Operational Legitimacy of Industrial Control Actions: The Liscere Framework

Bruno Salmazo
Liscere
June 2026 · Liscere Technical Report · LTR-2026-02 · v1

Abstract

Industrial control protocols such as Modbus/TCP authenticate neither the sender nor the appropriateness of a command [1, 2]. An action can therefore be valid at the protocol level, issued over an authorised channel, and still be inappropriate for the operational state of the plant. This report presents the Liscere framework, a protocol-agnostic model for evaluating an industrial control action along four dimensions: the protocol operation, the automation artefact it affects, the operational context in which it occurs, and the applicable policy constraint. The framework reconstructs an observed protocol event as an industrial action, resolves the affected artefact, enriches the action with declared operational context, evaluates it against explicit policy, and returns a decision together with an evidence record. The model is defined above protocol-specific encodings, with Modbus/TCP as its first instantiation. An initial feasibility demonstration is reported separately [11]; this report sets out the framework, its definitions, and its scope. A peer-reviewed treatment is in preparation. We are explicit about the boundary between what the model defines and what the current implementation evaluates.

Keywords: operational technology security, industrial control systems, contextual action evaluation, operational legitimacy, policy-based decision, Modbus/TCP.

1. Introduction

Operational technology (OT) networks were historically isolated and optimised for availability and determinism rather than for security. Field protocols such as Modbus/TCP reflect that heritage: requests are unauthenticated and unencrypted, and any station able to reach a device may issue a well-formed command that the device will execute [1, 2]. As OT and IT networks converge, the assumption of a trusted perimeter no longer holds.

A recurring lesson from real incidents is that damage need not involve malformed traffic or protocol violations. Stuxnet manipulated controllers using legitimate-looking commands [4, 6], and process-aware attacks issue individually valid actions whose harm becomes apparent only at the level of the physical process [9]. Much of the intrusion-detection literature for industrial systems focuses on detecting anomalous or malformed traffic [7, 8, 10]. Comparatively little addresses a complementary question: given a well-formed and authorised action, is it appropriate for the current operational context?

This report makes the following contributions. It defines operational legitimacy and the problem of contextual industrial action evaluation (Section 2). It presents the Liscere action-evaluation chain and its four-dimension evaluation object (Section 3). It sets out the framework design, its principles, components, and decision logic (Section 4). It positions the framework relative to adjacent work (Section 5), and states its scope and limitations explicitly (Section 6). This is a conceptual exposition; the empirical feasibility demonstration is reported in LTR-2026-01 [11], and a peer-reviewed version is in preparation.

2. Problem Statement and Definitions

Operational technology systems interact with physical processes through controllers, actuators, supervisory applications, engineering workstations, and industrial communication protocols. In such environments, cyber operations are not merely information-processing events. They may change setpoints, trigger actuators,

modify controller configuration, alter process modes, or acknowledge alarms. The security meaning of these operations depends on the physical and operational context in which they occur.

Traditional network and protocol-level checks are necessary but insufficient. A message may be well formed, use an expected industrial protocol, and still be operationally inappropriate. Conversely, an unusual action may be legitimate under specific operating conditions such as commissioning, maintenance, recovery, or emergency handling. The central problem is therefore not only whether an action is valid at the protocol level, but whether it is contextually legitimate for the affected automation artefact.

Operational legitimacy means that an action is appropriate for the affected automation artefact under the current operational context and applicable policy constraints. We address the following question: given a protocol-level operation, the automation artefact it affects, the operational context in which it occurs, and the applicable policy constraints, should the resulting industrial action be allowed, alerted, denied, or escalated? We call this problem contextual industrial action evaluation.

The framework uses the following definitions.

An industrial action is a semantically reconstructed operation that may affect an industrial process, controller, automation artefact, or operational state. It may be carried by different protocol-level representations, including a Modbus register write, an OPC UA node write, a DNP3 operate command, or an IEC 60870-5-104 setpoint command. The action is not equivalent to the raw packet; the raw packet is evidence from which the action is reconstructed.

A protocol-level operation is the operation as represented by a specific industrial communication protocol. Protocol-level validity means that the operation conforms to the syntax and basic semantics of the carrier protocol. It does not imply operational legitimacy.

An automation artefact is the process, controller, or engineering object affected by an industrial action. Examples include process setpoints, actuator commands, alarm thresholds, interlocks, recipe parameters, operating mode selectors, safety-related thresholds, controller configuration variables, and unknown or unmapped registers, tags, or objects.

Operational context is the set of conditions that influence whether an industrial action is appropriate. Relevant dimensions may include process state, operating mode, actor role, timing, sequence position, maintenance state, artefact sensitivity, value semantics, mapping confidence, and evidence quality. The current Modbus/TCP baseline implements one declared context dimension, `maintenance_window`, and richer dimensions remain future work.

A policy constraint defines whether an industrial action is acceptable, suspicious, prohibited, or requires escalation under specified conditions. It may be derived from engineering limits, safety requirements, operational procedures, maintenance policies, actor-role permissions, site security policy, vendor documentation, process knowledge, formal specifications, or learned operational rules.

A decision output is the result of evaluating an industrial action against artefact, context, and policy constraints. The framework defines four outputs.

Decision	Meaning
`ALLOW`	The action is acceptable under the available artefact, context, and policy information.
`ALERT`	The action is suspicious, risky, unexpected, or policy-relevant and should be reported.
`DENY`	The action violates a policy constraint strongly enough that it should be blocked where enforcement is available.
`ESCALATE`	The action cannot be safely classified with the available evidence, or it affects a sensitive artefact requiring higher-level review.

Table 1: The four decision outputs defined by the framework.

The current implementation validates ALLOW and ALERT. DENY and ESCALATE are framework-level outputs and require separate enforcement or workflow semantics before they can be claimed as evaluated.

3. The Action-Evaluation Chain

Liscere evaluates industrial actions through a structured chain. Observed protocol events are reconstructed as industrial actions, resolved to affected automation artefacts, enriched with operational context, evaluated against policy constraints, and assigned an explicit decision with supporting evidence.

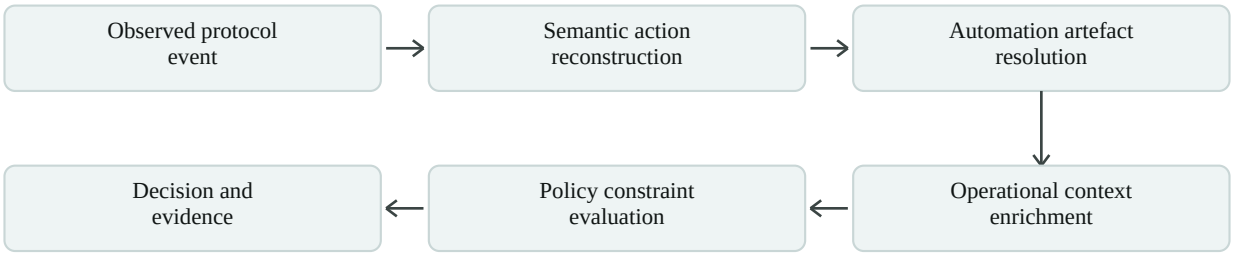

Figure 1: The Liscere action-evaluation chain. An observed protocol event is reconstructed as an industrial action, resolved to the affected automation artefact, enriched with declared operational context, evaluated against policy, and assigned a decision with an evidence record. The chain is protocol-agnostic, and Modbus/TCP is the first instantiation.

The conceptual evaluation unit relates four dimensions:

```
protocol operation  ×  automation artefact  ×  operational context  ×  policy constraint
                                              ↓
                                  decision + rationale + evidence
```

Three of these four dimensions describe the action: the protocol operation is the action at the wire level, the automation artefact is the logical asset it affects, and the operational context is the declared state of the plant. The fourth dimension, the policy constraint, evaluates the other three to yield a decision and an evidence record.

The chain is deliberately protocol-agnostic. Only the protocol-operation dimension is protocol-specific, and it is isolated behind a mapping from raw protocol fields to artefacts. Adding a protocol amounts to providing that mapping, leaving the context and policy machinery unchanged.

4. Framework Design

4.1 Design principles

Protocol-agnostic at the model level. The framework is defined above protocol-specific encodings. Industrial protocols are treated as carriers of industrial actions. Modbus/TCP is the first empirical baseline, not a claim of multi-protocol generality.

Artefact-centric semantics. The security meaning of an operation depends on the automation artefact it affects. A write is evaluated as a write to a setpoint, an actuator command, an alarm threshold, a configuration variable, a recipe parameter, or an unknown artefact.

Contextual legitimacy. An industrial action may be acceptable in one context and inappropriate in another. A configuration write may be acceptable during maintenance but suspicious during production; a setpoint change may be safe within engineering limits but policy-relevant outside them.

Policy-driven decisions. The framework evaluates declared policy constraints over reconstructed actions, artefacts, context, and evidence. Constraints may be derived from engineering limits, operational procedures, maintenance policies, documentation, process knowledge, formal specifications, access-control attributes, or learned operational rules.

Explainable decision outputs. Each decision should carry a label, the matched policy, the affected artefact, the context used, a rationale, and an evidence record. The baseline evaluates ALLOW and ALERT; DENY and ESCALATE require separate enforcement or workflow semantics.

4.2 Core components

The Protocol Event Observer captures protocol-level operations from packets, logs, command traces, proxy events, or parser outputs. The Semantic Action Reconstructor translates protocol events into candidate industrial actions by identifying operation type, target, value, and action class. The Automation Artefact Resolver maps the reconstructed action to the affected artefact using sources such as register maps, PLC tag databases, HMI configuration, engineering files, protocol object models, or scenario metadata; where no mapping is available, the artefact is classified as unknown. The Operational Context Enricher attaches context to the action, such as operating mode, actor role, timing, maintenance state, artefact sensitivity, value semantics, and mapping confidence; the current baseline implements only a declared `maintenance_window` value. The Policy Evaluation Engine evaluates policy constraints against the contextualised action. The Decision Generator produces the final decision and evidence record. The current instantiation runs in the OT Lab [12].

4.3 Decision logic

A simplified decision process is:

```
for each observed ProtocolEvent:
    reconstruct  IndustrialAction
    resolve      AutomationArtefact
    enrich with  OperationalContext
    select       applicable PolicyConstraints
    evaluate     constraints
    produce      ActionDecision
```

A deployment may return ALERT or ESCALATE for unknown mappings, ALERT or DENY for policy violations depending on enforcement availability, and ALLOW when all applicable constraints are satisfied. The current baseline validates ALLOW and ALERT.

5. Positioning

The framework sits alongside several established lines of work. Surveys of industrial anomaly and intrusion detection [8, 10] and introductions to industrial control networks [7] address whether traffic is anomalous or malformed. Process-aware analyses [9] reason about the effect of actions on the controlled process. Standards such as NIST SP 800-82 [5] and the IEC 62443 series [3] codify defence-in-depth, including zoning, monitoring, and least privilege, but do not themselves evaluate individual actions. Attribute-based access

control governs whether a subject may perform an action under given conditions.

Liscere offers a complementary framing. Rather than asking whether traffic is anomalous, or whether a subject is authorised, it asks whether an otherwise valid and authorised action is appropriate given a declared operational context, and it makes the decision, the matched policy, and the supporting evidence explicit. The framework does not claim novelty in semantic extraction, specification mining, protocol abstraction, access control, formal safe-action restriction, or process modelling. Its contribution is the explicit action-evaluation object and decision chain that combine artefact, context, policy, decision, and evidence around a reconstructed industrial action.

6. Scope and Limitations

The framework is protocol-agnostic at the model level, but only Modbus/TCP is instantiated; multi-protocol validation is future work. One declared operational-context dimension, `maintenance_window`, is implemented; richer context dimensions remain future work. ALLOW and ALERT are implemented and validated, while DENY and ESCALATE are framework-level and require separate enforcement or workflow semantics. The context is declared, not inferred. Policy rules and address-to-artefact mappings are defined manually in the current instantiation. The framework is observational and applies no production enforcement; it complements, and does not replace, protocol checks, anomaly detection, access control, and formal safe-action restriction within a defence-in-depth posture [3, 5]. The empirical feasibility demonstration is limited to a controlled process model and is reported in LTR-2026-01 [11]. A peer-reviewed treatment is in preparation.

7. Conclusion and Outlook

We presented the Liscere framework, a protocol-agnostic model that evaluates an industrial control action across four dimensions and returns an explicit decision with an evidence record. Modbus/TCP is the first instantiation, and an initial feasibility demonstration is reported separately [11]. Future work will broaden the context model beyond a single declared dimension, add further protocols through the protocol-operation mapping, explore inferred rather than declared context, implement and evaluate the DENY and ESCALATE outputs, and move from constructed scenarios to a quantitative evaluation with adversarial and false-positive analysis.

References

[1] Modbus Organization. Modbus Application Protocol Specification V1.1b3, 2012.

[2] Modbus Organization. Modbus Messaging on TCP/IP Implementation Guide V1.0b, 2006.

[3] International Electrotechnical Commission. IEC 62443-3-3:2013, Industrial communication networks, Network and system security, Part 3-3: System security requirements and security levels. IEC, 2013.

[4] N. Falliere, L. O. Murchu, and E. Chien. W32.Stuxnet Dossier, version 1.4. Symantec Security Response, 2011.

[5] K. Stouffer, M. Pease, C. Tang, T. Zimmerman, V. Pillitteri, S. Lightman, A. Hahn, S. Saravia, A. Sherule, and M. Thompson. Guide to Operational Technology (OT) Security. NIST Special Publication 800-82, Revision 3, 2023.

[6] R. Langner. Stuxnet: Dissecting a Cyberwarfare Weapon. IEEE Security & Privacy, 9(3):49-51, 2011.

[7] B. Galloway and G. P. Hancke. Introduction to Industrial Control Networks. IEEE Communications Surveys & Tutorials, 15(2):860-880, 2013.

[8] I. Garitano, R. Uribeetxeberria, and U. Zurutuza. A Review of SCADA Anomaly Detection Systems. In Soft Computing Models in Industrial and Environmental Applications (SOCO), Advances in Intelligent and Soft Computing, vol. 87, pages 357-366. Springer, 2011.

[9] A. A. Cárdenas, S. Amin, Z.-S. Lin, Y.-L. Huang, C.-Y. Huang, and S. Sastry. Attacks Against Process Control Systems: Risk Assessment, Detection, and Response. In ACM ASIACCS, pages 355-366, 2011.

[10] S. McLaughlin, C. Konstantinou, X. Wang, L. Davi, A.-R. Sadeghi, M. Maniatakos, and R. Karri. The Cybersecurity Landscape in Industrial Control Systems. Proceedings of the IEEE, 104(5):1039-1057, 2016.

[11] B. Salmazo. Context-Aware Evaluation of Industrial Control Actions: A Modbus/TCP Baseline in the OT Lab. Liscere Technical Report LTR-2026-01, 2026.

[12] Liscere. OT Lab: a controlled laboratory for industrial action evaluation. https://github.com/LiscereSecurity/OT-Lab, 2026.

Appendix A. The evaluation object, illustrated

The following illustrates how the model characterises a single action across its four dimensions. It is an illustration of the model, not a result; the empirical evaluation of such scenarios is reported in LTR-2026-01 [11].

Dimension	Value
Protocol operation	Write single register (function code `0x06`) to a mapped address.
Automation artefact	A high-alarm setpoint, exposed as the artefact `ALARM_HI_SP` and marked sensitive.
Operational context	The declared `maintenance_window`, true or false.
Policy constraint	A sensitive configuration write is permitted during a declared maintenance window and is otherwise reported.

Table 2: The evaluation object, illustrated across its four dimensions.

Under this model, the same write resolves to different decisions according to the declared context: the policy reports the write outside a maintenance window and permits it within one. The action is identical; the declared context is the only difference.