement path.

Table 6. TRL assessment summary.

Block	TRL	Validation Evidence	Experimental Boundary
AAS semantic layer (BaSyx)	4–5	Rule-management benchmark over 1000 iterations; integration with controllers and HIL asset.	Evaluated in laboratory conditions, not in a multi-site industrial deployment.
SDN/P4 control loop	4–5	End-to-end policy installation through P4Runtime on emulated and HIL communication paths.	Tested on a limited topology with two switches and controlled traffic patterns.
HIL industrial asset (RevPi)	5	Industrial-grade controller running representative workloads and exposing AAS-mediated telemetry.	Validated on a single HIL asset rather than a heterogeneous production cell.
5G connectivity subsystem	3–4	Orchestration validated with UERANSIM and Open5GS.	RAN and Core components are virtualized; commercial 5G hardware was not used.
Anomaly-detection logic	3–4	Threshold-based detector evaluated on one sustained DoS scenario and one transient benign scenario.	No large labeled dataset or multi-class attack evaluation is included.
Composite system	4 (→5)	Integrated AAS, SDN/P4, 5G-emulated, and HIL validation in a relevant laboratory testbed.	System-level readiness is limited by the virtualized 5G subsystem and controlled detection scenarios.

More advanced detection logic can be integrated at the same architectural point, but the choice of detector is deployment-dependent. Data-driven alternatives require stronger assumptions: reconstruction models and one-class methods rely mainly on representative nominal data [25,61,62], while supervised classifiers require labeled benign and attack traces, which are often unavailable in OT environments. Lightweight unsupervised methods, such as Isolation Forests, can reduce computational cost and improve interpretability, but their effectiveness depends on the separability of anomalies from normal operational transients [63]. Accordingly, the detector is not considered a generic replaceable block with equivalent alternatives. It is a deployment-specific component whose design depends on telemetry availability, computational constraints, and the class of attacks to be detected. The architectural interface, however, remains unchanged: the detector consumes AAS-exposed telemetry and triggers mitigation through the existing AAS-to-SDN/P4 enforcement path.

The resulting loop should be interpreted as a supervisory security loop, not as a real-time process-control loop. Its purpose is to detect abnormal cyber–physical conditions and enforce network-level mitigation, rather than to stabilize physical dynamics at sub-second time scales. The loop latency includes telemetry sampling, the confirmation window used by the anomaly detector, AAS request processing, controller-side policy handling, and P4Runtime rule installation. This timing is compatible with the mitigation of sustained security events, such as volumetric or resource-exhaustion attacks, but it is not appropriate for immediate actuation or stability-critical control. Future extensions should therefore evaluate shorter sampling intervals, adaptive confirmation windows, and mitigation policies explicitly designed for cyber–physical workflows requiring tighter synchronization.

7.3. Scalability

Scalability in the proposed architecture mainly concerns the AAS-based management path and the SDN/P4 enforcement path. An AAS operation is processed by the BaSyx infrastructure, dispatched to the corresponding operation handler, translated by the Network

Manager into a network policy, and finally installed by the SDN/P4 controller through P4Runtime. The OT data plane is involved only after the rule has been installed, through match–action processing in the programmable switch. Therefore, the AAS layer does not add per-packet processing overhead to industrial traffic; its impact is associated with the rate and concurrency of management operations. As the number of assets increases, the relevant scaling factors are the number of registered AASs and submodels, the frequency of operation invocations, the rate of telemetry reads, and the number of active policies that must be translated into P4 table entries. This is different from data-plane scalability, which depends on switch table capacity, rule granularity, and P4Runtime update rate. The current prototype evaluates this mechanism on a small topology, so it validates the control-loop implementation but does not quantify these factors under large-scale concurrency. The IT/control-plane placement of the AAS infrastructure also leaves deployment flexibility. BaSyx registries, dashboards, and historical asset information can be hosted on plant servers, edge platforms, or cloud infrastructure. Components involved in time-sensitive enforcement, such as the Network Manager and the SDN/P4 controller, should remain close to the plant network when low latency or local availability is required. This separation is consistent with the implemented architecture, where AAS services expose asset-level operations while the programmable switches remain responsible for packet forwarding.

In the current prototype, telemetry correlation is implemented through periodic retrieval of network counters and machine-side metrics. This mechanism is appropriate for the evaluated use case, where the number of monitored assets is limited and the sampling interval is fixed. At a larger scale, however, unchanged per-asset polling would increase management traffic and registry/API load. A more scalable implementation should therefore rely on selective monitoring, event-triggered updates, and aggregation of counters at the controller or switch level.

Policy management represents a second scalability dimension. In the prototype, policy updates are serialized by the Network Manager before being forwarded to the SDN controller. This avoids inconsistent updates in the evaluated scenarios. In larger deployments, where multiple AAS instances or applications may request overlapping network actions, the Network Manager should also provide policy arbitration by detecting conflicting traffic selectors and applying deterministic priorities among isolation, maintenance, and ordinary connectivity rules.

7.4. Industry 5.0 Implications

Although the prototype is primarily designed to validate AAS-driven network control in an Industry 4.5 setting, its mechanisms address three core Industry 5.0 dimensions: resilience, human-centricity, and sustainability. These dimensions are not treated as independent add-ons, but as consequences of the same architectural principle: asset-level semantics are exposed through the AAS and translated into programmable network behavior.

The resilience dimension is represented by the closed-loop mitigation mechanism. In the current implementation, the response to a confirmed anomaly is a binary drop rule installed through the SDN/P4 enforcement path. This action provides a deterministic validation of the AAS-to-P4 loop, but the same mechanism can support more selective responses. Since the mitigation is expressed as an AAS-level policy before being translated into data-plane rules, the action can be extended from simple isolation to rate limiting, traffic mirroring, redirection to an inspection path, or time-bounded quarantine. This is relevant for industrial resilience because the objective is not only to stop malicious traffic, but also to preserve the essential communication needed for diagnosis, safety monitoring, and recovery.

The human-centric dimension is illustrated by the maintenance use case. The operator does not manually configure 5G connectivity, OT isolation, and rollback as separate low-level network actions. Instead, the maintenance state is requested through an asset-level AAS operation, and the infrastructure translates this request into the required network configuration. This reduces the operational burden on human operators and makes the resulting network state explicit and auditable. In a production deployment, the same mechanism can be extended with role-based authorization, operator confirmation for disruptive actions, policy expiration times, and rollback conditions. These extensions would preserve human control over safety-relevant decisions while keeping the underlying network orchestration automated.

The sustainability dimension is not directly implemented in the current prototype, but it can be integrated through the same AAS-based abstraction. An energy-aware or environmental submodel could expose information such as energy consumption, carbon intensity, or preferred low-energy connectivity options for network and edge resources.

The prototype, therefore, does not claim to implement a complete Industry 5.0 system. Its contribution is more specific: it demonstrates a programmable and semantic control layer through which Industry 5.0 requirements can be expressed and enforced at the network level. Resilience is addressed through closed-loop mitigation, human-centricity through auditable maintenance operations, and sustainability through the possibility of adding environmental constraints to AAS-based policy decisions.

7.5. Standardization Perspective

The prototype also exposes a standardization gap in current AAS-based industrial modeling. While AAS specifications provide a general structure for representing industrial assets and their submodels, there is no commonly adopted AAS submodel dedicated to programmable network control. In this work, the OT network is therefore modeled through implementation-specific AASs and submodels. This is sufficient to validate the proposed architecture, but it limits portability across different plants, controllers, and network-programming technologies.

Following the SDN principle adopted in the architecture, we argue that network-related AAS standardization should distinguish between control-plane and data-plane functions. A first reusable submodel should describe the Network Control Plane. Its role is to expose the operations and properties needed to express network intent, select or configure controllers, install or remove policies, and track policy state. Typical elements include policy identifiers, target assets, traffic selectors, requested actions, enforcement scope, priority, validity conditions, and installation status. This submodel would represent what the industrial system requires from the network, without binding the AAS interface to a specific enforcement technology.

A second reusable submodel should describe the Network Data Plane. Its role is to expose the observable and enforceable state of the forwarding infrastructure, including switches, ports, counters, flow statistics, table entries, and enforcement points. This submodel would support telemetry retrieval and verification of the rules installed by the control plane. In the prototype, this role is implemented through switch-level functions such as counter retrieval and through P4Runtime-based rule enforcement; in a standardized form, the same abstraction could be mapped to P4, OpenFlow, or other programmable forwarding technologies.

The separation between these two submodels is important because control and data planes evolve differently. The control-plane submodel captures asset-level network intent and policy lifecycle information, whereas the data-plane submodel captures the state and telemetry of the forwarding devices. Keeping these concerns separate makes the

AAS representation more reusable: the same control-plane intent can be translated by the Network Manager into different enforcement backends, while the data-plane submodel can expose technology-specific measurements through a common asset-oriented interface.

This standardization direction is consistent with IEC 63278, which provides the structural basis for AASs and their submodels [16]. The contribution suggested by this work is not a new network protocol, but the definition of two network-oriented AAS submodels: one for the programmable network control plane and one for the programmable network data plane. Such submodels would make asset-aware connectivity, security policies, telemetry, and operational constraints portable across SDN/P4, OpenFlow, 5G, DetNet, and APN-based infrastructures [34–36,64].

8. Conclusions

This paper presented an AAS-driven network control abstraction for converged OT/IT environments. The proposed approach models the OT network as an industrial asset and uses Asset Administration Shells to expose network-control operations, telemetry, and asset-related context through a standardized semantic layer. These AAS-level operations are translated into programmable P4 actions, enabling forwarding policies and mitigation rules to be installed at runtime.

The main contribution is the operational coupling between industrial asset semantics and programmable network enforcement. Rather than treating the network as a passive communication substrate, the proposed architecture makes network behavior controllable through the same asset-oriented abstraction used to represent industrial devices. This coupling enables three capabilities demonstrated in the validation use cases: dynamic customization of forwarding rules, controlled 5G-assisted maintenance with OT isolation and rollback, and anomaly mitigation based on the correlation between AAS-exposed machine and network telemetry.

The evaluation was conducted on a hybrid testbed combining a containerized network environment with a hardware-in-the-loop extension based on Revolution Pi equipment. Across the three use cases, the prototype demonstrates that AAS operations can be used to configure forwarding behavior, coordinate controlled 5G-assisted maintenance, and trigger network-level mitigation from cross-layer telemetry correlation. These results support the feasibility of the proposed abstraction at laboratory scale and justify a TRL 4 assessment, with partial evidence toward TRL 5 for the components exercised with industrial hardware.

Finally, the work identifies a standardization direction for network-oriented AAS models. The implemented architecture separates network control-plane functions from data-plane state and telemetry, following the same separation adopted by SDN-based network design. In this sense, the contribution is not only an implementation of AAS-based network control, but also a first step toward a more portable representation of programmable industrial networks through AAS submodels.

Author Contributions: Conceptualization, R.B., A.M., L.R. and F.C.; methodology, R.B., A.M., L.R. and F.C.; software, R.B.; validation, R.B., A.M. and L.R.; investigation, R.B., A.M. and L.R.; resources, A.M., L.R. and F.C.; data curation, R.B.; writing—original draft preparation, A.M., L.R. and F.C.; writing—review and editing, M.P., R.G., A.M., L.R. and F.C.; visualization, R.B., A.M., L.R. and F.C.; supervision, A.M., R.B., M.P. and F.C.; project administration, A.M., R.B., M.P. and F.C. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: The raw data supporting the conclusions of this article will be made available by the authors on request.

Acknowledgments: This study was carried out in the framework of the projects Cri4.0 (CUP: E37G22000490007) and IGNITE5.0 (CUP: E77G22000640003), co-funded by the European funds of the Emilia-Romagna Region, ERDF Regional Programme 2021–2027, Priority 1 Research, Innovation and Competitiveness, Action 1.1.2. This study was also partially funded by the PRIN 2022 Empowering awareness with a digital twin for mission-critical communications (EDIT4MCC)–Codice Progetto 2022Z4N84X_001 CUP J53C24002800006.

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

5G	Fifth-Generation mobile network
AAS	Asset Administration Shell
APN	Application-aware Networking
API	Application Programming Interface
CPU	Central Processing Unit
CUPS	Control–User Plane Separation
DetNet	Deterministic Networking
DoS	Denial of Service
DSR	Design Science Research
gRPC	Google Remote Procedure Call
GUI	Graphical User Interface
HIL	Hardware-in-the-Loop
HMI	Human–Machine Interface
HTTP	Hypertext Transfer Protocol
I4.0	Industry 4.0
IAM	Identity and Access Management
ICS	Industrial Control System
IDS	Intrusion Detection System
IEC	International Electrotechnical Commission
IP	Internet Protocol
IPv4	Internet Protocol version 4
IT	Information Technology
LTE	Long-Term Evolution
MITM	Man-in-the-Middle
MQTT	Message Queuing Telemetry Transport
OT	Operational Technology
P4	Programming Protocol-independent Packet Processors
PDU	Protocol Data Unit
PLC	Programmable Logic Controller
QoS	Quality of Service
RAMI 4.0	Reference Architecture Model for Industrie 4.0
RAN	Radio Access Network
REST	Representational State Transfer
RevPi	Revolution Pi
SCADA	Supervisory Control and Data Acquisition
SDN	Software-Defined Networking
TRL	Technology Readiness Level
UE	User Equipment
UI	User Interface

Appendix A. Prototype Screenshots

This appendix provides representative screenshots (Figures A1–A3) of the implemented BaSyx-based prototype, illustrating the AAS interface and the exposed asset information used during the validation scenario.

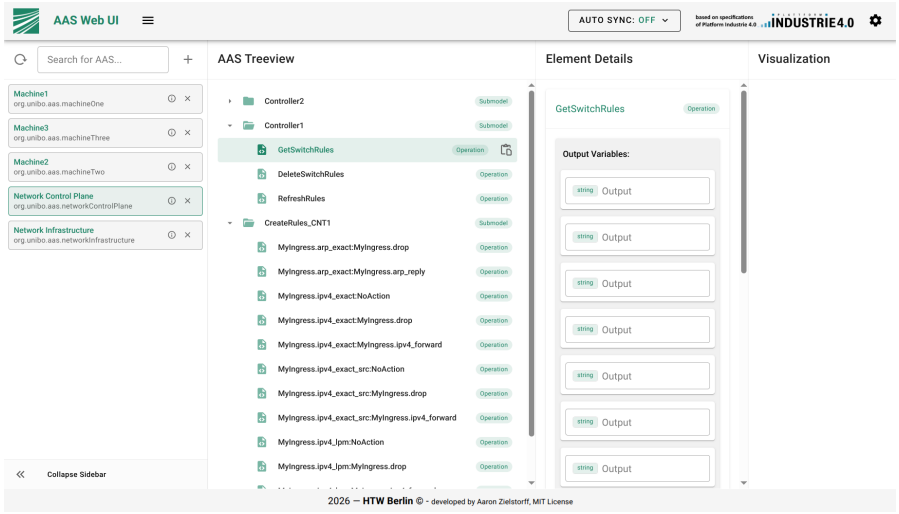

Figure A1. Network control-plane AAS prototype interface.

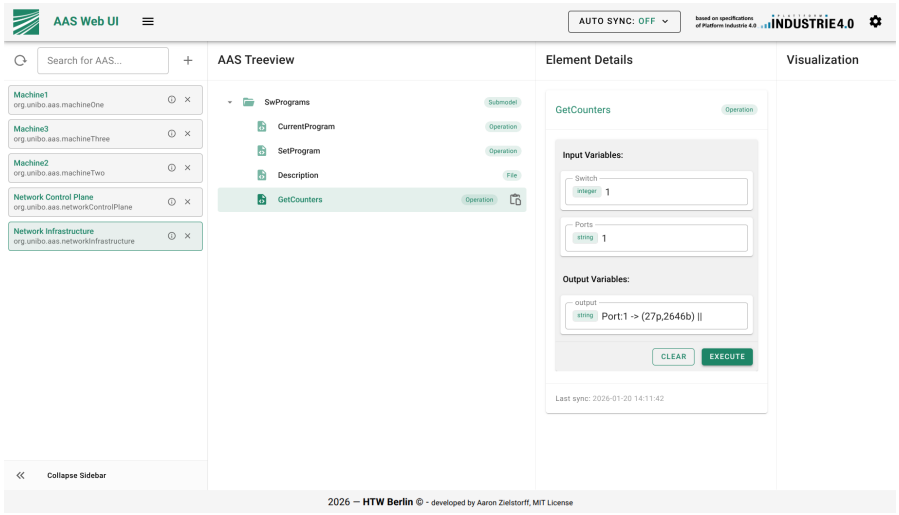

Figure A2. Network data-plane AAS prototype interface.

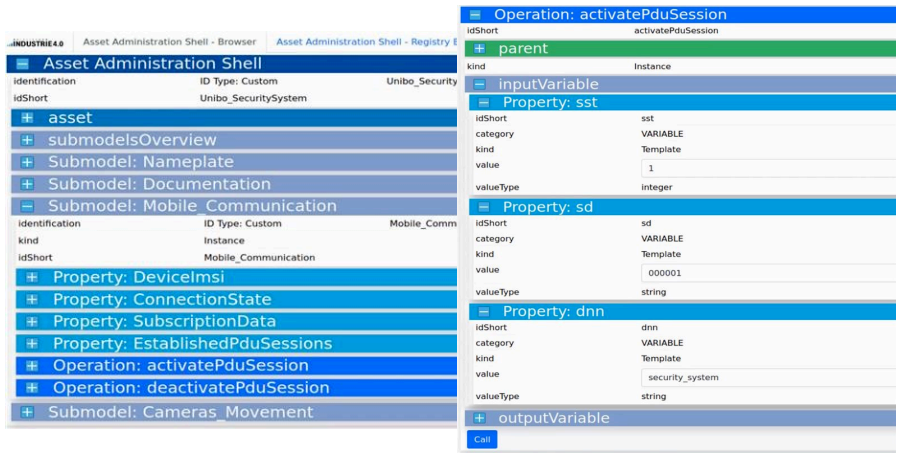

Figure A3. Asset AAS prototype interface with 5G connectivity submodel.

References

1. Berardi, D.; Callegati, F.; Giovine, A.; Melis, A.; Prandini, M.; Rinieri, L. When Operation Technology Meets Information Technology: Challenges and Opportunities. *Future Internet* **2023**, *15*, 95. [CrossRef]
2. Gartner. Operational Technology (OT)—Gartner Glossary. 2022. Available online: https://www.gartner.com/en/information-technology/glossary/operational-technology-ot (accessed on 22 June 2026).
3. Lasi, H.; Fettke, P.; Kemper, H.G.; Feld, T.; Hoffmann, M. Industry 4.0. *Bus. Inf. Syst. Eng.* **2014**, *6*, 239–242. [CrossRef]
4. Ghoson, N.H.; Meyrueis, V.; Benfriha, K.; Guiltat, T.; Loubere, S. A review on the static and dynamic risk assessment methods for OT cybersecurity in industry 4.0. *Comput. Secur.* **2025**, *150*, 104295.
5. Hankel, M.; Rexroth, B. *The Reference Architectural Model Industrie 4.0 (Rami 4.0)*; ZVEI: Frankfurt am Main, Germany, 2015; Volume 2, pp. 4–9.
6. Tantik, E.; Anderl, R. Integrated data model and structure for the Asset Administration Shell in Industrie 4.0. *Procedia Cirp* **2017**, *60*, 86–91. [CrossRef]
7. Wagner, C.; Grothoff, J.; Epple, U.; Drath, R.; Malakuti, S.; Grüner, S.; Hoffmeister, M.; Zimermann, P. The role of the Industry 4.0 asset administration shell and the digital twin during the life cycle of a plant. In *Proceedings of the 2017 22nd IEEE International Conference on Emerging Technologies and Factory Automation (ETFA), Limassol, Cyprus, 12–15 September 2017*; IEEE: New York, NY, USA, 2017; pp. 1–8.
8. McKeown, N. Software-defined networking. *INFOCOM Keynote Talk* **2009**, *17*, 30–32.
9. Bosshart, P.; Daly, D.; Gibb, G.; Izzard, M.; McKeown, N.; Rexford, J.; Schlesinger, C.; Talayco, D.; Vahdat, A.; Varghese, G.; et al. P4: Programming protocol-independent packet processors. *ACM SIGCOMM Comput. Commun. Rev.* **2014**, *44*, 87–95. [CrossRef]
10. Hevner, A.R.; March, S.T.; Park, J.; Ram, S. Design Science in Information Systems Research. *MIS Q.* **2004**, *28*, 75–105. [CrossRef]
11. Peffers, K.; Tuunanen, T.; Rothenberger, M.A.; Chatterjee, S. A Design Science Research Methodology for Information Systems Research. *J. Manag. Inf. Syst.* **2007**, *24*, 45–77. [CrossRef]
12. Plattform Industrie 4.0. *Details of the Asset Administration Shell – Part 1: The Exchange of Information Between Partners in the Value Chain of Industrie 4.0*; Technical Report; Plattform Industrie 4.0: Berlin, Germany, 2020.
13. Lu, Q.; Zhu, D.; Li, M. Intelligent manufacturing with asset administration shell: A bibliometric and knowledge map analysis. *Digit. Eng.* **2026**, *9*, 100077. [CrossRef]
14. Hosseini, A.M.; Sauter, T.; Kastner, W. Towards adding safety and security properties to the Industry 4.0 Asset Administration Shell. In *Proceedings of the 2021 17th IEEE International Conference on Factory Communication Systems (WFCS), Linz, Austria, 9–11 June 2021*; IEEE: New York, NY, USA, 2021; pp. 41–44.
15. Hosseini, A.M.; Sauter, T.; Kastner, W. A safety and security reference architecture for asset administration shell design. In *Proceedings of the 2022 IEEE 18th International Conference on Factory Communication Systems (WFCS), Pavia, Italy, 27–29 April 2022*; IEEE: New York, NY, USA, 2022; pp. 1–8.
16. International Electrotechnical Commission. IEC 63278-1:2023 Asset Administration Shell for Industrial Applications—Part 1: Asset Administration Shell Structure. 2023. Available online: https://webstore.iec.ch/en/publication/65628 (accessed on 28 May 2026).
17. Hauser, F.; Häberle, M.; Schmidt, M.; Menth, M. P4-ipsec: Site-to-site and host-to-site vpn with ipsec in p4-based sdn. *IEEE Access* **2020**, *8*, 139567–139586.
18. AlSabeh, A.; Khoury, J.; Kfoury, E.; Crichigno, J.; Bou-Harb, E. A survey on security applications of P4 programmable switches and a STRIDE-based vulnerability assessment. *Comput. Netw.* **2022**, *207*, 108800. [CrossRef]
19. Pittalà, G.F.; Rinieri, L.; Al Sadi, A.; Davoli, G.; Melis, A.; Prandini, M.; Cerroni, W. Leveraging Data Plane Programmability to enhance service orchestration at the edge: A focus on industrial security. *Comput. Netw.* **2024**, *246*, 110397. [CrossRef]
20. Borsatti, D.; Davoli, G.; Cerroni, W.; Callegati, F. Service Function Chaining Leveraging Segment Routing for 5G Network Slicing. In *Proceedings of the 2019 15th International Conference on Network and Service Management (CNSM), Halifax, NS, Canada, 21–25 October 2019*; IEEE: New York, NY, USA, 2019; pp. 1–6.
21. De Benedictis, A.; Flammini, F.; Mazzocca, N.; Somma, A.; Vitale, F. Digital Twins for Anomaly Detection in the Industrial Internet of Things: Conceptual Architecture and Proof-of-Concept. *IEEE Trans. Ind. Inform.* **2023**, *19*, 11553–11563. [CrossRef]
22. Varghese, S.A.; Dehlaghi Ghadim, A.; Balador, A.; Alimadadi, Z.; Papadimitratos, P. Digital Twin-based Intrusion Detection for Industrial Control Systems. In *Proceedings of the 2022 IEEE International Conference on Pervasive Computing and Communications Workshops and Other Affiliated Events (PerCom Workshops), Virtual Event, 16–18 March 2022*; IEEE: New York, NY, USA, 2022; pp. 611–617. [CrossRef]
23. Alcaraz, C.; López, J. WASAM: A Dynamic Wide-Area Situational Awareness Model for Critical Domains in Smart Grids. *Future Gener. Comput. Syst.* **2014**, *30*, 146–154. [CrossRef]
24. Alem, S.; Espes, D.; Nana, L.; Martin, E.; De Lamotte, F. A novel bi-anomaly-based intrusion detection system approach for industry 4.0. *Future Gener. Comput. Syst.* **2023**, *145*, 267–283.

25. Fung, C.; Srinarasi, S.; Lucas, K.; Phee, H.B.; Bauer, L. Perspectives from a comprehensive evaluation of reconstruction-based anomaly detection in industrial control systems. In *Proceedings of the European Symposium on Research in Computer Security, Copenhagen, Denmark, 26–30 September 2022*; Springer: Berlin/Heidelberg, Germany, 2022; pp. 493–513.
26. Zhang, H.; Xiao, X.; Mercaldo, F.; Ni, S.; Martinelli, F.; Sangaiah, A.K. Classification of ransomware families with machine learning based onN-gram of opcodes. *Future Gener. Comput. Syst.* **2019**, *90*, 211–221.
27. Rossi, D.; Tontini, G.; Borsatti, D.; Callegati, F. Integration of 5G connectivity with the Asset Administration Shell in Industry 4.0. In *Proceedings of the 2022 13th International Conference on Network of the Future (NoF), Ghent, Belgium, 5–7 October 2022*; IEEE: New York, NY, USA, 2022; pp. 1–3. [CrossRef]
28. Perez Olaya, S.S.; Kulasekara Pallewaththe Kankanamge, H.; Cainelli, G.P.; Gambal, B. Management of Industrial 5G Networks over Asset Administration Shell. In *Proceedings of the 2024 IEEE 20th International Conference on Factory Communication Systems (WFCS), Toulouse, France, 17–19 April 2024*; IEEE: New York, NY, USA, 2024; pp. 1–4. [CrossRef]
29. Puligheddu, C.; Ashdown, J.; Chiasserini, C.F.; Restuccia, F. SEM-O-RAN: Semantic O-RAN slicing for mobile edge offloading of computer vision tasks. *IEEE Trans. Mob. Comput.* **2023**, *23*, 7785–7800.
30. Busacca, F.; Faraci, G.; Grasso, C.; Palazzo, S.; Schembra, G. Designing a multi-layer edge-computing platform for energy-efficient and delay-aware offloading in vehicular networks. *Comput. Netw.* **2021**, *198*, 108330.
31. Busacca, F.; Galluccio, L.; Palazzo, S. A marketplace model for drone-assisted edge computing in 5G scenarios. *Comput. Netw.* **2022**, *219*, 109453. [CrossRef]
32. Chen, X.; Zhang, H.; Wu, C.; Mao, S.; Ji, Y.; Bennis, M. Optimized computation offloading performance in virtual edge computing systems via deep reinforcement learning. *IEEE Internet Things J.* **2018**, *6*, 4005–4018. [CrossRef]
33. Yu, H.; Wang, H.; Tiwari, D.; Li, J.; Park, S.J. Stellaris: Staleness-Aware Distributed Reinforcement Learning with Serverless Computing. In Proceedings of the SC24: International Conference for High Performance Computing, Networking, Storage and Analysis, Atlanta, GA, USA, 17–22 November 2024.
34. IETF. Application-Aware Networking (APN) Working Group. 2026. IETF Datatracker. Available online: https://datatracker.ietf.org/wg/apn/about/ (accessed on 28 May 2026).
35. IETF Deterministic Networking Working Group. Deterministic Networking (DetNet) Working Group Charter. 2026. Available online: https://datatracker.ietf.org/wg/detnet/charter/ (accessed on 28 May 2026).
36. Finn, N.; Thubert, P.; Varga, B.; Farkas, J. RFC 8655: Deterministic Networking Architecture. IETF RFC 8655. 2019. Available online: https://www.rfc-editor.org/info/rfc8655/ (accessed on 22 June 2026).
37. Graessler, I.; Pöhler, A. Integration of a digital twin as human representation in a scheduling procedure of a cyber-physical production system. In *Proceedings of the 2017 IEEE International Conference on Industrial Engineering and Engineering Management (IEEM)*; IEEE: New York, NY, USA, 2017; pp. 289–293.
38. Cheng, Z. Human digital twin with applications. In Proceedings of the 7th International Digital Human Modeling Symposium (DHM 2022) and Iowa Virtual Human Summit 2022, Iowa City, IA, USA, 29–31 August 2022.
39. International Telecommunication Union. Recommendation ITU-T L.1333: Carbon Data Intensity for Network Energy Performance Monitoring. 2022. Available online: https://www.itu.int/rec/T-REC-L.1333-202209-I (accessed on 28 May 2026).
40. International Telecommunication Union. Recommendation ITU-T L.1470: Greenhouse Gas Emissions Trajectories for the Information and Communication Technology Sector Compatible with the UNFCCC Paris Agreement. 2020. Available online: https://www.itu.int/ITU-T/recommendations/rec.aspx?id=14084 (accessed on 28 May 2026).
41. International Telecommunication Union. Recommendation ITU-T L.1480: Enabling the Net Zero Transition–Assessing How the Use of Information and Communication Technology Solutions Impacts Greenhouse Gas Emissions of Other Sectors. Recommendation ITU-T L.1480 (07/2025), in Force. 2025. Available online: https://www.itu.int/rec/T-REC-L.1480-202507-I (accessed on 28 May 2026).
42. International Telecommunication Union. ITU-T Study Group 5: Environment, Climate Action, Circular Economy and Electromagnetic Fields. 2026. Available online: https://www.itu.int/en/ITU-T/studygroups/2025-2028/05/Pages/default.aspx (accessed on 28 May 2026).
43. Patzer, F.; Volz, F.; Usländer, T.; Blöcher, I.; Beyerer, J. The industrie 4.0 asset administration shell as information source for security analysis. In *Proceedings of the 2019 24th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA), Zaragoza, Spain, 10–13 September 2019*; IEEE: New York, NY, USA, 2019; pp. 420–427.
44. Rose, S.; Borchert, O.; Mitchell, S.; Connelly, S. Zero Trust Architecture. Special Publication 800-207, National Institute of Standards and Technology, 2020. Available online: https://csrc.nist.gov/pubs/sp/800/207/final (accessed on 22 June 2026).
45. Williams, T.J. The Purdue enterprise reference architecture. *Comput. Ind.* **1994**, *24*, 141–158. [CrossRef]
46. Dolev, D.; Yao, A. On the security of public key protocols. *IEEE Trans. Inf. Theory* **1983**, *29*, 198–208. [CrossRef]
47. Abbas, S.G.; Ozmen, M.O.; Alsaheel, A.; Khan, A.; Celik, Z.B.; Xu, D. SAIN: Improving ICS Attack Detection Sensitivity via State-Aware Invariants. In *Proceedings of the 33rd USENIX Security Symposium (USENIX Security 24), Philadelphia, PA, USA, 11–13 August 2024*; Association for Computing Machinery: New York, NY, USA, 2024; pp. 6597–6613.

48. Ike, M.; Phan, K.; Sadoski, K.; Valme, R.; Lee, W. Scaphy: Detecting modern ics attacks by correlating behaviors in scada and physical. In *Proceedings of the 2023 IEEE Symposium on Security and Privacy (SP), San Francisco, CA, USA, 22–25 May 2023*; IEEE: New York, NY, USA, 2023; pp. 20–37.
49. Pickren, R.; Shekari, T.; Zonouz, S.; Beyah, R. Compromising industrial processes using web-based programmable logic controller malware. In Proceedings of the Network and Distributed System Security (NDSS) Symposium, San Diego, CA, USA, 26 February–1 March 2024.
50. Erba, A.; Tippenhauer, N.O. Assessing model-free anomaly detection in industrial control systems against generic concealment attacks. In *Proceedings of the 38th Annual Computer Security Applications Conference, Austin, TX, USA, 5–9 December 2022*; Association for Computing Machinery: New York, NY, USA, 2022; pp. 412–426.
51. Industrie4.0. Basyx Explained. Available online: https://wiki.basyx.org/en/latest/content/introduction/basyx_explained.html (accessed on 22 June 2026).
52. 3GPP. 3GPP TS 23.214: Architecture Enhancements for Control and User Plane Separation of EPC Nodes (Release 19). 2025. Available online: https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=3077 (accessed on 22 June 2026).
53. 3GPP. 3GPP TS 23.401: General Packet Radio Service (GPRS) Enhancements for Evolved Universal Terrestrial Radio Access Network (E-UTRAN) Access (Release 19). 2025. Available online: https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=849 (accessed on 22 June 2026).
54. 3GPP. 3GPP TS 23.501: System Architecture for the 5G System (5GS) (Release 19). 2025. Available online: https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=3144 (accessed on 22 June 2026).
55. Liatifis, A.; Sarigiannidis, P.; Argyriou, V.; Lagkas, T. Advancing SDN from OpenFlow to P4: A Survey. *ACM Comput. Surv.* **2023**, *55*, 1–37. [CrossRef]
56. Bonofiglio, G.; Iovinella, V.; Lospoto, G.; Di Battista, G. Kathará: A container-based framework for implementing network function virtualization and software defined networks. In Proceedings of the *NOMS 2018—2018 IEEE/IFIP Network Operations and Management Symposium, Taipei, China, 23–27 April 2018*; IEEE: New York, NY, USA, 2018; pp. 1–9.
57. The P4 Language Consortium. P4Runtime Specification, Version 1.5.0. 2026. Available online: https://p4lang.github.io/p4runtime/spec/v1.5.0/P4Runtime-Spec.html (accessed on 22 June 2026).
58. Aheleroff, S.; Xu, X.; Zhong, R.Y.; Lu, Y. Digital Twin as a Service (DTaaS) in Industry 4.0: An Architecture Reference Model. *Adv. Eng. Inform.* **2021**, *47*, 101225. [CrossRef]
59. Liu, M.; Fang, S.; Dong, H.; Xu, C. Review of Digital Twin About Concepts, Technologies, and Industrial Applications. *J. Manuf. Syst.* **2021**, *58*, 346–361. [CrossRef]
60. Xu, H.; Wu, J.; Pan, Q.; Guan, X.; Guizani, M. A survey on digital twin for industrial internet of things: Applications, technologies and tools. *IEEE Commun. Surv. Tutor.* **2023**, *25*, 2569–2598. [CrossRef]
61. Munir, M.; Siddiqui, S.A.; Chattha, M.A.; Dengel, A.; Ahmed, S. FuseAD: Unsupervised anomaly detection in streaming sensors data by fusing statistical and deep learning models. *Sensors* **2019**, *19*, 2451. [CrossRef] [PubMed]
62. Ruff, L.; Kauffmann, J.R.; Vandermeulen, R.A.; Montavon, G.; Samek, W.; Kloft, M.; Dietterich, T.G.; Müller, K.R. A Unifying Review of Deep and Shallow Anomaly Detection. *Proc. IEEE* **2021**, *109*, 756–795. [CrossRef]
63. Liu, F.T.; Ting, K.M.; Zhou, Z.H. Isolation Forest. In *Proceedings of the 2008 Eighth IEEE International Conference on Data Mining, Pisa, Italy, 15–19 December 2008*; IEEE: New York, NY, USA, 2008; pp. 413–422. [CrossRef]
64. International Telecommunication Union. ITU-T Study Group 13: Future Networks and Emerging Network Technologies. Study Period 2025–2028. 2026. Available online: https://www.itu.int/en/ITU-T/studygroups/2025-2028/13/Pages/default.aspx (accessed on 28 May 2026).