No Need for Details: Effective Anomaly Detection for Process Control Traffic in Absence of Protocol and Attack Knowledge

Franka Schuster
Brandenburg University of Technology
Cottbus-Senftenberg
Germany
franka.schuster@b-tu.de

Hartmut König
Brandenburg University of Technology
Cottbus-Senftenberg
Germany
hartmut.koenig@b-tu.de

Abstract

The rapidly expanding landscape of attack vectors on cyber-physical systems (CPS) has led to the proposal of various attack detection methods for this area. Most approaches focus on analyzing time series of data from physical processes. However, the availability of such well-prepared data is not guaranteed in most infrastructures. In contrast, relatively few approaches address the direct analysis of network traffic, which is the natural basis for interaction between CPS devices. In this paper, we examine traffic-based methods using data flows, packets, and packet sequences as monitoring base. We include the packet payload in the analysis in a protocol-agnostic manner. This offers the possibility to apply the approach in different networks independently of the used CPS technologies or processes. We use one-class machine learning methods applied on only normal traffic in the training phase. This allows us to configure the detection capabilities independently of attack knowledge or given attack examples. Besides the evaluation regarding detection capability and efficiency, we further examine the potential of the protocol-agnostic models for a transfer on foreign detection scenarios.

CCS Concepts

• **Security and privacy** → **Intrusion detection systems**; • **Networks** → **Cyber-physical networks**; • **Computing methodologies** → *Machine learning*.

Keywords

Protocol-agnostic Traffic Analysis, Anomaly Detection, Process Control Networks, Industrial Control Systems

ACM Reference Format:
Franka Schuster and Hartmut König. 2024. No Need for Details: Effective Anomaly Detection for Process Control Traffic in Absence of Protocol and Attack Knowledge. In *The 27th International Symposium on Research in Attacks, Intrusions and Defenses (RAID 2024), September 30–October 02, 2024, Padua, Italy*. ACM, New York, NY, USA, 20 pages. https://doi.org/10.1145/3678890.3678932

This work is licensed under a Creative Commons Attribution International 4.0 License.

RAID 2024, September 30–October 02, 2024, Padua, Italy
© 2024 Copyright held by the owner/author(s).
ACM ISBN 979-8-4007-0959-3/24/09
https://doi.org/10.1145/3678890.3678932

1 MOTIVATION

As the backbone of many critical infrastructures, cyber-physical systems (CPS) have come into focus as a potential attack target [54]. Therefore, identifying malicious activities on these networks, which mainly rely on operational technologies (OT)[1] to control complex physical processes, is an issue of paramount importance. As a result, various detection approaches focusing on OT have been proposed [95, 96]. The vast majority of methods presented so far assumes that time-series of data from the OT processes, such as sensor and actuator states, are available for the analysis [96]. These approaches rely on the existence of well-structured input datasets that have already been preprocessed regarding operational events by an external system. This expectation, however, is not fulfilled in most real-world networks.[2] The previously rarely pursued alternative, the direct analysis of network traffic, has several advantages. These are ❶ detecting anomalous communication activity before the impact on OT processes becomes visible, ❷ the independence of any external information sources apart from network traffic, ❸ no instance (and potential point of failure or corruption) between network activities (carrying also process activities) and the detection system, so that ❹ it can potentially be applied to various OT networks. Existing traffic-based systems though suffer from at least one of the three following disadvantages. They use algorithms that rely on all classes including malicious traffic for the configuration of the detection, thus neglecting permanently changing attack variants to evade identification. Either the approach or the evaluation scenario do not allow conclusions about the applicability to a larger number of (OT) networks [42]. This applies to all approaches that are based on protocol knowledge or are only evaluated on traffic dominated by a specific OT protocol. In addition, the evaluation is limited to the set of correctly identified input instances, while the question remains whether the detection pipeline, which is complex in many cases, can keep up with the given input rates.

Contribution. In this paper, we propose a traffic-based detection approach that overcomes these limitations. It is based on requirements that we have previously identified by analyzing OT traffic characteristics [76]. Our goal is to create a method with maximum local and global effectiveness that analyzes all network traffic without protocol or port filtering and is applicable to any OT network.

[1] In the following we use the term OT as synonym for CPS or ICS (Industrial Control System) because cyber is no well-defined term and not every control process has an industrial purpose.

[2] This is one of our fundamental findings from years of work in the field, for which no public sources exist to date.

- We present a flexible packet parsing that allows the packet payload to be incorporated in the detection analysis without knowledge of the underlying protocol stack. It can be applied to any OT (or even non-OT) traffic, either based on IP-protocols or on those directly transmitted over Ethernet.
- We propose a protocol-agnostic anomaly detection approach which, unlike previous approaches, can be applied without making any assumptions regarding the OT network's characteristics, e.g., about the dominance of an OT protocol. We demonstrate this by applying the proposed protocol-independent anomaly detection method to four public OT datasets and show that it achieves the same or better results for which multiple protocol-specific and/or more complex methods were required previously.
- We explore a potential that arises through the protocol-agnostic traffic analysis: We examine the feasibility of transfer learning in the area of OT attack detection by using models trained on traffic in one OT network for the detection in foreign networks.
- To move beyond the academic context to real-world application, as recommended [4, 83], we also perform all experiments on traffic traces taken from three productive critical OT infrastructures.

The remainder of the paper is organized as follows. In Section 2, we reason design decisions for the detection and, based on prior research, derive open questions. Next, in Section 3, the traffic perspectives and algorithms are disussed that were investigated for the protocol-agnostic detection. After introducing the datasets used for evaluation in Section 4, we present and discuss the results achieved in Section 5. We conclude the paper in Section 6 by a summary of the main takeaways.

2 PROBLEM AND OPEN QUESTIONS

Starting with a brief summary of the conditions in OT networks, we reason the requirements for effective attack detection. To implement such a detection, we need to address a number of questions that have not been answered in previous studies, yet.

2.1 OT Networks in a Minimal Nutshell

OT networks are used widely. They form the backbone of most critical infrastructures, such as the energy and water supply, why they have to meet highest requirements regarding availability and reliability. They typically operate 24/7. The options for setup changes needed to integrate new features or conduct experiments are extremely limited. The networks are relatively heterogeneous in terms of the physical processes implemented and the devices used. The interaction is often based on proprietary, non-standardized communications [84]. Many devices have been developed with the sole aim of performing a specific control task in an expected setup lacking fault tolerance, built-in security, and flexibility for new network conditions [6, 84]. These are also the reasons why these networks are currently only protected from the outside world, if at all, using firewalls. There is no internal security monitoring, which is overdue, especially given the importance of critical infrastructures, in creating multi-level security. An internal attack detection must take all these factors into account.

2.2 Requirement-driven Detection Design

The effectiveness of a detection approach lies in the ability to correctly identify both normal and anomalous activity. Two aspects are important here: Firstly, all network events that can indicate attacks should be incorporated. We refer to this as the detection's *local effectiveness*. Secondly, an effective detection approach should further be created in such a way that it can be used without any information or assumptions about the network, i.e., to be applicable to as many infrastructures as possible, what we call *global effectiveness*. If a method is to be applicable in real time, its effectiveness is also influenced by the efficiency of the algorithms. If an approach is designed for maximum effectiveness, but is not fast enough to capture all activity, not all events can be analyzed. They have to be dropped, leading to blindness of the method regarding these activities. Therefore, our approach pursues all three goals.

Anomaly detection without attack(er) knowledge. There are two basic paradigms for attack detection: Misuse detection by using attack signatures as detection base and anomaly detection as orthogonal approach based on a definition of normal activities [7, 40, 83][3]. Numerous works [2, 3, 9, 32, 34, 91, 98] misleadingly refer to anomaly detection when they actually implement a third type by using all classes (normal *and* anomalous activities) for setting up the detection [40]. Systems that depend on malicious activity for configuration have significant drawbacks. ❶ They are always biased (or even overfitted) by the provided attack knowledge and are thus blind against other types. ❷ They cannot detect new (zero-day) attacks which is crucial with the attack landscape's dynamics nowadays. ❸ They further cannot be applied if malicious activity is not available at all. For OT networks, only few realistic attack activities have been recorded yet [6], especially in terms of their diverse nature. Usually they also cannot be generated under realistic conditions due to the aforementioned operational contraints. For this reason, anomaly detection, i.e., identifying attacks as deviations from normality, is the only practicable scheme here. At the same time, detecting attacks based on knowledge derived only from the class of normal traffic is more challenging than including anomalies as a second class during training [7, 40].

Traffic as input instead of process data. The research on OT attack detection can be divided into process-based approaches that analyze preprocessed data from control processes and communication-based ones taking network traffic as input. Process-based schemes dominate the field [42, 95, 96], although they come with a number of serious issues (cf. Appendix A.1 for a more extensive argumentation). ❶ They mostly perform comparatively simple checks [95], which are since decades a built-in functionality of the already established monitoring necessary to run the control process(es), without measuring the benefit of the new detection on top. ❷ They expect a *historian*, i.e., a time-series OT database with a large amount of well-structured and collected process data as working base, which is neither available in most networks (cf. Footnote 2) nor can be established by the operators due to stringent warranty restrictions from system vendors. ❸ In the lack of standards for logging process data (e.g., format and granularity), this reliance on preprocessed data means that a process-based solution built for one network is typically not transferable to other networks. ❹ They monitor log

[3] The terms signature-based and anomaly-based refer to the same differentiation.

files (or databases) of activities reported by the OT devices and therefore can only detect what already has caused anomalies (and potential damage) at the process level. Consequently, we advocate communication- or traffic-based approaches, respectively, in the following, whose advantages were already mentioned in Section 1.

Modeling normality by machine learning rather than rules or state machines. For the definition of normality, which is necessary for anomaly detection, rules, state machines, and models built using machine learning are conceivable. Although activities in OT are considered comparatively homogeneous (or regular) [13, 24, 40, 55, 96], activities are still too variable to model normality sufficiently by rules or state machines [76]. This may be feasible for a single OT application [11, 12, 24, 29] or for certain traffic aspects [66], but with a rising amount of monitored applications or traffic data, the two approaches suffer from rule or state explosion with negative effects on practicability and efficiency. One-class machine learning algorithms, in contrast, are designed to extract the properties relevant to represent a class from highly variable input data, which is why we prefer them for modeling normal network traffic. Unlike rules and state machines they are black boxes, but can be post-explained by explanation algorithms [36, 50]. We weight efficiency over transparency here, as a transparent model is of no use if it cannot efficiently solve the detection task.

Complete traffic analysis in a protocol-agnostic manner. For maximum global effectiveness, the detection process should be applicable without prior knowledge about the used OT (and non-OT) protocols, the deployed OT devices, or any other traffic characteristics. For maximum local effectiveness the whole traffic, i.e., every flow and Ethernet frame, is presented to the detector without filtering for certain criteria, such as OT protocols or specific ports. This meets the facts that ❶ OT communication is often realized by several (OT) protocols instead of a single dominant one [53, 76], ❷ is mixed with very different amounts of communication using standard information technology (IT) protocols [53, 76], and ❸ this standard IT communication can also be a target of attacks to damage the OT network. From each Ethernet frame, which we refer to as packet in the following, the content of the highest network layer available is incorporated as undecoded byte sequence in the analysis. Thus, also sophisticated attacks on the payload's process data can be detected.

2.3 Current State of Traffic-based Detection Schemes for OT

We summarize prior research in the field using the collection of [96] supplemented by eight further approaches recently published at top security venues [8, 10, 25, 52, 74, 86, 89, 97].

Specific approaches. There are eight approaches using benign and malicious traffic and 19 tackling the greater challenge of using only one class [7, 40], the normal traffic, to set up the detector. There is no approach that applies more than one perspective for monitoring the network traffic (*flows*, *single packets*, or *packet sequences*) which limits the range of detectable attacks from the outset (cf. Section 3.1). We present a closer look at the detection methods in Table 1, which reveals that only five one-class approaches were evaluated on public datasets which is necessary to relate the results to previous and new approaches. Moreover, only five of the 19 approaches were evaluated on heterogenous traffic traces (*#public*[4] and *#real datasets*) with different OT protocols (*#OT protocols*). Consequently, the local and especially global effectiveness, motivated in Section 2.2 including the transferability of the approaches to other OT networks, was not evaluated either. In fact, so far only one approach [8] aims to analyze packet payloads in a protocol-agnostic manner, as we do. In addition, many works lack an evaluation whether the detection is sufficiently efficient for data rates encountered in OT networks. Especially for for packet-wise analyses, an evaluation of the processing speed (efficiency) should prove the suitability for real-world networks. So far, however, only three of 14 one-class approaches (partly) discuss this issue (*speed evaluation*). Finally, only for a single one-class approach [24] the implementation is published to share the results and efforts made with the intrusion detection community.

Protocol-agnostic reuse of existing approaches. In [96], it has been proposed to decouple existing approaches from domain-specific OT protocols to use them outside of their restricted initial design. We fully share the motivation and appreciate the work. Strength of the work is to create a base for offline comparisons and reevaluations of approaches scientifically. Unfortunately, it cannot help to design a performant system for the real world. The overhead using the method is only very briefly discussed. It is estimated by 10% for a single approach [24], assessed as the most complex one [96], without providing a nominal analysis rate that could be related to traffic rates in real networks as identified previously [53] and in Section 4.3. A second major hurdle for practical use is the effort necessary to integrate new abstractions to support a further protocol. This seems reasonable given the assumption that networks are usually dominated by one or two OT protocols. Unfortunately, this expectation persists, although it is not true, as shown previously [53] and observable in our real-live captures [76].

Novelty of our work. Unfortunately, the only existing protocol-agnostic approach so far [8] has several clear drawbacks. It relies on the construction of hidden semi-Markov models, Gaussian mixture models, *and* probalistic suffix trees for the detection. The computational complexity is given as quadratic to the states of the Markov models and the size of the suffix tree's range. We present our approach because we consider it desirable to use at most one of already highly advanced algorithms with high efficiency rather than a combination of three complex methods with limited performance. Additionaly, the results cannot be verfied or used[5] in order to create a practicable method for the requirements of OT networks stated in 2.2. In contrast, we are the first to provide a protocol-agnostic approach including material for comparative analyses and source code for further developments.

2.4 Open Questions

Our goal is to develop a traffic-based anomaly detection approach with maximum local and global effectiveness without any information or assumptions about the network. We break the evaluation of feasibility down to the following questions.

[4] We use # as shorter form for *number of.*

[5] The implementation is not available.

Table 1: Traffic-based works with evaluation characteristics indicating local (# OT protocols) and global effectiveness (# datasets). Two-class approaches use benign and malicious traffic for configuration, one-class approaches only rely on normal traffic.

	two-class approaches								*one-class approaches*																			
	[69]	[80]	[25]	[86]	[67]	[3]	[14]	[70]	[90]	[5]	[29]	[99]	[11]	[12]	[89]	[10]	[48]	[23]	[52]	[24]	[46]	[100]	[74]	[97]	[47]	[51]	[8]	**here**
flows	●		●	●				●	●	●							●		●						●			●
single packets		●			●	●	●											●					●	●		●	●	●
packet sequence											●	●	●	●	●	●				●	●	●						●
# public datasets					1	1[6]	1	1							1[7]		1	1					2				3	**4**
# real datasets			1					1			1		1	3	1	2	1		1	1		9		2	1			**3**
# OT protocols	1	1	1	2	1	2	1	1	1	x[8]	1	1	1	1	2	1	3	1	1	1	1	1	2	6	1	1	3	**4+**
protocol-agnostic																											●	●
speed evaluation	●		●			●											●	●				●	●		●			●
public material					●											◗				●	◗							●

Legend: ●: matching characteristic; ◗: partly available

Q1: Can the protocol-agnostic approach cope with previous, more individual approaches? In order to determine which detection success can be achieved by a protocol-agnostic approach, we model packet payloads as byte sequences without effort to decode [10, 89] or reverse-engineer payload contents [38]. We evaluate this approach using four public datasets with well-labeled validation traffic. For each dataset, we evaluate the local effectiveness (results per dataset) of the protocol-agnostic detection and compare it with all previous results achieved on that dataset. Since the datasets include four OT protocols, we conclude to the global effectiveness, i.e., the general applicability to OT networks.

Q2: Is the approach sufficiently efficient to cope with networks' data rates? As argued in Section 2.2, effectiveness also relies on efficiency. In our approach the analysis of every packet of the whole, unfiltered network traffic combined with *machine learning on packet payloads* represents a non-negligible effort. Although an OT protocols' payload is not interpreted, the packet is partly decoded to extract general transport information. We measure if such a detection design is feasible or traffic has to be dropped with negative impact on effectiveness (due to blindness regarding the dropped traffic). Here, the processing times for each step of the training and detection pipeline are collected and correlated to the packet rates of each analyzed dataset. Since network traffic from real OT sites with different process purpose are part of the evaluation, the results also allow conclusions on the feasibility of the approach for traffic volumes in real OT networks. Apart from previous results for a byte-mapping approach [74] that focused only on packet parsing times, such a comprehensive analysis has neither been done before nor with the inclusion of several real OT datasets.

Q3: What is the potential of transfer learning for anomaly detection in OT traffic? The protocol- and process-agnostic nature of the detection approach opens new possibilities to use detection models even more effectively among several networks. Hence, to evaluate the global effectiveness of the method, we go a step further and investigate its potential regarding transfer learning [88]. Transfer learning in OT networks has the advantage of knowledge exchange between infrastructures by reusing already trained models of normality in other infrastuctures. This could help to decrease false-alarm rates, an inherent problem of anomaly detection, and to shorten training times required for each deployment. It can be implemented in different ways using *pre-training* on one dataset and application of the resulting model on other data, or using *self-learning* methods which train the model on the target data supplemented by knowledge from a foreign setup [101]. In this paper, we examine to what extent models trained on a dataset can be used for detection on traffic from another dataset and how this is affected by the same or another protocol mix.

Q4: How do different types of traffic perspectives affect the effectiveness and efficiency of learning-based anomaly detectors? Network traffic can be prepared for monitoring in various ways. Additionally, there are several options of algorithms usable for one-class training of the detector. We are therefore conducting a series of experiments with three kinds of traffic mappings and four machine learning algorithms. We aim to find out whether there are certain suitable settings that perform well independent of the respective network (here: among the datasets). This allows the numerous possibilities for future improvements to be narrowed down to the most promising setups.

3 TECHNICAL REALIZATION

The design of the detection method presented here is based on requirements for attack detection procedures given by operators of round-the-clock energy supply networks. Table 2 summarizes the requirements with the associated design and deployment decision.

Table 2: Operational requirements with design decisions for a convenient and widespread deployment.

operational requirement		**design and deployment decision**
no host changes	→	network traffic as input
no (or minimal) network changes	→	switch mirror port as input source
no interaction with the network	→	passive operation
independence from OT technology	→	protocol-agnostic procedure
independence from physical processes	→	process data not incorporated
self-adjustment	→	use of machine learning
no availability of attack examples	→	one-class training of the detector

The method was therefore created for the following scenario. It is placed on dedicated hardware connected to the mirror port of a switch in an OT network. The entire traffic is continuously analyzed and relevant data for identifying attacks is extracted from every

[6] This work was evaluated on two datasets but one [37] is not accessible anymore.
[7] This work was evaluated on the infrastructure of SWaT [28] but the traffic used is not public.
[8] The protocol(s) are not named.

flow and packet. This information is continuously conversed into instances (samples) of a meaningful and efficient format, which are used to model the detection knowledge using machine learning in a training phase. The knowledge is afterwards used in the detection phase to differentiate benign from malicious communication.

3.1 Feature Parsing and Sample Creation

There are two traditional perspectives on network traffic, in particular for intrusion detection. We consider both because they differ in their strengths and weaknesses.

Flow perspective. The analysis of network flows, defined as a sequence of packets sharing five attributes[9], focuses on quantitative traffic characteristics. The investigation is limited to the meta attributes of the packets, which allows for very efficient monitoring. The volume-based analysis of traffic characteristics further enables the detection of load-based attacks (Denial-of-Service attacks), which are hardly detectable by per-packet analyses. On the other hand, false-data injection attacks cannot be detected without evaluating packet contents. We assign **33 flow features** to each flow sample reflecting common flow properties for quantifying and characterizing network flows [94]. They have emerged in the past in conjunction with flow export standards (sFlow [68], NetFlow [15], or IPFIX [16]) and have proven to be useful for attack detection (cf. works operating on flows in Table 1). They consist of eleven parameters measured in three ways, i.e., per (bidirectional) flow, forward, and backward direction: *duration* in milliseconds, *number of packets*, *number of bytes*, *minimum, mean, and maximum packet size*, *standard deviation of the packet size*, *minimum, mean, and maximum packet inter-arrival time*, and the *standard deviation of the packet inter-arrival time* in milliseconds. Although this implies redundancy, we do not want to predetermine (and limit) which aspects might be particularly suitable for modeling normal traffic.

Packet perspective. The limitations of flow analysis require packet analysis. We apply deep packet inspection (DPI) to each Ethernet frame to extract meta data and payload prefixes. The meta data include *source and destination MAC addresses, packet size, number of packet layers, EtherType, payload size*, and a *hash value of the payload*. We consider MAC addresses in the modeling because otherwise all packets of the traffic would be presented to the machine learning algorithms without a criterion to distinguish communication relationship. When providing them, the algorithm can still decide to what extent the addresses are relevant for modeling the traffic. From the payload itself, a fixed-length prefix of 57 bytes is extracted and used for detection without decoding. If the payload is shorter the vector positions that are not required are set to zero. The length of the payload results from the target total length of the packet samples of 64 minimized by the length for the seven meta data features. We consider the sample size of 64 to be a good trade-off to model meta data and a representative part of the payload with a limited probability of sparse zero-filled samples. In addition, it is square and can be transferred to samples of quadratic dimension, which is useful as potential input for autoencoder structures. The hash value of the payload complements the byte-wise payload features in two ways. First, it represents an efficient characterization of the complete packet payload beyond the considered prefix. In addition, it expresses the payload in a single feature. This allows us to differentiate the complete payloads, whereas the 57 byte features are necessary to finer model and evaluate single payload byte positions. The features of packet samples and the reasons for their choice are summarized in Table 3. Packet-based analysis enables a very precise monitoring of the data traffic, which in turn usually involves considerable effort for packet decoding. This can be challenging, potentially a bottleneck at high packet rates. To detect Man-in-the-Middle attacks [64, 65] or the manipulation of process data encapsulated in packets [26], such an analysis is necessary.

Table 3: Features in packet samples with reason for choice.

feature(s)	reason
source MAC address	reflect the corresponding communication relation
destination MAC address	
packet size	reflect the structure of the packet
number of packet layers	
EtherType	
payload size	reflects the amount of the highest-layer payload within the packet structure
payload hash value	characterizes the whole payload
first 57 payload byte values	byte-wise reflection of the payload prefix

Packet sequence samples. In addition to observing flows and individual packets, sequences of packets are also essential for detecting attacks. By observing them, further deviations in the communication flow can be detected that are neither characterized by changing load or communication partners (evident in flow characteristics) nor by unusual packet data (recognizable in the individual packet analysis). These are (stealthy) replay attacks, Denial-of-Service and Man-in-the-Middle attacks [64, 65] in which the attacker injects packets using the address of a legitimate communication partner. A sequence length of three packets represents the minimal context of a packet, i.e., the previous and next packet in the traffic capture. Using longer sequence lengths for training and detection may lead to the modeling of less important relationships and thus to noise in the learning and decision process. Furthermore, the degree of redundant information and the number of different samples increases. We analyzed the latter and found that in the case of the seven datasets, the diversity of the samples increases by 19-43% from 3- to 5-grams. Higher diversity of inputs also means higher complexity of the models, such as larger tree structures to represent all inputs when using Isolation Forest, with negative impact on efficiency for the training and detection phase. We therefore decided to work with 3-packet samples. They are created by applying a sliding window to the observed sequence of packets. Here, 14 bytes of payload are extracted to form 63-dimensional vectors that reflect the same seven features ($3 \times (7 + 14) = 63$) as the single-packet samples, in addition to the packet payload.

Support of proprietary and non-IP-based protocols. Support for potentially all OT protocols in the monitored network is ensured by two properties of the protocol-agnostic parsing process that address two challenges in monitoring OT communications: Firstly, certain OT protocols, such as Profinet or GOOSE, can be applied without the use of the IP protocol. A flexible payload extraction method was implemented to support such protocols. As

[9] These are source and destination IP addresses, IP protocol (number), source and destination ports.

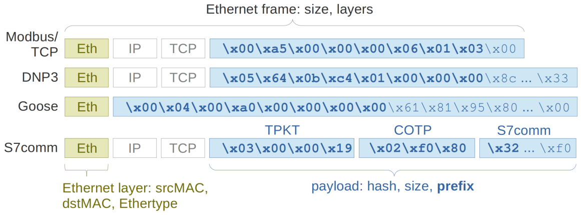

Ethernet frame: size, layers
Modbus/ TCP
Eth
IP
TCP
\x00\xa5\x00\x00\x00\x06\x01\x03\x00
DNP3
\x05\x64\x0b\xc4\x01\x00\x00\x00\x8c ... \x33
Goose
\x00\x04\x00\xa0\x00\x00\x00\x00\x61\x81\x95\x80 ... \x00
TPKT
COTP
S7comm
\x03\x00\x00\x19
\x02\xf0\x80
\x32 ... \xf0
Ethernet layer: srcMAC, dstMAC, Ethertype
payload: hash, size, prefix

Figure 1: Packet parsing on different OT protocol stacks: Data beyond known transport layers is considered as payload (blue), from which a fixed byte number (bold) is analyzed.

Table 4: Selected one-class algorithms with reason for choice.

	modeling idea	reason
Elliptic Envelope algorithm [73, 77]	covariance-based	fast detection
One-class Support Vector Machine [75, 79]	kernel-based	modeling structural data
Isolation Forest algorithm [49, 78]	tree-based	modeling statistical data
Convolutional Autoencoder [30, 85]	neural network	inherent feature extraction, ability to model any function

depicted in Figure 1, the developed parser analyzes each Ethernet frame seen on the network layer by layer and takes the data field of the highest available layer as payload. Secondly, in addition to using well-known protocols, there are countless process implementations that use proprietary protocols. To be able to incorporate the payload transmitted by these protocols it is taken as a sequence of bytes from which a prefix is used during detection.

3.2 One-class-based Detector Modeling

The goal is to create a system that derives its configuration solely by observing normal traffic from the network it is attached to, as reasoned in Section 2.2. Therefore, only machine learning based on one class is relevant here. We do not only examine one algorithm, but several to compare results between different types of algorithm for the following reasons: ❶ In general, one cannot infer a particularly suitable machine learning algorithm from a given problem. ❷ We do not want to unnecessarily limit the possibilities for good detection results from the outset. Since the algorithms differ in the underlying concepts, different results must be expected for the same input. ❸ Comparisons across multiple algorithms can provide researchers with a rationale for future detection decisions. From previous experiments, we selected four algorithms, each based on a different modeling idea, as still debatable number. These are Elliptic Envelope algorithm (envelope), One-class Support Vector Machine (OCSVM), Isolation Forest algorithm (iforest), and Convolutional Autoencoder (CAE). They are summarized in Table 4. Further information regarding the algorithms and investigated hyperparameter space for model creation can be found in Section A.2 in the appendix. The algorithms have not been altered in order to use them.

4 TRAFFIC DATASETS FOR EVALUATION

The selection of the datasets is affected by our research goals: ❶ To assess the protocol independence, a representative number of different OT protocols must be included. ❷ For a serious evaluation of the detection capability, a groundtruth defined by a third party (here the authors of the public datasets) is required. Network traffic needs to be labeled at the packet level to precisely distinguish normal and anomalous activity, which excludes many public datasets. ❸ To assess the suitability of the approach for real data rates, we include traffic captured from real OT networks. Consequently, we use traffic from public testbeds and real infrastructures. Public datasets serve as a reference point for comparing our results with past and future approaches. Traffic from real infrastructures permits us to relate the results to real-world conditions.

4.1 Public Datasets

Several public OT datasets are available [17]. In keeping with our focus on traffic-based detection schemes, we examined the datasets regarding usability for a sufficient evaluation of such systems. This includes ❶ the availability of unpreprocessed (raw) network traffic, ❷ the presence of anomalous traffic in addition to normal one, and ❸ a labeling of these anomalies with sufficient granularity, i.e., on packet level. We found four datasets that meet these criteria (see Table 12 in the appendix). They include four OT protocols: Modbus/TCP, DNP3, GOOSE, and S7comm.

Lemay dataset. The captures of Lemay and Fernandez [43, 44] were collected from a simulation of a power grid control system using the protocol Modbus/TPC. From the six normal capture files we use the one referred to as Run1_6RTU because it was taken from a setting that incorporates six remote terminal units (RTUs), for which also the attack traffic was generated. We consider this as relevant normal traffic for training because one would also place, train, and apply a detector in an identical network setup in a real scenario. All five parts available were used as validation traffic, whereas the labeling of the largest one[10] with a packet share of about 40% was corrected in March 2023 (after we identified and reported an apparently wrong labeling to the authors). Before it could not or only erroneously be applied for evaluations. The malicious activities consist of command and control, penetration attack, and covert channel attack traffic.

Water distribution testbed (WDT) dataset. This dataset also contains Modbus/TCP and origins from a testbed that emulates a water distribution process by connecting a real subsystem to a simulated one using hardware-in-the-loop technology [22, 31]. The process implements the water flow among eight tanks using four programmable logic controllers (PLCs), valves, pumps, pressure, and flow sensors.

DNP3 dataset. These traces from the Queensland University of Technology of Brisbane (QUT) [62, 71] come from a testbed of energy transmission stations using the protocols DNP3 and GOOSE. The malicious activities are reconnaissance actions, data injection, masquerading, packet flooding, Man-in-the-Middle communication, and packet replay.

S7comm dataset. This dataset, also published by QUT, [63, 72] is from a testbed emulating a subprocess of a mining refinery based on communications using the S7comm protocol. The attacks manipulate the process through respective messages, e.g., switching on and off the conveyor belts or water tanks either by targeted messages or even by flooding of the corresponding process commands.

[10] characterization_modbus_6RTU_with_operate_labeled.csv

4.2 Real Datasets

In addition, we used three datasets from real OT networks. The traces were recorded in January 2023 in three different networks of a productive coal-fired power generation infrastructure, which belongs to the class of 500 MW units. For each capture, the tool `tcpdump`[11] was installed on a separate device and attached to a central switch of the respective networks with activated port mirroring[12]. The infrastructure is protected by a demilitarized zone (DMZ) and consists of several plants. Each plant controls a high number of subprocesses, such as the production of steam, for the overall power generation process. Figure 2 shows the three capture points. All three traces were recorded in parallel during normal plant operation. As these networks are subject to the conditions described in 2.1, no anomalies could be generated.

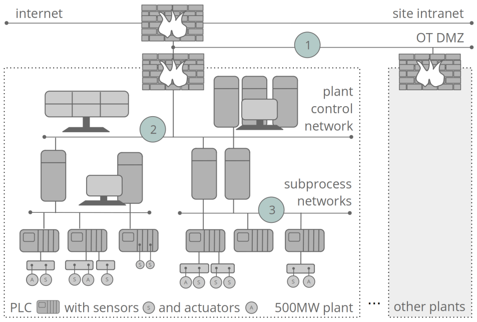


Figure 2: Real infrastructure with three capture points.

DMZ dataset. This trace was captured at infrastructure level in the DMZ of the power generation site. As perimeter network to the infrastructure, it has the highest load of all three traces regarding flow and packet rate. It possesses the highest number of communication devices and the most diverse protocol mix, including a lot of standard IT protocols.

Plant control dataset. This capture reflects OT traffic at plant level. It was taken from the network at that point where the data streams of all generation sub-networks are combined and prepared for the process monitoring. This includes routines for automatic control of process interactions and for providing all relevant events and process states to the visualization in the plant's control room. The trace also includes various protocols used in standard IT communications rather than exposing specific prevailing OT protocols.

Steam production dataset. A subprocess of the power generation is the steam production to operate a turbine. The third trace is taken from the network controlling this process. In the subnetwork 41 devices interact as part of the steam control system to regulate and optimize the firing of several steam generators. The steam is then fed into a turbine to generate electricity. The dominant OT protocol is Modbus/TCP.

4.3 Characteristics and Labeling

Dataset characteristics. Each public dataset consists of a part with only normal traffic for training the detector and a part with mixed traffic for validating it afterwards. The three real datasets contain only normal traffic. The training part precedes the validation part. The characteristics of all parts are summarized in Table 5. From each part, flow and packet samples were derived according to the steps presented in Section 3. The numbers of the samples extracted during the training and validation periods correspond to the flow and packet numbers given in the table. Additional facts about the involved OT protocols and three of the four public datasets can be found in [17].

From packet labels to flow labels. Each packet of the public datasets was labeled by the authors as normal or malicous. To also examine the feasibility of applying our detection method to flows, a respective labeling of flows as groundtruth was required. For this, we derived flow labels from the packet label files by defining each flow as anomalous if the data contained an anomalous packet related to this flow.

Use of real datasets without anomalies. We considered a subsequent integration of artificial anomalies into the captured real datasets. We decided against it because any choice and implementation of attacks is subjective and can be criticized, and results on artificial anomalies can only be used to a very limited extent to infer the detection capability in the origin network. In contrast, the anomalous traffic of the public datasets was generated by other scientists in the respective network setup.

5 EXPERIMENTS AND RESULTS

The approach is evaluated on the network traffic of the seven introduced datasets. We first explain the design of the experiments, the measured criteria, and the identification of the investigated detection models. Then we answer the four initially stated questions regarding effectivity, efficiency, the potential of model transfer, and the effects of different traffic perspectives on the anomaly detection. We present the results and compare them with previous detection approaches where applicable.

5.1 Experimental Procedure

The design of the experiments was driven by two objectives. Firstly, the training procedure and measurements should be as close as possible to the intended application scenario. Secondly, the evaluation should be free from common pitfalls identified in evaluations of learning-based security systems [4]. Hence, the following procedure was applied for each constellation of traffic mapping and machine learning algorithm.

Step 1: Training phase. The model was trained on the entire traffic designated for training by presenting all training samples, i.e., flow samples, packet samples, or 3-packet samples, to the machine learning algorithm.

Step 2: Detection phase on training traffic. Then, the training set itself was presented to the trained detector. We did not split the training traffic to potentially shuffle it and to calculate cross-validation rates. Although common practice, this would be inconsistent with the introduced application scenario consisting of a designated training and consecutive detection phase for traffic with a given temporal order. In addition to correctly classifying new traffic, the detector should also recognize as much traffic as possible from the training phase as normal. Therefore, we tested

[11] www.tcpdump.org

[12] High security regulations of this critical infrastructure do not allow us to publish the traffic traces due to non-disclosure agreements.

Table 5: Dataset characteristics and number of validation samples: For labeled data normal and anomalous ones, and for real data the total number (*total*) and the number of samples that are not present in the training set (*new in valid.*) are stated.

	public testbed or simulation data								*traffic from real infrastructure*					
	Lemay		**WDT**		**DNP3**		**S7comm**		**DMZ**		**plant control**		**steam production**	
OT protocol(s)	Modbus/TCP		Modbus/TCP		DNP3, GOOSE		S7comm		*none*		*none*		Modbus/TCP	
characteristics of training and validation sets														
	training	validation	training	validation	training	validation	training	validation	training	validation	training	validation	training	validation
duration (hh:mm:ss)	00:59:17	00:21:40	00:57:03	01:36:04	24:01:25	23:59:59	08:18:56	09:00:25	00:59:49	00:59:49	00:59:50	00:59:51	00:57:12	00:57:22
duration (s)	3,557	1,300	3,423	5,764	86,485	86,399	29,936	32,425	3,589	3,589	3,590	3,591	3,432	3,442
# flows	13,626	2,455	17,989	46,234	558	624	235	272	20,338	19,452	4,209	4,138	82,305	81,925
# packets	134,690	30,063	7,757,289	16,549,425	422,200	1,567,529	421,080	1,802,757	5,147,077	5,106,618	145,647	145,322	602,404	600,692
# bytes (10^3)	15,017	2,909	506,750	5,242,365	57,611	127,081	57,809	173,836	1,522,958	1,539,162	21,999	22,095	46,687	46,622
∅packet size	111	97	65	317	136	81	137	96	296	301	151	152	78	78
# flows / sec	4	2	5	8	0.01	0.01	0.01	0.01	6	5	1	1	24	24
# pkts / sec	38	23	2266	2871	5	18	14	56	1434	1423	41	40	176	175
# bytes / sec (10^3)	4	2	148	910	1	1	2	5	424	429	6	6	14	14
# MAC addresses	17	9	7	9	9	13	8	13	58	55	33	33	41	40
breakdown of the validation set														
	normal	anomalous	normal	anomalous	normal	anomalous	normal	anomalous	total	new in valid.	total	new in valid.	total	new in valid.
# flow samples	2,435	20	22,867	23,367	601	23	251	21	19,452	7,092	4,138	2,937	81,925	61,496
# packet samples	21,939	8,124	9,851,133	6,698,292	1,556,112	11,417	1,137,294	665,463	5,106,618	3,239,591	145,322	39,431	600,692	26,469
# 3-packet samples	21,802	8,259	9,850,523	6,698,900	1,544,883	22,644	438,013	1,364,742	5,106,616	3,810,640	145,320	127,735	600,690	182,760

the detector on the training set, determined the number of correctly classified normal samples as *true negatives* (TN) and the false alarms as *false positives* (FP). We derived the training accuracy[13] as:

$$tacc = \frac{TN}{TN + FP}$$

Step 3: Detection phase on validation traffic. Finally, the validation set was presented to the trained detector. In the case of the public datasets, this part contains normal and anomalous traffic. For each sample, the decision was compared with the actual label of the sample. Besides the TN and FP from the normal samples, the number of correctly identified and undetected malicious samples were determined from the anomalous samples as *true positives* (TP) and *false negatives* (FN). From this, we calculated the balanced accuracy as measure on the validation set, which normalizes the imbalance between normal and anomalous sample sets:

$$vacc = \frac{1}{2}\left(\frac{TN}{TN + FP} + \frac{TP}{TP + FN}\right)$$

To have a single measure that expresses the same importance to the potential re-recognition of the training set as to the detection capability in the monitoring phase, we combine the two measures by equally weighting them:

$$bacc = \frac{tacc + vacc}{2}$$

The validation part of the real datasets only contains normal traffic. Here, the adaptions $vacc^-$ and $bacc^-$ are defined by:

$$vacc^- = \frac{TN}{TN + FP} \quad , \quad bacc^- = \frac{tacc + vacc^-}{2}$$

Model identification. We performed the protocol-agnostic modeling of normal traffic as detection base by the conversion of network traffic to samples as explained in Section 3.1 for each dataset presented in Section 4. Considering the four one-class learning algorithms (cf. Section 3.2), this resulted in 84 setups defined by the dataset, mapping type, and the algorithm. For each setup, we trained and evaluated a series of models. We carefully reviewed the hyperparameters of these algorithms and examined 100 autoencoder designs and 1000 hyperparameter constellations for the remaining three algorithms for each sample type. A list of the hyperparameters with corresponding value range is given in Table 11.

The recommended procedure for model selection is to split the data available for training into a part that is actually used for training numerous models with different hyperparameter constellations, and a separate part that is used to prevalidate all trained models and to select the best one for the actual evaluation. This procedure is effective for models that are created using multiple classes. For our one-class case, however, it is not possible to determine a suitable model in this way. It can only show which models generalize well from seen normal traffic (in the training phase) to potentially unseen normal traffic in the prevalidation phase. However, this procedure does not provide any indication of how well the models can recognize anomalies, the class that is not present in the prevalidation phase. Cross-validation cannot help here either. Thus, it is not possible to make a good decision for a model, since theoretically all good models can still be overfitted to normal traffic and are not able to detect any anomalies. However, to show the *potential* of anomaly detection for OT networks, which is the goal of this work, we have therefore evaluated *all* models (not just a preselected one) on the mixed traffic part and determined as best the one that can recognize both parts well, i.e., has the best *bacc* value.

5.2 Detection Effectiveness (Q1)

In order to determine the detection success when using protocol-agnostic OT traffic inputs, we identified the model with the maximum accuracy *bacc* from each of the 84 setups (cf. Section 5.1). By using this value for selection, we deliberately assigned the same importance to the recognition of the training set as to the detection

[13] Accuracy measures are used because the measures *precision, recall*, and consequently *f-score* are not applicable to the sets without anomalies (all sample sets for training and all sets derived from the real datasets).

capability in the monitoring phase. This is because every detector, regardless of whether it is specifically designed for OT, must be able to recognize far more normal traffic in its lifetime than attacks or misconfigurations. And this normal traffic was very likely already part of the training phase, especially in view of the homogeneity associated with OT [24, 40, 96]. The following analyses, however, focus on the balanced accuracy on the validation set (*vacc*) designated for testing the detectors, which is in line with usual evaluations.

Specifics of the WDT dataset. In the course of the experiments, we noticed that many models generated on packet samples of the WDT dataset achieved a *vacc* value of 75%. This seems to be a natural limit. We found that roughly 15% of the samples composed of anomalous labeled packets were also contained in the training set created from normal samples. Due to these *label mismatches*, the reachable detection rate is limited from the outset. Either the WDT dataset is labeled by a procedure that hits too many normal packets in parallel to an attack phase or flooding is performed by replaying legitimate packets. There are three ways to limit the negative effects of label mismatches on the performance of models. These are *label learning with label noise*, *label cleaning*, and *label noise identification* [56, 81]. We applied the last one and created a fixed validation set in which the anomalous packet samples that are also part of the training set were relabeled as normal ones. To give a fair measure for the models found, we also determined the effectiveness measures for this set as $vacc^f$ and $bacc^f$. For comparability with other research results, however, our discussions focus on results with the originally labeled sample set.

5.2.1 Single-model Results. Table 6 summarizes the balanced accuracy results for the 48 setups, defined by dataset, sample type, and machine learning algorithm, examined for the four public datasets. In the appendix also the precision, recall, and F1-score are provided for these models (cf. Table 13) in A.4 along with an explanation, why these measures are less suitable than balanced accuracy for the given detection problem. As no anomalies were available for the evaluation of the 36 real-traffic models, only the recognition of normal traffic ($vacc^-$) could be measured for them. Since this represents only half of the coin regarding local effectiveness, the results are only given for information in the appendix (cf. Section A.5). For these models built from real traffic, however, at least the *global* effectiveness can be completely determined. This is examined in Section 5.4 by using the models on the validation traffic of the four public datasets that contain anomalies. In Figure 3 the balanced accuracy results on the validation sets (*vacc*) are additionally visualized for all setups on public and real traffic. According to Table 6 twenty of the 48 models on public traffic have a *vacc* over 0.90, eleven over 0.95, and six over 0.98. The overall best models with an accuracy of 1.00 were created when traffic flow samples were used. This was two times the case, both for the dataset S7comm.

Sample types and algorithms. For flow samples the Isolation Forest algorithm is particular suited. It gives the best results for three of the four datasets. In contrast, it is the weakest algorithm on samples from single packets or packet sequences. Here, the Elliptic Envelope algorithm was in six of eight cases the best method per combination of dataset and sample type.

Attack-level results. We further analyzed the degree to which attacks and the individual attack types of the four public datasets are detected. Table 7 summarizes the results using the best model identified per dataset and mapping presented in Table 6. For maximum traceability, the names of the anomaly types are given as used in the original data. For the Lemay dataset all attack types have a medium intensity and are detected by all models. In case of the WDT dataset, all attacks are very intensive indicated by the high mean number of malicious samples per attack type, with the exception of several targeted scans. Here, 4 of 5 attack types are detected 100% by at least one of the three best models. Only the unspecified type *anomaly* can only be detected in 98% of the occurences. What is remarkable are the almost opposite results when using single-packet and 3-packet samples on the attack types *anomaly* and *scan*, which is a point we will further investigate. In contrast, the DNP3 dataset consists nearly exclusively of very subtle attacks. Here, 13 of 16 attack types contain in average less than three packets per attack. From these 13 types more than the half can be detected well. Unlike the WDT dataset, the models do not perform well on four of six types of Man-in-the-Middle attacks. As expectable, fine-grained injection attacks represent a difficulty for the protocol-agnostic detection. However, the results also show that a detection of such subtle attacks, in which the modification of payload data must be identified, are possible with protocol-agnostic models. Likewise, the S7comm attack types are very subtle and only differ significantly in their number. Nevertheless, all 22 types are perfectly recognized using single-packet samples.

False-alarm rates. Given the packet rates in the datasets of up to 2871 packets per second (cf. Table 5), even a very small proportion of misclassifications can lead to a very high number of false alarms. In our scenario, the false-positive rate can be decreased by a procedure neither limiting the detection sensitivity nor the speed for reaction to an attack. Although our approach for continuous detection initially evaluates every flow and every packet, not every anomaly has to be reported individually. As long as it is not defined too long, anomalies can be collected over a certain period, duplications can be eliminated and the number of alarms reduced. In critical infrastructures, automatic responses to attacks, such as immediate disconnections of systems from the network, are generally not installed, as this would jeopardize rather than protect the required 24/7 availability. The established response and reporting procedures are therefore human-driven. As the speed of human reception and decision-making is limited, extending the reporting time to an appropriate period does not compromise the immediate human reaction to malicious events. We therefore examined the aggregation of false alarms for periods of 1 to 300 seconds. The effects for using packet and 3-packet samples as input are summarized in Figure 4. The concrete hourly rate of false alarms for an aggregation period of 30 seconds is stated in Table 8. The false-alarm rates on flow samples could not be reduced by aggregation. Since their frequency is lower, the initial and aggregated rates correspond to the numbers in Table 8.

5.2.2 Discussion. Although the detection performance and false-alarm rate achieved is not perfect, it should be evaluated in terms of the combination of challenges that the detection approach takes up. The detection capability was generated solely from training on normal traffic in order to be independent of any attack examples and to be prepared for zero-day attacks. Every flow and packet is part of

Table 6: Single-model effectiveness on the public datasets: Balanced detection accuracy for validation samples (*vacc*) and training and validation samples combined (*bacc*) of the best model identified for each setup of traffic perspective, algorithm, and dataset. The best model per sample type is printed in bold, the overall best model for the dataset is additionally underlined.

		flow samples				packet samples				3-packet samples			
		envelope	iforest	OCSVM	CAE	envelope	iforest	OCSVM	CAE	envelope	iforest	OCSVM	CAE
Lemay	*vacc*	0.8487	**0.9479**	0.8477	0.8227	0.5401	0.5405	**0.9454**	0.9374	0.9324	0.9694	0.9348	**0.9795**
	bacc	0.9223	0.9664	0.8967	0.8846	0.7655	0.7673	0.9216	0.9169	0.9331	0.9532	0.9196	0.9535
WDT	*vacc*	0.8467	**0.8544**	0.8468	0.8258	**0.7572**	0.7524	0.7541	0.7523	**0.7595**	0.7568	0.7200	0.7204
	bacc	0.9233	0.9260	0.9234	0.8839	0.8739	0.8617	0.8658	0.8614	0.8774	0.8707	0.8591	0.8602
	$vacc^f$	0.9471	0.9581	0.9471	0.9262	0.9088	0.9001	0.9033	0.8999	0.8337	0.8297	0.7840	0.7846
	$bacc^f$	0.9735	0.9778	0.9735	0.9341	0.9497	0.9356	0.9404	0.9352	0.9145	0.9071	0.8911	0.8923
DNP3	*vacc*	0.8790	**0.9026**	0.8004	0.7979	**0.9395**	0.7510	0.5441	0.5668	**0.9566**	0.8539	0.8139	0.5877
	bacc	0.9368	0.9378	0.9002	0.8989	0.9441	0.8435	0.7615	0.6555	0.9763	0.9056	0.8257	0.7435
S7comm	*vacc*	0.9960	0.9920	**1.0000**	**1.0000**	**0.9962**	0.9216	0.5000	0.7602	**0.9809**	0.9684	0.9347	0.9736
	bacc	0.9958	0.9896	1.0000	1.0000	0.9907	0.8110	0.7500	0.8604	0.9898	0.9707	0.9673	0.9789

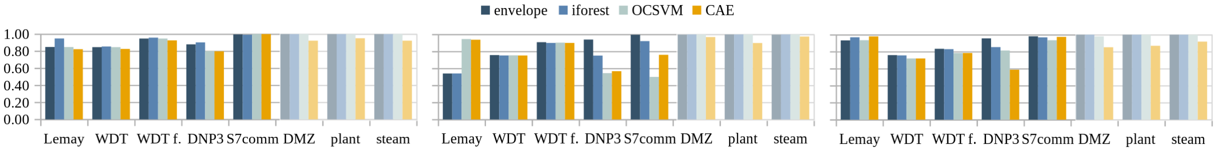


Figure 3: Validation accuracy (*vacc*) per model for flow samples, packet samples and 3-packet samples (from left to right), with datasets lacking anomalies distinguished by less opacity.

Table 7: Ratio of detected attacks and intensity per attack type for all public datasets.

	Lemay						WDT					
	1_moving_two_files_modbus_6RTU	2_exploit_ms08_netapi_modbus_6RTU_with_operate	3_CnC_uploading_exe_modbus_6RTU_with_operate	4_characterization_modbus_6RTU_with_operate	5_send_a_fake_command_modbus_6RTU_with_operate	**total**	DoS	MITM	anomaly	physical fault	scan	**total**
#attacks	4	10	2	59	1	76	5	2	266	8	30	311
detection ratio per sample type												
flows	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.45	0.50	0.80	0.50
packets	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.08	1.00	1.00	0.83
3 pkts.	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.98	1.00	0.00	0.89
intensity of the attack type in # malicious packets per attack												
min.	5	1	28	1	10		87k	150k	1	94k	1	
max.	41	487	93	199	10		2971k	197k	334k	310k	1	
mean	19	120	61	114	10		753k	174k	5023	155k	1	

	DNP3																
	ARP_poisoning	DNP3_reconnaissance	MITM_forwarding_from_attacker	MITM_forwarding_to_attacker	MITM_from_attacker	MITM_hijack_injection	MITM_modification_ImmedFreezeNR	MITM_to_attacker	injection_ColdRestart	injection_FreezeObj	injection_WarmRestart	master_flooding_freeze	master_masquerading	master_replay	master_replay_flooding	nmap_reconnaissance	**total**
#attacks	125	5	353	97	15	15	6	7	6	18	3	4026	142	206	4544	12	9580
detection ratio per sample type																	
flows	–	1.00	–	–	–	–	–	–	–	–	–	1.00	1.00	1.00	1.00	1.00	1.00
packets	1.00	1.00	0.89	0.62	0.93	0.40	0.00	0.29	0.00	0.00	0.00	1.00	1.00	1.00	1.00	0.00	0.99
3 pkts.	1.00	1.00	0.41	0.29	0.33	0.53	0.17	0.71	0.33	0.56	0.67	1.00	1.00	1.00	1.00	0.83	0.97
intensity of the attack type in # malicious packets per attack																	
min.	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	
max.	7	3	12	10	8	2	2	4	1	1	1	2	2	2	2	217	
mean	1	2	3	2	4	1	2	1	1	1	1	1	1	1	1	34	

	S7comm											
	ChangeLowerThreshold	ChangeUpperThreshold	ChangeUpperThreshold_Flooding	ConveyorBeltGateChangeDirection	ConveyorBeltGateChangeDirection_Flooding	ConveyorBeltOff	ConveyorBeltOff_Flooding	ConveyorBeltOn	ConveyorBeltOn_Flooding	ConveyorBeltReset	WaterTankOnManu	WaterTankOnManu_Flooding
#attacks	6	6	18909	8	22178	6	56357	18	118386	10	8	17608
detection ratio per sample type												
flows	1.00	–	–	1.00	–	–	1.00	1.00	1.00	1.00	–	–
packets	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
3 pkts.	0.33	0.33	1.00	1.00	0.99	1.00	0.99	1.00	0.99	1.00	1.00	0.99
intensity of the attack type in # malicious packets per attack												
min.	1	1	1	1	1	1	1	1	1	1	1	1
max.	1	1	2	1	1	1	1	1	1	1	1	1
mean	1	1	1	1	1	1	1	1	1	1	1	1

	S7comm												total
	ReactorOff	ReactorOff_Flooding	ReactorOn	ReactorOn_Flooding	WaterTankOff	WaterTankOff_Flooding	WaterTankOnAuto	WaterTankOnAuto_Flooding	EmergencyStop	GlobalReset	**total**		
#attacks	4	57931	28	80225	4	85158	10	208596	2	4	665k		675k
detection ratio per sample type													
flows	–	1.00	1.00	–	–	–	1.00	1.00	1.00	1.00	1.00		0.999
packets	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00		0.999
3 pkts.	1.00	0.99	0.93	0.99	1.00	0.99	1.00	0.99	1.00	1.00	0.99		0.989
intensity of the attack type in # malicious packets per attack													
min.	1	1	1	1	1	1	1	1	1	1			
max.	1	1	1	1	1	1	1	1	1	1			
mean	1	1	1	1	1	1	1	1	1	1			

Legend: –: attack type not visible in flows; k: 10^3

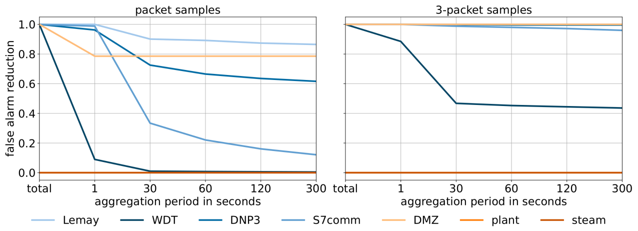


Figure 4: False-alarm drop for different aggregation periods.

Table 8: False-alarm rates per hour for the best model per dataset (highlighted in Tables 6 and 14) and sample type when using an aggregation period of 30 seconds.

	Lemay	WDT	DNP3	S7comm	DMZ	plant	steam
flow samples	10	52	5	0	0	2	0
packet samples	100	404	1506	312	11	0	0
3-packet samples	688	9682	502	492	3; 26	0	0

the detection for maximum local effectiveness. They are prepared as samples without any knowledge or assumption about the protocol stack, without neglecting the analysis of user data. The fine-grained analysis of packets and packet sequences is accompanied by an immense number of samples, from which an enormous number of correct classifications are simultaneously achieved despite the initially remaining false-alarm rate. An alternative approach to reduce false alarms is to define a threshold for the number of anomalies

Table 9: Prior results on traffic from public datasets (best stated in case of several results) compared to our work (best *vacc* from Table 6) for the respective detection measures.

	Lemay						WDT				DNP3		S7comm	
	[8]	[93]	[1]	[92]	[74]	**here**	[48]*	[24]*	[21]	**here**	[8]	**here**	[8]	**here**
accuracy	0.99	0.72	0.99	0.94	1.00	**0.97**	0.75	0.75	0.86	**0.85**	0.77	**0.95**	0.97	**1.00**
f-score	(AUC)	0.91	0.99	0.74	1.00	**0.95**	0.19	0.02	0.76	**0.83**	(AUC)	**0.75**	(AUC)	**1.00**
precision				0.74		**0.92**	0.42	0.80		**0.99**		**0.63**		**1.00**

Results for approaches marked with * are taken from [96].

that must be reached in order to generate an alarm. We deliberately do not pursue this approach, as this method is associated with blindness to all attacks below the threshold value. With our approach, an attacker cannot infiltrate the system with little activity, as every specific attack trace (anomalous sample) is detected and alerted, only the duplication of alerts is catched.

5.2.3 *Comparison with Previous Works.*

In order to further assess the results regarding local effectiveness on the public datasets we identified previous detection approaches applied to them. For this, we reviewed every publication citing the papers that introduced the datasets and identified 23 publications providing detection measures on the traffic parts of at least one of the four datasets. We do not go into 13 of the methods [2, 9, 27, 32–34, 41, 45, 82, 87, 91, 96, 98], as they do not address the greater challenge of using only one class for setting up the detection [7, 40]. The works that pursue a one-class-based detection, as we do, are summarized in Table 9.

Lemay dataset. As argued in Section 4.1, all approaches evaluated on the Lemay dataset before April 2023 could not (properly) consider 40% of the validation packets we used in our experiments because not all of them had correct labels. Therefore, better results using anomaly detection (based on single-class training) on that dataset [1, 8, 74] cannot be directly related to our results.

WDT dataset. In the case of the WDT dataset, a nominal better accuracy (by 0.01) is only reached by fusing features from the physical processes with additional features extracted from network traffic to work with these *cyber-physical fusion features* [21]. Consequently, this approach is not purely communication-based as we realize it for the reasons we gave in Section 2.2 and the drawbacks of detection approaches based on process data further substantiated in Section A.1 in the appendix.

DNP3 and S7comm datasets. The only one-class approach [8] applied to both captures is also designed as protocol-agnostic procedure incorporating payload. It is evaluated on three of the public datasets (Lemay, DNP3 and S7comm). Unfortunately, the detection results are only provided as receiver operating characteristic (ROC) curve and area under the curve (AUC) for true-positive (detected attack data) to false-positive (false alarms) relations, from which the true performance of the classifier, including low or high precision, cannot be inferred [4]. Furthermore, the approach relies on the construction of hidden semi-Markov models, Gaussian mixture models, *and* probalistic suffix trees for the detection. Since the computational complexity is stated as quadratic to the states of the Markov models and size of the suffix tree's range, the advantage of combining these three complex models over the use of already available and sophisticated (machine learning) routines is not clear.

5.2.4 *Ensemble Detection Effectiveness.*

As we could not identify a single dominant machine learning algorithm for the protocol-agnostic anomaly detection, we also carried out a comprehensive evaluation of ensembles of these models in order to determine if their combination outperforms single models. For each dataset, all models were used in parallel and different decision strategies were measured as provided in Section A.6 in the appendix. The main result was that ensembles are not more effective than the best single model per dataset, but they can grant this property if the best model of a group of models is not known in advance. In practise, this is the case as long as the models have not been confronted with attacks and their individual performance could not been identified yet.

5.3 Detection Efficiency (Q2, Q4)

The aim of the efficiency analysis is to assess whether the effectiveness of the approach is compromised by insufficient speed. This would be the case if the detection process were too slow to process all traffic in real time. Monitoring data would be discarded regularly and the process would partially be blind against attacks. The autoencoder measurements were carried out on an AMD machine with 5700 bogomips per core (AMD EPYC 7443 24-Core Processor) with 264 GB RAM. All other measurements were made on an AMD machine with 7786 bogomips per core (AMD Ryzen Threadripper PRO 3955WX) with 64 GB RAM. All experiments were executed using a prototype implementation not yet realizing multi-threading. We discuss the mean times. This does not consider traffic peaks, but it is sufficient to evaluate whether the processing speed and data rates correlate in such a way that the system with a buffering of traffic peaks could process the whole traffic of the network.

5.3.1 *Results on Packet Samples.*

It is sufficient to limit the evaluation on the processing of the most complex sampling procedure, which is the single-packet sampling. The effort is much higher than for the analysis of fewer flows with also fewer features. The results are also valid for 3-packet samples because a 3-packet sample is created with every observed single packet (with its two predecessors). For the 3-packet variant, the necessary parsing (of maximum 11 payload bytes) and concatenation with prior samples is less costly than sampling the larger payload prefix in the course of single-packet sampling (57 bytes). Figure 5 summarizes the packet sample processing load in percentage. It is calculated as fraction from the number of packet samples that can be processed per second by the used architecture and the packet rate in the datasets, i.e., the number of packet samples that have to be processed per second for real-time capability of the detection. The processable number of packet samples per second was derived by the time needed to process a single packet sample, observable in Table 16 in Section A.7.

Datasets. The most demanding public dataset is WDT, the real dataset with highest packet rate is the DMZ trace. The first one corresponds to 2871 packets per second, the second one to 1423 packets (cf. Table 5). In the case of the WDT dataset, the number of packet samples to be trained per second exceeds the capability of the detection system by a factor of 15. For the DMZ trace, an overload of 27 to 333% was observed. The prediction procedure of the models is less intensive than the fitting step, so that the detection process is only too slow for the two datasets when using the Isolation Forest algorithm. For the five remaining datasets, the

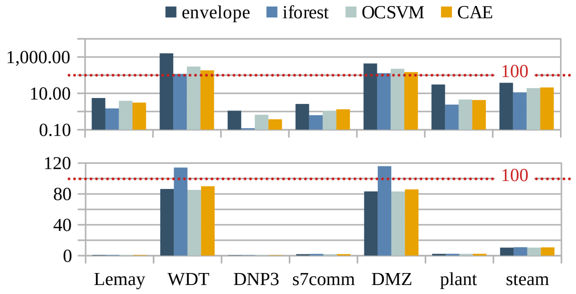


Figure 5: Packet sample processing load in % while model training (top) and prediction of validation samples (bottom), as summary of the measurements provided in Section A.7.

training and the detection can be performed in real time using any of the four algorithms. Here, the number of processable packet samples is (far) below the packet rate of the respective dataset.

Algorithms. The dominant factors for the sample processing times are determined by the model fitting step in the training phase and the prediction performed by the model in the detection phase. The Elliptic Envelope algorithm, which is the most represented one among the best models in Section 5.2, is also the most time-consuming algorithm regarding model training. The algorithm with the most efficient fitting procedure is Isolation Forest. Regarding prediction, however, the tree-based algorithm is the less efficient one for six of the datasets. The One-class Support Vector Machine followed by the Elliptic Envelope are the fastest algorithms.

5.3.2 Discussion. From a practical perspective, only the detection (prediction) must be real-time capable. When no continuous learning is applied, the training can be performed offline, eventually even apart from the network using traffic captures for the machine learning on dedicated training servers. In critical infrastructures, several aspects speak in favor of carrying out the detection phase on the premises[14]. This requires the algorithms to keep up with data rates using hardware that is realistic to be placed in the infrastructure. The results, with the exception of 2/28 of the cases, show that the protocol-agnostic detection scheme, even at the stage of a single-thread implementation, executed on such hardware can completely analyze the traffic of OT networks. In 24/28 cases the experiment workstation was below 11% on average for fulfilling the detection task. We conclude that a self-learning anomaly detection including payload in a protocol-agnostic manner, especially when extended to a hardware-efficient multi-threading application, can analyze data rates of OT networks in real time without dropping traffic. This demonstrates that the local and global effectiveness of the proposed approach is not compromised by insufficient efficiency.

5.3.3 Comparison with Previous Works. To the best of our knowledge, a comparable measurement of individual processing steps with relation to real OT data rates has not been performed, yet. Relevant papers addressing performance issues are rare (cf. Table 1) and did not go beyond quantifying a single step, such as parsing times [74].

5.4 Model Transfer Potential (Q3, Q4)

The protocol-agnostic nature of the detection approach raises the question to what extent models trained on one dataset can be used for attack detection on traffic from another dataset. If there is a reuse potential, a knowledge transfer among infrastructures would be possible and the training-based configuration of the detection process could be minimized. For answering the question, we used the models identified in Section 5.2 to classify the associated sample sets from all other six datasets. We applied the already trained models to imitate the scenario that a detector trained in one network is applied in another one. All 504 possible transfer scenarios (7 × 6 dataset combinations, three sample types, and four machine learning algorithms) were examined. In addition to the overall results, we examined two related subquestions. (1) Does the transferability of models depend on the similarity of the protocol mixes present in the model's source dataset and the transfer target dataset and (2) do the models created from the real datasets possess the capability to correctly detect anomalies in foreign data? The last question arises from the fact that the real datasets could only partially be evaluated due to the impossibility of generating anomalies in their networks.

Overall transfer results. The results are summarized in Figure 6 and Table 10. The dominant learning algorithm in the experiments with an accuracy of higher 90 % was the OCSVM method, with 34 occurences, followed by the Elliptic Envelope algorithm with 22 models and the Isolation Forest procedure with 19 representatives. In 33 cases of this group (roughly 6.5 % from all experiments), even a perfect balanced accuracy of 100 % was reached. Among these models, 10 Isolation Forest models and 10 OCSVM models were present, followed by 7 Convolutional Autoencoders. From the amounts collected in Table 10 we conclude that there is a clear potential for the reuse of one-class trained attack detectors in OT networks when traffic is analyzed in a protocol-agnostic manner. Nevertheless, the accuracy range of a model when applied to foreign data is very high in most cases, as it can be seen in Figure 6. Unfortunately it cannot (yet) be concluded that a relatively high minimum accuracy is guaranteed for a certain combination of sample type and algorithm. Here, further tuning, such as a little re-learning in the new infrastructure, could help lower the hurdles.

Sensitivity to protocol mix similarity. We investigated the transfer scenarios where source and target network are dominated by the same OT protocol, but could not identify a positive transfer effect. We provide the results in Section A.8 in the appendix.

Table 10: Amount of transfer scenarios per sample type and reached validation accuracy *vacc*.

vacc ≥	flows	packets	3 packets	total	% scenarios
0.85	52	24	24	100	19.84
0.90	51	21	15	87	17.26
0.95	46	12	15	73	14.48
0.98	36	12	13	61	12.10
1.00	22	4	7	33	6.55

Anomaly detection capability of the models built from real traffic. We analyzed the transfer scenarios with a model trained on a real network and applied to a public dataset that provides anomalies for evaluation. For all real datasets (DMZ, plant control,

[14] These include data protection and independence from external systems for security-relevant operations.

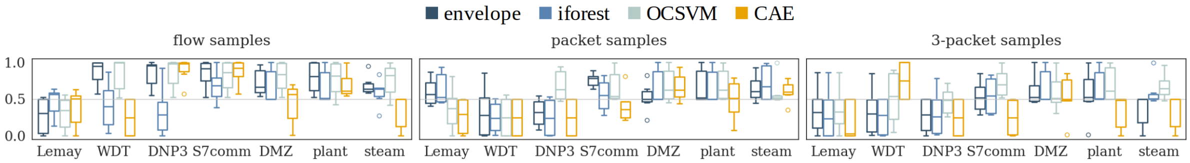


Figure 6: All model transfers: Detection capability (*vacc*) of models trained on the indicated dataset and applied to the validation sample sets of all other datasets. It is summarized for the sample types (subfigures) and the four models trained per dataset (x-axis of each subfigure) how each model performs on the same sample type of the other six datasets.

and steam production), a model was found that could be proven at least on one testbed dataset to be capable to effectively distinguish normal traffic from anomalies. In total, a model was found for each of the real datasets, for which a detection capability higher 90% was achieved. This proves the feasibility of the protocol-agnostic, one-class trained anomaly detection also for real traffic, what was not possible to show in Section 5.2 due to the lack of real anomalies.

6 CONCLUSIONS

In this paper, we present a protocol-agnostic anomaly detection approach for OT traffic. It can be applied ❶ without any assumptions about the network, OT technology, or the OT processes installed ❷ directly on a network switch without dependence on further external (logging) systems, ❸ using only normal traffic for modeling the detection logic addressing the absence of attack examples in most real networks and the ability to detect zero-day attacks, ❹ to detect not only comparatively easily observable DoS attacks but also fine-grained injection attacks by analyzing also packet payloads in a protocol-agnostic manner, and ❺ without blindness because the detection is applied to every flow and packet of the complete traffic mix without any filtering. We have evaluated the approach for three traffic perspectives (network flows, packets, and packet sequences) with four machine learning algorithms on seven datasets including four public and three real OT traffic traces. The outcome of the analyses referring to the initial questions are:

Detection effectiveness (Question 1). We did an evaluation of 84 combinations of traffic perspective, machine learning algorithm, and dataset. For the four public datasets, we measured in 20 of 48 setups a balanced accuracy higher 0.90, for eleven higher 0.95, and for six higher 0.98. These results show that the same (or even better) detection capability is achieved, for which multiple protocol-*specific* and more complex methods were required previously (cf. Section 5.2.3). Besides the local effectiveness, the results on several, very diverse datasets also demonstrate the global effectiveness of the protcol-agnostic detection. We further analyzed which heavy and subtle attack types are detected to which degree and found that also fine-grained injection attacks can be detected in a protocol-agnostic manner (cf. Section 5.2.1). We then worked out which false-alarm rates the remaining false decisions still correspond to, also for real traffic rates, and proposed and demonstrated a measure for minimizing them without reducing the detection capability.

Detection efficiency (Question 2). We further analyzed if the local effectiveness could be compromised by a too slow detection process, using the most expensive type of traffic samples. We measured the time consumption of each step of the traffic preparation and analysis, which has neither been done before at this granularity nor with the inclusion of several real datasets. We worked out that, although a significant amount of packet payload is part of the detection process, it can be implemented with real-time capability (cf. Section 5.3). Hence, the detection is not limited by dropped traffic.

Model transfer to foreign traffic (Question 3). The protocol-agnostic nature of the detection approach allows to potentially use models trained in an OT network (here on one dataset) for the detection in foreign networks (on other datasets). This transferability was examined for all 504 potential transfer scenarios (7 × 6 dataset combinations, 3 sample types, and 4 algorithms). In 17% of the cases a balanced accuracy higher 0.90 and in 6.5% 1.0, i.e., the perfect classificiation of normal and anomalous traffic, was achieved. From these numbers we conclude a clear potential of the reuse of one-class trained attack detectors in OT networks if the traffic is analyzed in a protocol-agnostic manner. However, we could not (yet) prove a beneficial effect on the transfer if realized with a similar protocol mix in the source and target traffic.

Effect of different traffic perspectives (Question 4). Most of the best single models have been derived from flow and 3-packet samples (cf. Table 6). The flow perspective is also particularly beneficial with regard to the transfer of models (cf. Table 10). Unfortunately, flow samples are not applicable on traffic transmitted without the IP layer, which causes blindness of this perspective regarding GOOSE traffic and related attack types, for instance, as observable in Table 7. Regarding the sample types including payload, we found out that packet sequences are more beneficial for detection purposes than modeling single packets. They have a higher count in the best models per dataset (cf. Table 6). Their detection success for some attack types, however, is inverse to that when using single packets (cf. Table 7). Hence, single packets and packet sequences as detection perspectives should be combined.

After measuring the local and global effectiveness and efficiency of our protocol-agnostic detection scheme in this paper, we will in a next step compare the models regarding the reflected traffic knowledge to provide explanations for alarms to potential users.

Material. We provide relevant material for comparative analyses and further developments online: https://wissenwerk.de/raid2024

Acknowledgments

We like to thank Andreas Paul and René Larisch for fruitful discussions. Special thanks go to Heiko Kanisch for his trust and relentless efforts to bridge the gap between science and real world. Without him, this research would not have been possible. Nor would this be the case without Franka's muse and favorites.

References

[1] Simon Duque Anton, Lia Ahrens, Daniel Fraunholz, and Hans Dieter Schotten. 2018. Time is of the Essence: Machine Learning-based Intrusion Detection in Industrial Time Series Data. In *IEEE International Conference on Data Mining Workshops (ICDMW)*. IEEE, 1–6.

[2] Simon Duque Anton, Suneetha Kanoor, Daniel Fraunholz, and Hans Dieter Schotten. 2018. Evaluation of Machine Learning-based Anomaly Detection Algorithms on an Industrial Modbus/TCP Data Set. In *International Conference on Availability, Reliability and Security (ARES)*. 1–9.

[3] Simon D. Duque Anton, Sapna Sinha, and Hans Dieter Schotten. 2019. Anomaly-based Intrusion Detection in Industrial Data with SVM and Random Forests. In *International Conference on Software, Telecommunications and Computer Networks (SoftCOM)*. IEEE, 1–6.

[4] Daniel Arp, Erwin Quiring, Feargus Pendlebury, Alexander Warnecke, Fabio Pierazzi, Christian Wressnegger, Lorenzo Cavallaro, and Konrad Rieck. 2022. Dos and Don'ts of Machine Learning in Computer Security. In *USENIX Security Symposium (USENIX Security)*. 3971–3988.

[5] Rafael Ramos Regis Barbosa, Ramin Sadre, and Aiko Pras. 2012. Towards Periodicity based Anomaly Detection in SCADA Networks. In *IEEE International Conference on Emerging Technologies & Factory Automation (ETFA)*. IEEE, 1–4.

[6] Deval Bhamare, Maede Zolanvari, Aiman Erbad, Raj Jain, Khaled Khan, and Nader Meskin. 2020. Cybersecurity for Industrial Control Systems: A Survey. *Computers & Security* 89 (2020), 101677.

[7] Monowar H. Bhuyan, Dhruba Kumar Bhattacharyya, and Jugal K. Kalita. 2013. Network Anomaly Detection: Methods, Systems and Tools. *IEEE Communications Surveys & Tutorials* 16, 1 (2013), 303–336.

[8] Jun Cai, Qi Wang, Jianzhen Luo, Yan Liu, and Liping Liao. 2021. CapBad: Content-agnostic, Payload-based Anomaly Detector for Industrial Control Protocols. *IEEE Internet of Things Journal* 9, 14 (2021), 12542–12554.

[9] Guoyan Cao, Zhaowen Feng, and Tongli Wang. 2020. A Semi-supervisory Anomaly Detection Method for Industrial Networks Security. In *IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. IEEE, 2125–2129.

[10] Marco Caselli, Emmanuele Zambon, Johanna Amann, Robin Sommer, and Frank Kargl. 2016. Specification Mining for Intrusion Detection in Networked Control Systems. In *USENIX Security Symposium (USENIX Security)*. 791–806.

[11] Marco Caselli, Emmanuele Zambon, and Frank Kargl. 2015. Sequence-aware Intrusion Detection in Industrial Control Systems. In *ACM Workshop on Cyber-Physical System Security (CPSS)*. 13–24.

[12] Marco Caselli, Emmanuele Zambon, Jonathan Petit, and Frank Kargl. 2015. Modeling Message Sequences for Intrusion Detection in Industrial Control Systems. In *IFIP 11.10 International Conference on Critical Infrastructure Protection (ICCIP)*. Springer, 49–71.

[13] Steven Cheung, Bruno Dutertre, Martin Fong, Ulf Lindqvist, Keith Skinner, and Alfonso Valdes. 2007. Using Model-based Intrusion Detection for SCADA Networks. In *SCADA Security Scientific Symposium 2007*.

[14] Ankang Chu, Yingxu Lai, and Jing Liu. 2019. Industrial Control Intrusion Detection Approach based on Multiclassification GoogLeNet-LSTM Model. *Security and Communication Networks* 2019 (2019), 1–11.

[15] Benoit Claise. 2004. Cisco Systems Netflow Services Export Version 9.

[16] Benoit Claise, Brian Trammell, and Paul Aitken. 2013. Specification of the IP Flow Information Export (IPFIX) Protocol for the Exchange of Flow Information.

[17] Mauro Conti, Denis Donadel, and Federico Turrin. 2021. A Survey on Industrial Control System Testbeds and Datasets for Security Research. *IEEE Communications Surveys & Tutorials* 23, 4 (2021), 2248–2294.

[18] Corinna Cortes and Vladimir Vapnik. 1995. Support-vector Networks. *Machine Learning* 20 (1995), 273–297.

[19] Tiago Cruz. 2018. Modbus TCP SCADA #1 Dataset. https://github.com/tjcruz-dei/ICS_PCAPS/releases/tag/MODBUSTCP%231. Online. Accessed: 2024-03-26.

[20] Cybersecurity and Infrastructure Secruity Agency (CISA). 2023. Alert ICSA-23-017-01. https://www.cisa.gov/news-events/ics-advisories/icsa-23-017-01. Online. Accessed: 2023-04-29.

[21] Yan Du, Yuanyuan Huang, Guogen Wan, and Peilin He. 2022. Deep Learning-based Cyber-Physical Feature Fusion for Anomaly Detection in Industrial Control Systems. *Mathematics* 10, 22 (2022), 4373.

[22] Luca Faramondi, Francesco Flammini, Simone Guarino, and Roberto Setola. 2021. A Hardware-in-the-Loop Water Distribution Testbed Dataset for Cyber-Physical Security Testing. *IEEE Access* 9 (2021), 122385–122396.

[23] Cheng Feng, Tingting Li, and Deeph Chana. 2017. Multi-level Anomaly Detection in Industrial Control Systems via Package Signatures and LSTM Networks. In *IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*. IEEE, 261–272.

[24] Benedikt Ferling, Justyna Chromik, Marco Caselli, and Anne Remke. 2018. Intrusion Detection for Sequence-based Attacks with Reduced Traffic Models. In *International GI/ITG Conference on Measurement, Modelling and Evaluation of Computing Systems*. Springer, 53–67.

[25] David Formby, Preethi Srinivasan, Andrew M. Leonard, Jonathan D. Rogers, and Raheem A. Beyah. 2016. Who's in Control of Your Control System? Device Fingerprinting for Cyber-Physical Systems. In *Network and Distributed System Security Symposium (NDSS)*.

[26] Wei Gao and Thomas H. Morris. 2014. On Cyber Attacks and Signature based Intrusion Detection for Modbus based Industrial Control Systems. *Journal of Digital Forensics, Security and Law* 9, 1 (2014), 3.

[27] Ying Gao, Jixiang Chen, Hongyue Miao, Binjie Song, Yiqin Lu, and Weiqiang Pan. 2022. Self-learning Spatial Distribution-based Intrusion Detection for Industrial Cyber-Physical Systems. *IEEE Transactions on Computational Social Systems* 9, 6 (2022), 1693–1702.

[28] Jonathan Goh, Sridhar Adepu, Khurum Nazir Junejo, and Aditya Mathur. 2016. A Dataset to Support Research in the Design of Secure Water Treatment Systems. In *International Conference on Critical Information Infrastructures Security (CRITIS) (Lecture Notes in Computer Science, Vol. 10242)*. Springer, 88–99.

[29] Niv Goldenberg and Avishai Wool. 2013. Accurate Modeling of Modbus/TCP for Intrusion Detection in SCADA Systems. *International Journal of Critical Infrastructure Protection (IJCIP)* 6, 2 (2013), 63–75.

[30] Jiuxiang Gu, Zhenhua Wang, Jason Kuen, Lianyang Ma, Amir Shahroudy, Bing Shuai, Ting Liu, Xingxing Wang, Gang Wang, Jianfei Cai, et al. 2018. Recent Advances in Convolutional Neural Networks. *Pattern Recognition* 77 (2018), 354–377.

[31] Simone Guarino, Luca Faramondi, Roberto Setola, and Francesco Flammini. 2021. A Hardware-in-the-Loop Water Distribution Testbed (WDT) Dataset for Cyber-Physical Security Testing. https://dx.doi.org/10.21227/rbvf-2h90. IEEE Dataport. Online. Accessed: 2022-10-10.

[32] Jun Han, Dongdong Huo, Wenqian Zhang, and Yazhe Wang. 2022. A Blockchain based Federal Abnormal Detection Method for Power Dispatching Systems. In *International Conference on Renewable Energy and Power Engineering (REPE)*. IEEE, 56–61.

[33] Gu Haoran, Lai Yingxu, Yipeng Wang, Jing Liu, Sun Motong, and Mao Beifeng. 2022. DEIDS: A Novel Intrusion Detection System for Industrial Control Systems. *Neural Computing & Applications* 34, 12 (2022), 9793–9811.

[34] Yibo Hu, Dinghua Zhang, Guoyan Cao, and Quan Pan. 2019. Network Data Analysis and Anomaly Detection using CNN Technique for Industrial Control Systems Security. In *IEEE International Conference on Systems, Man and Cybernetics (SMC)*. IEEE, 593–597.

[35] SANS Institute. 2013. CyberCity Dataset. https://assets.contentstack.io/v3/assets/blt36c2e63521272fdc/bltff8e7c1232f3bcbc/5fbd7be072a3526f28dbed75/sansholidayhack2013.pcap. Online. Accessed: 2024-03-26.

[36] Arthur S. Jacobs, Roman Beltiukov, Walter Willinger, Ronaldo A. Ferreira, Arpit Gupta, and Lisandro Z. Granville. 2022. AI/ML for Network Security: The Emperor has no Clothes. In *ACM SIGSAC Conference on Computer and Communications Security (CCS)*. 1537–1551.

[37] DFKI Kaiserslautern. 2023. IUNO Project Website with Datasets. https://projects.dfki.uni-kl.de/IUNO. Offline. Access tried: 2023-04-29.

[38] Stephan Kleber, Frank Kargl, Milan State, and Matthias Hollick. 2022. Network Message Field Type Clustering for Reverse Engineering of Unknown Binary Protocols. In *IEEE/IFIP International Conference on Dependable Systems and Networks Workshops (DSN-W)*. IEEE, 80–87.

[39] Moshe Kravchik and Asaf Shabtai. 2018. Detecting Cyber Attacks in Industrial Control Systems using Convolutional Neural Networks. In *Workshop on Cyber-Physical Systems Security and Privacy*. 72–83.

[40] Dominik Kus, Eric Wagner, Jan Pennekamp, Konrad Wolsing, Ina Berenice Fink, Markus Dahlmanns, Klaus Wehrle, and Martin Henze. 2022. A False Sense of Security? Revisiting the State of Machine Learning-based Industrial Intrusion Detection. In *ACM on Cyber-Physical System Security Workshop (CPSS)*. 73–84.

[41] Hassan Fareed M. Lahza. 2019. *Designing a Feature Construction and Selection Approach for Machine Learning-based Intrusion Detection in Industrial Control System Networks.* Ph. D. Dissertation. Queensland University of Technology.

[42] Olav Lamberts, Konrad Wolsing, Eric Wagner, Jan Pennekamp, Jan Bauer, Klaus Wehrle, and Martin Henze. 2023. [SoK] Evaluations in Industrial Intrusion Detection Research. *Journal of Systems Research* 3, 1 (2023).

[43] Antoine Lemay. 2016. Modbus Dataset from CSET 2016. https://github.com/antoine-lemay/Modbus_dataset. Online. Accessed: 2023-04-01.

[44] Antoine Lemay and José M. Fernandez. 2016. Providing SCADA Network Data Sets for Intrusion Detection Research. In *Workshop on Cyber Security Experimentation and Test (CSET)*. USENIX Association.

[45] Hongwei Li and Danai Chasaki. 2021. Ensemble Machine Learning for Intrusion Detection in Cyber-Physical Systems. In *IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*. IEEE, 1–2.

[46] Chih-Yuan Lin and Simin Nadjm-Tehrani. 2018. Understanding IEC-60870-5-104 Traffic Patterns in SCADA Networks. In *ACM Workshop on Cyber-Physical System Security (CPSS)*. 51–60.

[47] Chih-Yuan Lin and Simin Nadjm-Tehrani. 2019. Timing Patterns and Correlations in Spontaneous SCADA Traffic for Anomaly Detection. In *Symposium on Research in Attacks, Intrusions and Defenses (RAID)*. 73–88.

[48] Chih-Yuan Lin, Simin Nadjm-Tehrani, and Mikael Asplund. 2018. Timing-based Anomaly Detection in SCADA Networks. In *International Conference on Critical Information Infrastructures Security (CRITIS)*. Springer, 48–59.

[49] Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. 2008. Isolation Forest. In *IEEE International Conference on Data Mining*. IEEE, 413–422.

[50] Scott M. Lundberg and Su-In Lee. 2017. A Unified Approach to Interpreting Model Predictions. *Advances in Neural Information Processing Systems (NIPS)* 30 (2017).

[51] Leandros A. Maglaras and Jianmin Jiang. 2014. Intrusion Detection in SCADA Systems using Machine Learning Techniques. In *Science and Information Conference*. IEEE, 626–631.

[52] Chen Markman, Avishai Wool, and Alvaro A Cardenas. 2017. A New Burst-DFA Model for SCADA Anomaly Detection. In *Workshop on Cyber-Physical Systems Security and Privacy*. 1–12.

[53] Stefan Mehner, Franka Schuster, and Oliver Hohlfeld. 2022. Lights on Power Plant Control Networks. In *International Conference on Passive and Active Network Measurement (PAM)*. Springer, 470–484.

[54] Thomas Miller, Alexander Staves, Sam Maesschalck, Miriam Sturdee, and Benjamin Green. 2021. Looking Back to Look Forward: Lessons Learnt from Cyberattacks on Industrial Control Systems. *International Journal of Critical Infrastructure Protection (IJCIP)* 35 (2021), 100464.

[55] Robert Mitchell and Ing-Ray Chen. 2014. A Survey of Intrusion Detection Techniques for Cyber-Physical Systems. In *ACM CSUR*, Vol. 46. ACM.

[56] Nicolas M. Müller and Karla Markert. 2019. Identifying Mislabeled Instances in Classification Datasets. In *International Joint Conference on Neural Networks (IJCNN)*. IEEE, 1–8.

[57] David Myers. 2018. QUT S7 Communication. https://cloudstor.aarnet.edu.au/plus/index.php/s/9qFfeVmfX7K5IDH. Offline. Accessed: 2023-03-19.

[58] Gorby Kabasele Ndonda. 2019. HVAC Dataset. https://github.com/gkabasele/HVAC_Traces. Online. Accessed: 2024-03-26.

[59] NETRESEC. 2015. Capture files from 4SICS Geek Lounge. https://www.netresec.com/?page=PCAP4SICS. Online. Accessed: 2024-03-26.

[60] NETRESEC. 2015. S4x15 ICS Village PCAP Files. https://www.netresec.com/?page=DigitalBond_S4. Online. Accessed: 2024-03-26.

[61] Singapore University of Technology and Design (SUTD). 2021. Datasets Collection inlcuding Electric Power and Intelligent Control (EPIC) and Secure Water Treatment (SWaT). https://itrust.sutd.edu.sg/itrust-labs_datasets/dataset_info. Online. Accessed: 2024-03-26.

[62] Queensland University of Technology Information Security. 2017. DNP3 Cyber-Attack Datasets. https://github.com/qut-infosec/2017QUT_DNP3. Online. Accessed: 2023-04-30.

[63] Queensland University of Technology Information Security. 2017. SCADA Network Attack Datasets and Process Logs. https://github.com/qut-infosec/2017QUT_S7comm. Online. Accessed: 2023-04-30.

[64] Christopher Parian, Terry Guldimann, and Sajal Bhatia. 2020. Fooling the Master: Exploiting Weaknesses in the Modbus Protocol. *Procedia Computer Science* 171 (2020), 2453–2458.

[65] Andreas Paul, Franka Schuster, and Hartmut König. 2013. Towards the Protection of Industrial Control Systems – Conclusions of a Vulnerability Analysis of Profinet IO. In *International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment (DIMVA)*. Springer, 160–176.

[66] Andreas Paul, Franka Schuster, and Hartmut König. 2023. Whitelisting for Characterizing and Monitoring Process Control Communication. In *International Conference on Network and System Security (NSS)*. Springer, 23–45.

[67] Rocio Lopez Perez, Florian Adamsky, Ridha Soua, and Thomas Engel. 2018. Machine Learning for Reliable Network Attack Detection in SCADA Systems. In *International Conference on Trust, Security and Privacy in Computing and Communications/12th IEEE International Conference on Big Data Science and Engineering (TrustCom/BigDataSE)*. IEEE, 633–638.

[68] Peter Phaal, Sonia Panchen, and Neil McKee. 2001. InMon Corporation's sFlow: A Method for Monitoring Traffic in Switched and Routed Networks.

[69] Stanislav Ponomarev and Travis Atkison. 2015. Industrial Control System Network Intrusion Detection by Telemetry Analysis. *IEEE Transactions on Dependable and Secure Computing* 13, 2 (2015), 252–260.

[70] Panagiotis Radoglou-Grammatikis, Panagiotis Sarigiannidis, George Efstathopoulos, Paris-Alexandros Karypidis, and Antonios Sarigiannidis. 2020. DIDEROT: An Intrusion Detection and Prevention System for DNP3-based SCADA Systems. In *International Conference on Availability, Reliability and Security (ARES)*. 1–8.

[71] Nicholas R. Rodofile. 2018. *Generating Attacks and Labelling Attack Datasets for Industrial Control Intrusion Detection Systems.* Ph. D. Dissertation. Queensland University of Technology.

[72] Nicholas R. Rodofile, Thomas Schmidt, Sebastian T. Sherry, Christopher Djamaludin, Kenneth Radke, and Ernest Foo. 2017. Process Control Cyber-Attacks and Labelled Datasets on S7Comm Critical Infrastructure. In *Australasian Conference on Information Security and Privacy (ACISP)*. Springer, 452–459.

[73] Peter J. Rousseeuw and Katrien Van Driessen. 1999. A Fast Algorithm for the Minimum Covariance Determinant Estimator. *Technometrics* 41, 3 (1999), 212–223.

[74] Peter Schneider and Konstantin Böttinger. 2018. High-performance Unsupervised Anomaly Detection for Cyber-Physical System Networks. In *Workshop on Cyber-Physical Systems Security and Privacy*. 1–12.

[75] Bernhard Schölkopf, Robert C. Williamson, Alex Smola, John Shawe-Taylor, and John Platt. 1999. Support Vector Method for Novelty Detection. *Advances in Neural Information Processing Systems (NIPS)* 12 (1999).

[76] Franka Schuster and Hartmut König. 2024. Questioning the Myth: Investigating ICS Traffic Homogeneity from an Anomaly Detection Perspective. In *International Conference on Critical Information Infrastructures Security (CRITIS)*. Springer, accepted for publication.

[77] Scikit-learn. 2023. Implementation of Elliptic Envelope Algorithm. https://scikit-learn.org/stable/modules/generated/sklearn.covariance.EllipticEnvelope.html. Online. Accessed: 2023-07-26.

[78] Scikit-learn. 2023. Implementation of Isolation Forest Algorithm. https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.IsolationForest.html. Online. Accessed: 2023-07-12.

[79] Scikit-learn. 2023. Implementation to Approximate the Solution of a Kernelized One-class SVM. https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.SGDOneClassSVM.html. Online. Accessed: 2023-07-26.

[80] Wenli Shang, Junrong Cui, Ming Wan, Panfeng An, and Peng Zeng. 2016. Modbus Communication Behavior Modeling and SVM Intrusion Detection Method. In *International Conference on Communication and Network Security (ICCNS)*. 80–85.

[81] Karishma Sharma, Pinar Donmez, Enming Luo, Yan Liu, and I. Zeki Yalniz. 2020. Noiserank: Unsupervised Label Noise Reduction with Dependence Models. In *European Conference on Computer Vision (ECCV)*. Springer, 737–753.

[82] Chuan Sheng, Yu Yao, Wei Yang, Ying Liu, and Qiang Fu. 2019. How to Fingerprint Attack Traffic Against Industrial Control System Network. In *International Conference on Industrial Artificial Intelligence (IAI)*. IEEE, 1–6.

[83] Robin Sommer and Vern Paxson. 2010. Outside the Closed World: On Using Machine Learning for Network Intrusion Detection. In *IEEE Symposium on Security and Privacy (S&P)*. IEEE, 305–316.

[84] Keith Stouffer, Joe Falco, Karen Scarfone, et al. 2011. Guide to Industrial Control Systems (ICS) Security. *NIST Special Publication* 800, 82 (2011), 16–16.

[85] Tensorflow. 2023. Keras API for Constructing Artificial Neural Networks. https://www.tensorflow.org/api_docs/python/tf/keras. Online. Accessed: 2023-07-26.

[86] Asuka Terai, Shingo Abe, Shoya Kojima, Yuta Takano, and Ichiro Koshijima. 2017. Cyber-Attack Detection for Industrial Control System Monitoring with Support Vector Machine Based on Communication Profile. In *European Symposium on Security and Privacy Workshops (EuroS&PW)*. IEEE, 132–138.

[87] Preetha Thulasiraman. 2022. Cyber Analytics for Intrusion Detection on the Navy Smart Grid using Supervised Learning. In *IEEE International Systems Conference (SysCon)*. IEEE, 1–7.

[88] Lisa Torrey and Jude Shavlik. 2010. Transfer Learning. In *Handbook of Research on Machine Learning Applications and Trends: Algorithms, Methods, and Techniques*. IGI global, 242–264.

[89] David I. Urbina, Jairo A. Giraldo, Alvaro A. Cardenas, Nils Ole Tippenhauer, Junia Valente, Mustafa Faisal, Justin Ruths, Richard Candell, and Henrik Sandberg. 2016. Limiting the Impact of Stealthy Attacks on Industrial Control Systems. In *ACM SIGSAC Conference on Computer and Communications Security (CCS)*. 1092–1105.

[90] Alfonso Valdes and Steven Cheung. 2009. Communication Pattern Anomaly Detection in Process Control Systems. In *IEEE Conference on Technologies for Homeland Security*. IEEE, 22–29.

[91] Jan Vavra and Martin Hromada. 2018. Comparative Study of Feature Selection Techniques Respecting Novelty Detection in the Industrial Control System Environment. *Annals of DAAAM* 29 (2018).

[92] Jan Vavra and Martin Hromada. 2019. Evaluation of Data Preprocessing Techniques for Anomaly Detection Systems in Industrial Control System. *Annals of DAAAM* 30 (2019).

[93] Jan Vávra, Martin Hromada, Luděk Lukáš, and Jacek Dworzecki. 2021. Adaptive Anomaly Detection System based on Machine Learning Algorithms in an Industrial Control Environment. *International Journal of Critical Infrastructure Protection (IJCIP)* 34 (2021), 100446.

[94] Gernot Vormayr, Joachim Fabini, and Tanja Zseby. 2020. Why Are My Flows Different? A Tutorial on Flow Exporters. *IEEE Communications Surveys & Tutorials* 22, 3 (2020), 2064–2103.

[95] Konrad Wolsing, Lea Thiemt, Christian van Sloun, Eric Wagner, Klaus Wehrle, and Martin Henze. 2022. Can Industrial Intrusion Detection Be SIMPLE? In *European Symposium on Research in Computer Security (ESORICS)*. Vol. 13556. Springer Nature Switzerland, 574–594.

[96] Konrad Wolsing, Eric Wagner, Antoine Saillard, and Martin Henze. 2022. IPAL: Breaking up Silos of Protocol-dependent and Domain-specific Industrial Intrusion Detection Systems. In *International Symposium on Research in Attacks, Intrusions and Defenses (RAID)*. ACM, 510–525.

[97] Christian Wressnegger, Ansgar Kellner, and Konrad Rieck. 2018. Zoe: Content-based Anomaly Detection for Industrial Control Systems. In *IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*. IEEE, 127–138.

[98] Tao Yang, Yibo Hu, Yang Li, Wei Hu, and Quan Pan. 2019. A Standardized ICS Network Data Processing Flow with Generative Model in Anomaly Detection.

IEEE Access 8 (2019), 4255–4264.

[99] Man-Ki Yoon and Gabriela F. Ciocarlie. 2014. Communication Pattern Monitoring: Improving the Utility of Anomaly Detection for Industrial Control Systems. In *NDSS Workshop on Security of Emerging Networking Technologies*.

[100] Jeong-Han Yun, Yoonho Hwang, Woomyo Lee, Hee-Kap Ahn, and Sin-Kyu Kim. 2018. Statistical Similarity of Critical Infrastructure Network Traffic based on Nearest Neighbor Distances. In *International Symposium on Research in Attacks, Intrusions, and Defenses (RAID)*. Springer, 577–599.

[101] Barret Zoph, Golnaz Ghiasi, Tsung-Yi Lin, Yin Cui, Hanxiao Liu, Ekin Dogus Cubuk, and Quoc Le. 2020. Rethinking Pre-training and Self-training. *Advances in Neural Information Processing Systems (NIPS)* 33 (2020), 3833–3845.

A ADDITIONAL INFORMATION AND RESULTS

A.1 Problems with Detection Approaches based on Process Data

There are a number of serious issues with detection schemes based on process data.

Issue 1: Unclear benefit compared to the already established process monitoring. Every OT (sub-)system deployed for controling a sufficiently complex process already contains a monitoring appliance. This is a basic function necessary for operators to monitor and control the process. Otherwise, it cannot operate reliably. When reviewing process-based detection approaches, we could not find any discussion regarding the benefits of the proposed attack detection systems compared to the built-in monitoring functionality of already deployed process control systems. Many complex detection approaches mainly identify activity which can also be determined by simple heuristics [95]. Since such comparatively simple checks are usually a built-in functionality of process monitoring systems, as a decades long standard, the benefit of an additional monitoring function on top is questionable. The advantage of communication-based approaches, in contrast, lies in the analysis of the communication activities underlying the process.

Issue 2: Absence of sufficiently prepared process data in real infrastructures. Based on years of experience with real-life infrastructures we can say with certainty: One cannot assume a *historian*, i.e., time-series OT databases with a large amount of well-structured and collected data, in real infrastructures, as it is provided by the most prominent testbed SWaT [28]. A single plant may contain dozens of subsystems. Many of them produce only error logs that do not contain a single concrete process value. They further store logs locally so that a person has to go to the system console to have a look at the logs. The nice idea of a central point holding a multitude of sufficiently precise information for external usage is unfortunately an illusion. One reason are stringent warranty restrictions imposed by system vendors which severely limit the possibilities of system operators to attach deployed systems to a global log system.

Issue 3: Downstream detection based on logging instead of directly analyzing OT activities. The term *historian* for time-series databases in OT infrastructures already hints to the drawback of process-based detection systems analyzing their data. The available records contain information about process events or states that have been reported by the OT equipment in the past. Although this past may not be long ago, information processing in a *historian* always takes place after the activity happened, whereas a communication-based detection system directly monitors the network activities, such as physical value settings. Therefore, they have a clear advantage regarding the detection time. Furthermore, recently disclosed vulnerabilities [20] in *historians* call into question the reliability of these systems.

Issue 4: Detection methods built on non-standardized input. Even if log data was available to the expected extent on each system, there would be another problem. There is no homogenous standard for producing process log data regarding granularity, structure, and meaning of the data items. This makes every detection system built on top of the logging of data to a detection system that only relates to the respective individual log setting. Network traffic, in contrast, already is a well-defined, consistent, and fine-granular source (with flows or packets as informational items) incorporating generally agreed information structures explored for decades in form of standardized flow characteristics [15, 16, 68] and protocol stacks of packets. Therefore, traffic-based detection systems can potentially be applied in every other network.

A.2 Used Machine Learning Algorithms

From preliminary experiments, we select the following machine learning algorithms from four different families.

Covariance-based: Elliptic Envelope (envelope). The idea of the algorithm is to fit a probability distribution to given input interpreted as data points, which is in our case a Gaussian one. Then, a covariance matrix is identified with the lowest possible determinant that still covers a defined ratio of the data points. All uncovered points are considered as outliers. Although we do not assume a Gaussian distribution of the input data, we examine how well this algorithm performs for our problem.

Kernel-based: One-class Support Vector Machine (OCSVM). This algorithm is an adaption of the original Support Vector Machine method [18] to be usable for one-class learning. Input samples with *n* features are considered as data points in *n*-dimensional space. For these points a hyperplane is determined that separates all points from the origin with maximum distance between hyperplane and origin, for which the data points usually must be transmitted in a higher-dimensional space. After training, samples situated between the origin and the hyperplane are considered as anomalies. Points lying on the hyperplane are called support vectors, which led to the algorithm's name.

Tree-based: Isolation Forest (iforest). This algorithm identifies anomalies by two quantitative properties: as being fewer and having attribute values different to normal instances. An ensemble of isolation trees is then built in such a way that anomalies are isolated closer to the root of each isolation tree. Thus, anomalies are those cases that have short average path lengths in the trees.

Neural-Network-based: Convolutional Autoencoder (CAE). After investigating the design options for artificial neural networks (ANN), we decided to examine a network type which incorporates so-called convolutions [30, 85]. Here, the complexity of single matrix calculations is decreased by separately applying them to smaller parts of the input data. We further decided to structure the ANN as autoencoder because the encoder part realizes an implicit feature extraction through sample compression, which can be beneficial for the detection success. Our decision was further strengthened by

Table 11: Probed hyperparameter space per classifier.

hyperparameter	**parameter space**
Elliptic Envelope	
ratio of points to be included	[0.001, 0.9999]
ratio of outliers expected during training	[1/all samples, 0.1]
Isolation Forest	
number of trees	[1, 200]
sample number for training a tree	[200, all samples]
ratio of outliers expected during training	[1/all samples, 0.1]
One-class Support Vector Machine	
kernel type	[linear; polynomial; radial basis function; sigmoid function]
ratio of support vectors to training samples	[0.0001, 1]
number of training samples	[0.00001, 1]
Convolutional Autoencoder	
number of convolutional layers	[1, 2]
size of convolutions	[2, 4]
activation function	[relu; sigmoid; swish]
number of feature maps	[16; 28; 40; 52; 64; 86]
number of dense layers	[2, 5]
dropout rate	[0.0, 0.9]

results regarding OT process data. It has been shown that 4-layer convolutional autoencoder (CAE) designs applied to OT sensor data are nine to 36 times faster (with at least the same detection accuracy) than designs incorporating LSTM[15] cells [39], which due to their novelty have strongly been represented in research recently. We use the autoencoder for classification by deriving the decision for a class based on the reconstruction error determined between the output (reconstructed samples) of the autoencoder and the original samples used as input. A list of the hyperparameters used for tuning the models with corresponding value range is given in Table 9.

A.3 Additional Information to the Public Datasets

We reviewed available public datasets with network traffic and summarize them in Table 12.

Sufficient dataset labeling. We consider a dataset as sufficiently labeled if the authors either directly provide a flag to each network packet of the traffic or the attack periods are given so precisely that such a labeling can be derived by the users. For the QUT S7 (Myers) dataset, for instance, it was not possible for us to assign the attack phases to certain packets due to inconsistent timestamp deviations.

Parts used for training and validation. We used the following captures, stated per dataset, in our work. Information to the Lemay dataset are already provided in Section 4.1.

Table 12: Reviewed datasets with raw OT network traffic.

	year	**labeled**	**OT protocol(s)**
CyberCity [35]	2013		Modbus, Ethernet/IP
4SICS [59]	2015		Modbus
S4x15 ICS [60]	2015		Modbus, BACnet
Lemay [43]	2016	●	**Modbus**
QUT S7comm [63]	2017	●	**S7comm**
QUT DNP3 [62]	2018	●	**DNP3, GOOSE**
Modbus SCADA #1 [19]	2018		Modbus
QUT S7 (Myers) [57]	2018	●[16]	S7comm
HVAC [58]	2019		S7comm
SWaT [61]	2020		EtherNet/IP, CIP
EPIC [61]	2021		Modbus, IEC61850
WDT [31]	2021	●	**Modbus**

WDT datatset. We used normal.pcap for training and the remaining capture files as validation traffic. The attacks included Man-in-the-Middle, Denial-of-Service, and scanning activities.

DNP3 dataset. We used control/testing/frequent/slave.pcap as normal traffic for training and attacks/testing/frequent/slave.pcap for validation.

S7comm dataset. Our experiments were conducted on the traffic captured in the files called master.pcap. The one taken from the control set was used for training, the one belonging to the part called s7_process_attacks for validation.

A.4 Detection Effectiveness (Q1) on the Public Datasets

Besides the balanced detection accuracy provided in Table 6 we further summarize the precision, recall, and F1-score of the detection models in Table 13. This allows us to relate the performance of the models to other research results using these measures. They are defined as follows:

$$precision = \frac{TP}{TP + FP} \quad , \quad recall = \frac{TP}{TP + FN}$$

$$F1\text{-}score = 2 \cdot \frac{precision \cdot recall}{precision + recall}$$

At the same time, we point out the limitations of these metrics to assess our models, which is why we use the balanced accuracy to evaluate them in the main part of our work. These are:

- The measures precision, recall, and F1-score completely disregard true negatives, i.e., correctly detected normal samples. However, the addressed detection scenario, like most problems, is very imbalanced in the real world, since attacks are usually exceptions and normal communication strongly predominates. Therefore, the correct classification of normality plays a particularly important role and should be included in the evaluation accordingly.
- Since the three measures do not include the (correct or incorrect) classification of *all* samples, they should only be used with balanced datasets because they are misleading for

[15] Short form for Long Short-term Memory.

[16] We omit this dataset because timestamp deviations do not allow the use of this dataset in a reliable manner.

the measurement of datasets with imbalanced class membership. The balanced accuracy used in the main part, in contrast, is always meaningful, as it equally incorporates the classification of all samples.

- Precision, recall, and F1-score are primarily influenced by the amount of true positives (TP), i.e., correctly detected anomalous samples. However, the detection of an attack does not require to detect *every* associated anomalous sample. For our approach, the detection of a single anomalous sample is sufficient. Therefore, the three TP-driven metrics can be low and the model can still detect all attacks. Potentially low values should therefore be considered in combination with the ratio of detected attacks provided in Table 7.

We deliberately refrained from first creating an artificially balanced problem from the given datasets, as this does not help to find practicable solutions for the imbalanced real scenario. Instead, we deliberately wanted to test the approach on the actual conditions. This is why we use the balanced accuracy to measure the effectiveness of the detection models in the main part instead of precision, recall, and F1-score.

A.5 Detection Effectiveness (Q1) on the Real Datasets

The experiments with the real datasets are summarized in Table 14. They included 36 setups defined by the dataset, the sample type, and the machine learning algorithm. Due to the absence of anomalous network traffic in the real datasets, only one side of the coin can be evaluated here, which is a correct classification of normal network traffic. In this respect, the results for the algorithms Elliptic Envelope, Isolation Forest, and OCSVM can be considered as very good. Here, a validation accuracy higher 0.99 was measured in 26 of 27 cases. The perfect value of 1.00 was reached in 12 cases. The CAE performs comparatively poorly with the real datasets, regardless of the sample type used. Only one third of the models have a *vacc* value higher 0.95, none a value higher 0.98.

A.6 Ensemble Detection Effectiveness (Q1)

The fact that each of the examined machine learning algorithms showed at least once an accuracy on the validations set *vacc* greater than 0.98 and all algorithms are present among the best model of each dataset made us curious for further investigations. We wanted to find out whether the detection results could generally be improved by using ensembles of models instead of single models. Thus, we determined collective decisions for each constellation of dataset and sample type using all four identified machine learning models in parallel. The investigated ensemble stategy considers single predictions as votes. Each prediction of a sample's class as anomaly is counted as a vote for the class *anomalous traffic*. The ensemble prediction for the anomalous class regarding a sample is then derived from the respective count of the single models' predictions on the same sample towards this class. Table 15 summarizes the results of the 21 constellations of dataset and sample type. Measurements were made for the cases (strategies) if the ensemble's anomaly prediction is based on at least one (1/4), two (2/4), three (3/4) or four (4/4) anomaly predictions from the four single models corresponding to the sample type and dataset.

Local effectiveness. The subject of the experiments was to find out if a voting point can be identified where in most of the 21 constellations the false classifications among the models can be balanced to gain a higher validation accuracy from the ensemble compared to the best single model. Unfortunately, in most constellations no significant advantage can be observed by using all four identified models for a constellation as ensemble to just using the identified best single model. With a single vote as a sufficient criterion for the ensemble's anomaly decision, the least tolerant model dominates the overall vote negatively by its number of false positives. Starting with two out of four necessary votes for the anomaly decision, this disadvantage is eliminated for most constellations, but also no increase in the validation accuracy *vacc* and no reduction in the false classification rate can be observed. For three constellations (packet samples on Lemay and S7comm, and 3-packet samples on S7comm), the false-negative (missed anomalies) rate rises noticeably indicated by increased overall false rates with hardly any decrease in false alarms. Consequently, the local effectiveness of a protocol-agnostic detection is not necessarily improved by using ensembles of protocol-agnostic detection models. A well-fitted single model may be sufficient with the advantage of only having to configure and operate one model for detection. At the same time, the ensembles are no worse than the best individual model in most cases. This is important for the application in real-world conditions. Here, the quality of a single model is not known until it was exposed to anomalous network activity. By using ensembles across several algorithms, poorer models can be compensated and a similar detection level can be ensured as for the best individual model, without having to know it in advance.

A.7 Background Results for Detection Efficiency (Q2)

The diagrams given in Section 5.3 are derived from the measurements on the public and real datasets provided in Table 16.

A.8 Transfer Sensitivity to Protocol Mix Similarity (Q3)

For this aspect, we took a closer look at the transfer scenarios where source and target network use the same dominant OT protocol (Modbus/TCP), which applies to the public datasets Lemay and WDT, as well as the real dataset steam production. The results of the 72 experiments (3 × 2 dataset combinations, three sample types, and four machine learning algorithms) are illustrated in Figure 7. In a small number of 5 experiments (approximately 7%) a balanced accuracy of higher 90% was achieved. Here, flow and packet samples are equally represented, each with two models. The Elliptic Envelope algorithm has two occurences, all others have one. A perfect balanced accuracy of 100% was achieved in only two cases (about 3%), one time with flow samples and one time on single-packet samples. In the two experiments the OCSVM algorithm and an Autoencoder were applied. Consequently, a lower fraction of models achieves an accuracy between 90% and 100% when compared to all transfer scenarios. Hence, the experiments with a Modbus dataset as source and target dataset surprisingly turn out to be comparatively weak. A dependence of the transfer

Table 13: Single-model effectiveness on the public datasets: Precision, recall, and F1-score of the best model identified for each setup of traffic perspective, algorithm, and dataset.

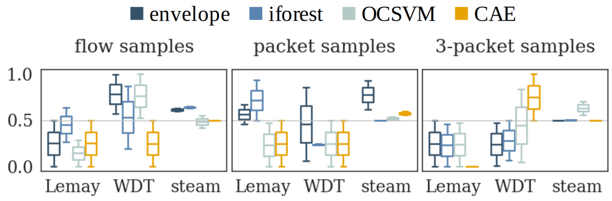
		flow samples				**packet samples**				**3-packet samples**			
		envelope	iforest	OCSVM	CAE	envelope	iforest	OCSVM	CAE	envelope	iforest	OCSVM	CAE
Lemay	precision	0.7000	0.6429	0.5600	0.5417	0.9692	0.9970	0.9855	0.9838	0.9781	0.9826	0.9728	0.9252
	recall	0.7000	0.9000	0.7000	0.6500	0.0812	0.0811	0.8957	0.8804	0.8723	0.9453	0.8789	0.9893
	F1-score	0.7000	0.7500	0.6222	0.5909	0.1498	0.1500	0.9385	0.9292	0.9222	0.9636	0.9235	0.9562
WDT	precision	0.9990	0.9930	0.9996	0.9248	0.9747	0.9262	0.9418	0.9249	0.9868	0.9533	0.9921	0.9999
	recall	0.6942	0.7140	0.6940	0.7107	0.5237	0.5338	0.5305	0.5341	0.5238	0.5315	0.4425	0.4410
	F1-score	0.8192	0.8307	0.8192	0.8037	0.6813	0.6773	0.6787	0.6772	0.6843	0.6825	0.6120	0.6121
	precision^f	0.9990	0.9930	0.9996	0.9152	0.9643	0.9019	0.9222	0.9004	0.9822	0.9424	0.9907	0.9999
	recall^f	0.8949	0.9204	0.8946	0.9066	0.8281	0.8309	0.8304	0.8310	0.6730	0.6783	0.5705	0.5693
	F1-score^f	0.9441	0.9553	0.9442	0.9109	0.8910	0.8649	0.8739	0.8643	0.7987	0.7888	0.7241	0.7255
DNP3	precision	0.2299	0.1643	0.2192	0.2105	0.1728	0.1278	0.1041	0.0085	0.6334	0.4619	0.2035	0.0177
	recall	0.8696	1.0000	0.6957	0.6957	0.9112	0.5286	0.0943	0.9415	0.9211	0.7203	0.6661	0.9376
	F1-score	0.3637	0.2822	0.3334	0.3232	0.2905	0.2058	0.0990	0.0168	0.7506	0.5629	0.3118	0.0347
S7comm	precision	0.9130	0.8400	1.0000	1.0000	0.9875	0.7951	TP=0	0.5496	0.9966	0.9896	0.9992	0.9847
	recall	1.0000	1.0000	1.0000	1.0000	1.0000	0.9930	0.0000	1.0000	0.9722	0.9688	0.8716	0.9955
	F1-score	0.9545	0.9130	1.0000	1.0000	0.9937	0.8831	TP=0	0.7093	0.9842	0.9791	0.9310	0.9901

Table 14: Single-model effectiveness on the real datasets: Detection accuracy for training samples ($tacc$), validation samples ($vacc$) and both ($bacc$) of the best model identifed for each setup of traffic perspective, algorithm, and dataset. The best model per sample type is printed in bold, the best model for the dataset is also underlined.

		flow samples				**packet samples**				**3-packet samples**			
		envelope	iforest	OCSVM	CAE	envelope	iforest	OCSVM	CAE	envelope	iforest	OCSVM	CAE
DMZ	$tacc$	0.9999	1.0000	1.0000	0.9279	0.9980	0.9999	0.9998	0.9698	0.9999	0.9999	0.9806	0.8454
	$vacc^-$	1.0000	**1.0000**	**1.0000**	0.9223	0.9979	**0.9999**	0.9998	0.9689	**0.9999**	0.9999	0.9798	0.8521
	$bacc^-$	0.9999	1.0000	1.0000	0.9251	0.9980	0.9999	0.9998	0.9693	0.9999	0.9999	0.9802	0.8488
plant control	$tacc$	0.9997	0.9995	1.0000	0.9527	0.9999	0.9999	1.0000	0.9031	0.9999	0.9999	1.0000	0.8956
	$vacc^-$	**1.0000**	0.9997	0.9995	0.9507	1.0000	1.0000	**1.0000**	0.8987	**1.0000**	0.9999	0.9999	0.8693
	$bacc^-$	0.9998	0.9996	0.9997	0.9517	0.9999	0.9999	1.0000	0.9009	0.9999	0.9999	0.9999	0.8824
steam production	$tacc$	0.9999	0.9999	1.0000	0.9399	0.9998	1.0000	1.0000	0.9999	0.9999	0.9999	1.0000	0.9253
	$vacc^-$	0.9999	0.9999	**1.0000**	0.9223	0.9979	**1.0000**	**1.0000**	0.9754	0.9999	0.9999	**1.0000**	0.9183
	$bacc^-$	0.9999	0.9999	1.0000	0.9311	0.9989	1.0000	1.0000	0.9877	0.9999	0.9999	1.0000	0.9218

The best models were determined using more decimal places than those shown.

Figure 7: Modbus-related model transfers: Detection capability ($vacc$) of models trained on the indicated dataset and applied to the validation sample sets of the other Modbus datasets.



potential on the protocol mix similarity of the source and target network could not be observed.

Table 15: Ensemble detection results (in %): Validation accuracy (*vacc*), false classifications (F) and false positives only (FP) if anomaly prediction relies on at least one (1/4), two (2/4), three (3/4) or four (4/4) anomaly predicitions from the single models.

		flow samples				packet samples				3-packet samples			
		1/4	2/4	3/4	4/4	1/4	2/4	3/4	4/4	1/4	2/4	3/4	4/4
Lemay	*vacc*	99.77	94.77	79.79	72.37	94.51	93.77	55.16	52.90	97.75	97.05	95.20	91.61
	F	0.45	0.53	0.73	0.69	3.21	3.59	24.28	25.45	2.77	1.98	2.92	4.84
	FP	0.45	0.45	0.41	0.24	0.40	0.35	0.08	0.00	2.48	0.57	0.45	0.37
WDT	*vacc*	83.39	84.62	84.67	84.68	75.22	75.24	75.31	75.81	75.99	75.89	72.03	71.75
	F	16.71	15.54	15.49	15.48	20.62	20.59	20.48	19.68	19.81	19.59	22.65	22.86
	FP	2.96	0.21	0.04	0.01	1.76	1.72	1.55	0.33	1.18	0.25	0.05	0.00
DNP3	*vacc*	88.43	86.65	80.78	82.12	59.56	92.36	73.56	54.67	58.49	95.10	89.26	78.35
	F	22.28	13.62	8.81	6.25	80.28	4.29	2.45	0.68	77.31	2.62	1.18	0.86
	FP	22.28	13.14	7.69	5.13	80.28	4.20	2.08	0.02	77.24	2.51	0.89	0.23
S7comm	*vacc*	99.20	99.60	100.00	100.00	69.29	99.16	99.35	50.00	96.07	98.83	97.90	92.97
	F	1.47	0.74	0.00	0.00	38.74	1.05	0.63	36.91	1.97	1.20	3.02	10.58
	FP	1.47	0.74	0.00	0.00	38.74	1.05	0.37	0.00	1.88	0.27	0.08	0.02
DMZ	*vacc*	92.23	100.00	100.00	100.00	96.83	99.84	99.99	100.00	84.24	98.96	99.99	100.00
	F	7.76	0.00	0.00	0.00	3.17	0.16	0.00	0.00	15.76	1.04	0.00	0.00
	FP	7.76	0.00	0.00	0.00	3.17	0.16	0.00	0.00	13.06	0.00	0.00	0.00
plant control	*vacc*	95.07	99.92	100.00	100.00	89.87	100.00	100.00	100.00	86.93	99.99	99.99	100.00
	F	4.93	0.07	0.00	0.00	10.13	0.00	0.00	0.00	13.06	0.00	0.00	0.00
	FP	4.93	0.07	0.00	0.00	10.13	0.00	0.00	0.00	15.76	1.04	0.00	0.00
steam production	*vacc*	92.23	99.98	100.00	100.00	97.54	99.79	100.00	100.00	91.83	99.99	100.00	100.00
	F	7.76	0.01	0.00	0.00	2.45	0.20	0.00	0.00	8.17	0.00	0.00	0.00
	FP	7.76	0.01	0.00	0.00	2.45	0.20	0.00	0.00	8.17	0.00	0.00	0.00

Table 16: Efficiency: Packet sample processing times in milliseconds with real-time processing load in % and the sample number that could be processed on top (positive value) or exceed the detection speed (negative value).

				training	validation		
		sampling	**fitting**	**pred.**	**pred.**	**%load**	**buffer/s**
Lemay	envelope	0.2526	1.4356	0.2707	0.2687	0.62	3698
	iforest	0.2523	0.3866	0.3110	0.3200	0.74	3101
	OCSVM	0.2526	1.0065	0.2526	0.2675	0.62	3714
	CAE	0.2526	0.8047	0.2668	0.2840	0.65	3497
WDT	envelope	0.3066	6.9608	0.3104	0.3004	86.27	457
	iforest	0.3061	0.5135	0.3793	0.3969	113.97	☹ -352
	OCSVM	0.3066	1.2933	0.3066	0.2964	85.09	503
	CAE	0.3066	0.8073	0.3248	0.3128	89.83	325
DNP3	envelope	0.2283	2.1934	0.2295	0.2275	0.41	4376
	iforest	0.2274	0.2387	0.2340	0.2333	0.42	4268
	OCSVM	0.2283	1.3284	0.2284	0.2264	0.41	4399
	CAE	0.2283	0.7343	0.2445	0.2415	0.43	4122
S7comm	envelope	0.3144	1.8438	0.3187	0.3126	1.75	3143
	iforest	0.3144	0.4404	0.3627	0.3820	2.14	2562
	OCSVM	0.3144	0.7576	0.3145	0.3115	1.74	3154
	CAE	0.3144	0.9326	0.3310	0.3284	1.84	2988
DMZ	envelope	0.5587	3.0171	0.5598	0.5840	83.12	289
	iforest	0.5584	0.8902	0.7928	0.8139	115.79	☹ -194
	OCSVM	0.5587	1.5457	0.5587	0.5829	82.97	292
	CAE	0.5587	1.0155	0.5799	0.6029	85.77	236
plant control	envelope	0.5276	7.3354	0.5399	0.5469	2.19	1788
	iforest	0.5273	0.5770	0.5532	0.5706	2.28	1712
	OCSVM	0.5276	1.1015	0.5276	0.5457	2.18	1792
	CAE	0.5276	1.0262	0.5480	0.5624	2.25	1738
steam production	envelope	0.5592	2.1173	0.5650	0.5862	10.26	1531
	iforest	0.5592	0.6333	0.5918	0.6153	10.77	1450
	OCSVM	0.5592	1.0793	0.5593	0.5849	10.24	1534
	CAE	0.5592	1.1742	0.6071	0.6054	10.59	1477