# Poster: Facilitating Protocol-independent Industrial Intrusion Detection Systems

Konrad Wolsing
Fraunhofer FKIE & RWTH Aachen University
konrad.wolsing@fkie.fraunhofer.de

Eric Wagner
Martin Henze
Fraunhofer FKIE
{firstname.lastname}@fkie.fraunhofer.de

## ABSTRACT

Cyber-physical systems are increasingly threatened by sophisticated attackers, also attacking the physical aspect of systems. Supplementing protective measures, industrial intrusion detection systems promise to detect such attacks. However, due to industrial protocol diversity and lack of standard interfaces, great efforts are required to adapt these technologies to a large number of different protocols. To address this issue, we identify existing universally applicable intrusion detection approaches and propose a transcription for industrial protocols to realize protocol-independent semantic intrusion detection on top of different industrial protocols.

## CCS CONCEPTS

• **Security and privacy → Intrusion detection systems**; • **Networks → Cyber-physical networks**; *Network monitoring*.

## KEYWORDS

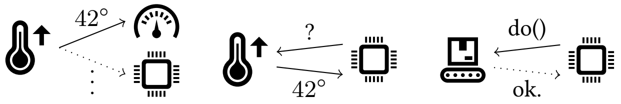
Intrusion Detection, IDS, Industrial Protocols, CPS, IEC-60870-5-104, Modbus, NMEA 0183

**ACM Reference Format:**
Konrad Wolsing, Eric Wagner, and Martin Henze. 2020. Poster: Facilitating Protocol-independent Industrial Intrusion Detection Systems. In *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security (CCS '20), November 9–13, 2020, Virtual Event, USA.* ACM, New York, NY, USA, 3 pages. https://doi.org/10.1145/3372297.3420019

## 1 MOTIVATION

Incidents such as Stuxnet [4] or the attack on the Ukrainian power grid [13] highlight the necessity to secure cyber-physical systems (CPS) against attacks, especially since these systems are progressively integrated to the Internet. Since existing industrial hardware can not be replaced or updated easily, network separation, firewalls, or VPNs promise to prevent such attacks. However, as sophisticated attackers still might circumvent protective security measures, intrusion detection systems (IDS) are important to detect remaining attacks.

Indeed, various detection mechanisms have been proposed for various industrial use cases. Especially in CPSs, IDSs can leverage physical process knowledge for anomaly detection since, e.g., the procedures for a given production plant are known. Thus, attacks can be detected by, e.g., modeling communication patterns with automata [2, 6, 9, 15], finding invariants for physical processes [5], detecting critical states with logic formulas [8], fingerprinting physical processes [1], comparing the current physical process state to a simulation [3], or analyzing packet time intervals [14]. While these mechanisms could, in principle, be widely applicable as they only depend on process variables or communication patterns, their implementations are usually confined to a specific industrial protocol.



**(a) Push** **(b) Request/response** **(c) Write or call**

**Figure 1: Industrial protocols communication behavior can be abstracted to these three scenarios.**

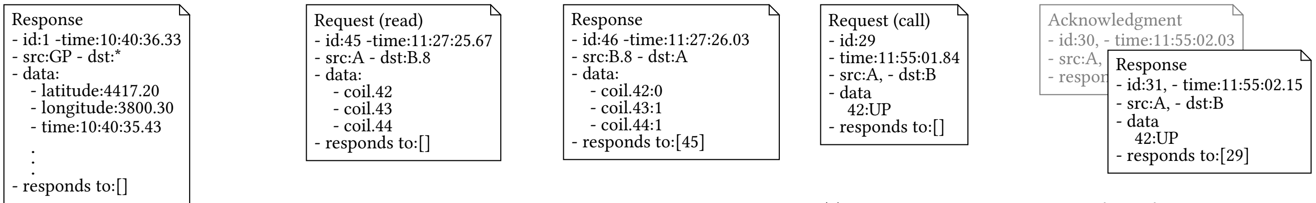
Historically, industrial protocols were designed for unique application requirements, resulting in a variety of special-purpose protocols. This diversity hinders applying IDS techniques to different protocols. As detection mechanisms typically only operate on communication patterns, or on the current physical state, we propose to transcribe industrial protocol messages into abstract information objects as the foundation for protocol-independent intrusion detection.

Optimally, a universal and widely deployed industrial protocol would obviate the need to adapt IDSs with a protocol transcription. But, since CPS communication requirements are largely diverse (due to varying constraints on devices, real-time requirements, or the need for authenticated or encrypted communications), such a general industrial protocol seems utopistic. Even if unifying the industrial protocol landscape becomes feasible in the future, IDSs are retrofitted for existing CPSs where specific protocols are already in use and irreplaceable. Our approach of transcribing industrial protocols into abstract protocol messages helps to realize protocol-independent industrial IDSs. Therefore, it is important to preserve essential communication features throughout the transcription process, such as communication patterns and message relationships.

Despite their huge diversity, transcribing industrial protocols is possible as they share essential commonalities like supervision and control of physical processes. First, we observe a shift towards IP-based communication for industrial protocols, easing passive network monitoring. Furthermore, as shown in Fig. 1, industrial protocols' communication behavior can be abstracted into three distinctive patterns: (a) pushing messages to single or multiple devices, (b) request and response, or (c) calls with an optional response. Finally, while protocol specifications dedicate great efforts to value

*CCS '20, November 9–13, 2020, Virtual Event, USA*

ACM ISBN 978-1-4503-7089-9/20/11.
https://doi.org/10.1145/3372297.3420019



Response
- id:1 -time:10:40:36.33
- src:GP - dst:*
- data:
  - latitude:4417.20
  - longitude:3800.30
  - time:10:40:35.43

  ⋮
- responds to:[]

Request (read)
- id:45 -time:11:27:25.67
- src:A - dst:B.8
- data:
  - coil.42
  - coil.43
  - coil.44
- responds to:[]

Response
- id:46 -time:11:27:26.03
- src:B.8 - dst:A
- data:
  - coil.42:0
  - coil.43:1
  - coil.44:1
- responds to:[45]

Request (call)
- id:29
- time:11:55:01.84
- src:A, - dst:B
- data
  42:UP
- responds to:[]

Acknowledgment
- id:30, - time:11:55:02.03
- src:A,
- respon

Response
- id:31, - time:11:55:02.15
- src:A, - dst:B
- data
  42:UP
- responds to:[29]

**(a) NMEA position update**
$GPGLL,4417.20,N,3800.30,W,
104035.43,A,A*43

**(b) Modbus read coil**
*A* ⟶ *B*: txID:237, unitID:8, funcCode:1, addr:42, quantity:3
*B* ⟶ *A*: txID:237, unitID:8, funcCode:1, values:[0, 1, 1]

**(c) IEC-60870-5-104 commands and responses**
*A* ⟶ *B*: IoA: 42, cause: Act; command: step, args: UP
*B* ⟶ *A*: IoA: 42, cause: ActCon; command: step, args: UP
*B* ⟶ *A*: IoA: 42, cause: ActTerm; command: step, args: UP

**Figure 2: Our concept for transcribing industrial protocols allows us to transfer protocol messages of widely different industrial protocols and communication patterns into unified information objects.**

| | References | [9] | [2] | [5] | [8] | [1] | [15] | [6] | [3] | [14] |
|---|---|---|---|---|---|---|---|---|---|---|
| | Protocol-independent | ○ | ◐ | ● | ○ | ● | ◐ | ○ | ◐ | ● |
| Required Information | Communicating entity | ● | ○ | ○ | ● | ○ | ○ | ● | ● | ● |
| | Message type | ● | ● | ○ | ● | ○ | ○ | ● | ○ | ● |
| | Accessed values | ● | ● | ○ | ● | ● | ● | ● | ● | ● |
| | Process values | ○ | ● | ● | ● | ● | ● | ○ | ● | ○ |
| | Timing information | ○ | ● | ● | ○ | ● | ● | ○ | ● | ● |

● : yes ◐ : partly ○ : no

**Table 1: Comparison of required information for industrial IDS detection mechanisms.**

encoding, the actual underlying data types can be condensed to a minimal, unified set across all protocols, as IDSs do not care about, e.g., endianness or bit-length of integers.

Although possible in theory, adopting IDS approaches developed for one industrial protocol to others requires manual effort for each individual protocol. To overcome this issue and provide a foundation for protocol-independent IDSs, we make two contributions: First, we provide an overview of industrial IDS detection techniques and the data they utilize for detection (§2). Second, we propose the concept of transcribing industrial protocols into a common representation as input for protocol-independent IDSs (§3) and discuss the challenges and potential of our design (§4). To the end, we conclude with a presentation of future work (§5).

## 2 INDUSTRIAL INTRUSION DETECTION

IDSs for industrial networks leverage anomalies, e.g., in device communication or physical process state, to detect attacks [10]. Early approaches spot malicious messages or malformed packets using simple static rules for traditional IDSs, e.g., Snort [16] or Suricata [12]. Relying on protocol-specific protocol fields, they can neither detect sophisticated and multi-stage attacks nor be transferred to other protocols.

Recent IDSs focus on detecting sophisticated semantic or sequence attacks. To this end, they leverage sensor values, actuator states, process knowledge, or recurrent communication sequences. In an initial survey (Tab. 1), we identify detection mechanisms underlying these industrial IDSs that do not depend on protocol-specific characteristics, but: (i) involved *communicating entities*, (ii) *message types* (e.g., read request), (iii) *accessed values* (e.g., temperature sensor), (iv) actual *process values*, and (v) related *timing information*.

While these detection mechanisms developed for specific industrial protocols can theoretically also be applied to other protocols, actually doing so today requires manual effort for each individual detection mechanism and industrial protocol [8]. Consequently, the entirety of industrial systems does not benefit from improvements in detecting advanced semantic attacks developed for one industrial protocol.

## 3 IDEA: PROTOCOL TRANSCRIPTION

Although showing vast diversity, industrial protocols still share similar functionality, i.e., communicating measurements and commands, theoretically allowing to transfer IDS functionality across protocols. Existing approaches such as PLC4X [7] show the potential of providing a unified API for actively querying different industrial protocols. But as shown before, IDSs are capable of analyzing further information such as network-wide communication patterns between individual devices. To realize IDSs, we need to go one step further and extract information on communication patterns, sensor readings, and commands directly from passive network captures. Passive monitoring at network switches with dedicated hardware does not impact the CPSs performance nor does it introduce additional delay to the system. Thus, our core idea is to extract messages from network flows, transfer them into unified information objects, and use them as IDS input.

When transcribing network flows into information objects, it is crucial to retain inherent industrial communication patterns (cf. Fig. 1 and Tab. 1). To this end, we model communication using abstract *request*, *response*, and optional *acknowledgment* objects, which we enrich with meta information such as source, destination, and receive time. For *push* communication (depicted with NMEA 0183, Fig. 2a), we use response objects to encode the pushed measurements. In contrast, *request and response*-based protocols (depicted with Modbus, Fig. 2b) require request and response objects, and we introduce identifiers (`id`) to link requests with subsequent responses. In *write or call*-based communication (depicted with IEC-60870-5-104, Fig. 2c), we encode the write or call operation in request objects using the name of an object or method and create response objects for return values.

Besides the need to encode communication patterns, we observe that industrial protocols often bundle multiple measurement points or control commands into a single protocol message. E.g., NMEA encodes six data points in one GPS position update message (cf. Fig. 2a), where each data point comprises a name and a value. We desist from introducing a common naming scheme for object names and instead suggest to derive these naturally but consistently from each protocol's specification. The resulting unified information objects can be used to realize protocol-independent IDS detection

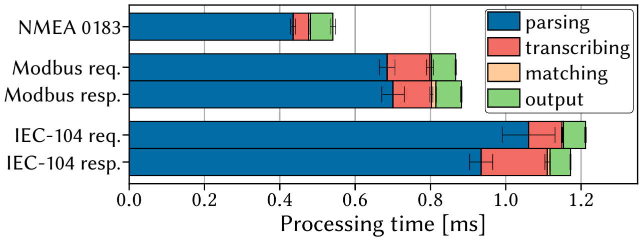


**Figure 3: Packet processing time of our prototype for** 10000 **packets per message type each (conf.** 0.95**).**

mechanisms for *all* industrial protocols since all relevant information types are preserved (cf. Tab. 1).

## 4 DESIGN CHALLENGES & POTENTIAL

To identify design challenges and future potential of our idea to transcribe industrial protocols, we built a prototypical transcription tool that currently supports NMEA 0183, Modbus, and IEC-60870-5-104. While these protocols are simple to transcribe, protocols such as OPC UA demand additional effort, especially if encrypted. Nonetheless, the selected protocols, stemming from vastly different domains, highlight the abstraction potential of industrial protocols.

The first design challenge arises from information discrepancies between request and response messages with identical purposes across different protocols. E.g., Modbus responses only contain process values and do not repeat the requested registers. This information has to be interfered from the preceding request. Thus, we have to follow the transcription of incoming packets with a protocol-specific packet matching phase. This matching is reused later on to identify and analyze communication patterns accurately.

Secondly, transferred data and their identifiers do not always match nicely with the physical state they represent. For example, Modbus might transfer two 16-bit registers representing a single 32-bit value. Adapting transformation rules, similar to those presented by Zehnder et al. [18], solves this and allows to prepare extracted data for further processing.

Our overview of IDSs in §2 highlights the need for different representations of captured information. Common schemes used by IDSs are time series of specific sensor data and commands, global process states over fixed or variable time slices, or localized views on a process' state. To support a wide range of (future) representations, we propose to offer exchangeable representations of the transcribed information.

Furthermore, identifying the optimal trade-off between the protocol abstraction-level and information loss is not trivial. Including every single networking artifact (e.g., TCP retransmissions, or IP fragmentation) might blow up abstraction complexity with only marginal information gains. Contrary, acknowledgments are of interest; however, IEC-60870-5-104 acknowledges on three different layers of the communication stack. We omit non-application layer acknowledgments and transcribe only semantic confirmations to, e.g., commands. Moreover, the influence of information loss or potential mistranscribing on IDS performance is still to explore.

Finally, we evaluated the initial performance of our prototype on a single physical core of an Intel i7-9850H CPU. From the results in Fig. 3, we already conclude a sufficient performance for many industrial use cases [11]. As most of the time is spent by Scapy parsing incoming packets, increasing the performance further is possible through implementation improvements or parallelization of the parser and transcriber. With the increasing use of more than one IDS in a single network [17], our approach can even improve current IDSs' performance as the expensive parsing of captured traffic only has to be executed once.

## 5 CONCLUSION & FUTURE WORK

While today's industrial IDS approaches could be capable of detecting cyberattacks against various CPS, they are often tailored to individual industrial protocols. To obviate the need to apply IDS approaches for each industrial protocol individually, we propose a concept for transcribing industrial protocols: By converting network flows into abstract request and response message objects and retaining relevant communication relationships, we provide a basis for widely-applicable and protocol-independent IDS.

In the next step, we plan to validate that IDSs based on transcribed network traffic yield similar detection rates as custom-built IDSs. Afterward, the detection mechanism can be applied to different protocols to evaluate how well this adoption performs. Since IDSs may also leverage process knowledge that cannot be inferred from protocol messages, underlying process models might potentially have to be relearned.

Our approach can thus serve as a decent foundation to realize protocol-independent industrial IDSs as it transparently maintains important communication features such as communication patterns and message relationships. Consequently, we enable the development and deployment of IDSs which target *all* industrial protocols.

## REFERENCES

[1] Chuadhry M. Ahmed et al. 2020. Process Skew: Fingerprinting the Process for Anomaly Detection in Industrial Control Systems. In *ACM WiSec*.
[2] Marco Caselli et al. 2015. Sequence-aware Intrusion Detection in Industrial Control Systems. In *ACM CPSS*.
[3] John H. Castellanos and Jianying Zhou. 2019. A Modular Hybrid Learning Approach for Black-Box Security Testing of CPS. In *ACNS*. Springer.
[4] Nicolas Falliere et al. 2011. W32.Stuxnet Dossier. *Symantec White Paper*.
[5] Cheng Feng et al. 2019. A Systematic Framework to Generate Invariants for Anomaly Detection in Industrial Control Systems. In *NDSS*.
[6] Benedikt Ferling et al. 2018. Intrusion Detection for sequence-based attacks with reduced traffic models. In *MMB*.
[7] The Apache Software Foundation. 2020. *PLC4X*. https://plc4x.apache.org/
[8] Igor N. Fovino et al. 2010. Modbus/DNP3 State-based Intrusion Detection System. In *IEEE AINA*.
[9] Niv Goldenberg and Avishai Wool. 2013. Accurate Modeling of Modbus/TCP for Intrusion Detection in SCADA Systems. *IJCIP* 6, 2.
[10] Martin Henze et al. 2020. Poster: Cybersecurity Research and Training for Power Distribution Grids – A Blueprint. In *ACM CCS*.
[11] Jens Hiller et al. 2018. Secure Low Latency Communication for Constrained Industrial IoT Scenarios. In *IEEE LCN*.
[12] BooJoong Kang et al. 2016. Towards A Stateful Analysis Framework for Smart Grid Network Intrusion Detection. In *ICS-CSR*.
[13] Robert M. Lee et al. 2016. Analysis of the Cyber Attack on the Ukrainian Power Grid. *E-ISAC* 388.
[14] Chih-Yuan Lin et al. 2017. Timing-based Anomaly Detection in SCADA Networks. In *CRITIS*. Springer.
[15] Qin Lin et al. 2018. TABOR: A Graphical Model-based Approach for Anomaly Detection in Industrial Control Systems. In *ACM AsiaCCS*.
[16] Thomas Morris et al. 2012. A Retrofit Network Intrusion Detection System for MODBUS RTU and ASCII Industrial Control Systems . In *IEEE HICSS*.
[17] Fredrik Valeur et al. 2004. A Comprehensive Approach to Intrusion Detection Alert Correlation. *IEEE TDSC* 1, 3.
[18] Philipp Zehnder et al. 2020. StreamPipes Connect: Semantics-Based Edge Adapters for the IIoT. In *ESWC*. Springer.