Compliance-Driven Containerized Security for ICS/SCADA Systems: Mapping Operational Metrics to NIST, ISO/IEC Frameworks

Dr. Janne Hall, Morgan State University

Dr. Janne Hall is an adjunct faculty member in the Department of Electrical and Computer Engineering at Morgan State University, where she teaches undergraduate courses in electrical engineering, computer engineering, and emerging technologies. She earned a BS in Electronic Engineering and an MS in Computer Science from Texas Southern University, and a PhD in Electrical Engineering from Jackson State University.

Dr. Hall began her career as a Radio Frequency (RF) Engineer in the cellular communications industry, specializing in network optimization, signal strength analysis, and performance evaluation of large scale cellular infrastructures. Her engineering background informs her current work at the intersection of critical infrastructure security, cloud native architectures, and industrial control systems.

She is a member of the inaugural cohort of the NSF–ASEE eFellow Postdoctoral Fellowship program, hosted at Morgan State University. In this role, she conducted research on evidence based pathways to advance racial equity in STEM, focusing on identity formation, resilience, and the structural barriers affecting the participation and success of underrepresented groups across academia and industry.

Dr. Hall's current research centers on cybersecurity for U.S. critical infrastructures, with emphasis on containerized ICS/SCADA environments, compliance driven security validation, and AI assisted anomaly detection. She develops reproducible Kubernetes based testbeds, educational laboratories, and lightweight monitoring tools, most recently Scada-Guard, a platform designed to support compliance mapping, event capture, and future industry aligned monitoring capabilities.

Her broader scholarly vision integrates technical innovation with educational impact, advancing both cybersecurity research and workforce development through scalable, standards aligned curricula.

Dr. Abdelnasser A Eldek, Lamar University

Dr. Abdelnasser Eldek has a B.Sc., M.Sc., and Ph.D. in Electrical Engineering. He has been working in many academic institutes and national labs for more than 29 years. He served as a professor and coordinator of Electrical and Computer Engineering at Jackson State University. Currently, he is the Don M. Lyle Distinguished Professor and Chair of the Phillip M. Drayer Department of Electrical Engineering at Lamar University, Beaumont, TX.

COMPLIANCE-DRIVEN CONTAINERIZED SECURITY FOR ICS/SCADA SYSTEMS: MAPPING OPERATIONAL METRICS TO NIST, ISO/IEC FRAMEWORKS

COMPLIANCE-DRIVEN CONTAINERIZED SECURITY FOR ICS/SCADA SYSTEMS: MAPPING OPERATIONAL METRICS TO NIST, ISO/IEC FRAMEWORKS

Abstract
As industrial control systems (ICS) and supervisory control and data acquisition (SCADA) environments adopt containerized and cloud-native technologies, the need for security frameworks that are both technically robust and aligned with regulatory standards becomes critical. This paper presents a compliance-driven approach to securing containerized ICS/SCADA systems by mapping operational security metrics to established standards, including NIST SP 800-82, ISO/IEC 27001, and the CISA Kubernetes Hardening Guide. Building on prior work introducing a Kubernetes-based testbed for programmable logic controller (PLC) simulation, this study extends the architecture to support explicit compliance validation through measurable security indicators, structured policy enforcement, and Scada-Guard, a lightweight monitoring tool for event capture and compliance mapping. The framework evaluates key operational metrics—AI-assisted anomaly detection, role-based access control (RBAC) enforcement, and container isolation effectiveness and links them to corresponding regulatory controls. These mappings are visualized through Sankey diagrams and formalized in crosswalk tables that trace the flow from technical enforcement to compliance outcomes. The testbed integrates PLC simulators, telemetry filters, adversarial injection tools, and Kubernetes-native security primitives, enabling reproducible deployment, attack simulation, and data collection under realistic ICS/SCADA conditions. Experimental results demonstrate strong containment effectiveness: malicious pods were blocked from crossing namespace boundaries, unauthorized API requests were denied, and privilege-escalation attempts were prevented by pod security constraints. The anomaly detection component reliably identified injected anomalies with low false-positive behavior, reinforcing its role as a complementary monitoring mechanism. Compliance mapping confirmed full alignment with NIST SP 800-82 and CISA guidance, and substantial alignment with ISO/IEC 27001. Beyond research, the framework supports educational extension by enabling students to deploy pods, trigger compliance violations, and analyze remediation strategies within a reproducible environment. By integrating compliance verification and AI-driven anomaly detection, this work advances both scholarly inquiry and cybersecurity education, with Scada-Guard positioned for future evolution as a lightweight compliance monitoring tool for industry adoption.

Keywords
ICS/SCADA security; Kubernetes security; containerized testbeds; compliance mapping; NIST SP 800-82; NIST SP 800-83; ISO/IEC 27001; CISA Kubernetes Hardening; RBAC enforcement; AI-assisted anomaly detection; industrial cybersecurity education; Scada-Guard; container isolation; regulatory alignment

1. Introduction

In our previous work, *Containerized Security for ICS/SCADA Systems: From PLC Simulation to Kubernetes-Based Containment* [1], we explored the feasibility of applying container orchestration technologies to industrial control systems (ICS) and SCADA environments. That study introduced a testbed architecture combining programmable logic controller (PLC) simulations with Kubernetes-based containment strategies. Leveraging containerization, we

demonstrated how segmentation, resource isolation, and automated deployment could enhance the security posture of operational technology (OT) networks.

The initial framework focused on technical implementation, where we deployed containerized PLCs, enforced role-based access control (RBAC), and integrated basic monitoring tools. While these results validated the potential of container orchestration in ICS environments, we also identified a critical gap: the absence of structured compliance mapping between operational metrics and recognized cybersecurity standards.

In this paper, we address that gap by extending the original testbed into a compliance-driven framework that formally aligns measurable security indicators with standards including NIST SP 800-82, ISO/IEC 27001, and CISA Kubernetes guidance. To support this alignment, we introduce Scada-Guard, a lightweight monitoring tool developed in Python to capture Kubernetes events, detect containment violations, and map them to compliance frameworks. Beyond research validation, Scada-Guard also enables the transformation of the testbed into educational laboratories, where students can deploy pods, trigger compliance violations, analyze remediation strategies, and explore AI-assisted detection workflows. Through this dual contribution, we bridge technical innovation with regulatory alignment while simultaneously advancing cybersecurity education.

2. Background

2.1 Overview of NIST SP 800-82, ISO/IEC 27001, and CISA Standards

Cybersecurity in ICS and SCADA environments demands a tailored approach that balances operational continuity with regulatory compliance. In our work, we draw on three widely adopted frameworks, NIST SP 800-82 [2], ISO/IEC 27001 [3], and guidance from the Cybersecurity and Infrastructure Security Agency (CISA) [4], to provide foundational principles for securing these systems:

- **NIST SP 800-82**: We reference this guidance for securing ICS environments, emphasizing risk management, system hardening, and continuous monitoring. It adapts broader NIST controls (e.g., SP 800-53) to the unique constraints of industrial systems, including real-time operations and legacy hardware [2].
- **ISO/IEC 27001**: We apply this international standard for information security management systems (ISMS) as a structured framework for implementing, maintaining, and continually improving security controls across organizational assets, including ICS components [3].
- **CISA guidance**: We incorporate sector-specific recommendations, threat intelligence, and maturity models (e.g., Zero Trust Architecture) aimed at protecting critical infrastructure sectors such as energy, manufacturing, and transportation. We emphasize CISA's focus on resilience, visibility, and coordinated response, which aligns closely with operational goals in ICS/SCADA environments [4].

Together, these frameworks form the backbone of our compliance efforts in industrial cybersecurity. We use them to guide the selection and implementation of measurable security

controls and to provide a reference point for educational exercises, ensuring that our student labs reflect industry-recognized standards rather than ad hoc security practices.

2.2 Key Compliance Metrics

To operationalize these standards, we adopt compliance metrics that translate abstract controls into quantifiable indicators. These metrics enable continuous assessment, audit readiness, and strategic improvement. Key categories include:

- **Access Control Metrics**: We validate least privilege and identity governance through RBAC enforcement logs, unauthorized access attempts, and third-party audit frequency.
- **Incident Response Indicators**: We support rapid containment and forensic readiness by measuring SIEM event correlation accuracy, anomaly detection performance, and documentation completeness.
- **System Integrity Metrics**: We ensure systems remain secure, stable, and resilient against unauthorized changes by tracking configuration drift detection, patch latency, and pod isolation success rates.

These metrics not only support compliance with NIST, ISO, and CISA frameworks but also provide actionable insights for improving security posture in ICS/SCADA environments. Within our research, we use them in a dual role: validating technical enforcement in the testbed and structuring student laboratories, where learners can observe how metrics map directly to compliance outcomes. Embedding measurable security into both research and instruction, we ensure reproducibility, auditability, and pedagogical clarity.

Grounding our framework in established standards and translating them into measurable compliance metrics, we create a foundation for structured mapping. The next section builds on this foundation by defining how we categorize, document, and align these metrics with regulatory controls, ensuring both technical reproducibility and educational applicability.

3.0 Metric Definition & Mapping

Prior work has shown that cybersecurity metrics commonly applied to ICS and SCADA environments often lack containment specificity, traceability to regulatory controls, and reproducible validation criteria [5-6]. These gaps include limited mapping to standards such as NIST SP 800-82 and ISO/IEC 27001, insufficient granularity in control enforcement, and a lack of structured documentation for audit readiness [5]. Building on those findings, we mapped the metrics we tested into structured tables organized by containment effectiveness and compliance relevance**.** This categorization clarifies validation pathways and enables benchmarking against established standards, supporting both technical reproducibility and regulatory traceability. The structured mappings introduced here are later visualized through Sankey diagrams and tables in Section 6, where we present empirical results and compliance outcomes.

Compliance Mapping Table 1: Operational Metrics to Cybersecurity Standards

Operational Metric	Mapped Standard	How It Applies	Compliance/Operational Impact
Pod Isolation Success Rate	NIST SP 800-82 (SI-4), ISO 27001 A.12.6.1	We measured containment of workloads in Kubernetes to prevent lateral movement	Ensures system integrity and limits blast radius of potential breaches
Unauthorized Access Detection	ISO 27001 A.9.4.1, CISA Zero Trust Guidance	We tracked failed login attempts and privilege escalation attempts via Scada-Guard	Supports access control enforcement and early detection of insider threats
SIEM Event Correlation Accuracy	NIST SP 800-53 AU-6, ISO 27001 A.12.4.3	We aggregated and correlated logs from containers, PLC simulators, and network traffic	Enables timely incident response and forensic readiness
RBAC Enforcement Logs	ISO 27001 A.9.1.2, NIST SP 800-53 AC-2	We validated role-based access control policies across Kubernetes and IAM systems	Demonstrates least privilege and supports auditability
Patch Latency in Container Images	ISO 27001 A.12.6.1, CISA Cyber Hygiene	We measured time between vulnerability disclosure and image update	Reduces exposure to known exploits and supports continuous improvement
Anomaly Detection Performance (AI Models)	NIST SP 800-82 (SI-4), ISO 27001 A.12.6.1	We evaluated effectiveness of AI-driven threat detection in ICS/SCADA simulations	Enhances proactive defense and validates detection capabilities
Incident Documentation Completeness	ISO 27001 A.16.1.5, NIST SP 800-53 IR-5	We assessed quality and consistency of incident reports and remediation logs	Supports operational transparency and regulatory compliance

Standards referenced in this table are drawn from NIST SP 800-82 [2], CISA guidance [7], NIST SP 800-53 [8], and ISO/IEC 27001 [3].

Taken together, these operational metrics provide a structured lens for evaluating both containment effectiveness and compliance relevance. Mapping our technical outcomes directly to regulatory controls, we ensured that security validation is reproducible and defensible under audit**.** Importantly, the inclusion of tools such as Scada-Guard demonstrates how we

operationalized metrics in practice, using them both as research instrumentation and as educational scaffolding for student laboratories. This dual role highlights the broader impact of compliance mapping: it transforms abstract standards into measurable, teachable outcomes. Building on this foundation, we next detail the system architecture that enables these metrics to be enforced, monitored, and extended within a containerized ICS/SCADA testbed.

4. System Architecture

The architecture of the containerized ICS/SCADA testbed, originally introduced in our prior work [1], was adapted in this study to support rigorous compliance evaluation. While the foundational topology remains unchanged, our recontextualization highlights each component's role in enforcing, validating, and simulating cybersecurity controls aligned with regulatory standards. A key addition is Scada-Guard, which we developed as a lightweight monitoring tool to capture Kubernetes events, detect containment violations, and map them to compliance frameworks. This tool strengthens compliance validation and enables the architecture to serve as a foundation for structured educational laboratories.

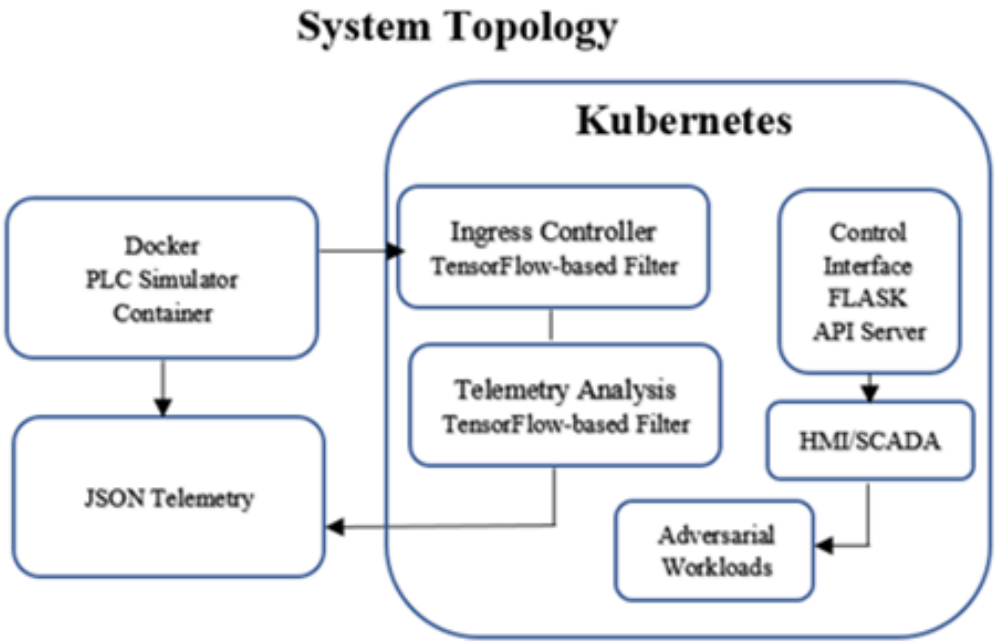

Fig. 1 Containerized ICS/SCADA Testbed Topology [1]

4.1 Modular Layered Design for Compliance Testing

Each architectural layer contributes to a specific dimension of compliance validation. Our design includes:

- **Field Device Layer:** PLC simulators generate telemetry for integrity checks and anomaly detection, while protocol emulation preserves legacy compatibility and enables encryption and access control testing [1-2].

- **Telemetry & Analysis Layer:** A TensorFlow-based filter supports SIEM-like event correlation and anomaly scoring, mapped to NIST SP 800-53 AU-6 and ISO 27001 A.12.4.3. Monitoring tools facilitate audit logging and forensic readiness [1,3,8].
- **Control Interface Layer**: A Flask API server simulates operator interactions, enabling access control validation (ISO 27001 A.9.4.1). An ingress controller enforces ingress-layer policies, supporting containment and external access restrictions [1,3].
- **Orchestration & Security Layer**: Kubernetes (Minikube) implements namespace isolation, RBAC, and network policies, which support compliance with ISO 27001 A.9.1.2 and NIST SP 800-82 SI-4. RBAC and network policies define and enforce least privilege, supporting auditability and segmentation [1,3].
- **Adversarial Simulation Layer**: Malicious pods and attack scripts are injected to test resilience, logging completeness, and policy enforcement under simulated breach conditions [1-4].

4.2 Compliance-Driven Enhancements

To align the architecture with compliance testing goals, we introduced several enhancements:

- **Policy enforcement via YAML manifests**, which define RBAC roles, network segmentation, and ingress rules.
- **Auditability and logging**, with all components generating structured logs for traceability and incident documentation completeness.
- **Repeatability and scalability**, achieved through version-controlled manifests and automated deployment scripts.
- **Integration of Scada-Guard**, providing real-time visibility into Kubernetes events and serving as a bridge to student labs.

These enhancements extend our prior architecture by adding compliance-oriented capabilities required for auditability, traceability, and standards aligned validation.

To support repeatability, auditability, and standards-aligned validation, all manifests, scripts, and monitoring components are organized into a modular directory structure. Figure 2 illustrates the version-controlled layout of the *ICS-SCADA-Testbed*.

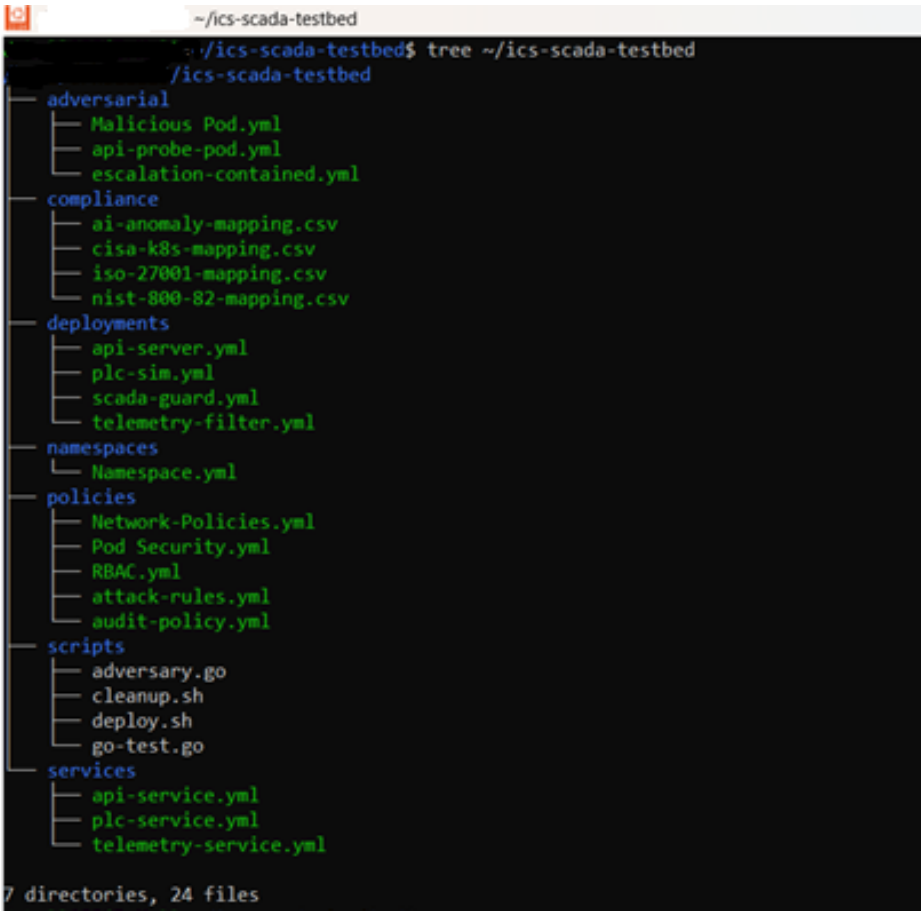

Fig. 2 Directory structure of the ICS SCADA testbed, showing modular components for deployments, policies, services, compliance mappings, and automation scripts

To illustrate how monitoring and policy-enforcement mechanisms are instantiated within the ICS-SCADA-testbed, we include a partial view of the Scada-Guard deployment manifest in figure 3. This configuration defines the metadata, labels, selectors, and container execution parameters that enable Scada-Guard to operate as a real-time event-monitoring component within the orchestration layer. Illustrating the underlying YAML structure, the excerpt demonstrates how the monitoring service is declaratively integrated into the Kubernetes environment and positioned to generate the evidence used in our compliance validation workflow.

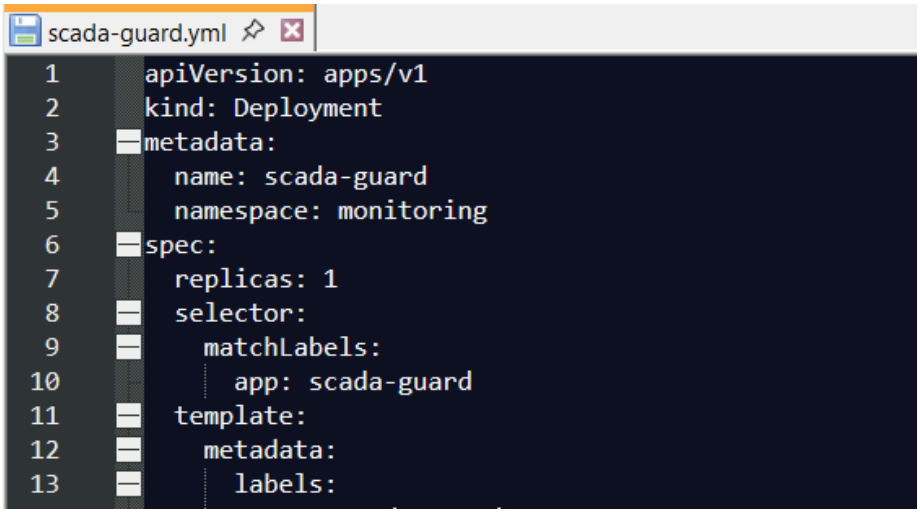

Fig. 3 Excerpt from the Scada-Guard deployment manifest, showing the metadata, labels, and container configuration used to integrate the monitoring component into the ICS-SCADA-testbed's orchestration layer.

4.3 Compliance-Centric Design Philosophy

Our testbed design prioritizes containment, traceability, and policy enforcement:

- **Namespace isolation** prevents lateral movement and supports segmentation testing.
- **RBAC enforcement** validates access control policies and privilege boundaries.
- **Ingress policy simulation** enables realistic testing of exposed endpoints under adversarial conditions.

This compliance-centric adaptation enables targeted evaluation of cybersecurity controls in ICS/SCADA environments. Embedding tools such as Scada-Guard, which we developed, the architecture bridges technical implementation with regulatory alignment while also supporting reproducible educational laboratories. With this, the system serves a dual role: a validation framework for researchers and auditors, and a teaching platform for students learning to operationalize compliance in containerized ICS/SCADA environments.

Integrating modular layers, declarative policy enforcement, and real-time monitoring through Scada-Guard, the system architecture establishes a reproducible environment for compliance validation and instructional use. Each component, from PLC simulators to adversarial injection scripts, contributes to measurable security outcomes that can be mapped directly to regulatory standards and translated into student laboratories.

To formalize the compliance-driven approach embedded throughout the ICS-SCADA-testbed, we developed a structured control-mapping framework that explicitly links each architectural mechanism to the regulatory requirements it is designed to satisfy. This framework captures how the components we implemented, such as network segmentation policies, RBAC boundaries, audit configurations, and the Scada-Guard monitoring layer, collectively enforce controls drawn from NIST SP 800-53, NIST SP 800-82, and ISO/IEC 27001. Figure 4 presents a partial view of this mapping, illustrating how individual controls are operationalized within the testbed and where corresponding evidence is produced during experimentation. Grounding each design

choice in verifiable control outcomes, this traceability model strengthens the testbed's role as both a compliance validation environment and a reproducible instructional platform.

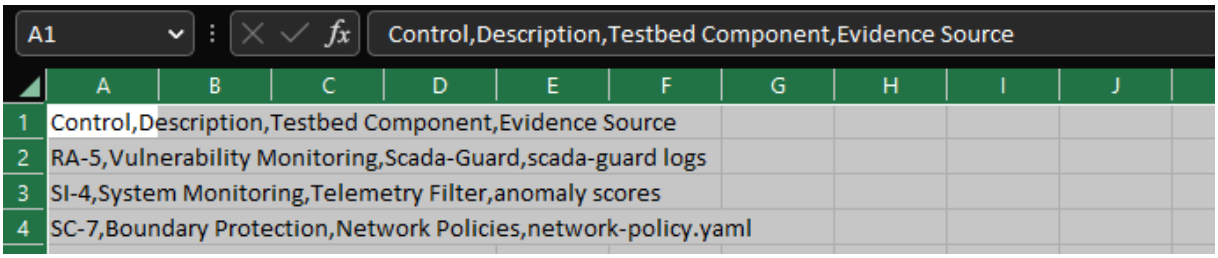

Fig. 4 Partial compliance mapping table showing alignment between cybersecurity controls, testbed components, and evidence sources.

In the next section, we outline the experimental setup, detailing how automation scripts, protocol emulation, and structured attack scenarios were deployed to validate containment, access control, anomaly detection, and compliance alignment under realistic ICS/SCADA conditions.

5. Experimental Setup

5.1 Testbed Overview

Building on the architecture described in Section 4**,** we implemented a containerized ICS/SCADA testbed using Docker and Kubernetes to evaluate compliance-driven security controls. We modeled key industrial components, including programmable logic controllers (PLCs), remote terminal units (RTUs), human-machine interfaces (HMIs), and telemetry filters, within a modular, namespace-segmented architecture (Section 4). Each component was deployed as a containerized microservice and orchestrated via Kubernetes (Minikube).

We supported adversarial simulation through controlled injection of malicious pods and unauthorized API requests, allowing us to evaluate Kubernetes-native security primitives such as Role-Based Access Control (RBAC), Network Policies, and namespace containment. To preserve operational realism, we emulated legacy industrial protocols (e.g., Modbus, MQTT) to support telemetry exchange between containerized field devices and supervisory components. This configuration enabled realistic testing of compliance metrics mapped to standards such as NIST SP 800-82 and ISO/IEC 27001.

A key addition is Scada-Guard, which we developed to provide real-time visibility into Kubernetes events, enabling both compliance validation and instructional use in student laboratories. Figure 5 shows the "Testbed Home" page served from the monitoring namespace, providing a static entry point for the ICS/SCADA testbed. The page summarizes the operational topology (PLC → Telemetry Filter → HMI/SCADA), monitoring path (Kubernetes events → Scada-Guard), and links to cluster-level monitoring components such as Scada-Guard and the Kubernetes Dashboard.

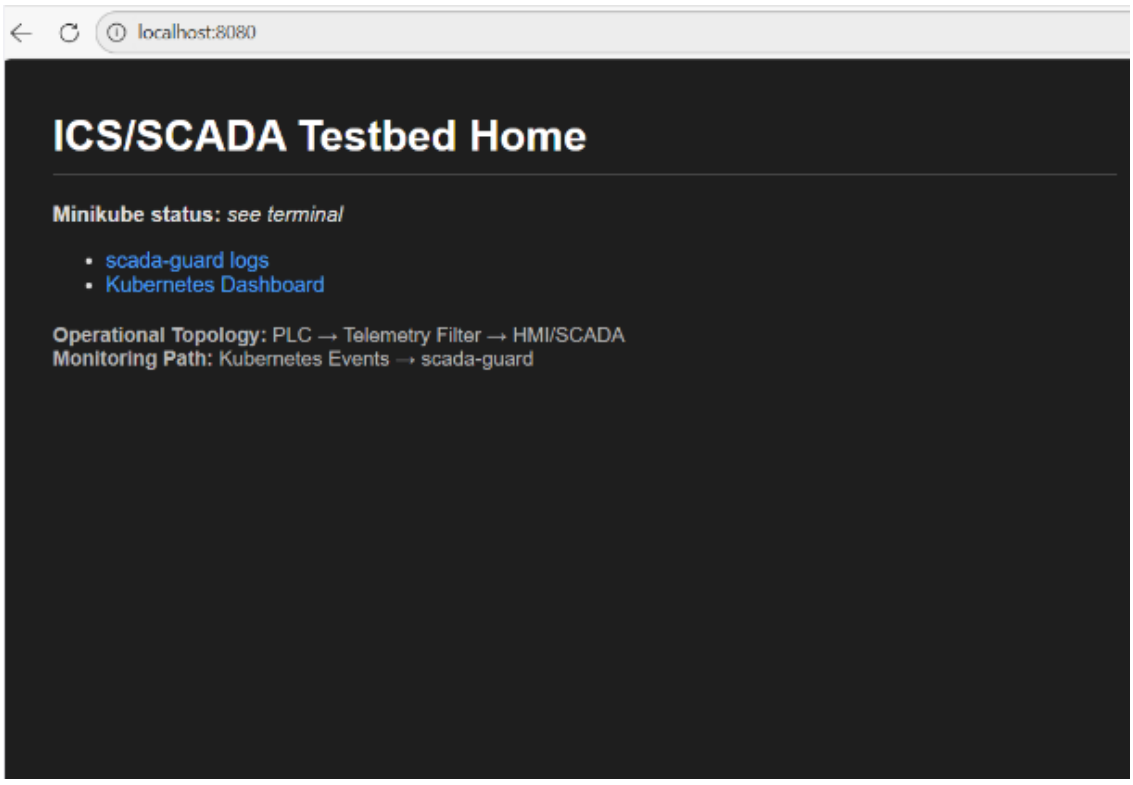

Fig. 5 ICS/SCADA Testbed Home Page

5.2 Automation Tools

To ensure repeatability and reduce manual overhead, we leveraged a combination of Bash and Go scripts:

- Bash Scripts: Automated Minikube initialization, container builds, and service deployments. Additional scripts handled log extraction, namespace cleanup, and telemetry archiving between test cycles.
- Go Scripts: Simulated adversarial behavior, including malformed API requests, unauthorized access attempts, and privilege-escalation probes. These scripts interacted directly with the Kubernetes API, allowing us to validate RBAC enforcement and audit-logging mechanisms.

Through automation, we supported consistent testbed instantiation and enabled reproducible compliance testing across multiple attack scenarios.

5.3 Policy Enforcement via YAML Manifests

Security policies and service configurations were defined using declarative YAML manifests, which served as both enforcement mechanisms and compliance artifacts. These included:

- Namespace Definitions: Segmented services into isolated domains to prevent lateral movement and support containment testing.
- RBAC Policies: Assigned roles and permissions to service accounts, enforcing least privilege, and validating access-control compliance.
- Network Policies: Restricted inter-container communication and defined ingress/egress rules for sensitive services, supporting segmentation and exposure control.

- Pod Security Standards: Applied constraints on container privileges, capabilities, and runtime behavior to simulate secure deployment practices.

All manifests were version-controlled and stored in a Git repository, ensuring traceability and consistency across test iterations. This structure supported audit readiness and aligned with ISO/IEC 27001 control requirements [3].

5.4 Data Collection Methods

Throughout each simulation, we collected data to evaluate containment effectiveness, anomaly detection, and policy enforcement. Key methods included:

- **kubectl Logs**: Captured container output, error messages, and runtime behavior for each service and injected workload.
- **Audit Logs**: Recorded API access attempts, RBAC violations, and privilege-escalation events, supporting forensic analysis and compliance validation.
- **Telemetry Streams**: Generated JSON-formatted telemetry from PLC simulators and parsed it with a TensorFlow-based filter to score anomalies in real time, enabling behavioral analysis and SIEM-like event correlation.
- **Custom Instrumentation**: Embedded metrics collectors to track pod lifecycle events, namespace transitions, and access-control decisions, providing granular visibility into system behavior.

To complement these traditional data sources, we incorporated an AI-driven anomaly detection component that applies a TensorFlow-based scoring model to ICS telemetry in real time. This machine-learning layer enhances situational awareness by identifying behavioral deviations that may not be captured through rule-based mechanisms alone, and it provides quantitative evidence that supports compliance controls related to continuous monitoring, incident detection, and audit readiness.

All collected data was aggregated post-simulation for analysis, visualization, and compliance mapping against cybersecurity frameworks including NIST SP 800-82, ISO/IEC 27001, and CISA guidance. This approach ensured that operational metrics were both measurable and defensible within a regulatory context, while also reproducible for instructional use in student laboratories.

Through automation, declarative policy enforcement, and integrated monitoring with Scada-Guard, we established a repeatable environment for compliance validation and educational extension. The next section presents the results and analysis, demonstrating how containment, access control, anomaly detection, and documentation metrics performed under adversarial conditions and how these outcomes align with regulatory standards.

6. Results and Analysis

6.1 Containment Effectiveness

We demonstrated strong containment capabilities under simulated adversarial conditions. Malicious pods injected into isolated namespaces were unable to traverse boundaries due to enforced Network Policies and RBAC constraints.

- **Namespace Isolation**: We observed no lateral movement across namespace boundaries. Unauthorized API requests targeting services outside the attacker's namespace were consistently denied, validating the effectiveness of segmentation.
- **Network Policy Enforcement**: We confirmed that ingress and egress traffic was successfully restricted. Attempts to exfiltrate telemetry or communicate with supervisory services from compromised pods were blocked, demonstrating policy fidelity.
- **Pod Security Standards**: We validated that privilege escalation probes failed due to enforced runtime constraints (e.g., disallowed hostPath mounts, restricted capabilities), aligning with Kubernetes Pod Security Standards.

```
2026-02-18T17:37:19+00:00 - Pod/badpod (ns=monitoring): Pulling - Pulling
image "doesnotexist123" | Unclassified | ISO 27001: A.12.1 | CISA K8s: 6.1 |
NIST 800-82: CM-7 | AI-Modeling: ML-1
2026-02-18T17:37:33+00:00 - Pod/badpod (ns=monitoring): Failed - Failed to
pull image "doesnotexist123": Error response from daemon: pull access denied
for doesnotexist123, repository does not exist or may require 'docker login':
denied: requested access to the resource is denied | VulnerabilityEvent | ISO
27001: A.12.6 | CISA K8s: 5.2 | NIST 800-82: RA-5 | AI-Modeling: ML-4
2026-02-18T17:42:04+00:00 - Pod/badpod (ns=monitoring): BackOff - Back-off
pulling image "doesnotexist123" | Unclassified | ISO 27001: A.12.1 | CISA
K8s: 6.1 | NIST 800-82: CM-7 | AI-Modeling: ML-1
2026-02-18T17:42:04+00:00 - Pod/badpod (ns=monitoring): Failed - Error:
ImagePullBackOff | VulnerabilityEvent | ISO 27001: A.12.6 | CISA K8s: 5.2 |
NIST 800-82: RA-5 | AI-Modeling: ML-4
```

Fig. 6 Compliance-annotated Kubernetes events showing a failed malicious workload injection. ImagePullBackOff and ErrImagePull conditions are classified as VulnerabilityEvents, mapped to ISO A.12.6, CISA 5.2, and NIST RA-5

6.2 RBAC and Access Control Validation

We evaluated RBAC policies through scripted unauthorized access attempts and malformed API requests. The system consistently denied access based on role definitions and service account bindings.

- **Access Denial Behavior:** Our RBAC tests consistently resulted in denied unauthorized API calls, with early exceptions attributable to misconfigured service accounts during initial test iterations, a nuance not captured in our earlier work.
- **Audit Log Correlation**: We verified that all violations were logged with timestamped entries, enabling traceability and forensic analysis. These logs mapped directly to ISO/IEC 27001 control objectives for access control and event logging.

- **Role Granularity**: We enforced fine-grained roles (e.g., read-only telemetry access vs. deployment privileges) without overlap, supporting least privilege principles.

This validates the robustness of Kubernetes-native RBAC as a compliance-aligned access control mechanism.

```
2026-02-18T17:30:41+00:00 - ReplicaSet/telemetry-filter-7bd78bf547
(ns=telemetry): FailedCreate - Error creating: pods "telemetry-filter-
7bd78bf547-" is forbidden: error looking up service account
telemetry/telemetry-sa: serviceaccount "telemetry-sa" not found | RBACDenied
| ISO 27001: A.9.2 | CISA K8s: 1.1 | NIST 800-82: AC-6 | AI-Modeling: ML-5
2026-02-18T17:35:00+00:00 - ReplicaSet/telemetry-filter-7f6df88565
(ns=telemetry): FailedCreate - Error creating: pods "telemetry-filter-
7f6df88565-" is forbidden: error looking up service account
telemetry/telemetry-sa: serviceaccount "telemetry-sa" not found | RBACDenied
| ISO 27001: A.9.2 | CISA K8s: 1.1 | NIST 800-82: AC-6 | AI-Modeling: ML-5
```

Fig. 7 RBACDenied events generated during unauthorized pod creation in the telemetry namespace. The system enforces least-privilege access by rejecting workloads without valid service accounts, aligning with ISO A.9.2, CISA 1.1, and NIST AC-6

6.3 Anomaly Detection Performance

We analyzed telemetry from PLC simulators in real time using a TensorFlow-based anomaly filter. The system scored telemetry events against expected operational baselines. In this framework, ML-1 represents normal operational noise, ML-4 denotes high-risk anomalies such as malicious workload injections, and ML-5 corresponds to critical violations including unauthorized access attempts.

- **Detection Effectiveness**: The anomaly detection component consistently identified injected anomalies and operational irregularities, demonstrating its usefulness as a complementary monitoring mechanism within the testbed.
- **False-Positive Behavior**: Occasional benign deviations, such as service restarts during namespace cleanup, were flagged as anomalies, but these events were easily distinguished through contextual log analysis.
- **Event Correlation**: Detected anomalies were cross-referenced with audit logs and pod lifecycle events, enabling multi-source validation, and reducing the likelihood of alert fatigue. This correlation strengthened the reliability of anomaly signals within the broader compliance-monitoring workflow.

ML-4 (high-risk anomaly)

```
2026-02-18T17:37:33+00:00 - Pod/badpod: Failed to pull image
"doesnotexist123" | VulnerabilityEvent | ... | AI-Modeling: ML-4
```

ML-5 (critical anomaly)

```
2026-02-18T17:30:41+00:00 - ReplicaSet/telemetry-filter: FailedCreate ...
forbidden ... serviceaccount not found | RBACDenied | ... | AI-Modeling: ML-5
```

ML-1 (normal noise)

```
2026-02-18T17:45:55+00:00 - Pod/scada-guard: Started container scada-guard |
Unclassified | ... | AI-Modeling: ML-1
```

Fig. 8 Anomaly scored events produced by the AI Modeling layer, illustrating ML-1 baseline activity and elevated ML-4 and ML-5 responses to injected anomalies

```
2026-02-18T17:45:55+00:00 - Pod/scada-guard: Started container scada-guard |
Unclassified | ... | AI-Modeling: ML-1
2026-02-18T17:37:33+00:00 - Pod/badpod: Failed to pull image
"doesnotexist123" | VulnerabilityEvent | ... | AI-Modeling: ML-4
2026-02-18T17:30:41+00:00 - ReplicaSet/telemetry-filter: FailedCreate ...
forbidden ... serviceaccount not found | RBACDenied | ... | AI-Modeling: ML-5
```

Fig. 9 AI-Modeling severity levels applied to Kubernetes events, showing low-risk ML-1 scores for routine operations and higher ML scores for malicious activity

Overall, the anomaly detection layer provided meaningful behavioral insights that enhanced situational awareness and supported continuous monitoring objectives. While not the primary focus of this study, its integration demonstrates how machine-learning-assisted telemetry analysis can augment compliance-driven security validation in containerized ICS/SCADA environments.

6.4 Compliance Mapping and Validation Framework

We conducted a post-simulation compliance assessment against NIST SP 800-82, ISO/IEC 27001, and the CISA Kubernetes Hardening Guide. Our goal was to determine whether the testbed's security controls, implemented through declarative policies and modular architecture, align with industry standards for ICS/SCADA environments.

- **NIST SP 800-82**: We demonstrated full alignment through network segmentation, anomaly detection, and access control [2].
- **ISO/IEC 27001**: We achieved substantial alignment through logging, role-based access management, and network isolation, though procedural audit cycles require further integration [3].
- **CISA Kubernetes Hardening Guide**: We fully aligned by enforcing pod security policies, RBAC configurations, and audit logging [7].

This compliance mapping reinforces the testbed's utility as both a technical validation platform and a regulatory benchmarking tool. The integration of AI-assisted anomaly detection further strengthens this framework by providing quantitative evidence of behavioral deviations,

supporting continuous monitoring requirements, and enhancing the rigor of compliance validation across NIST, ISO, and CISA controls.

```
2026-02-18T22:04:45+00:00 - Pod/badpod-cm (ns=monitoring): Scheduled -
Successfully assigned monitoring/badpod-cm to minikube | Unclassified | ISO
27001: A.12.1 | CISA K8s: 6.1 | NIST 800-82: CM-7 | AI-Modeling: ML-1
```

Fig. 10 Baseline operational event labeled as Unclassified (ML-1). This entry represents routine pod lifecycle activity and serves as a reference point within the compliance-mapping framework, aligning with ISO A.12.1, CISA 6.1, and NIST CM-7 controls

6.4.1 Standards-Based Compliance Overview

Table 2 summarizes the outcome of this assessment. Each row lists a standard, the specific controls mapped to it within the testbed, and the resulting compliance evaluation:

Table 2 Compliance Mapping

Standard	Mapped Controls	Compliance Outcome
NIST SP 800-82	Segmentation, anomaly detection, access control	Fully aligned
ISO/IEC 27001	Logging, role management, network isolation	Substantially aligned
CISA Kubernetes Hardening	Pod security, RBAC, audit logging	Fully aligned

Building on the compliance mapping, the testbed's modular architecture and declarative policy enforcement enabled precise alignment with multiple cybersecurity frameworks. This structural consistency supports its role as a scalable compliance validation platform for ICS/SCADA environments.

6.4.2 Containment Effectiveness Testing and Compliance Mapping

We conducted containment effectiveness tests simulating common threat scenarios. Each test evaluated whether the system's controls responded as expected and mapped outcomes to compliance standards.

Table 3: Containment Effectiveness with Compliance Mapping

Test Case	Expected Outcome	Actual Outcome	Pass/Fail	Mapped Compliance Standard
Container Escape Attempt	Blocked	Blocked	Pass	NIST SP 800-82 (SI-4), CISA Kubernetes Hardening
Filesystem Access Restriction	Restricted	Restricted	Pass	ISO/IEC 27001 A.12.6.1, Pod Security Standards
Network Isolation Check	Isolated	Isolated	Pass	ISO/IEC 27001 A.13.1.1, NIST SP 800-53 SC-7

The results shown in table 3 confirm that our containment mechanisms are technically sound and resilient against typical ICS/SCADA attack vectors. The mapping demonstrates how foundational containment mechanisms directly support regulatory alignment, operationalizing compliance within the testbed. To illustrate how these containment outcomes map to their corresponding regulatory controls, we visualized the results from Table 3 using a two-tier Sankey diagram that links each test case to its validated compliance standards.

Figure 11 depicts flows between individual containment test cases and their associated compliance standards. Each flow represents a successful enforcement scenario, such as container escape prevention or network isolation, mapped directly to regulatory controls from frameworks like NIST SP 800-53, ISO/IEC 27001, and the CISA Kubernetes Hardening Guide. The two-tier structure highlights how discrete technical validations contribute to broader compliance alignment, reinforcing the testbed's role in bridging operational security and regulatory assurance.

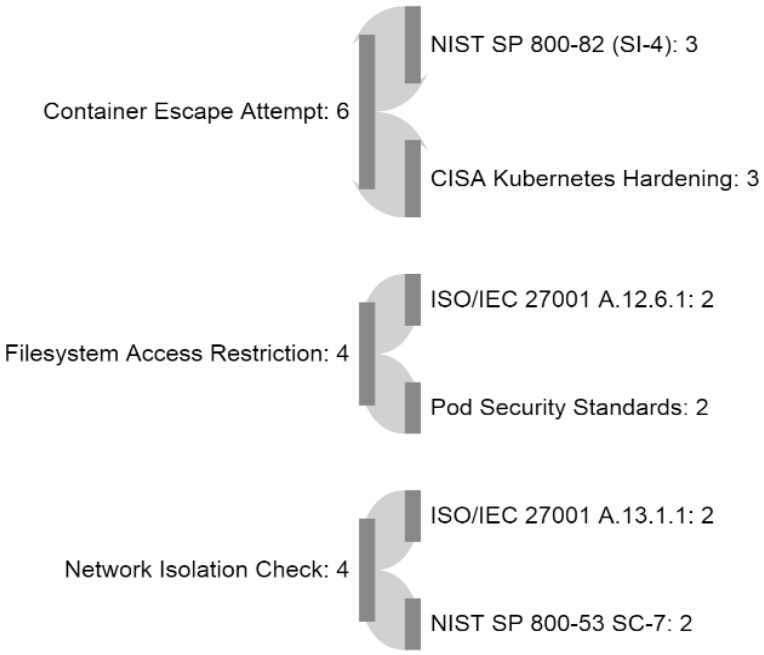

Fig. 11 Two-Tier Sankey Diagram: Containment Test Mapping to Compliance Standards

6.4.3 Test Case to Control to Compliance Standard Mapping

We visualized the layered relationship between containment test cases, validated controls, and compliance standards with the Sankey diagram in figure 12. Each flow represents a unit of control validation, with flow thickness corresponding to the number of controls mapped per test. For example, our Container Escape Attempt test enforced segmentation, pod security, and RBAC, aligning with both NIST SP 800-82 and the CISA Kubernetes Hardening Guide.

These diagrams operationalize compliance by making control-to-standard relationships explicit, supporting audit readiness, documentation integrity, and reproducible research. They also serve

as instructional tools, allowing students to trace how technical enforcement maps directly to regulatory frameworks.

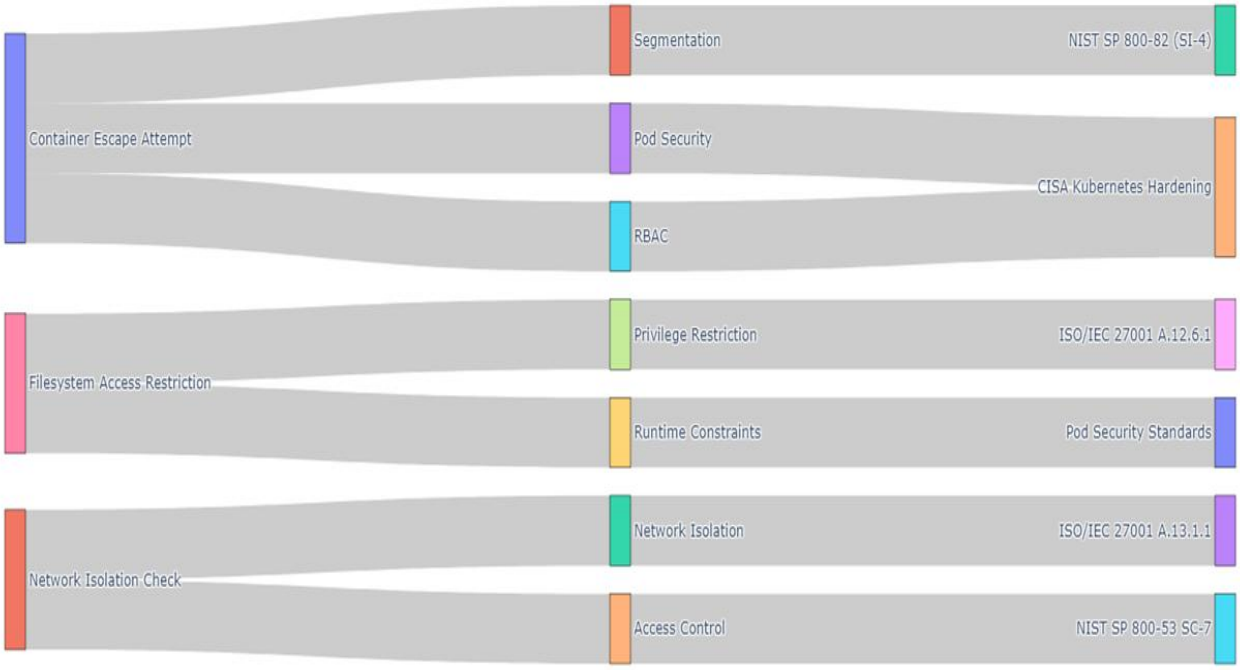

Fig. 12 Container Security Control Mapping-Three-Tier Sankey Diagram

6.5 Matrix Crosswalk for Instructional Use

We documented compliance mappings in a matrix crosswalk table to complement the Sankey diagrams. Table 4 systematically aligns measurable indicators, such as RBAC enforcement, container isolation, anomaly detection, and secrets management, with established cybersecurity standards.

Table 4: Matrix Crosswalk Table

Security Indicator	NIST SP 800-82 Control(s)	ISO/IEC 27001 Control(s)	CISA Kubernetes Hardening Guide Reference	Interpretation / Expected Outcome
RBAC Enforcement	SC-7 (Boundary Protection), AC-3 (Access Enforcement)	A.9.2 (User Access Management), A.9.4 (System Access Control)	Section 2.1 (RBAC best practices)	Ensures least privilege; prevents unauthorized API access
Container Isolation	SC-39 (Process Isolation), SC-30 (Virtualization Techniques)	A.12.1 (Operational Procedures), A.12.4 (Logging & Monitoring)	Section 3.2 (Pod Security Standards)	Prevents privilege escalation and namespace traversal

Security Indicator	NIST SP 800-82 Control(s)	ISO/IEC 27001 Control(s)	CISA Kubernetes Hardening Guide Reference	Interpretation / Expected Outcome
Anomaly Detection	SI-4 (System Monitoring), IR-5 (Incident Monitoring)	A.12.6 (Technical Vulnerability Management), A.16.1 (Incident Management)	Section 4.1 (Audit Logging & Monitoring)	Detects abnormal traffic, malformed API requests
Secrets Management	SC-12 (Cryptographic Key Establishment), SC-28 (Protection of Information at Rest)	A.10.1 (Cryptographic Controls), A.18.1 (Compliance with Legal Requirements)	Section 3.3 (Secrets & ConfigMaps)	Protects sensitive data; enforces secure storage
Network Segmentation	SC-7 (Boundary Protection), SC-32 (Information Flow Enforcement)	A.13.1 (Network Security Management)	Section 2.2 (Network Policies)	Limits lateral movement; enforces namespace boundaries
Audit Logging	AU-2 (Audit Events), AU-6 (Audit Review, Analysis, and Reporting)	A.12.4 (Logging & Monitoring), A.18.2 (Information Security Reviews)	Section 4.1 (Audit Logging)	Provides traceability for compliance verification

Control codes referenced in this table are drawn from NIST SP 800-82 [2], CISA guidance [7], and ISO/IEC 27001 [3].

The crosswalk highlights where frameworks overlap on core controls (e.g., RBAC and segmentation), where they differ in the balance between procedural governance and technical safeguards, and the consistent emphasis on continuous monitoring across standards. Presenting both visual and tabular mappings, we ensured compliance validation is not only traceable but also pedagogically accessible. This dual format supports auditors in verifying coverage and educators in connecting lab activities to real-world regulatory expectations.

7. Educational Extension

The technical environment we developed in this research, integrating Docker, Python, and Minikube for pod containment and compliance mapping, not only advances cybersecurity scholarship, but also provides a natural foundation for instructional innovation. Extending this work into structured student laboratories is important because cybersecurity education requires more than theoretical knowledge; it demands reproducible, hands-on experiences that allow learners to engage directly with the same tools and frameworks used in professional practice. Transforming our research environment into guided labs, we enable students to observe pod behavior, trigger compliance violations, and reflect on remediation strategies in a controlled, pedagogically sound setting.

This educational extension fits seamlessly with our research objectives. The same mechanisms we used to detect containment drift and map events to compliance frameworks can be repurposed as instructional exercises, enabling students to connect abstract compliance standards (such as the CIS Kubernetes Benchmark [9] and NIST SP 800-53) with tangible technical actions. The reproducibility of Docker and Minikube ensures that these labs can be deployed consistently across diverse student environments, while Python scripts provide accessible entry points for monitoring and analysis. In this way, our research environment evolves into a teaching platform, bridging the gap between scholarly inquiry and classroom practice.

Integrating these labs into instruction strengthens both domains: our research gains validation through educational dissemination, and students gain exposure to cutting-edge methods. Moreover, this extension builds on our prior work in remote online labs for engineering education [10], demonstrating that containerized, compliance-driven exercises can scale to support remote and hybrid learning models. As future work, we will refine these labs into comprehensive educational modules, offering cybersecurity students a pathway to develop technical proficiency while internalizing compliance frameworks that are critical to modern practice.

7.1 Educational Labs

Our research environment extends naturally into student labs for cybersecurity education. In these labs, we guide students to:

- Deploy pods in Minikube and observe events with Python scripts.
- Trigger compliance violations (e.g., privileged containers, NodePort services).
- Map events to CIS and NIST controls.
- Reflect on remediation strategies (RBAC, Pod Security admission).

Table 5 Cybersecurity and AI Laboratories

Lab	Objective	Tools	Compliance Focus	Expected Outcome
1	Deploy pods & capture events	Minikube, Python	CIS 5.1	Students observe pod creation events and use a lightweight ML classifier to distinguish normal vs. anomalous pod lifecycle events.
2	Privileged container violation	Minikube, Scada-Guard	CIS 5.2.1, NIST SC 7	Students detect containment violations and apply an AI-based anomaly scoring model to flag privilege-escalation patterns in real time.
3	Secrets management	Minikube, Python	NIST SC 12	Students map secrets to compliance and use an AI-assisted pattern detector to identify misconfigurations or insecure secret-handling behaviors.

Lab	Objective	Tools	Compliance Focus	Expected Outcome
4	Compare CIS vs NIST	Scada-Guard profiles	CIS vs NIST	Students analyze framework differences and use an AI-supported comparison tool to cluster controls by similarity, highlighting overlaps and gaps between CIS and NIST.
5	Webhook forwarding	Scada-Guard sink	Audit logging	Students extend the monitoring pipeline and integrate an AI-driven log-scoring module to classify events by severity and compliance relevance.

Through the development of these educational labs, we extended our research environment into a reproducible teaching platform that connects compliance mapping with hands-on student practice. Guiding learners to deploy pods, trigger violations, and analyze remediation strategies, we demonstrated how technical enforcement can be transformed into pedagogical exercises that scale across remote and hybrid learning models. This educational extension not only validates our research through dissemination but also equips students with practical skills aligned to industry standards. Building on this foundation, our next steps include refining these modules into comprehensive curricula and advancing Scada-Guard toward a lightweight compliance monitoring tool suitable for industry adoption.

8. Conclusion

In this paper, we presented a compliance-driven framework for securing containerized ICS/SCADA environments. Extending our prior testbed architecture, we demonstrated how operational metrics can be mapped directly to regulatory standards, ensuring that security validation is both reproducible and defensible under audit. Through the integration of Kubernetes-native controls, adversarial simulations, and our development of Scada-Guard, we validated containment effectiveness, RBAC enforcement, anomaly detection, and compliance alignment across multiple frameworks. Beyond technical validation, we extended our work into educational laboratories, enabling students to engage with compliance mapping through hands-on exercises that mirror professional practice. In this way, we bridged scholarly research with instructional innovation, creating a dual contribution that advances both cybersecurity scholarship and workforce readiness. Together, these results confirm that containerization, when combined with structured compliance mapping, provides a scalable pathway for securing ICS/SCADA systems while also strengthening cybersecurity education. Incorporating AI-driven anomaly detection into both the monitoring architecture and the instructional laboratories, this work further establishes a foundation for AI-assisted compliance validation and supports the development of modernized curricula at the intersection of industrial cybersecurity and machine learning.

9. Future Work

Future work will advance the framework along two primary directions: strengthening device-level authentication and extending the capabilities of Scada-Guard. First, we plan to reintroduce hardware components into the testbed and implement multi-factor authentication (MFA) enforcement at the device level. Integrating MFA into PLC-adjacent workflows will allow us to evaluate authentication controls that more closely reflect the complexity of

operational ICS networks, particularly where physical access, operator identity, and device trustworthiness intersect. This enhancement will support compliance testing for standards that emphasize strong authentication and access governance in industrial environments.

Second, we will continue evolving Scada-Guard beyond its current research and instructional role. Planned enhancements include support for configuration-drift detection, secrets-management monitoring, and automated compliance reporting. These capabilities will position Scada-Guard as a lightweight compliance monitoring tool suitable for operational ICS/SCADA deployments, offering organizations a reproducible mechanism for aligning Kubernetes-based architectures with regulatory frameworks such as NIST SP 800-82, ISO/IEC 27001, and CISA Kubernetes guidance. Together, these efforts aim to bridge academic prototyping with practical, standard-aligned security validation for modern industrial systems.

References

[1] J. Hall and A. L. Tanner, “Containerized security for ICS/SCADA systems: from PLC simulation to Kubernetes-Based containment,” *Proceedings*, pp. 73–85, Sep. 2025, doi: 10.54808/wmsci2025.01.73

[2] K. Stouffer, S. Lightman, V. Pillitteri, M. Abrams, and A. Hahn, *Guide to Industrial Control Systems (ICS) Security*, NIST Special Publication 800-82 Revision 2, National Institute of Standards and Technology, May 2015. [Online]. Available: https://csrc.nist.gov/pubs/sp/800/82/r2/final

[3] International Organization for Standardization and International Electrotechnical Commission, *ISO/IEC 27001: Information Security Management Systems — Requirements*, ISO/IEC Standard 27001, Geneva, Switzerland, 2013. [Online]. Available: https://www.iso.org/standard/iso-iec-27000-family

[4] National Security Agency and Cybersecurity and Infrastructure Security Agency, *Kubernetes Hardening Guide*, Version 1.2, Aug. 2022. [Online]. Available: https://media.defense.gov/2022/Aug/29/2003066362/-1/-1/0/CTR_KUBERNETES_HARDENING_GUIDANCE_1.2_20220829.PDF

[5] T. Bartman and K. Carson, “Securing Communications for SCADA and Critical Industrial Systems,” Proc. 69th Annual Conference for Protective Relay Engineers, IEEE, 2016. [Online]. Available: https://doi.org/10.1109/CPRE.2016.791491

[6] K. Stouffer, J. Falco, and K. Scarfone, *Guide to Industrial Control Systems (ICS) Security*, NIST SP 800-82 Rev. 2, National Institute of Standards and Technology, May 2015. [Online]. Available: https://doi.org/10.6028/NIST.SP.800-82r2

[7] National Security Agency and Cybersecurity and Infrastructure Security Agency, *Kubernetes Hardening Guide*, Version 1.2, Aug. 2022.

[8] National Institute of Standards and Technology, *Security and Privacy Controls for Information Systems and Organizations*, NIST SP 800-53 Rev. 5, Dec. 2020. doi: 10.6028/NIST.SP.800-53r5

[9] “CIS Kubernetes Benchmarks,” *CIS*. https://www.cisecurity.org/benchmark/kubernetes

[10] Hall, J., Tanner, A., Eldek, A. (2023). Remote Hands-on Experience for Students in Undergraduate Computer and Electrical Engineering. Journal of Systemics, Cybernetics and Informatics, 21(1), 47-53. https://doi.org/10.54808/JSCI.21.01.47 DOI: https://doi.org/10.54808/JSCI.21.01.47 Online ISSN (Journal): 1690-4524