On the Generalizability and Transferability of Industrial Intrusion Detection Research

Von der Fakultät für Informatik der RWTH Aachen
University zur Erlangung des akademischen Grades eines
Doktors der Naturwissenschaften genehmigte Dissertation

vorgelegt von

Master of Science

Konrad Richard Vinzenz Werner Wolsing

aus Bergisch Gladbach

Berichter:

Prof. Dr.-Ing. Klaus Wehrle
Prof. Dr.-Ing. Thorsten Strufe

Tag der mündlichen Prüfung: 16.01.2026

Abstract

The increasing automation of Industrial Control Systems (ICSs) has brought significant advantages across various industrial domains such as water treatment, power grids, and manufacturing. Simultaneously, the accompanying digitization of ICSs has led to increasing surfaces for cyberattacks. While preventive security measures to thwart these attacks exist, they can prove challenging to retrofit into legacy systems, making specialized Industrial Intrusion Detection Systems (IIDSs) a viable alternative. Addressing the specific needs of unique and diverging ICS applications, a large research community has gathered, inventing novel and effective detection methodologies yet focusing on specific industrial protocols or domains. However, as two ICSs are rarely identical, IIDSs are expected to generalize to different use cases or even transfer to new domains instead of being built for a single purpose. While the protocols and physical processes underlying all ICSs show remarkable similarity concerning their functionality or predictability, current IIDS research remains fragmented, as we show, rather than providing generalizable solutions.

This dissertation addresses this overarching challenge by investigating the feasibility of protocol- and domain-independent intrusion detection. Through a Systematic Mapping Study (SMS), five critical issues hindering effective IIDS research were identified, including (i) disjoint research areas for each ICS domain, (ii) inefficiencies in evaluation methodologies to holistically capture the capabilities of new IIDSs, (iii) a strong focus of research on complex detection methodologies with little justification, (iv) few efforts to transfer existing achievements to all ICS domains, and (v) little considerations on implications to select and deploy an IIDS.

To overcome these limitations, this dissertation introduces the Industrial Protocol Abstraction Layer (IPAL), a novel framework that proposes a standardized data representation for industrial intrusion detection. Thereby, IPAL facilitates the generalizability, transferability, and comparability of IIDSs across different ICS environments and among research. Furthermore, this dissertation presents new lightweight yet effective detection methodologies disproving the current misconception of complexity in research. Ultimately, since deploying just the single-best IIDS can be disadvantageous with respect to optimal detection performance, we successfully explore techniques for ensemble learning to streamline multiple IIDSs' capabilities to enhance detection performance while reducing false positives.

These findings demonstrate that intrusion detection can be both efficient and adaptable to ICS domains or leveraged communication technologies. On the one hand, our contributions support a more coherent and effective IIDS research landscape. On the other hand, we pave the way for sustainable and practical implementations of IIDSs in real-world deployments. Thereby, we make IIDSs more accessible for ICSs to protect critical infrastructure more efficiently against harmful cyberattacks.

Kurzfassung

Die zunehmende Automatisierung industrieller Kontrollsysteme (ICS) hat in verschiedenen Domänen wie der Trinkwasseraufbereitung, den Stromnetzen und der Fertigung erhebliche Verbesserungen gebracht. Gleichzeitig führt die damit einhergehende Digitalisierung zu einer größeren Angriffsfläche für Cyberangriffen. Zwar können präventive Sicherheitsmaßnahmen solche Angriffe vereiteln, jedoch kann es sich als schwierig erweisen, diese in bestehende Systeme nachzurüsten. Als sinnvolle Alternative bieten sich daher spezialisierte Intrusion Detection Systeme (IDS) an. Um den spezifischen Anforderungen aller ICS-Domänen gerecht zu werden, hat sich eine Forschungsgemeinschaft gebildet, die neuartige und wirksame IDS entwickelt. Jedoch spezialisiert sich die Forschung dabei auf bestimmte Industrieprotokolle oder Domänen. Da jedoch zwei ICS selten identisch sind, ist es sinnvoll, dass ein IDS potenziell für verschiedene Domänen nutzbar ist, anstatt nur für einen einzigen Zweck entwickelt zu sein. Während die Kommunikationsprotokolle und physikalischen Prozesse, die allen ICS zugrunde liegen, bemerkenswerte Ähnlichkeiten in Bezug auf ihre Funktionalität oder Vorhersagbarkeit aufweisen, bleibt die aktuelle IDS-Forschung, wie wir zeigen, fragmentiert, anstatt allgemeingültige Lösungen zu bieten.

Darum befasst sich diese Dissertation mit der übergreifenden Herausforderung von protokoll- und domänenunabhängiger Angriffserkennung. Durch eine systematische Literaturrecherche wurden fünf kritische Punkte identifiziert, die einer effektiven IDS-Forschung im Wege stehen: (i) nicht zusammenhängende Forschungsbereiche für jede ICS-Domäne, (ii) ineffiziente Evaluierungsmethoden, um die Fähigkeiten neuer IDS ganzheitlich zu bewerten, (iii) ein starker Fokus der Forschung auf komplexe Erkennungsmethoden, (iv) geringe Bemühungen, erzielte Ergebnisse auf alle ICS-Domänen zu übertragen, und (v) seltene Überlegungen zu den Implikationen für die Auswahl und den Einsatz eines IDS.

Um diese Herausforderungen anzugehen, wird in dieser Dissertation der Industrial Protocol Abstraction Layer (IPAL) vorgestellt, ein neuartiges Framework, das ein standardisiertes Datenformat für industrielle Angriffserkennung vorschlägt. Dadurch erleichtert IPAL die Generalisierbarkeit, Übertragbarkeit und Vergleichbarkeit von IDS in verschiedenen ICS-Umgebungen und in der Forschung. Darüber hinaus werden in dieser Dissertation neue, minimalistische und dennoch effektive Erkennungsmethoden vorgestellt, welche die derzeitige Notwendigkeit für Komplexität in der Forschung widerlegen. Da der Einsatz eines einzelnen IDS im Hinblick auf eine bestmögliche Erkennungsleistung nachteilig sein kann, erforschen wir Techniken wie das Ensemble-Lernen, um mehrerer IDS zu fusionieren und so die Erkennungsleistung zu verbessern und gleichzeitig die Zahl der Fehlalarme zu verringern.

Unsere Ergebnisse zeigen, dass Angriffserkennung sowohl effizient als auch generalisierbar sein kann, unabhängig von der ICS-Domäne oder der verwendeten Kommunikationstechnologie. So fördern unsere Beiträge einerseits eine kohärentere und effektivere Forschungslandschaft. Andererseits ebnen wir den Weg für einfachere Integration von IDS. Somit machen wir IDS für ICS zugänglicher, um letztendlich kritische Infrastrukturen effizienter vor schädlichen Cyberangriffen zu schützen.

Acknowledgments

The success of this dissertation is a collaborative effort made possible by a large number of people, including colleagues and students. I am grateful for all the enduring assistance and encouragement I have received over the last six years. These were essential drivers in enabling me to finish this dissertation.

I owe thanks to Klaus Wehrle, Thorsten Strufe, Dominique Unruh, and Ulrike Meyer for taking their time to read this dissertation and conduct the examination.

I would like to express my deepest gratitude to my closest colleagues, Martin Henze and Eric Wagner. You have accompanied me since day one and provided continuous support throughout the entire process, culminating in the final defense day. I remember endless discussions on the first IPAL paper drafts and tight submission deadlines. Despite these challenging phases, I could always count on your timely and valuable input, from which I have learned tremendously.

My next expression of gratitude is dedicated to my longtime colleagues Jan Bauer, Martin Serror, and Lennart Bader from the Fraunhofer FKIE Aachen office. While we worked on diverging topics, the mutual insights from the maritime and electrical domains have positively contributed to and enriched this dissertation. I am thankful that I could learn from your individual prior scientific experience, especially for your endless offers to engage in discussions or provide feedback of all kinds.

I am likewise thankful to my employer, Fraunhofer FKIE and Elmar Padilla, for providing me with the opportunity to freely pursue my research topics. I enjoyed working in this group of plentiful expertise, striving to make the world more secure every day. Special thanks also go to my group leader, Jan Bauer, with whom I have taken plenty of business trips and who supported me greatly.

At RWTH Aachen University, I found a connection with the Security and Privacy group at COMSYS, and I am thankful for your open-mindedness. I am especially thankful to Jan Pennekamp for his constant support during my time as an external researcher at COMSYS.

Another group of contributors are the students who participated through their bachelor's and master's theses. Antoine Saillard deserves special mention as a long-time research assistant. Frederik Basels and Luisa Lux, both former students, even became trusted colleagues by joining Fraunhofer FKIE.

Lastly, I thank my friends for providing distraction in countless nights with board games and my family for their mental support.

Overall, I could not have wished for better people to work with.

Published Papers

Parts of this dissertation are based on the following peer-reviewed papers that have already been published. All my collaborators are among my co-authors. A detailed attribution of contributions can be found below.

List of Publications

[LWW+23] Olav Lamberts, Konrad Wolsing, Eric Wagner, Jan Pennekamp, Jan Bauer, Klaus Wehrle, and Martin Henze. SoK: Evaluations in Industrial Intrusion Detection Research. *Journal of Systems Research (JSys)*, 3(1), 2023.

[SWWB24] Antoine Saillard, Konrad Wolsing, Klaus Wehrle, and Jan Bauer. Exploring Anomaly Detection for Marine Radar Systems. In *Proceedings of the 10th Workshop on the Security of Industrial Control Systems & of Cyber-Physical Systems (CyberICPS), co-located with the 29th European Symposium on Research in Computer Security (ESORICS)*, pages 361–381, 2024.

[WKW+24] Konrad Wolsing, Dominik Kus, Eric Wagner, Jan Pennekamp, Klaus Wehrle, and Martin Henze. One IDS Is Not Enough! Exploring Ensemble Learning for Industrial Intrusion Detection. In *Proceedings of the 28th European Symposium on Research in Computer Security (ESORICS)*, pages 102–122, 2024.

[WSB+22] Konrad Wolsing, Antoine Saillard, Jan Bauer, Eric Wagner, Christian van Sloun, Ina Berenice Fink, Mari Schmidt, Klaus Wehrle, and Martin Henze. Network Attacks Against Marine Radar Systems: A Taxonomy, Simulation Environment, and Dataset. In *Proceedings of the 47th Conference on Local Computer Networks (LCN)*, pages 114–122, 2022.

[WTvS+22] Konrad Wolsing, Lea Thiemt, Christian van Sloun, Eric Wagner, Klaus Wehrle, and Martin Henze. Can Industrial Intrusion Detection Be SIMPLE? In *Proceedings of the 27th European Symposium on Research in Computer Security (ESORICS)*, pages 574–594, 2022.

[WWB+24] Konrad Wolsing, Eric Wagner, Frederik Basels, Patrick Wagner, and Klaus Wehrle. Deployment Challenges of Industrial Intrusion Detection Systems. In *Proceedings of the 10th Workshop on the Security of Industrial Control Systems & of Cyber-Physical Systems (CyberICPS), co-located with the 29th European Symposium on Research in Computer Security (ESORICS)*, pages 453–473, 2024.

[WWH20] Konrad Wolsing, Eric Wagner, and Martin Henze. Poster: Facilitating Protocol-independent Industrial Intrusion Detection Systems. In *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security (CCS)*, pages 2105–2107, 2020.

[WWL+25] Konrad Wolsing, Eric Wagner, Luisa Lux, Martin Henze, and Klaus Wehrle. GeCos Replacing Experts: Generalizable and Comprehensible Industrial Intrusion Detection. In *Proceedings of the 34rd USENIX Security Symposium (USENIX Security)*, 2025.

[WWSH22] Konrad Wolsing, Eric Wagner, Antoine Saillard, and Martin Henze. IPAL: Breaking up Silos of Protocol-dependent and Domain-specific Industrial Intrusion Detection Systems. In *Proceedings of the 25th International Symposium on Research in Attacks, Intrusions and Defenses (RAID)*, pages 510–525, 2022.

Supervised Theses

Many of the initial ideas and concepts for these publications were first examined together with students in bachelor's or master's theses at the RWTH Aachen University. Theses relevant to this dissertation are listed below.

List of Supervised Theses

Antoine Saillard. *On the Generalizability of Process-Based Industrial Intrusion Detection Systems*. Advisers: Konrad Wolsing, Eric Wagner, and Martin Henze. Examiners: Klaus Wehrle and Ulrike Meyer. Bachelor's Thesis. RWTH Aachen University. April 2021.

Lea Thiemt. *Designing and Evaluating Minimalistic Industrial Intrusion Detection Approaches*. Advisers: Konrad Wolsing. Examiners: Klaus Wehrle and Ulrike Meyer. Bachelor's Thesis. RWTH Aachen University. October 2021.

Myrat Altyyev. *Closer to the Network: Deploying State-Based Industrial Intrusion Detection on Network Flows*. Advisers: Konrad Wolsing. Examiners: Klaus Wehrle and Ulrike Meyer. Master's Thesis. RWTH Aachen University. December 2021.

Dominik Kus. *Systematically Exploring the Fusibility of State-of-the-Art Industrial Intrusion Detection Systems*. Advisers: Konrad Wolsing, Eric Wanger, and Jan Pennekamp. Examiners: Klaus Wehrle and Elmar Padilla. Master's Thesis. RWTH Aachen University. February 2023.

Patrick Wagner. *Evaluating the Stability of Industrial Intrusion Detection Systems*. Advisers: Konrad Wolsing and Eric Wagner. Examiners: Klaus Wehrle and Sandra Geisler. Master's Thesis. RWTH Aachen University. May 2023.

Antoine Saillard. *Exploring the Effectiveness of Intrusion Detection for Marine Radar Systems.* Advisers: Konrad Wolsing. Examiners: Klaus Wehrle and Elmar Padilla. Master's Thesis. RWTH Aachen University. December 2023.

Luisa Lux. *Learning Physical Control Invariants for Industrial Intrusion Detection.* Advisers: Konrad Wolsing and Eric Wagner. Examiners: Klaus Wehrle and Ulrike Meyer. Master's Thesis. RWTH Aachen University. February 2024.

Attribution of Contributions

This dissertation is based on a compilation of ideas and content previously published in different papers (cf. List of Publications above). As the research presented here was conducted in collaboration with various colleagues and students over the years, we disclose the contributions of each contributor in the following. If not mentioned otherwise, I was the main contributor for the initial idea, realization and implementation, conducting evaluations, writing and publishing of the papers. If not mentioned otherwise, all co-authors contributed to reviewing, polishing, and editing the publications. Where possible, the publications are grouped into related topics.

IPAL [WWH20, WWSH22]. The Industrial Protocol Abstraction Layer (IPAL) resembles a central contribution and the foundation of this dissertation. IPAL is based on an observation by Martin Henze and myself that industrial protocols share similar features leveraged repetitively across IIDSs. Eric Wagner developed the first prototype of IPAL, which we presented in a poster pitching the idea of IPAL. The publication was written together with Eric Wagner and Martin Henze [WWH20]. Eric Wagner contributed the first draft of the design challenges and potential chapter. Over time, many persons have contributed to the IPAL project either to add new protocols, (re-)implement IIDSs, make more datasets and metrics usable, or to provide valuable software enhancements. All 18 contributors as of April 2025 are acknowledged in the IPAL software documentation [FC24]. In contrast, the publication for IPAL was written with a smaller team, including Eric Wagner, Antoine Saillard, and Martin Henze [WWSH22]. Antoine Saillard contributed by re-implementing further IIDSs relevant to this publication as part of his bachelor's thesis, whereas Eric Wagner was, among others, especially involved in writing the draft for the chapter on the motivation for protocol-independent intrusion detection. Later, Frederik Basels re-implemented the Invariant IIDS [FPMC19] in IPAL contained in this dissertation as part of a practical project course. IPAL impacts large parts of this dissertation but is mainly condensed into Chapter 4 and partially in Section 3.5. The evaluation of Section 4.3.3 was developed as a part of Myrat Altyyev's master's thesis.

IIDS Evaluations [LWW+23]. Olav Lamberts came up with the idea to compare different evaluation metrics that measure the detection performance of an IIDS as a practical project and continued this work in a seminar course. One outcome of his practical work is the foundation for the evaluation tool ipal-evaluate, which is incorporated into the IPAL software suite. More importantly, Olav Lamberts took over the work-intensive task of conducting the SMS and was partially assisted by Jan Bauer. The survey data and seminar text by Olav Lamberts built the draft for the subsequent publication, which was written collaboratively with all authors. I was responsible for evaluating the obtained survey data and leading the writing process of the publication based on Olav Lamberts' seminar text. Martin Henze and Jan Pennekamp provided valuable feedback and helped edit the publication. Eric Wagner additionally assisted in evaluating the data. The results of this work form the basis of Chapter 3.

SIMPLE [WTvS+22]. My idea of assessing how effective simple IIDSs can be for intrusion detection was examined in a bachelor's thesis by Lea Thiemt. For the publication, the team was augmented by Eric Wagner, Christian van Sloun, and Martin Henze. Eric Wagner provided the drafts for the introduction and background sections, whereas Christian van Sloun was responsible for the design chapter. Martin Henze revised the publication and provided valuable feedback. Section 5.1 presents the results and impacts of this research, and parts of the initial literature analysis are discussed in Section 3.5.2.

GeCo [WWL+25]. The idea, design, first implementation, and experiments of GeCo were derived in collaborative meetings between Luisa Lux, Eric Wagner, and myself and later extended by myself after the finished master's thesis by Luisa Lux. Eric Wagner and Luisa Lux were involved in writing the draft for the design chapter of GeCo, and Luisa Lux provided the draft for the conclusion. Eric Wagner and Luisa Lux assisted in conducting and evaluating the user study. Martin Henze joined the collaborators for recruiting participants for the survey, writing up the publication, polishing and providing feedback. GeCo is embedded in Section 5.2.

Radar Security [WSB+22, SWWB24]. The contributions to the radar security topic of this dissertation are two-fold. The initial ideas and implementations for the RadarSec-Lab and the Radar Attack Tool (RAT) were developed in a practical project course by Antoine Saillard and Jan Bauer. I entered the project to enhance the testbed for dataset recording and later led the publication process. The first publication [WSB+22] was created in a collaborative effort by a large team of the practical project course's supervisors and members of the Fraunhofer FKIE, all listed as co-authors of the publication. Jan Bauer provided the first outline and

the paper structure. Mari Schmidt contributed the background on radar communication protocols, and Ina Fink wrote the related work chapter. Christian van Sloun and Eric Wagner worked on the taxonomy for attacks against radar systems.

The second contribution used in the dissertation leverages this environment to examine the effectiveness of Intrusion Detection Systems (IDSs). Antoine Saillard worked on this topic for his master's thesis and was responsible for the implementation, evaluation, and first draft of the publication. The designs of the IIDSs were invented in collaborative meetings during the master's thesis. Antione Saillard worked on the evaluation and wrote the first draft of the publication [SWWB24], which was reworked and finalized by Jan Bauer and myself. Both works are jointly embedded into Section 5.3 as one contribution of this dissertation.

Deployment Experiments [WWB+24]. The practical experiments of Section 3.5.2.4 and Section 5.1.6, centered around the stability of hyperparameters, are based on a master's thesis by Patrick Wagner. He is also responsible for the foundations of the ipal-tune tool necessary for this contribution. In contrast, the evaluation from Section 6.1 was implemented and conducted by Frederik Basels. Together with Eric Wagner, who provided valuable feedback, I joined these two research activities in a publication about the deployability of IIDSs.

IIDS Ensembles [WKW+24]. The idea to combine IIDSs and thereby increase their detection performance was born in a practical project course and a subsequent publication by Dominik Kus investigating the inabilities of misuse-based IIDSs to detect novel attacks [KWP+22a]. His following master's thesis, which was mainly concerned with the combination of misuse-based IIDSs and presented as a poster [KWP+22b], laid the foundations for combining anomaly detection-based IIDSs. The initial ideas examined in his master's thesis were greatly enhanced and extended by myself after the master's thesis and summarized in a joint publication [WKW+24]. Dominik Kus contributed to the publication by working on related work and revising the publication. Eric Wagner, Jan Pennekamp, and Martin Henze collaborated on paper writing by providing feedback and proofreading. This paper resembles the last contribution of this dissertation for Chapter 6.

Contents

1

Introduction

The automation of physically demanding and repetitive work by offloading labor-intensive tasks from humans to machines is a continuous endeavor in the history of humanity [Ben96]. Historically, automation dates back more than 2000 years to ancient inventions such as early vending machines that were used to hand out holy water or to operate water-powered organs [Shu11, Gri19]. In contrast, today's *Industrial Control Systems (ICSs)* are able to take on a wide range of tasks such as purifying and distributing our drinking water, electric power generation and delivery, manufacturing and transporting necessary goods, or even automated navigation and steering of vessels across oceans. These achievements in automation lay the foundation for our modern society and grant humanity tremendous freedom in our daily lives. As a consequence, humanity increasingly relies on this high level of automation to the point of being entirely dependent on it, as is the case for electrical power [Els12]. Thus, ICSs' faultless operation gains utmost importance, which can be achieved through the advancements in digitization and networking. But, this trend likewise opens the doors for cybercriminals aiming to disrupt such facilities and endanger the environment or our society [ACZ20, MSM+21, SMLC21].

Usually, ICSs should be protected against cyberattacks through preventive security measures [SB21]. But, retrofitting, e.g., encryption or authentication, can be difficult for existing ICSs [AWT+20, DLP+22, WRW+23] as it may require shutting down the plant or exchanging deprecated hardware. Moreover, preventive measures can be imperfect or contain implementation errors and thus might be circumvented by attackers (in the future) [MC99]. As an alternative method, this dissertation is concerned with research on improving the cybersecurity of ICSs by means of *intrusion detection*. An Intrusion Detection System (IDS) unveils the presence of a cyberattack [Wan09, SB21], e.g., by monitoring network traffic and alerting the operators. The intuition behind intrusion detection is that damage to an ICS can be contained or optimally even prevented if a threat is detected quickly. To identify attacks effectively, researchers invent plenty of methods to realize the automatic training of

intrusion detection models [MC14, DHX+18, GUC+18, HYL+18, LXC+21, ZWF+21, UJJM22], e.g., by learning an ICS's normal behavior and alerting any deviations.

To narrow down the topic of this dissertation, we begin with an introduction to ICSs in Section 1.1 and a motivation for using IDSs to strengthen ICSs' cybersecurity in Section 1.2. Afterward, Section 1.3 provides an overview of IDS research in the ICS sector. We concretize the scope of this dissertation and formulate the overarching research question in Section 1.4, centered around how generalizable and transferable IDS solutions are between the various ICS use cases and how we can make them more accessible to ICSs. Lastly, Section 1.5 provides an outlook on the following chapters of this dissertation.

1.1 The Digitization of Industrial Control Systems

Before we delve into the cybersecurity issues of Industrial Control Systems (ICSs), we take a look at what an ICS is and how it has evolved into today's versatile digital systems [KL14]. We begin with a definition by the National Institute of Standards and Technology (NIST) stated in their Guide to Operational Technology Security:

> *"An ICS consists of combinations of control components (e.g., electrical, mechanical, hydraulic, pneumatic) that act together to achieve an industrial objective (e.g., manufacturing, transportation of matter or energy)" [SLP+13].*

The first parts of an ICS are the *sensors and actuators* connecting the ICS to the physical environment. An example of such control components could be an actuator in form of a valve opening a water faucet and a sensor measuring the water level in a tank. Since sensors and actuators are just standalone instances, they cannot achieve industrial objectives, e.g., maintaining a certain water level. To this end, control components connect them, forming a *control loop* between the control components [DSW76]. Control loops usually first collect measurements with sensors, then process the gathered information about the (physical) *process state*, and lastly command the actuators accordingly. Since the sensors and actuators, as well as associated control components of a larger physical process may be installed in spatially divided locations, they have to be networked and communicate with each other.

Until the 19th century, process control was conducted by workers. They manually controlled and supervised the process near the devices themselves, which was a resource-intensive task [Pit23a]. Central control rooms were the next evolution, bundling all control instruments in one place [Pit23b]. Although they solved the previous disadvantages, they were only flexible to a limited extent if changes had to be made to the process. Finally, the progressing advancements in Information Technology (IT) paved the way for realizing distributed industrial control networks. With the help of digital communication technologies [GH13], they provide flexibility and enhanced remote steering capabilities for ICSs.

In Section 1.1.1, we look at the applications of modern ICSs enabled through the advancements in IT. Section 1.1.2 then highlights how digital communication technologies shape how ICS's control components and control units are interconnected.

1.1.1 Modern Applications of Industrial Control Systems

As the digitization of ICSs progressed in the 20th century, the Programmable Logic Controller (PLC) was invented to take over the process control [Pit23b]. A PLC is essentially a ruggedized and programmable computer that commands control components in ICSs [SLP+13]. Such special and reliable devices are necessary since industrial facilities are built for long lifetimes (up to 30 years) [SHH+21]. Multiple PLCs can be connected with each other in a complex, interconnected system. The flexibility achieved through this digitization and interconnection enables many use cases. In the following, we present relevant industrial domains for this dissertation.

Common applications for ICSs encompass the purification of public water, its distribution, and finally the treatment of wastewater [RAMH18]. Similarly, the distribution of gas and fuels through networks of pipelines, as well as their refining processes, is automated and closely monitored [MTT15]. The same holds for the generation and delivery of electric power to industries and households [BSL+23]. Besides the usual suspects for automation in manufacturing and chemical processing facilities, there are other domains, such as public transport [LKP+19] or maritime shipping [BUH+22], that make use of the same principles as in larger ICS facilities but on a smaller scale. For example, vessels generate their own power and navigate semi-autonomously as a tightly integrated ICS that is monitored and controlled from a central place on a vessel's bridge by the crew [BUH+22].

All domains have in common that they rely on the advances in digitization and connectivity [KL14, MMA+16, DLP+22]. For example, access to the Internet enables use cases, such as remote monitoring and maintenance. In contrast, early ICSs were operated without digital connections to the outside world [MMA+16]. Projecting these trends into the future, ICSs will grow even more interconnected than they are today. Trends such as Industry 4.0 forecast that facilities will become more flexible in how they operate [LFK+14]. For example, ICSs are no longer restricted to producing just a single good in a fixed quantity. Further ideas envision coupling multiple facilities to allow enhanced collaboration [PBD+21]. These trends have in common that they rely on digital systems and reliable communication methods.

1.1.2 Digital Communication in Industrial Control Systems

An ICS requires real-time communication that is robust to failures and reliable even in harsh environmental conditions [GH13]. One early method for connecting control components and implementing process control in the late 19th century was pneumatic control systems [Pit23a]. Pneumatic pipes were first replaced with electrical wires [Pit23b] before digital communication systems took over. But, the requirements for communication in locally restricted facilities differ from those of the electrical power distribution domain overseeing large areas of countries. Thus, adopting existing IT communication technologies was historically not directly possible due to ICSs' unique properties. This varying set of requirements has led to the formation of a large pool of specialized and domain-dependent industrial protocols [GH13].

Nowadays, ICSs still work with these once proprietary, serial, or local protocols that are adopted on top of IT network protocols such as the Internet Protocol (IP). For example, Modbus was adapted to ubiquitous Ethernet networks, resulting in ModbusTCP [Mod18]. Therefore, these once proprietary and local network protocols became more open, enabling greater interconnection of industrial networks [MMA+16, DLP+22], e.g., for remote monitoring. While the pairing with connectivity to the Internet enables unique applications, this openness requires industrial networks to be protected against malicious intruders [HLLL17].

1.2 Methods to Protect ICSs Against Cyberattacks

The rather short list of exemplary use cases and their future paths in Section 1.1.1 already show that ICSs are and will remain a fundamental building block for our modern society, and failure to any of them can have severe consequences for our human society. For example, power outages for extended periods of time [Els12, LAC16], destruction of gas pipelines [Ics12], or contamination of drinking water [Vas23], as happened in the past, have significantly raised public and political awareness with regard to the cyber risks ICSs are exposed to. The growing number of observed cyber incidents with detrimental environmental damage and risk to human life [HEF18, Mor19, MSM+21] highlight the importance and urgency of adequately protecting ICSs. Consequently, many sectors where ICSs are in use are classified as critical infrastructures that demand special protection [AZ15].

In this dissertation, we refer to an attacker that aims to disrupt the physical process of an ICS [MIT24], be it to harm its business, endanger the environment, or people. Our *attacker model* assumes that the intruder has already gained access to the ICS's communication, its control components, or control units (we do not consider the initial intrusion). Thereby, they can issue harmful commands via the ICS's communication network to other control components. The effect of an attack then affects the data perceived by the ICS as either sensor or actuator values are inaccurate.

To understand the consequences of such attacks, we showcase the dangers and properties of cyberattacks for ICS in Section 1.2.1. Aiming to avert cyberattacks, we discuss what makes cybersecurity for ICSs challenging and how it differs from cybersecurity for IT systems in Section 1.2.2. Since preventive security measures introduced in Section 1.2.3 may not be perfect, Section 1.2.4 motivates the need for and the rationale behind intrusion detection — the topic of this dissertation.

1.2.1 The Dangers and Properties of Cyberattacks on ICS

The rapid digitization of ICSs has disadvantages, as facilities initially designed to operate as air-gapped systems become increasingly connected to the (public) Internet. While modern IT-based communication networks open the path for various use cases and optimizations, ICSs inherit the same cybersecurity issues that traditional IT systems face. Consequently, there is an escalating rise in cyberattacks across all industrial domains [ACZ20, MSM+21, SMLC21].

Early on in 2007, researchers proved with the Aurora generator test [Wei16] that through the coupling of digital and physical systems, as achieved by ICSs, cyberattacks can now have severe physical consequences such as the destruction of power grid components. Not much later, in 2010, the prominent cyberattack, Stuxnet, was developed with the intention of destroying the centrifuges of the Iranian nuclear program [FMC11]. Despite being an air-gapped facility, the attack managed to reach the inner boundaries of the ICS. Besides such highly coordinated and likely state-level attacks, the hurdles for attackers do not have to be that high in general as industrial devices can also be found freely accessible on the Internet [DLP+22]. Note that cyberattacks are caused intentionally and do not occur randomly as faults (e.g., wear and tear) or mistakes (e.g., human error) [GUC+18].

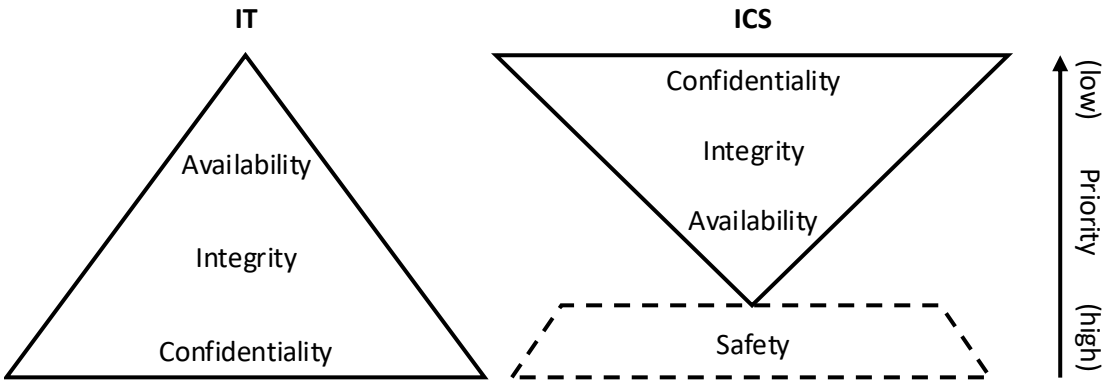
When considering the time course of a cyberattack, it becomes evident that attacks are not a single event but rather a process. As summarized by the MITRE ATT&CK framework [ABS20] and the MITRE ICS Matrix [MIT24], cyberattacks traverse different phases that each take variable amounts of time. Attacks usually, but do not necessarily, begin with a reconnaissance phase during which the attackers gather information about their target and continue with initial access by, e.g., means of an exploit. Afterward, attackers may move through the network to reach additional targets or perform privilege escalation attacks to widen their access. Only after such preparation steps does the actual impact take place, such as performing modifications to a physical process. Also, as can be seen from Stuxnet, inflicting physical damage is not always instantaneous. Therefore, there are opportunities for defenders to mitigate such an attack at an earlier stage where no or just minimal physical damage has been conducted yet. Traces of an attack can be found in, e.g., network patterns or deviating process behavior, disclosing an attacker's activities and presence.

One property of attacks that is special in ICS compared to IT systems is that cyberattacks are usually specially targeted to a single facility or use case. In contrast, IT systems leverage common software, e.g., web servers, across millions of Internet-accessible devices. Thus, malware can leverage vulnerabilities broadly across many devices, e.g., encrypt their data and extort ransom money. But, to conduct *physical* damage to an ICS, knowledge about its inner workings may be required, such as in the case of Stuxnet [FMC11]. Since ICSs differ from one to another, the attacks also have to be adopted according to the unique scenario of their target.

Regardless of the methodology, all attacks pose a severe risk to the process, businesses, environment, and society. Moreover, the widely deployed legacy devices in ICSs can not effectively withstand such attacks as they were historically not meant to implement crucial cybersecurity measures [CDV13]. Thus, to prevent further incidents in the future, solutions for mitigating cyberattacks are required.

1.2.2 Means of Cybersecurity for ICS

The risks associated with the rise of cyberattacks, especially against critical infrastructure, are well recognized by legislation and standardization organizations. For example, the European Union published the NIS-2 regulation [NIS], demanding enhanced cybersecurity efforts for various important domains. Targeted explicitly for



(a) Security principles in IT. **(b)** Security principles in ICS.

Figure 1.1 Cybersecurity is concerned with three properties of technical systems, namely their confidentiality, integrity, and availability. Their importance depends on the target application. In ICS applications (cf. Figure 1.1b), the priority is ordered differently. Moreover, the safety of ICSs may even dominate and outweigh any cybersecurity measure [KPH+15].

ICS, the IEC-62443 standard [IEC13] publishes a list of requirements that has to be fulfilled to strengthen cybersecurity. Yet, simply adopting cybersecurity solutions from conventional IT to ICS is not always trivial.

Cybersecurity in the IT domain is concerned with three properties: confidentiality, integrity, and availability [SB21] (cf. Figure 1.1a). Confidentiality is concerned with making data exchanged, processed, or stored only accessible to entities eligible to obtain the information. In contrast, integrity ensures that changes to information without authorization are detectable. Lastly, availability states that adversaries cannot easily disrupt a service and the service remains accessible to authorized entities.

While ICSs also recognize these properties, they are not only prioritized in a different order, but also overshadowed by the safety property [KPH+15] (cf. Figure 1.1b). Ensuring operational and process safety is the utmost priority in an ICS to avoid damaging the process, environment, or humans. For example, an emergency shutdown due to a shortage of a power line must be prioritized before cybersecurity measures. The three conventional measures are perceived with different importance [CDV13] since availability, relatable to safety, is inevitable in critical infrastructures such as the power grid. Integrity is still necessary to detect unintentional changes to or manipulations of, e.g., measured data. Confidentiality follows last. While the data inside an ICS is usually still considered confidential, confidentiality is less important compared to the other properties.

To realize and protect these security principles, various categories of security measures exist one can choose from to protect a system, e.g., preventive and detective measures [SB21]. While preventive measures aim to mitigate an attack in the first place, they may be imperfect, as cyberattacks might circumvent them or inapplicable to an existing system with old hardware. As one alternative, detecting all attacks slipping through any other security measures is equally relevant. Only then can other measures react to them. For example, containing the damage, performing system cleanups, or conducting forensic analyses to understand the weaknesses in the current system and to enhance other measures in the future. In the following,

preventive measures are briefly explained in Section 1.2.3 and Section 1.2.4 motivates the idea and rationale behind detective measures.

1.2.3 Preventive Security Measures

As shown in the past, even entirely air-gapped systems have been compromised already (cf. Section 1.2.1). To counter these persisting security issues, ICS operators have access to a wide range of solutions to mitigate cyberattacks by means of preventive measures [SB21]. Their goal is to proactively prevent vulnerabilities and reduce the likelihood of cyberattacks occurring in the first place. Even standards exist for ICSs listing possible measures that should be fulfilled [IEC13]. Exemplary preventive security measures are the following. A first measure would be to perform network segmentation and separation so that initial access to the ICS network or the spread of attacks through the ICS network is hindered [GS12]. One crucial building block to achieving network security at boundaries can be a firewall. Next, implementing authentication, access control, and assigning specific roles to users are additional means to limit the actions a single entity can perform. Another example of preventive measures are secure variants of industrial communication protocols [CDT21, DLP+22], i.e., authentication, cryptographic integrity protection, or encryption to prevent unauthorized modifications to communication channels.

However, deploying preventive security measures can become difficult in practice. For example, access control can be unwanted for safety reasons in cases where operators would have to waste time authenticating themselves in a critical situation to avoid process damage. Also, while industries with long lifetimes (cf. Section 1.1.1) arise to save cost and maximize availability, they have the consequence that deployed hardware is often hardly updateable and, furthermore, only has minimal processing power. Deploying cryptographic protocols, such as VPN, can become difficult on ICS hardware [LKS19, SHH+21]. Thus, integrating secure communication protocols is not always trivial as they add bandwidth overhead and processing delays to a network that has to fulfill real-time requirements [SHH+21]. Less intrusive methods to retrofit preventive security measures through additional devices that authenticate traffic between them [TS08, LGH+21, HBW21], or the embedding of authentication data directly in communications [WRW+23] have also had little impact in practice, as commercial plug-and-play solutions are hardly available.

Overall, there exist great opportunities to protect ICSs from malicious intentions of attackers by a variety of solutions. Yet, implementing them in practice can be difficult for various reasons. Furthermore, as the continuation of successful cyberattacks demonstrates, even the best preventive measures can not provide perfect cybersecurity. Therefore, orthogonal security measures have to be considered additionally or as compensating measures to achieve comprehensive defense-in-depth.

1.2.4 Detective Security Measures

Intrusion detection automatically uncovers cyberattacks or other suspicious activity by monitoring a system's behavior, e.g., with respect to its communication [Wan09,

SB21]. Intrusion detection is particularly attractive for ICSs due to their passive nature. In contrast, retrofitting (active) security measures, i.e., integrity protected or authenticated communication protocols, often results in costly hardware or software modifications, significant downtime, or the need to redo acceptance tests [IEC24]. Moreover, detective measures can be implemented as complementary security in addition to preventive measures and are even demanded by current security standards [IEC13, NIS, SLP+13] or for critical infrastructures (KRITIS) in Germany.

The core assumption behind Intrusion Detection Systems (IDSs) in general is that cyberattacks lead to distinctively different (anomalous) system behavior. Their task is to identify such malicious behavior that indicates the presence of an ongoing cyberattack in near real-time by constant monitoring. Since cyberattacks are performed in stages (cf. Section 1.2.1), an IDS can indicate the first signs of a compromise, even before physical damage is dealt in an ICS scenario. For example, without intrusion detection, stealthy attacks can remain undetected for a long time, as in the Stuxnet attack. An IDS shortens the time attackers can operate undetected by alerting the system operators. Operators can then conduct further investigations or automatically prevent an attack likely even before physical damage is performed, e.g., by blocking specific network traffic. Detecting and mitigating a cyberattack early also reduces the total cost caused by the attack on average [IBM24].

Traditionally, IDSs were designed for IT environments, i.e., offices, servers, or datacenter networks, with largely fluctuating traffic and usage patterns [SP10]. Even though this makes it challenging to define benign activity comprehensively, there is a decent understanding of how individual attacks are performed on these networks. This is the case since they leverage similar software, e.g., web servers, with similar vulnerabilities. Thus, standard methods to reliably detect such (known) attacks are *signature- or rule-based* approaches, which compare user and network behavior against a database of attack indicators. There exist well-established implementations such as YARA [Alv08], Suricata [Sur21], Zeek [Zeek] (previously Bro [Som03]), and Snort [Roe99]. In contrast, *anomaly detection* learns the normal behavior of a system and alerts any deviations from it.

However, established intrusion detection solutions from the IT domain are not as effective in industrial contexts for various reasons [ZHX+15, AWBJ21]. First, attacks against industrial networks are often specifically designed for a single or few targets (cf. Section 1.2.1) and hardly generalize across domains (as, e.g., malware does), making it hard to catch them by predefined rules. Next, ICSs are susceptible to cyberattacks targeting the physical process [UGC+16, ACZ20], which causes physical damage to the process, facility, or environment. Furthermore, industrial settings are susceptible to subtle attacks [UGC+16, ZHW+22] where attackers send legitimate traffic (as allowed by rules) but at the "wrong" time to trip the system into a dangerous (physical) state. For example, a command closing the flood gate of a dam is legit but not during flooding season. Lastly, ICSs' reliance on unique (real-time) hardware, such as PLCs and domain-specific communication protocols, makes establishing unified solutions challenging. Thus, IDSs designed for typical IT networks cannot simply be transferred to ICSs [ZHX+15, AWBJ21].

Consequently, intrusion detection has to be thought of differently for ICS. Specialized *Industrial* Intrusion Detection Systems (IIDSs) address this gap by providing unique solutions for legacy industrial deployments [DHX+18, GUC+18]. Despite their differences from IT systems, ICSs likewise feature unique opportunities to detect intrusions and anomalies through different means. First, ICSs are inherently more structured such that this knowledge can be integrated into the detection methodologies. Likewise, their rather deterministic industrial tasks lead to more predictable communication patterns due to the heavy use of repetitive machine-to-machine communication [HHS+18]. Also, the tight coupling between the industrial processes and the physical environment motivates a new class of *process-aware IIDSs*. This class incorporates information from the physical process and/or communication level to uncover highly specialized and subtle attacks and anomalies [UGC+16]. Therefore, IIDS approaches typically "learn" a model of benign behavior for a specific facility and alert deviations from this model (anomaly detection). Such models may incorporate additional system knowledge, e.g., that certain actuator combinations never occur, resulting in an IIDS that does not necessarily require attack knowledge.

One downside of intrusion detection is that the detection methodologies are not perfect because benign and anomalous behavior cannot be strictly distinguished. Thus, an IIDS can struggle in these overlapping situations. This situation inevitably leads to false positives [AAM22], e.g., situations where the IDS suspects a cyberattack, but the observed situation is actually legit, or false negatives (missed attacks). While false positives can be avoided by making the IDS less sensitive, this change can simultaneously result in more false negatives. Finding a middle ground between both extremes is the challenge of ongoing research.

1.3 Industrial Intrusion Detection Research

Research on IIDSs is influenced by several external factors, such as the plurality of ICS domains (cf. Section 1.1.1), ICSs' diversity in communication technologies (cf. Section 1.1.2), and distinct attack vectors of attackers (cf. Section 1.2.1). Consequently, new detection methodologies are developed in a way that they fit the unique needs of certain applications. This environment provides great opportunities for academia to invent appropriate IIDS solutions resulting in a research landscape that aims to protect vulnerable ICSs by means of intrusion detection.

Inventing new detection methodologies is not the only research activity as, e.g., benchmarking datasets are required to enable fair and objective comparisons between proposals. To present a holistic overview of IIDS research, we categorize the research activities along nine interdependent dimensions as visualized in Figure 1.2, similar to the categories considered by Umer et al. [UJJM22] to classify the different types of literature in their survey. Our categories involve the ICS scenario, the attacker model, the data type, the detection technique, the IIDSs' alerts, the benchmarking environment, the evaluation metrics, reactions to an alert, and the deployment of IIDSs. In the following, we briefly describe their individual purpose for IIDS research.

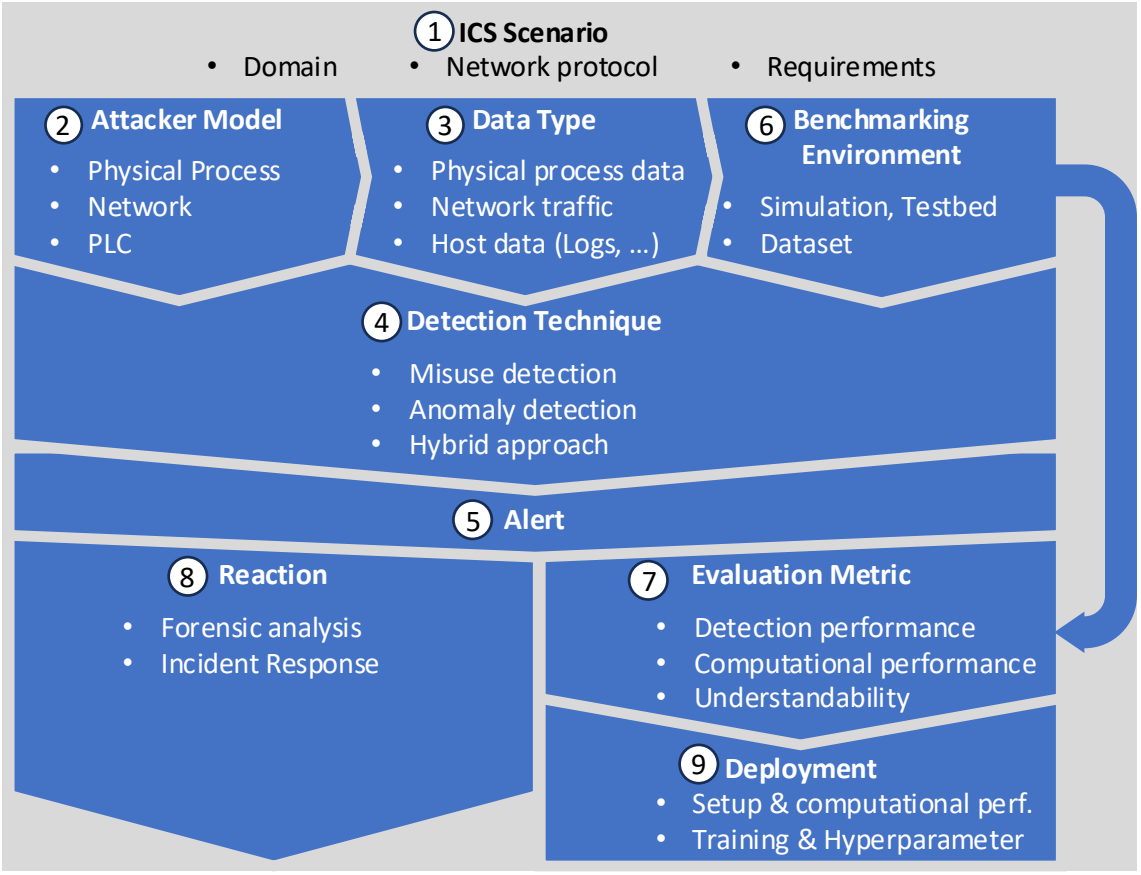


Figure 1.2 Industrial intrusion detection research can be assessed along nine essential dimensions influencing each other. All activities aim to develop powerful intrusion detection methods capable of protecting ICSs in real deployments.

One major factor influencing all other directions of IIDS research is the *ICS scenario* ①. As industries are diverse in domain and technology, such as network protocols [ORZ+24] or physical process, subsequent solutions must address their specifics. Thus, researchers tackle intrusion detection for various domains, such as water treatment facilities [RAMH18], gas distribution [PASE18], chemical processes [KLG15], the maritime domain [WRBW22], or power distribution grids [LNTA17]. The various surveys on these domains highlight not only the broad interest in intrusion detection across these domains but also its importance as a viable security solution.

Next, the *attacker model* ② (cf. also Section 1.2) influences which kind of attacks an intrusion detection system should be able to detect and potentially even differentiate. For example, while a DoS attack within the network can inhibit the real-time communication between PLCs, malware reprogramming a PLC can change the physical process. For an extensive list of possible attack models, please refer to the MITRE ATT&CK classification for ICS [ABS20, MIT24]. Overall, the attacker model greatly determines which *data type* ③ the IIDS monitors, since indications of the attack should show in this data. Common ones are network traffic, e.g., to detect a DoS attack, host data from SCADA systems or PLCs, e.g., to detect unwanted modifications to the devices, and physical process data, e.g., to find out whether the entire ICS operates as intended [LXC+21].

The main work of researchers goes into designing *detection techniques* ④, which can be categorized into misuse, anomaly, or hybrid detection [LDSR05, MC14]. While misuse detection (also referred to as knowledge-based or supervised detection [MC14]) identifies harmful actions based on (known) patterns of cyberattacks, anomaly detection instead specifies how the ICS behaves normally and alerts deviations from usual actions. Moreover, the detection technique is also influenced by the attacker model and data type. While attacks on the network can be detected on a per-packet basis, e.g., with deep-packet inspection [GW13], process-based detection leverages a broader view of an ICS's physical state, e.g., by analyzing whether the process moves towards a critical state that may lead to an incident [CCG+11]. The output of an IIDS is an *alert* ⑤ to indicate the suspected presence of a cyberattack. Yet, since the underlying detection mechanisms might not be perfect, it can be the case that cyberattacks are overseen or false alarms are emitted.

To validate the design and facilitate comparability of a newly proposed IIDS, their detection performance is evaluated with the help of suitable *benchmarking environments* ⑥ and *evaluation metrics* ⑦. Despite the data type, benchmarking environments for all kinds of industrial domains come in different forms, such as datasets, physical testbeds, simulations, or real facilities [KKG19, CDT21]. Each type has its trade-offs in terms of, e.g., accessibility, cost, or closeness to real deployments. Moreover, an IIDS's performance needs to be measured based on metrics, enabling scientific comparisons between proposals. In that regard, scientists can refer to a plethora of metrics that measure detection performance [Pow11], such as precision, recall, or the F1 score. But, evaluations may concern further aspects, such as the computational performance or understandability of alerts [SGAL22].

In practice, an IIDS alarm results in *reactions* ⑧ to mitigate an attack. To this end, operators may conduct (manual) forensic analyses to understand the cause for an alert [AJK+21] and ultimately mitigate the threat [SMLC21] by, e.g., applying firewall rules. Detection mechanisms can also be coupled to preventive measures for automated reactions called Intrusion Prevention Systems (IPSs) [SB21]. Those need, however, to be carefully designed for ICSs since simply blocking suspicious traffic in an industrial setting may cause more harm than the suspected attack itself. The handling of alarms, called alarm management, can also be found outside the IDS domain, e.g., on vessels in their engine rooms [SH24]. As crews may face up to a hundred hourly alarms from their monitoring systems [SH24], an IIDS should perform inherently well not to cause too many additional (false) alerts [AAM22].

The final dimension concerns transferring IIDSs to real-world *deployments* ⑨. This step should be considered already during research so that developed solutions are practicable and realizable in practice [Eta17]. Challenges during deployment could, e.g., be the computational performance of an IIDS or obtaining training data. Ultimately, research should thrive toward IIDSs that fulfill the needs of actual ICSs.

These various types of research activities centered around industrial intrusion detection raise the hope that ICSs are becoming more secure. However, researchers must take care that all the different streams of research actually converge in a tangible manner to yield a practical impact in the end.

1.4 How Can We Make IIDSs More Accessible for ICS?

One particularity of ICSs that affects the design of IIDSs is their diversity in domains and use cases (cf. Section 1.1.1), which attracts researchers with various backgrounds. Moreover, due to the myriad aspects concerning IIDS development (cf. Figure 1.2), research takes place at a multitude of dimensions. This environment constitutes a unique opportunity to tackle IIDS research from varying angles, transfer insights across industrial domains, and investigate their efficiency in real-world deployments.

However, the differences in ICS technologies result in complex IIDSs being developed that function well for one industry or protocol but are inapplicable elsewhere. This diversity segregates the research landscape, resulting in isolated silos, as we will show in Chapter 3. This segmentation and complexity slows down the applicability of IIDSs to different ICS scenarios and limits the accessibility of IIDSs for ICSs.

We are convinced that there exist fundamental similarities between ICSs for intrusion detection to remedy this situation. Since the underlying physical phenomena remain the same for all ICS, IIDSs can theoretically *generalize* to several protocols and *transfer* to multiple ICS use cases. To what extent this vision of generalizable and transferable IIDSs is actually feasible is the content of this dissertation. Additionally, by reducing inherent complexity and increasing comprehensibility for ICS operators, we seek to bring the research domain forward toward sustainable, coherent, and improved research methodologies. Thereby, we make IIDSs more accessible to ICSs and contribute to securing industrial facilities across various domains.

1.5 Outline

Chapter 2 introduces the necessary background knowledge about ICSs, intrusion detection, and relevant research assets such as testbeds and metrics used in this dissertation. Following up in Chapter 3, we start assessing the current state of IIDS research by means of a Systematic Mapping Study (SMS) and short experiments. The findings are then summarized in five issues (I1–I5) that the subsequent contributions of this dissertation address in the next three chapters. Our main contribution, the Industrial Protocol Abstraction Layer (IPAL), is introduced in Chapter 4. IPAL not only addresses the generalizability issues of IIDSs in the ICS domain but likewise serves as a research tool for the subsequent dissertation. Leveraging IPAL, we evaluate the generalizability and transferability of three novel IIDSs addressing inherent complexity and comprehensibility issues in Chapter 5. In our last contribution in Chapter 6, we take a closer look at which and how IIDSs should be deployed by means of ensemble learning. Finally, Chapter 7 ties together all the findings of this dissertation and recaps to which extent IIDS can now better generalize and transfer beyond the use cases examined in research and more effectively protect ICSs.

2

Background

Before we start with the contributions of this dissertation, this chapter introduces the reader to the necessary background knowledge. We begin with a closer investigation into the architecture and inner workings of Industrial Control Systems (ICSs) in Section 2.1. Following up, Section 2.2 explains the ideas and the concepts behind intrusion detection on a general level and concerning the specifics of intrusion detection in ICS. The last two sections are about the datasets (Section 2.3) and metrics (Section 2.4) used in this dissertation to assess the detection performance of Industrial Intrusion Detection Systems (IIDSs).

Note that the contents of this chapter are based in parts on existing publications. More precisely, Section 2.1.1 and Section 2.1.2 use parts of the background from [WWL+25]. Section 2.2.4 and Section 2.3 to Section 2.4 use parts of the background and appendix from [LWW+23].

2.1 Introduction to Industrial Control Systems

An ICS is a versatile system that implements automation in various domains (cf. Section 1.1). To get to know ICSs in more detail, we first introduce an exemplary ICS process in Section 2.1.1 that is used as a running example throughout this dissertation. Furthermore, digital control loops are explained in Section 2.1.2 as they represent the primary mechanism that implements the automation in an ICS. We continue with an overview of the digital architecture of ICSs in Section 2.1.3 since they follow a distinct hierarchical layout. With the basics established, the next topic concerns how connectivity and communication in an ICS environment are achieved. To this end, we first highlight the specifics of industrial protocols in general in Section 2.1.4 after which we concentrate on one selected protocol (Modbus). Finally, in Section 2.1.5, we take a look at how attackers can breach an ICS, explaining the attacker model assumed in this dissertation along with a representative example.

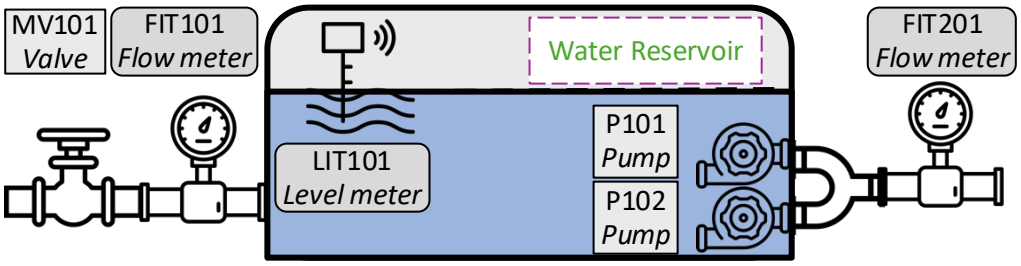


Figure 2.1 Exemplary ICS of a water reservoir to provide a constant water flow to subsequent processes. Sensors are indicated with rounded and actuators with regular rectangles.

2.1.1 ICS Example Process

ICSs are used to automate a wide range of scenarios (cf. Section 1.1.1), one of which are water treatment facilities. Such a scenario is also modeled by the prominent SWaT research testbed [GAJM16]. Throughout this dissertation, we use a simple subsystem of this testbed, a water storage tank, as an illustrative example.

The system, depicted in Figure 2.1, stores a supply of water in a tank for further processing. The tank is equipped with sensors and actuators to monitor and manipulate the physical state. The sensors are flow meters that measure the inflow (FIT101) and outflow (FIT201) of the water tank in cubic meters per hour. Additionally, a level meter (LIT101) measures the tank's water level in millimeters. The actuators involved in the process are a motorized valve (MV101) that controls the water inflow into the tank, a pump (P101) that forwards water for further processing as well as a backup pump (P102). All actuators can either be turned on or off (pumps) or be open or closed (valves). In this example, the industrial objective is to maintain a sufficient water supply and prevent the tank from draining or overflowing.

2.1.2 Digital Control Loops

Since sensors and actuators alone are not sufficient to accomplish complex industrial objectives, experts with knowledge of the ICS program Programmable Logic Controllers (PLCs) to automatize the process logic [GH13, KL14], as shown in Figure 2.2. The PLC continuously sends out commands to the actuators, obtains reports of their impact on the physical world back through sensor readings, and compares the effect with the desired setpoint, which is called a closed-loop system [DSW76].

Control loops can be as simple as our example process, checking whether the water level (LIT101) is low and opening the valve (MV101) until the tank is sufficiently filled. But, they can also control complex systems by leveraging multiple in- and outputs to model complicated functions [BRJ15]. One method to describe the physical process is through state-space models [Che84], which are mathematical equations predicting the evolution of the process. Considering our example and given the knowledge about the process and the sensor and actuator placements, the change in the water tank's level can be expressed as $\frac{\delta LIT101}{\delta t} = \frac{FIT101 - FIT201}{A}$ given the area A of the tank is known [APQ+20]. Leveraging this equation, the PLC can predict how the physical system behaves if, e.g., the valves or pumps are modified.

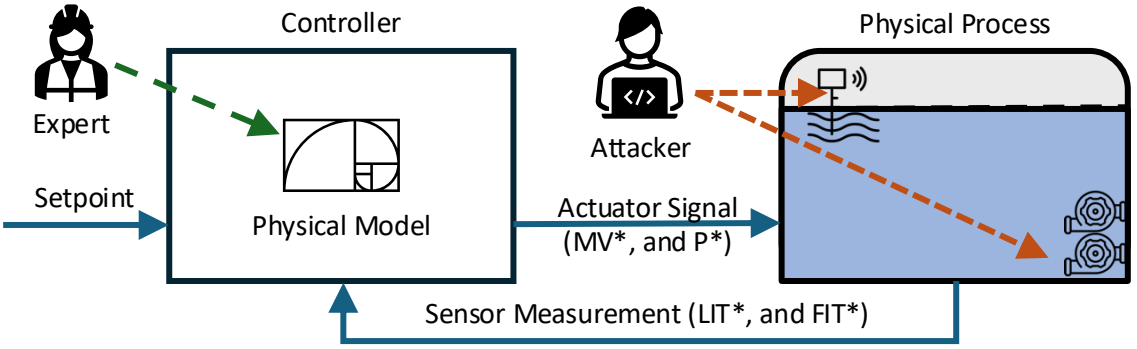


Figure 2.2 Digital control loops automatize physical processes according to a model provided by experts. Yet, ICSs' advances in digitalization pose vulnerabilities to cyberattacks.

2.1.3 ICS Architecture

To achieve an industrial objective, e.g., maintaining a certain water level, an ICS requires more components than just the sensors and actuators sketched in the example before. As visualized in Figure 2.3, an ICS comprises five layers that separate the individual tasks into different levels of automation [BKC+14, IEC16]. The following paragraphs explain the different layers from the bottom (production process) to the top layer (business planning and logistics) one by one.

Production process (Level 0). At the heart of an ICS is the physical process layer containing the interface to the real world. Compared to the controlled environments of data-centers, an ICS may be deployed anywhere, e.g., outside or on moving platforms such as trains, which requires specialized hardware [GH13]. Thus, industrial devices are ruggedized to withstand harsh environmental conditions such as temperature or vibrations. Also, ICSs are required to operate reliably for years, especially when interrupting the physical process for maintenance is cumbersome.

Sensing and Actuation (Level 1). The sensors and actuators of an ICS interface with the external world to measure or influence a (physical) property. Such a measurement could be the level of the water tank from our example, e.g., 800mm at 12:00. Similarly, the state of an actuator at 12:00 could be *on*. In this dissertation, we call such measurements a *process value* that are fed into an IIDS to classify whether such a (combination) of process values is benign or potentially anomalous.

Monitoring, Supervision, and Control (Level 2). Process control is achieved in the level two of the ICS hierarchy. It facilitates the PLCs implementing process control for a set of control components [KL14]. The individual control components are then connected to one PLC responsible for them. This is historically achieved through wires and setting or measuring a flow of electricity [Pit23b]. These signals range between 4 mA and 20 mA and need to be converted to the respective unit depending on the connected device.

Since ICS facilities can stretch over large areas, such as power grids, a single PLC usually takes over responsibility for a smaller part of an ICS. Likewise, complex processes can be a reason for segmenting the ICS through different PLCs. Then communication between the PLCs becomes essential, e.g., if a sensor value is necessary for multiple process stages [GAJM16, APM17] or for synchronization. To

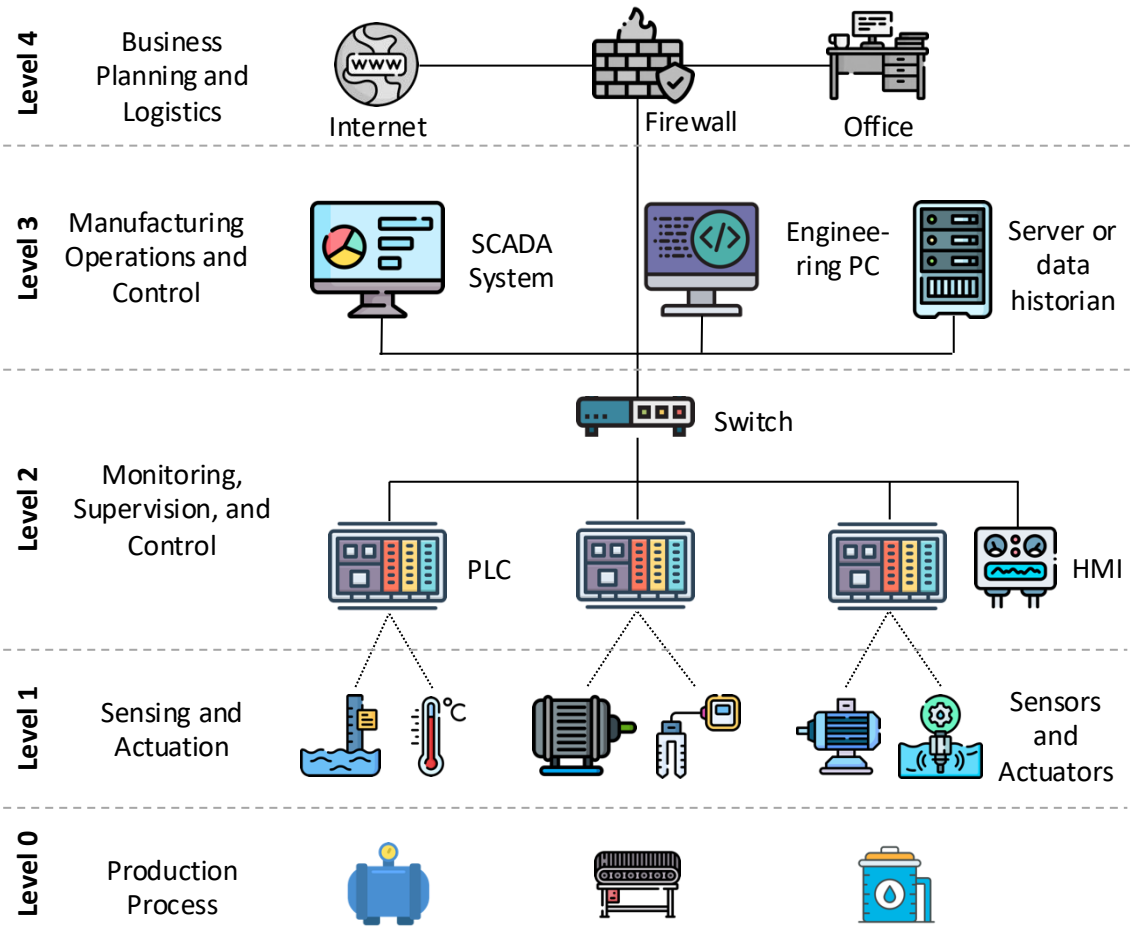


Figure 2.3 ICSs follow a hierarchical architecture where the physical process is on the bottom and higher layers provide more and more managing functionality. This figure is similar to Bangemann et al. [BKC+14] and the definition by the IEC 62264-3 standard [IEC16].

this end, the PLCs are connected through network switches with each other. But, processes may have strict temporal requirements, such as power plants to prevent shortages, down to the range of less than a milliseconds [GH13]. This has led to the development of industrial-specific network protocols such as Modbus. More details about these protocols follow in the next Section 2.1.4.

Lastly, Human-Machine Interfaces (HMIs) can be found on the process control layer, e.g., located in close vicinity to a process [KL14]. These serve as displays showing status information to an operator or accepting input to manually change the process.

Manufacturing Operations and Control (Level 3). While the assets on level two implement the majority of the automation functionality of an ICS, they are still dependent on the higher operation and control layer [KL14]. This level can be thought of as a control room serving as a central entity to supervise the industrial process and make adjustments to it. An adjustment could be, for example, speeding up the manufacturing process or shutting down the plant for maintenance. This level also facilitates engineering stations so that developers can optimize or change the automation programmed onto the PLCs. Lastly, a server may function as a collection point for historical data or perform ICS-wide analyses [KL14].

Business Planning and Logistics (Level 4). The highest layer of an ICS facilitates the business tasks of an ICS, e.g., its administration, managing logistics of resources and final products, or billing [KL14]. Optimally, this layer is separated from the industrial process, i.e., the lower layers, to mitigate unauthorized access to the process. But, modern ICSs may even make use of Internet-going connections, e.g., for remote monitoring of remote facilities without permanently stuffed control rooms. Thus, a complete separation between the industrial network and the public Internet is not always possible today anymore [DLP+22].

This hierarchical structure enables ICSs to adapt to many use cases and implementation sizes. Nonetheless, to connect all the layers and devices to each other, a reliable communication network is essential (cf. black lines in Figure 2.3).

2.1.4 Industrial Protocols

Industrial protocols enable communication between devices within the monitoring, supervision and control layer two, but also between layer two and the manufacturing operations and control layer three (cf. Figure 2.3). Yet, due to the underlying industrial processes, their protocols have to satisfy special requirements [GH13].

First, they need to have a high sensitivity against failures and a high reliability since industrial processes, such as the power grid, have to operate with nearly zero downtime. Next, while the communicated process data is lightweight, with just a few bytes, updates on such measurements are frequently required, especially in real-time applications. Consequently, round trip times can be below one millisecond [GH13]. At the same time, these protocols should be deterministic to avoid additional jitter that can disturb process stability. This unique set of requirements differs from Information Technology (IT) systems [GH13] where rather large data (images or videos) are exchanged over long distances with less restrictive latency requirements.

While industrial protocols achieved these requirements historically with, e.g., serial communication, nowadays, these protocols are often fully ported to the Ethernet or the Internet Protocol (IP). In this dissertation, we consider these newer variants of industrial protocols such as ModbusTCP [Mod18], EtherNet/IP, or S7. An overview on various industrial protocols is provided by Galloway et al. [GH13] and we present the ModbusTCP protocol in the following as one example.

2.1.4.1 Example of an Industrial Protocol (Modbus)

Modbus is intended as a general-purpose protocol to read and write data from a PLC [Mod18]. Its core idea is that the user of Modbus can directly access the memory of a device where the values are stored. For example, a user can obtain an array of bytes from a PLC's memory. Thereby, Modbus does not have to deal with interpreting the data nor standardize different data formats. The two rudimentary data formats of Modbus are coils (a single bit) or registers (2-byte values) [Mod18].

Since Modbus stems from an era of serial communication, it leverages a request and response communication pattern in which a primary communicates with one

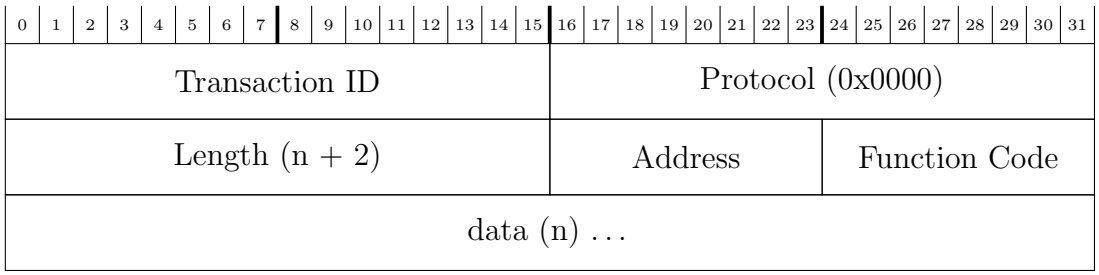


Figure 2.4 The header of the ModbusTCP protocol as defined in [Mod18].

or more secondary nodes [Mod18]. This simplifies the synchronization and reduces jitter, as would be introduced by free access to the communication medium by all participants. For example, if a HMI wants to display a new reading of the water level meter (LIT101), it first needs to know the memory address and the target PLC where this information is stored. The HMI then sends a Modbus request to the PLC, asking to retrieve the desired memory location. The PLC replies with the raw memory data and the HMI is in charge of interpreting the data. For example, converting a byte array into a float and interpreting it as a level measurement in millimeters. Modbus does not communicate how the data needs to be interpreted.

The protocol header of Modbus, as shown in Figure 2.4, comprises just eight bytes. The *transaction ID* serves as a unique identifier to match a Modbus request to its corresponding response. The *protocol* field is generally not used and is set to zero. More important is the *address* if multiple devices are connected to a single Modbus network. The *length* determines the number of bytes of the packet and is dependent on the payload (data). The last field, the *function code*, determines which action a Modbus packet triggers. For example, the function code number 1 (Read Coil Status) requests the value of a number of bits from the target device's memory [Mod18]. This can be used to, e.g., query whether a pump is running or not. The target device simply responds with the same transaction ID and function code but with the value of the requested bits as the payload.

Modbus can also be used on top of the TCP/IP protocols [Mod18]. While this might introduce additional latency or jitter, especially if the underlying network is congested, there are significant advantages. First, TCP/IP enables a flexible communication infrastructure for ICS through packet switching and high datarates compared to Modbus' serial communication. Furthermore, ModbusTCP-capable devices can become accessible from the Internet, e.g., for remote monitoring [DLP+22]. However, this has to be implemented carefully so as not to introduce vulnerabilities to the ICS, as ModbusTCP would expose the memory of a PLC to the public Internet without providing security features such as authentication (cf. Section 1.2.1).

2.1.5 Example Attack

The historical development of industrial protocols, their adaption to Internet-capable protocols, and their connectivity to the Internet results in their susceptibility to cyberattacks. In Section 1.2, we introduced which attacker model we assume in this dis-

sertation. In summary, we suspect an attacker to have access to at least the process control and supervision and control layers to disturb the physical process by manipulating the process values. This can be achieved from a position within the network by initiating illicit commands to PLCs [ACZ20]. Similarly, an attacker could manipulate the PLC's program such that the control loop behaves differently [ACZ20]. Nonetheless, the program in a PLC assumes that the measured sensor values and actuator signals are received correctly. But, an attacker can violate this assumption and thus change the control loop to their intention (cf. attacker in Figure 2.2).

For example, if the attacker manipulates the water level such that it shows a constant low value, the infill valve (MV101) will remain open. Then, the water level of the tank will slowly increase until it overflows. This might look like a failure of the level meter LIT101 at first sight rather than a cyberattack. Conversely, an attacker can also modify the actuator signals, e.g., manually closing the infill valve (MV101). At the same time, the attacker can pretend that the water level remains constant despite the water tank emptying. Thereby, the attacker could prevent an alarm from being raised, notifying the operators if the water tank reaches a critically low level.

IIDSs aim to detect such attacks before harmful damage can occur, e.g., before the tank overflows. In the following, we provide an introduction to intrusion detection.

2.2 Basics of Intrusion Detection

Since critical infrastructure and other ICSs face an increasing number of cyberattacks (cf. Section 1.2.1), adequate security measures are urgently required. As laid out in Section 1.2, we consider intrusion detection as the methodology for their protection in this dissertation as they promise to be a viable and retrofittable protection mechanism in ICSs [MC14, DHX+18, GUC+18, HYL+18, LXC+21, ZWF+21, UJJM22]. They can be a second line of defense if preventive measures are breached or a standalone solution when other measures are non-existent.

We want to emphasize that this dissertation distinguishes between two terminologies of intrusion detection, as already introduced in Section 1.2.4. The general class, named Intrusion Detection Systems (IDSs) in the following, denotes all kinds of intrusion detection, also covering classical IT environments without an implicit industrial context. IDSs can not only protect ICSs, but also offices, servers, datacenters, or IT networks from attacks, often leveraging signatures of known attacks as detection methodology. Some prominent solutions are YARA [Alv08], Suricata [Sur21], Zeek [Zeek], or Snort [Roe99]. While those IDSs can nonetheless be helpful for ICS use cases, they lack essential support for ICSs' unique attacks [ACZ20, MSM+21, SMLC21], such as against specific industrial protocols, industrial hardware, or the physical process. *Industrial Intrusion Detection Systems (IIDSs)*, the second class, close this gap as they are specifically designed for industrial purposes. Not only do they consider industries' network protocols [GW13, LNTA17], but they can also take physical phenomenons into account [FPMC19, QGS+20]. Thereby, they protect the automation of industrial processes. If we talk about *IIDSs*

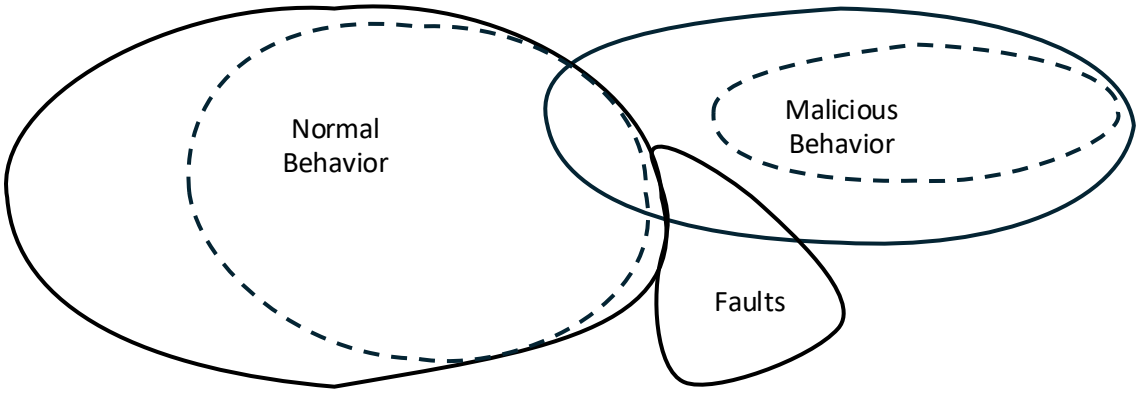


Figure 2.5 The task of intrusion detection is to separate normal from malicious behavior. But in practice, these regions might overlap, challenging the development of effective IDS solutions. Dotted lines indicate an IDS's training data, which is usually a subset of all possible behaviors.

in this dissertation, we refer to an approach specifically targeted for industries. Instead, the term *IDS* is used for intrusion detection in general, be it for traditional or industrial purposes.

To establish background knowledge on intrusion detection, the next sections introduce the philosophy behind intrusion detection in Section 2.2.1 and a mathematical formulation in Section 2.2.2. We continue with an overview of the different data types and detection methodologies to realize intrusion detection in Section 2.2.3. Afterward, we present two selected IIDS flavors in Section 2.2.4 as examples and compare them against each other to showcase how IIDSs function.

2.2.1 Philosophy of Intrusion Detection

The goal of intrusion detection is to indicate the presence of a cyberattack (with an alert) as early as possible to start mitigating the threat instead of it remaining unnoticed [Wan09, SB21]. The main intuition behind IDSs is that a cyberattack leads to unique system behavior that can be distinguished from legitimate behavior. Figure 2.5 visualizes the different cases that are relevant for intrusion detection.

If a system behaves as intended, we call this the *normal/benign behavior*. On the contrary, two important other cases need to be considered. The first case are *faults* where the system does not behave as intended [KL14]. Faults can be caused by equipment failure such as wear or tear over time. An ICS is usually already designed to detect and avoid faults and recover from them [HKKS10]. In contrast, malicious behavior is initiated by cyberattacks seeking to harm a system intentionally.

But, precisely defining these areas can be difficult. First, industrial processes and especially the (physically) measured values are prone to noise [AMO20]. This uncertainty makes the borders blurrier than anticipated in Figure 2.5. Second, there are areas of overlap, as it is not always clear whether a certain action should be considered benign or malicious, since this depends on external circumstances. For example, reprogramming a PLC's code can be normal behavior to optimize the con-

trol flow of an ICS but likewise be the cause of malware. For intrusion detection, this area of overlap is optimally as small as possible.

Another overlap occurs between faults and malicious behavior since a cyberattack can introduce the same effects as a fault [KL14]. For example, whether the measurement of a sensor yields false data due to a cyberattack or a fault is not necessarily directly apparent. While industries usually implement dedicated methods for Fault Detection, Isolation, and Recovery (FDIR) [HKKS10], an IDS could also indicate these situations [ANR17, PTS18, HL21]. Still, ICS operators are not necessarily averse to an IIDS that also recognizes faults. Our research focuses on malicious behavior induced intentionally by attackers and ignores fault detection.

Regarding intrusion detection, we consider a binary classification problem distinguishing between normal and malicious behavior. Since the use cases IDSs get deployed for can vary (cf. Section 1.1.1), an IDS has to undergo a training phase to map out the regions of benign and malicious behavior. Depending on the chosen detection methodology, samples of one or both regions are provided for training (cf. dashed circles in Figure 2.5). We presume that the data for training is labeled in the following, which can be accomplished for datasets, but be more challenging in practice. Also note that the training data does not have to cover all conditions that are to be expected (the dashed circles can be smaller than the entire area they represent). For example, recording data for all normal (edge-)cases of an ICS is hardly possible. Consequently, the detection methodology has to abstract from a smaller subset of training data to provide predictions even on unseen data.

These overlapping relationships pose a challenge to intrusion detection, fueling the research community to develop effective detection methodologies.

2.2.2 Notation and Formalism

Besides this informal introduction to the process of intrusion detection, we now formalize the procedures of training, evaluating, and configuring an IDS mathematically. The following notations are largely based on the work by Probst et al. [PBB19] from the machine learning domain and are adapted to our terminology.

Datasets. To measure an IDS's performance, research can refer to (public) evaluation datasets (cf. Section 2.3), which usually ship with or are split into dedicated parts for training (D_{train}) and testing (D_{test}). For simplicity of the definitions, we refer to the evaluation dataset in the following. A dataset contains a collection of features and their values X of some dimension k, e.g., representing protocol headers of network traffic. This dataset serves as the input for the IDS and is associated with labels Y, which indicate the expected outcome, e.g., benign 0 or malicious 1.

$$D = (X, Y) \subseteq \mathbb{R}^k \times \{0, 1\} \tag{2.1}$$

Hyperparameter. Prior to training an IDS, the underlying detection methodology may require configuration of its algorithm through a set of l hyperparameters, e.g.,

to determine some kind of detection threshold. To apply an IDS, one configuration θ has to be picked from the entire hyperparameter space Θ.

$$\theta = (\theta_1, \ldots, \theta_l)^\intercal \in \Theta \subseteq \mathbb{R}^l \tag{2.2}$$

Note that parameters may not be arbitrarily selectable as they can be restricted to specific ranges. Moreover, hyperparameters can also be ordinal or nominal, e.g., they are restricted to enabling or disabling a model's functionality.

IDS. An IDS is a function $\hat{f}$ trained with a fixed configuration θ and on a training dataset (D_{train}) to provide a verdict for an observed datapoint X, whether the ICS is under attack or not. If the IDS suspects an ongoing attack, we call this an alert.

$$\hat{f}_\theta^{D_{train}} : X \to Y \tag{2.3}$$

Note that X may not be contained in the training data (cf. dashed lines in Figure 2.5).

Metric. After the training of $\hat{f}$, an IDSs' detection performance can be measured with the help of suitable metrics, e.g., the F1 score, over a test dataset (D_{test}). Since plenty of metrics with distinct objectives exist (cf. Section 2.4), we define them as generic function m mapping to $\mathbb{R}$.

$$m : (D_{test}, \hat{f}) \to \mathbb{R} \tag{2.4}$$

Note that many metrics, but not all of them, are restricted to the interval $[0 - 1]$, where higher scores usually indicate a better-performing IDS.

2.2.3 Directions of Intrusion Detection Methodologies

Given the theoretical foundation of intrusion detection, we now introduce possible directions for the realization of an IDS. In general, an IDS depends on the *data type* (Section 2.2.3.1) on which it operates and the underlying *detection methodology* (Section 2.2.3.2) [MC14]. In the following, we introduce possible variants one by one, as shown in Table 2.1. All combinations between data types and detection methodology are possible in practice. Note that there may be more variants than listed in the following or even hybrid approaches [MC14], but those presented now resemble the important cases for this dissertation. We will conduct a more detailed survey on directions of IIDS research in Chapter 3. The cases shown here, serve as background knowledge for the reader.

2.2.3.1 Data Types

The first axis concerns the input data type, which is important for designing an IDS, as traces of an attack should be observable in the data. For example, while a reconnaissance attack on network services might be obvious to spot in network traffic, it becomes harder on process data. In the following, we distinguish between approaches considering network traffic, host data, and physical process data.

		Data Type		
		Host	**Network**	**Process**
Detection Technique	**Anomaly Detection**	Measure the execution time of a PLC cycle and alert deviations, e.g., if the program was changed [FB20].	Analyze the Inter-arrival Time (IaT) of Modbus packets to unveil insertion or flooding attacks [LNTA17].	Learn invariants that must be fulfilled, e.g., if a valve is open then there must be a flow of water [FPMC19].
	Misuse Detection	Detect control-logic injection attacks to a PLC by monitoring its memory [YKSA19].	Detect the usage of a legitimate or malicious Modbus function codes with an decision tree [PASE18].	Train a Support Vector Machine (SVM) with samples of benign and malicious process values [SMRM20].

Table 2.1 An IIDS can be distinguished by its detection method and the data type it operates on. Above, we provide an example of an IIDS for each of the six possible combinations.

Host data. Host data is one traditional class for IDSs from the IT domain [MC14]. Host data concerns actions that take place on a dedicated device, e.g., by inspecting access logs or modifications to files to unveil ransomware [vWW+25]. But, host data can also be found in ICSs to investigate whether industrial equipment behaves as intended. One example would be the IIDS by Frombey et al. [FB20], which determines whether the code of a PLC is executed as intended (cf. Table 2.1). Yet, host-based IIDSs are challenging to realize in ICS environments because the industrial devices have to remain real-time capable and are often restricted in computational and memory resources or have limited programming capabilities.

Network data. Looking at the industrial network can provide deep insights into an ICS as its traffic is often not encrypted [DLP+22]. Note that future ICS installations may incorporate encryption challenging network-based IIDSs. IIDSs operating on network traffic can thus be installed flexibly and passively within the network and do not interfere with the existing installation. By analyzing the network flows or performing deep-packet inspection [LNTA17, PASE18], such IDSs can reveal malicious connections, packets, or a Denial of Service (DoS) attack. In the case of ICSs, IIDSs that are specialized to industrial protocols can even take into account process knowledge. For example, the cycle of a PLC constitutes a repetitive network pattern of first reading the required sensor data and then issuing commands to the actuators [GW13]. Deviations from these patterns can indicate an attack.

Process data. Besides the previous two groups that are also common in IT systems [Roe99, Zeek, vWW+25], this third category is unique for the ICS use case. As ICSs are primarily concerned with the automation of physical processes, an IIDS could likewise monitor the physical process data to protect it regardless of the attack vector previously selected by the attackers. Such IIDSs monitor the physical state of an ICS to detect abnormal behavior, e.g., the upcoming overflowing of a water tank (cf. Section 2.1.5). The process data can either be provided in aggregated form from the manufacturing operations and control level or gathered from the observed network traffic between the PLCs on layer two (cf. Figure 2.3). Since this class contains aggregated physical data of the ICS and abstracts from other typical network traffic or host data features, it resembles a data type on its own.

2.2.3.2 Detection Techniques

Besides the data types that describe which features an IDS operates on, there are also different ways how to realize the detection mechanism. Consequently, the second axis, *detection technique*, can be broadly distinguished into misuse and anomaly detection [LDSR05] (cf. also Section 1.3). This difference shows in how the detection techniques use the data for their model generation. As shown in Figure 2.5, an IDS could require samples of normal behavior to understand how a system behaves, while other IDSs (additionally) require samples of attacks. The precise composition of the training data mainly depends on the type of the detection technique.

Anomaly detection. The first class implements anomaly detection using training data of only benign conditions [MC14]. Such an IDS can infer which behavior is normally intended and then alert all deviations as anomalies and suspected attacks. This methodology can be practical in scenarios where attack data is rare or difficult to obtain, as is the case in ICS [BSL+23]. In contrast, benign data can often be recorded simply during the normal ICS operation. This enables anomaly detection to detect even zero-day attacks that were unknown during the time of the ICS design [SB21]. Therefore, such IDSs could work well in an ICS context where attacks are usually specifically crafted and targeted for individual facilities and where benign behavior is usually well specified. Nonetheless, this method relies on a good set of training data, as any normal behavior not contained could inevitably be classified as an attack. Anomaly detection is realized with specialized One-Class Classifiers (OCCs), which have many opportunities for their realization in ICS due to their repetitive and predictable network [GW13, LNT18] and process patterns [GUC+18].

Misuse detection. In contrast, misuse-based IDSs additionally expect malicious behavior samples during training [MC14]. From these attack samples, they can infer what an attack would look like and aim to find similar attacks. Thereby, these systems can better distinguish between actual attacks and benign behavior. On the downside, misuse detection may struggle to detect unseen attacks that have not been observed in training [KWP+22a]. Moreover, obtaining attack samples can be complicated in certain deployments [SP10, AMM20]. For example, intentionally attacking ICSs can be complicated or even dangerous. This category often makes use of established supervised machine learning algorithms such as Random Forest (RF) or Support Vector Machine (SVM) that are specifically adapted for use in an ICS [PASE18]. While misuse detection is effective in IT environments with largely similar systems and vast amounts of known malware, their use is questionable in ICS due to a lack of generalization detecting only the learned attacks [KWP+22a].

2.2.4 Case-Study on Two IIDS Examples

Having laid out the motivation and scope of intrusion detection, this section presents a case-study on two IIDS proposals from literature and their detection techniques. These IIDSs represent diverging directions and were chosen for illustrative purposes providing the reader with background information on IIDS methodologies. The first IIDS analyzes network traffic with a misuse detection model (Section 2.2.4.1). The

second IIDS implements anomaly detection on process data (Section 2.2.4.2). In the end, we highlight the differences between these directions (Section 2.2.4.3).

2.2.4.1 An Example for Misuse-based Detection on Network Data

The first case-study concerns a misuse detection-based IIDS leveraging network data. Here, we consider a minimalistic ICS scenario where a PLC controls a sensor and an actuator with ModbusTCP. The PLC requests a new measurement from the sensor and, upon reception of the response, commands the actuator to change its setpoint appropriately. In this scenario, an attacker can ingest malformed packets to disrupt the actuator since ModbusTCP is not authenticated by default [DLP+22]. The goal of a network IIDS is to analyze the packets and indicate those that seem suspicious.

For example, two IIDSs implementing this direction are compared against each other by Perez et al. [PASE18]. They leverage supervised machine learning models, i.e., RFs or a SVM, to classify observed network packets. To this end, several features are extracted from each ModbusTCP packet, such as the source, destination, packet length, or setpoint values. Given a labeled training dataset with benign *and* malicious network packets, the RF can learn attack patterns by deriving suitable decision trees. Finally, the trained model is used for live detection and can indicate on a per-packet basis whether a packet is part of an attack or is benign.

2.2.4.2 An Example for Anomaly Detection on Process Data

The second case-study concerns an IIDS realizing anomaly detection and operates on a (system-wide) view of ICS's process data. As an ICS scenario, we refer to our example water tank (cf. Section 2.1.1). An attack against this system could result in a state where the inlet valve is open even though the maximum water threshold is reached leading to an overflow (cf. Section 2.1.5). The goal of a process-based IIDS is to indicate these unwanted states.

Feng et al. [FPMC19] present one exemplary approach tackling this problem. Their idea is to find invariants that must be fulfilled all the time. For the example scenario, we consider an equation stating that if the inlet valve is open, then we expect the water level to be below the maximum ($\texttt{MV101}\,\text{open} \Rightarrow \texttt{LIT101} < max$). Since this rule is always satisfied during normal behavior of the ICS, any violation resembles an unwanted situation. Given a large training set of *benign* ICS behavior, this IIDS can mine universally valid invariants and thus detect deviations without having to rely on sample attacks during training.

2.2.4.3 A Comparison Between Different IIDS Methodologies

As can be seen from the two showcased IIDSs, intrusion detection can take vastly different forms in the ICS domain. First, the considered network traffic-based approach detects attacks on a per-packet basis, e.g., with deep-packet inspection, while the process-based IIDS leverages a broader view of the ICS by analyzing the entire

physical process data. Consequently, both directions can detect widely different cyberattacks: network traffic-based IIDSs can indicate, among others, Machine-in-the-Middle (MitM), DoS attacks, or malicious access to devices while process-based IIDSs detect the consequence of an attack on the physical process irrespective of the prior attack vector. This difference is influenced by the data type an IIDS uses.

Independent of the data type, the underlying detection methodologies either train *only* on benign samples to derive a normality model and indicate deviations (anomaly detection, cf. Section 2.2.4.2) or learn a combination of benign *and* attack samples (misuse detection, cf. Section 2.2.4.1). In that regard, one disadvantage of misuse-based IIDSs is that they demand the ICS to be under attack for training, which may be difficult to record especially in safety-critical ICS deployments [BSL+23].

Note that IIDSs can also fall in between of these dimensions [UJJM22], i.e., making use of misuse- and behavior-based detection methodologies. This shows that not all IIDSs strictly follow the classification presented here and we refer the reader to other surveys which highlight the differences in greater depth [MC14, LXC+21, UJJM22].

Takeaway. There exist plenty of methodologies to realize intrusion detection for ICS scenarios. Thus, it is an interesting topic for research to develop novel approaches and compare the feasibility of existing and new IIDSs.

2.3 Datasets for IIDS Performance Evaluations

Researchers inventing a new IIDS want to measure its detection performance and compare it to related work. For this purpose, they can leverage testbeds or benchmarking datasets [CDT21]. Datasets are also common in other fields of intrusion detection, such as detecting Distributed Denial of Service (DDoS) attacks [SLHG19]. Yet, datasets must be designed carefully to yield results that represent actual use-cases [SGLG17]. Another evaluation methodology is testbeds [CDT21], which can be difficult to establish or access due to costly hardware or extensive setups.

This dissertation makes use of *datasets* recorded in industrial contexts, i.e., derived from ICS testbeds. Since most of the leveraged datasets in this dissertation are recorded on physical testbeds with actual hardware [MTT15, GAJM16, APM17, SLYK20], they provide great realism compared to simulation-based datasets. Also, this dissertation considers network and process datasets.

Regardless of the dataset type, a dataset must contain samples of benign system behavior and attacks (cf. Section 2.2.1). To this end, datasets are labeled such that the samples affected by attacks are known for the training or evaluation of an IIDS.

There exist great overviews on testbeds and datasets such as the one by Conti et al. [CDT21]. For this dissertation, we consider a selection of qualitative and regularly used datasets. Next, we provide a brief overview on the datasets used in this dissertation in Section 2.3.1. Afterward, Section 2.3.2 exemplarily presents one dataset in detail. Finally, we discuss the differences between the dataset types and their impact on IIDS evaluations in Section 2.3.3.

2.3.1 Datasets Used in this Dissertation

This dissertation considers public datasets available on the Internet. We selected these datasets as the result of our Systematic Mapping Study (SMS) on related work, which is included in the next Chapter 3. Most of these datasets are recorded on testbeds with ICS hardware [MTT15, GAJM16, APM17, SLYK20]. For detailed descriptions, please refer to the survey by Conti et al. [CDT21] or the respective publications. In the following, we present the five relevant ones for this work.

2.3.1.1 SWaT Dataset

The SWaT dataset [GAJM16] is based on the Secure Water Treatment testbed by iTrust in Singapore. This testbed represents a scaled-down physical testbed of a real water treatment plant containing six stages: a water intake to buffer water for further processing, initial chemical assessment and dosing of the water followed by filtration, dechlorination, inorganic purification, and a final disposal step. The first stage is identical to the example ICS shown in Figure 2.1. The entire underlying physical process involves 51 different sensors and actuators.

A PLC is associated to each stage of the testbed, which controls not only a single stage but also manages the synchronization between the stages. These PLCs are connected via an industrial network with each other, leveraging the EtherNet/IP protocol. Additionally, the whole process is monitored by a Supervisory Control and Data Acquisition (SCADA) system. It obtains a snapshot of the process values in one-second intervals.

The SWaT dataset itself was recorded on this testbed and multiple versions of it exist. This dissertation uses version A1&A2 from December 2015. During the dataset recording, the testbed ran for eleven days in total, of which the first seven contain normal operating behavior, and the latter four include the attacks for IIDS testing. In total, 36 different cyberattacks were launched as MitM attacker between any stage's PLC and the SCADA system. In the SWaT dataset, attacks solely affect physical modifications of the system, e.g., shutting down a pump, rather than network attacks such as scanning. Note that each attack of SWaT was recorded only once and is not repeated. Moreover, a single attack can last up to several hours.

While both network and process data were captured, we only leveraged the process data. The final pre-processed dataset is a collection of the physical states encompassing all sensor and actuator readings every second. Attacks within the datasets are labeled accordingly. The dataset is split into two files: one for training the IIDS on data without attacks encompassing 495.001 datapoints and a second file containing attacks with 449.919 datapoints.

2.3.1.2 WADI Dataset

The WADI dataset [APM17], also from iTrust, represents another testbed of a water domain meant to distribute filtered water across a distribution network, e.g., for a

city. The first two process stages of WADI simulate a primary and a secondary distribution grid, whereas the third stage manages the wastewater collection. While having fewer stages than SWaT, WADI contains 127 different physical measurement points. Again, the system is steered with a network of connected PLCs.

The recording of the WADI dataset leverages this testbed and was conducted over 16 days, of which 14 days serve as training data without attacks. On the remaining two days, 14 attacks were performed. In this dissertation, we use the WADI.A2 version of the dataset from November 2019. It comes with two files. First, a training file representing the 14 days of normal behavior with 784.571 datapoints in one-second intervals. The second testing file contains the attacks and is 172.801 datapoints long.

2.3.1.3 BATADAL Dataset

The last dataset of the iTrust family was developed for a competition to design detection algorithms called the BATtle of the Attack Detection ALgorithms [TGT+18]. The BATADAL dataset recorded for this competition also models a water distribution process similar to WADI, yet on a larger scale. It was generated with a simulation in Matlab based on a real-world water distribution network called C-Town, consisting of water storage tanks, pumps, valves, and many pipes to consumers. These assets are represented with 43 physical values contained in the dataset.

In contrast to the datasets before, BATADAL sampled the data in hourly intervals. Consequently, the training file with 8761 datapoints resembles about one year of the process. For the evaluations in this dissertation, we combine the two test files containing the attack samples, resulting in 6266 datapoints and 14 different attacks.

2.3.1.4 HAI Dataset

In contrast to the previous three datasets focusing on a single domain, the HAI dataset [SLYK20] resembles a multi-domain process. The first of the four processes is a boiler process meant to heat water and deliver it via a heat exchanger to a turbine. The turbine models the second process with a rotating motor. Additionally, a water treatment process models a water pump storage system with the help of two water tanks. All three physical components are tied together in a simulator modeling a power grid driven by either a steam turbine from process one and two or a hydro-water plant from process three. Thereby, the HAI testbed represents a more complex ICS than if these processes were considered in isolation.

The dataset recorded on this testbed consists of 79 process values measured in one-second intervals as with the SWaT and WADI datasets. We use the HAI dataset version 21.03, which contains three files to train an IIDS and five test files. In this version of the dataset, 50 different attacks are contained. Some of them target only a single physical value, while others are even launching multiple attacks at the same time. For training, we leverage the first training file with 216.001 entries, and for evaluation, we concatenate all five test files totaling 402.005 datapoints.

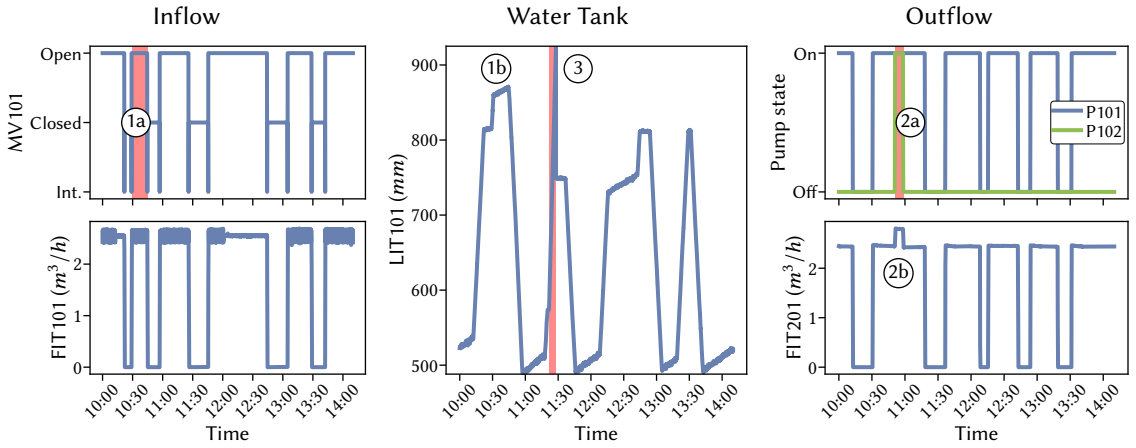


Figure 2.6 Extending the example from Figure 2.1, this figure shows a snippet of the SWaT dataset for the first process stage of the SWaT testbed, i.e., the water tank. The values of the process are depicted over time and red highlights show the areas where an attack took place.

2.3.1.5 Morris-Gas Dataset

In contrast to the previous datasets, this dataset resembles a recording of network traffic. Morris-Gas [MTT15] represents a testbed of a miniature gas-pipeline consisting of a pump, a solenoid pressure release valve, and a pressure sensor, which should maintain a constant pressure in the system. To this end, these devices are connected with Modbus to a central PLC, which implements the monitoring and control logic. In addition, a HMI serves as a display to monitor the process. The authors operated the system manually or in automatic mode during data capture.

Then, the authors implemented 35 attacks against the Modbus protocol, divided into seven categories that target the communication, e.g., via DoS, or the process itself by changing the target pressure value. The final dataset consists of 214.580 Modbus packets, of which 60.048 resemble randomly chosen attacks. Note that attacks are executed several times, sometimes even with slight variations in parameterization.

In contrast to the datasets before, each datapoint resembles a single Modbus packet modeled by generic Modbus features, such as a packet's function ID, and process-level information, e.g., the pressure measurements included in the payload, among others. Since no Modbus packet contains all process-data, datapoints contain missing values. Most importantly, only a single file exists and all packets are labeled individually for either IIDS training or evaluation. Thus, this dataset lacks a dedicated part with benign behavior, as was the case for all four previous datasets.

2.3.2 A Look Into the SWaT Dataset

Having introduced the relevant datasets of this dissertation on a high-level, we now present one of these datasets in more detail to understand its contents. To this end, Figure 2.6 depicts a snippet of the SWaT dataset. We limit the scope of the presented process values to those relevant to the first process, as explained in Figure 2.1.

The three columns of Figure 2.6 are arranged in such a way that they match the different parts of the process. The first column represents process values associated with the inflow to the water tank. The motorized valve (MV101) can be either opened, closed, or in an intermediate state. If it is opened, then the water tank gets filled with new water and the inflow rate into the tank is measured by a flow sensor (FIT101). This is also visible within the data. For example, when MV101 is set to the open position, the flow meter indicates a water flow of about $2.5m^3/h$. Moving on to the water tank shown in the middle column, its water levels increase when MV101 valve is opened. The fill level of the tank is measured in millimeters. Lastly, the third column depicts the outflow from the tank controlled by two pumps P101 and P102. If these pumps are running, this affects the water level and is visible in a positive flow measured by the second flow meter FIT201.

The example shown in Figure 2.6 contains three attacks, which are indicated with red highlights. The first attack against MV101 forces the valve to the open state, cf. 1a. Thereby, more water enters the water tank than planned for, cf. 1b. Under normal conditions, the maximum fill level of the water tank is only about $800mm$. Looking closely, we observe that the attack's effect persists even if the attack's label has ended. In this case, the water level in the tank remains too high for a while until it decreases to a normal level of about $800mm$ again. This goes to show that physical effects may persist even after an attack has ended.

The second attack, shown on the right, targets the backup outflow pump P102 by turning it on in addition to the main pump P101, cf. 2a. Consequently, the FIT201 sensor measures a spike in the outflow rate, cf. 2b. The third attack visible in this example targets the LIT101 sensor by artificially increasing its measurement, cf. 3.

These attacks already show that attacks in a physical system are not necessarily constrained to the place where the attack takes place. They rather propagate through the industrial process and can also harm other places. This might be helpful for intrusion detection as it increases the datapoints where anomalies can be observed. At the same time, as visible from the first and second attacks, an attack might need some time until it is dangerous. For example, the inflow valve MV101 needs to stay open for a certain time until the tank finally overflows. This gives an IIDS time to detect such attacks and ICS operators time to mitigate them.

After these attacks at around 11:30, the process continued to function as intended. Here, we can observe two important things for intrusion detection in ICS. First, the patterns during benign behavior are repetitive, such as the valves or pumps. Second, all values influence each other in a predictable way through the physical process behind it. Thus, repetitiveness and predictability are two important features of ICSs that provide great opportunities for intrusion detection.

Note that since first process of SWaT is confined to six process values, this part is easy to comprehend. Yet, other industrial processes, e.g., chemical processing [BRJ15], can become more difficult, especially if they do not behave this linearly anymore, as shown here. One example of a more complex dataset is the multi-domain dataset HAI also used in this dissertation (cf. Section 2.3.1.4).

2.3.3 A Comparison Between Different Datasets

When comparing the datasets' properties, it becomes apparent that they model different ICSs scenarios and attacks. Thus, they differ with respect to the scale, the domain of the testbed, and the provided data type.

Besides that, the composition of training and testing data is an equally important evaluation factor. In that regard, WADI's and BATADAL's low number of attacks without repetitions render them nearly useless to analyze an IIDS' capability to detect one precise attack class, whereas Morris-Gas repeatedly executed the same attacks. Conversely, Morris-Gas does not provide a dedicated set of benign-only data, which is necessary for behavior-based approaches.

Another factor affecting evaluations is the duration of an attack. For example, as a consequence of SWaT's large testbed and slow physical process cycle, we could observe that it takes some time after a cyberattack was initiated until effects become visible. These effects can exceed the actual attack until the process stabilizes again, making determining the range of attacks difficult for evaluations. Turin et al. [TETC20] also made this observation when analyzing different datasets.

In contrast, the Morris-Gas dataset expects the IIDS to precisely indicate each faulty network packet right at the time of observation. Thus, the datasets' labels differ strongly in interpretation. While for the Morris-Gas dataset, it is crucial to detect any network packet, for SWaT, one can argue that detecting an attack sometime during its execution suffices. Such effects strongly affect the interpretation of evaluation metrics, which we introduce in the following section. Overall, research has access to diverse datasets to develop and assess the feasibility of their IIDSs.

2.4 Evaluation Metrics for Detection Performance

Evaluating the performance of an IDS is of utmost importance to prove its effectiveness and compare it quantitatively against related works. Thus, metrics are essential to achieve objective comparisons among publications in research. In this section, we are concerned with the *detection* performance, which is usually expressed through a set of different metrics in intrusion detection [Pow11]. Detection performance metrics quantify the capabilities of an IDS, i.e., its ability to differentiate benign from malicious behavior (cf. Figure 2.5). Note that other aspects of an IDS, such as computational performance or comprehensibility, are further important aspects to measure yet not part of this section.

Scientific evaluations of IDSs are based on labeled benchmarking datasets (cf. Section 2.3), including samples of malicious behavior (cyberattacks) and benign behavior. After a training phase, e.g., to build a model of normal behavior or learn attack patterns, for each datapoint in the dataset, the known labels are compared to the output of the IDS (alarm or no alarm). The high-level goal of an IDS is to detect as many attack instances as possible while emitting few (false) alarms for benign behavior. Especially in ICSs, where cyberattacks are rare compared to benign behavior, false alarms should be minimal [Eta17, AAM22]. The detection performance

	Metric	Definition (Section)	Conf. Matr. TP	TN	FP	FN	Synonyms
Point-based	TPR	2.4.1.2	✓			✓	Recall, Sensitivity, Hit-Rate
	FNR	2.4.1.3	✓			✓	Miss-Rate
	TNR	2.4.1.4		✓	✓		Specificity, Slectivity
	FPR	2.4.1.5		✓	✓		Fall-out
	PPV	2.4.1.6	✓		✓		Precision, Confidence
	NPV	2.4.1.7		✓		✓	–
	Accuracy	2.4.1.8	✓	✓	✓	✓	Rand Index
	F1	2.4.1.9	✓		✓	✓	–
Time-aware	Detected Scenarios	2.4.2.1	✓				Scen.
	TPA	2.4.2.2	✓				–
	Detection Delay	2.4.2.3	✓			✓	–
	FPA	2.4.2.4			✓		–
	(e)Ta	2.4.2.5	✓	✓	✓	✓	eTaP, eTaR, eTaF1
	Affiliation	2.4.2.6	✓	✓	✓	✓	–

Table 2.2 Overview of the metrics used in this dissertation. We distinguish between point-based and time-aware metrics. Note that metrics may occur under different synonyms.

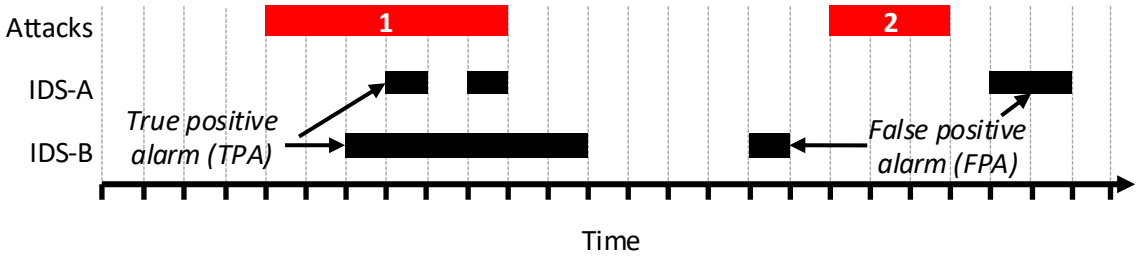


Figure 2.7 The metrics for detection performance compare the alerts of an IDSs (black) to the groundtruth attacks (red labels) from the benchmarking dataset.

of an IIDS can be expressed through a variety of metrics. To provide an overview, we present relevant and newer ones observed in related work in Table 2.2. A similar overview has already been introduced by Kim et al. [KCC+22].

Figure 2.7 provides two example alerts of IDSs. The red regions labeled with "1" and "2" mark the time during which the attacks took place. Below, we depict two fictional IDSs and their alerts in black. Judging which IDS is better is the task of different metrics. For example, IDS-A detects attack "1" later than IDS-B, but the alert of IDS-B stretches longer than the actual attack. Both IIDSs have two False-Positive Alarm (FPA). Yet, depending on the precise scenario and time-scale, the FPA of IDS-A might also be a late alarm resulting from attack "2". Table 2.3 contains the metrics used in this dissertation for the example shown in Figure 2.7.

Given the pool of metrics IDS authors can choose from, we provide a systematic overview of the different flavors. The following two sections present details about the most important metrics, covering well-established point-based metrics (Section 2.4.1) and promising time-aware metrics (Section 2.4.2). Please refer to Table 2.2 for synonyms by which these metrics are also known. Lastly, in Section 2.4.3, we look at two IIDSs from the literature and how they evaluated their approaches.

	TP	TN	FP	FN	Acc.	Prec.	Rec.	F1	eTaP	eTaR	eTaF1	Scen.	TPA	FPA
IDS-A	2	14	2	7	64.0	50.0	22.2	30.8	58.6	33.3	42.5	50.0	2	1
IDS-B	4	13	3	5	68.0	57.1	44.4	50.0	59.2	41.7	48.9	50.0	1	1

Table 2.3 The example of the two IDS alerts given in Figure 2.7 results in the following numbers for the metrics. Note that we display the metrics in percent points.

2.4.1 Point-based Metrics

The "traditional" way to evaluate IDSs is to utilize a benchmarking dataset, which includes a label for each entry in the dataset. The labels indicate whether an entry is benign or malicious, i.e., it corresponds to a (specific) cyberattack. Given the labels of a dataset and the outputs/alerts of an IDS, one can compare them point-to-point to estimate how often the output is correct (T) and how often it is false (F), i.e., deviating from the expected label. Various metrics then derive performance scores with different meanings, usually in the interval of $[0-1]$. We present those numbers as percent points in this dissertation, as shown in Table 2.3. For a detailed discussion on each metric, please refer to [Pow11, AQP+22].

2.4.1.1 Confusion Matrix

As the first performance indicators, one can count the occurrences of all four possible combinations between dataset labels and IDS outcomes: First, true negatives (TN) are all instances where the dataset label is benign, and the IIDS has not raised an alarm. Likewise, true positives (TP) correspond to those attack instances within the dataset that are correctly identified as attacks. In contrast, false negatives (FN) are attack instances that the IIDS does not detect. Lastly, false positives (FP) are false alarms triggered even though no attack has occurred. Counting all of these four possible outcomes across a dataset yields the confusion matrix, laying the foundation for many point-based metrics introduced in the following.

Yet these absolute numbers are difficult to compare if, e.g., the datasets vary in size. Since there is a desire to express performance with a single value irrespective of the dataset, there exist a large variety of *point-based* metrics derived from the confusion matrix [Pow11] (cf. Table 2.2). These express properties, such as its overall correctness (accuracy), the fraction of correct alarms (precision), or fraction of identified attacks (recall). Point-based metrics find wide application beyond IIDS research, e.g., machine learning, and thus are a natural choice for comparisons.

2.4.1.2 Recall / True Positive Rate (TPR)

This metric states how many attacks of the dataset are detected by an IIDS. Naturally, an IIDS has to detect as many attack instances of a dataset as possible.

$$\frac{TP}{TP+FN}$$

2.4.1.3 Miss-Rate / False Negative Rate (FNR)

FNR measures the fraction of missed attacks. Hence, a lower score is preferred.

$$\frac{FN}{TP + FN}$$

2.4.1.4 Specificity / True Negative Rate (TNR)

Since cyberattacks are rare, it is crucial that an IIDS does not trigger alarms during benign system behavior. Thus, TNR defines the fraction of correctly classified benign behavior. A high TNR score is preferential.

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

2.4.1.5 Fall-out / False Positive Rate (FPR)

IIDSs should only trigger an alarm in case of actual attacks. Therefore, FPR calculates the fraction of false alarms, which has to be as low as possible.

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

2.4.1.6 Precision / Positive Prediction Value (PPV)

When focusing on the alarms triggered by an IIDS, PPV defines the fraction of correctly detected attacks among all existing attacks in the dataset.

$$\frac{TP}{TP + FP}$$

2.4.1.7 Negative Prediction Value (NPV)

Contradicting the PPV metric, NPV counts the number of correctly classified negative predictions among all attack-free parts in the dataset.

$$\frac{TN}{TN + FN}$$

2.4.1.8 Accuracy

The first metric capturing the overall number of correct classifications is accuracy. The higher the accuracy score is, the more reliable the predictions of the IIDS are.

$$\frac{TP + TN}{TP + TN + FP + FN}$$

2.4.1.9 F1 score

There is an inherent tradeoff between achieving a maximal number of detected attacks (TPR) while reducing false positives (expressed by PPV as correct alarms). The F1 score combines both design goals into a single metric through the harmonic mean. Note that F scores with different TPR and PPV weightings exist [Pow11].

$$\frac{2TP}{2TP + FP + FN} = \frac{2Precision \cdot Recall}{Precision + Recall}$$

2.4.2 Time-aware Metrics

Point-based metrics are suitable when the benchmarking datasets' entries are independent. However, ICSs are inherently time-dependent since the current state of an ICS is always a result of the system's previous state (cf. Figure 2.2). Consequently, ICS datasets extracted from such systems also need to be considered in the aspect of time. For example, an alarm extending beyond an attack since the ICS has not yet recovered from the attack should be interpreted differently from a false alarm in the middle of normal behavior (cf. the first two alert of IDS-B in Figure 2.7). Furthermore, since point-based metrics consider every dataset instance independently, a long attack (cf. attack "*1*" in Figure 2.7) is weighted more than shorter ones. In such or similar scenarios, point-based metrics are skewed, which is already known in literature since 2014 by Gensler et al. [GS14].

Consequently, novel *time-aware* metrics tackle these flaws [GS14, LA15, TLZ+18, HYKK19, HNR22, HYKM22]. They usually define attacks and alarms as a continuous time range with start and end points. Alarm intervals should largely overlap with the attacks, i.e., an alarm is optimally expected immediately after the start of an attack and should stop in time after the attack phase.

2.4.2.1 Detected Scenarios (Scen.)

Unlike point-based metrics, time-aware metrics consider attacks as single instances, and thus, they do not necessarily care about when the attack is detected, how often it was indicated, or how large the coverage is. Therefore, detected scenarios states the fraction of independent attack instances detected by at least a single alarm. Given the example in Figure 2.7, both IDSs detect the first but not the last attack, and thus both score 0.5 in Scen. as they detect half of the attacks.

2.4.2.2 True Positive Alarms (TPA)

Another time-aware property of alerts is that an alert can also be considered as a single instance, changing the definition of a true positive. Since the beginning of an alert indicates the moment when the difference to normal behavior is sufficiently large to suspect an attack, this part can be more important than when the alert ends. Especially since the effect of attacks can continue even if the attack has ended, e.g.,

when the system has to stabilize (cf. Section 2.3.2). Thus, another time-aware metric is to consider alerts as one instance and count them as True-Positive Alarm (TPA) if the entire duration of the alert overlaps with at least one cyberattack. In the example from Figure 2.7, IDS-A has two TPA and IDS-B only one TPA.

2.4.2.3 Detection Delay

Nonetheless, for true alerts, early detection is still preferential in time-critical scenarios as this increases the time to respond to an attack. Thus, for all attacks in the dataset, the detection delay aggregates the time intervals between the start of an attack and the time of the first detection.

2.4.2.4 False Positive Alarms (FPA)

For similar reasons as described for the TPA metric, it can be misleading to count a continuous false alarm as multiple instances. For example, comparing a false alert that is ten seconds long with one that is just one second long, both still have to be investigated by an analyst regardless of their length. Consequently, False-Positive Alarm (FPA) counts the number of consecutive false alarms irrespective of their length. Both examples from Figure 2.7, therefore, yield one FPA each.

2.4.2.5 Enhanced Time-aware Precision and Recall (eTaP and eTaR)

In 2022, Hwang et al. [HYKM22] proposed their (enhanced) time-aware variants for classical point-based metrics, i.e., precision, recall, and F1, addressing known issues when adopting point-based metrics to time-aware evaluations. While point-based recall weights long attacks as more important, the new time-aware recall variant (eTaR) treats all consecutive attacks equally. To replace precision, eTaP implements diminishing returns for long-lasting alarms. Lastly, the new proposed eTaF score is defined in the same way as the regular F1 score (cf. Section 2.4.1.9) but leverages the substitute eTaP and eTaR metrics.

2.4.2.6 Affiliation Metrics

A similar approach was taken by Huet et al. [HNR22] with their affiliation metrics. Again, they consider alerts and the ground truth as continuous time ranges instead of independent points. Their approach associates each alert to the closest ground truth, called local affiliation, and then calculates the individual distances for precision and recall. What makes their approach interesting is, that the final result is normalized in comparison to an IDS that emits alerts at random. Finally, the time-aware affiliation precision and recall variants are averaged into the affiliation F score.

2.4.3 Case-Study on IIDS Evaluations

Extending the case-study on the two example IIDSs from Section 2.2.4, we take a closer look at how these two publications evaluated their approaches. Later on in Chapter 3 we more deeply analyze and criticize current evaluation methodologies.

Perez et al. [PASE18] evaluate the misuse-based network IIDSs with the help of the Morris-Gas dataset [MTT15]. Since their classifiers (RF and SVM) require examples of benign *and* malicious traffic for training, they shuffle the dataset randomly and split it into three parts for training, validation, and evaluation. After determining optimal hyperparameters on the validation part, they leverage the evaluation part to examine the IIDSs' performance. They compare the classifiers' output with the labels of the dataset and calculate the accuracy, precision, recall, and F1 score. In addition to these scores, Perez et al. provide a confusion matrix of all attack types for the best classifier (RF). Note that, the methodology of randomly shuffling the dataset has the drawback that the model has likely seen samples of all attacks during training and thus risks detecting only those later on instead of detecting novel attacks [KWP+22a]. Perez et al. do not compare their results to related work.

In contrast, Feng et al. [FPMC19] evaluate their approach on two distinct datasets (SWaT and WADI). As both datasets already ship with two parts, one containing solely benign system behavior and one with attacks (cf. Section 2.3), random shuffling or splitting the dataset is not necessary here. Moreover, this assures that the behavior-based IIDS only trains on attack-free data. As evaluation metrics, Feng et al. calculate recall, fall-out, as well as the number of detected scenarios. In addition, they provide a normalized version of recall that accounts for the different lengths of attacks in the dataset. Besides analyzing the performance of their IIDS, Feng et al. also compare their results to two different approaches.

As apparent from these two examples, IIDS approaches can be evaluated quite diversely, not only with respect to the underlying dataset, as analyzed in detail in Section 2.3 before, but also in terms of evaluation methodologies. This diversity involves the experiment design and conduction, e.g., performing a train/validation/test split or not requiring those at all. Thus, simply comparing the detection performance by metrics does not necessarily tell the entire story and other aspects, such as selected datasets or training procedures, have to be considered as well.

2.5 Summary

The field of industrial intrusion detection is closely related to regular intrusion research from the IT domain, as it shares many of its main principles, such as possible detection methodologies or evaluation metrics. However, the field of industrial intrusion detection significantly differs in how the detection methodologies are realized. Moreover, since the data types and use cases are unique, industrial intrusion detection research relies on dedicated datasets and testbeds. Consequently, IIDSs have developed into an independent research branch. In the next Chapter 3, we examine the current state, the success, and the insufficiencies of IIDS research.

3

Analyzing the State of IIDS Research

As a broad research area, Industrial Intrusion Detection Systems (IIDSs) attract researchers and industrial operators from diverse backgrounds (cf. Chapter 1). It thus comes as no surprise that, according to our subsequent literature research, at least 1109 distinct authors have published ideas for detection mechanisms between 2019 and 2021 alone. While their diverse backgrounds are beneficial to cover lots of different perspectives and ideas, the resulting fast-paced advancements also risk leading to inefficiencies. In that regard, worthwhile ideas may remain hard to identify. Furthermore, it could be unclear which improvements are suitable to close the gap to much-needed production-ready IIDSs. For example, specific detection methodologies have been independently proposed and evaluated for multiple Industrial Control System (ICS) domains in the past resulting in work being made twice [CZPK15, CZK15, FCCR18] and leading to less overall research output.

In this regard, meta-analyses of IIDS research already unveiled inefficiencies in the detection capabilities of published works [ET22] or criticized the conclusions drawn from scientific evaluation procedures [AQP+22, FSL+22, KWP+22a]. Simultaneously, we observe attempts to fix these issues, e.g., by reviewing representative benchmarking datasets [CDT21], or inventing specialized industrial metrics to accurately assess the "success" of an IIDS [LA15, TLZ+18, HYKK19, KYK20, JMAC20, HNR22, HYKM22, GZS+22, KCC+22]. However, related work so far still fails to objectively assess on a broad scale and with systematic methods what the state of IIDS research is and which factors limit its efficient progress.

With this chapter, Section 3.1 first identifies the need for action along related work. We then strive to close the outlined research gap on the evolution and composition of the IIDS research landscape. To this end, we design and conduct a Systematic Mapping Study (SMS) in Section 3.2, which is a variation of a systematic literature research more suited to obtain a broad view on a topic along potentially multiple research questions [Kit07]. With this methodology, we quantify the current state

of the research landscape, encompassing 609 papers from Section 3.3 to Section 3.5. From the resulting knowledge basis, we can draw a clear picture with respect to positive and negative developments as well as persistent flaws. Ultimately, in Section 3.6, our work allows us to identify five issues persisting in IIDS research that are worth tackling in this dissertation to catalyze the joint research efforts to protect industrial networks.

3.1 Limitations of Previous IIDS Surveys

Given the promises of IIDSs to detect sophisticated and safety-critical attacks on industrial networks and processes, a plethora of different research streams for IIDSs have been established (cf. Section 1.3). Due to the tremendous research efforts on this topic, different surveys already analyzed the current state of the art. To provide an overview of the existing surveys and criticism of the methodologies or evaluation techniques, we summarize them in Table 3.1 along five categories. These categories align to the topics listed in Figure 1.2 and presented in the following.

3.1.1 Attacks

ICSs have been attacked in the past (cf. Section 1.2.1), with reports dating back until 1988 [MSM+21]. Naturally, surveys have been conducted to summarize such incidents over time [ACZ20, MSM+21]. While these works make great efforts to document and uncover what went wrong, Sun et al. [SMLC21] take a different approach. Their paper [SMLC21] provides a detailed view of Programmable Logic Controllers (PLCs) by highlighting and categorizing their vulnerabilities and potential attack vectors. Working toward protecting these devices, Sun et al. promote intrusion detection as one building block for the mitigation of attacks.

3.1.2 Datasets and Testbeds

As the second category, we take a look at the datasets and testbeds since these are crucial for the development and, later, the evaluation of new detection methodologies. Here, surveys focused on summarizing available datasets and testbeds specifically designed for IIDS evaluations [KKG19, CDT21]. These efforts identify at least 61 testbeds and 23 benchmarking datasets that are publicly available [CDT21]. Yet, since these surveys focus solely on datasets, they lack an understanding of the actual application of datasets. As a rare exception, Balla et al. [BHIM22] analyzed dataset usage for deep learning detection methodologies. They observed a strong bias toward non-ICS datasets, such as the KDD dataset family representing a collection of datasets to evaluate Intrusion Detection Systems (IDSs) mostly from the Information Technology (IT) domain [DS15], with a usage of over 50 %.

Another direction of studies concerns the quality of the datasets [TETC20, LSAA23, SK25]. Turrin et al. [TETC20] analyzed three datasets with statistical methods to

Survey	Attacks	Datasets & Testbeds	Detection Methods	Evaluations and Metrics	Meta Survey	Systematic Approach
Sun et al. [SMLC21]	●					◐
Miller et al. [MSM+21]	●					○
Alladi et al. [ACZ20]	●					○
Conti et al. [CDT21]		●				○
Kavallieratos et al. [KKG19]		●				○
Lanfer et al. [LSAA23]		●				○
Turrin et al. [TETC20]		●				○
Schuster et al. [SK25]		●				○
Balla et al. [BHIM22]			●			○
Ding et al. [DHX+18]			●			○
Giraldo et al. [GUC+18]			●			○
Hu et al. [HYL+18]			●			○
Kaouk et al. [KFPG19]			●			○
Loukas et al. [LKP+19]			●			○
Luo et al. [LXC+21]			●			○
Mitchell et al. [MC14]			●			○
Rakas et al. [RSMP20]			●			●
Ramotsoela et al. [RAMH18]			●			○
Umer et al. [UJJM22]			●			●
Zhang et al. [ZWF+21]			●			○
Christen et al. [CHK24]				●		●
Fung et al. [FSL+22]				●		○
Gensler et al. [GS14]				●		○
Kus et al. [KWP+22a]				●		○
Powers et al. [Pow11]				●		○
Ahamed et al. [AMM20]					●	○
Alahmadi et al. [AAM22]					●	○
Apruzzese et al. [ALS23]					●	◐
Arp et al. [AQP+22]					●	◐
Etalle et al. [Eta17]					●	○
Olowononi et al. [ORL20]					●	○
Raman et al. [MAM21]					●	○
Sommer et al. [SP10]					●	○
Umer et al. [UJJM22]					●	●
Urbina et al. [UGC+16]					●	○

Table 3.1 While existing surveys comprehensively tackle the diverse topics relevant for intrusion detection, they often lack a systematic approach. Thereby, the actual usages across the ICS spectrum are not captured as current studies often focus on narrower domains.

judge their quality. To this end, they compare the differences between the training and testing parts. Overall, they found that BATADAL [TGT+18] is similar between the different dataset parts. In contrast, for SWaT [GAJM16] and WADI [APM17], they observed a few differences between the training and testing parts, which makes them challenging but nonetheless good datasets.

The primary concern with the current surveys enumerating existing datasets and testbeds is that they take a viewpoint from the published datasets. Thereby it remains unknown how often a single dataset is actually leveraged by the research community. Moreover, it is unknown whether research even leverages datasets to a great extent or instead rather considers private datasets or individual testbeds more suitable for their evaluations. Such aspects are not tackled in related work.

3.1.3 Detection Methodologies

Due to the diversity of ICSs, various surveys analyze the detection methodologies proposed in the research domain (cf. Table 3.1). While these cover up to 80 papers in a single survey [UJJM22], showing the huge attention industrial intrusion detection attracts, they often concern narrow research fields within the ICS domain. Related work includes specialized surveys considering just machine learning approaches [LXC+21, BHIM22, UJJM22], special domains such as for vehicles [LKP+19], or the water treatment domain [RAMH18], and even specific types of IIDSs like physics-based [GUC+18] or network-based approaches only [RAMH18, RSMP20]. Another drawback of the current surveys is the lack of systematic approaches to gathering the publications analyzed in the surveys. While Umer et al. [UJJM22] found and discussed 80 publications with a systematic approach, they limited their scope to IIDSs with the keyword "machine learning". Also, Rakas et al. [RSMP20], who conducted a systematic literature research, in the end, handpicked and analyzed just 17 publications.

These surveys provide great insights into established detection methodologies. However, by focusing on individual domains or specific types, the results are confined to those niches. Especially for the diverse ICS domains, with lots of use cases, a holistic overview across all domains is currently missing so far.

3.1.4 Evaluations and Metrics

Besides datasets or testbeds, the choice of metrics and evaluation methodologies are equally important when judging the performance of an IIDS. Without a dedicated focus on industrial intrusion detection, Powers [Pow11] provides an overview of different metrics and puts their expressiveness into context. Yet, the considered point-based metrics (cf. Section 2.4.1), e.g., accuracy or precision (also used in other domains such as machine learning), must be used carefully not to introduce any biases [Pow11, FSL+22]. In that regard, Christen et al. [CHK24] performed an in-depth analysis of the F1 score across many research disciplines.

For evaluations on time-series datasets, further challenges, such as an imbalanced representation of attacks, have to be considered [GS14, AQP+22]. Therefore, advanced time-aware metrics have been proposed [LA15, TLZ+18, KYK20, HYKK19, JMAC20, HNR22, HYKM22, GZS+22] (cf. Section 2.4.2). While this development promises to enhance the expressiveness of evaluations, their soundness, especially their use in research, remains mostly unexplored.

3.1.5 Meta Surveys

Finally, various meta surveys focus on pitfalls for industrial intrusion detection. Most of them stem from classical machine learning, which is not specific to industrial intrusion detection but is still highly relevant [SP10, AQP+22, ORL20, ALS23]. Other surveys highlight challenges when transferring IIDSs from research to actual

industrial deployments [SP10, AMM20, MAM21]. These problems include, e.g., inappropriate use of metrics [AQP+22], the dominance of lab-based datasets [RSMP20, AQP+22], analyses of dataset quality [TETC20, LSAA23], or concern the predominant focus on a few of the industrial domains and protocols [RSMP20]. With respect to network-based intrusion detection leveraging machine learning, not just for ICSs, Apruzzese et al. [ALS23] propose a new concept of pragmatic assessments. They advocate for evaluating more toward the values perceived by industrial operators actually implementing IDSs. Importantly, empirical data on the severity of the issues in the IIDS research landscape is not yet available.

3.1.6 Conclusion

The development of IIDSs can, in theory, be based on a solid foundation of well-documented attacks from the past, public datasets, advanced metrics, and sophisticated evaluation methodologies and does not lack ideas for the realization of detection methodologies. However, research lacks an understanding of the methodologies *actually* applied within it beyond individual criticism regarding isolated aspects.

Thus, there is a need to systematically consolidate the current state of industrial intrusion detection research to identify remedies against the status quo. We argue that by systematically analyzing the IIDS research landscape on a broad scale, as done in a SMS [Kit07], we can comprehensively tackle the overarching research questions what the state of IIDS research is. Answering this question lays the foundation to identify issues in IIDS research, enabling the different actors within the research community to focus their joint efforts on the overarching challenge of securing industrial deployments.

3.2 Designing a Systematic Mapping Study

The objective of this first contribution is to provide a systematic understanding of the IIDS research landscape's current status quo. While previous literature surveys already hints at prevalent issues, usually with topic-dependent studies (cf. Section 3.1), a holistic and large-scale analysis for *industrial* IDS research is missing.

Therefore, we strive to ascertain the state of IIDS research by conducting a SMS, a variation of a classical Systematic Literature Review (SLR) [Kit07], to obtain a large, qualitative, and unbiased collection of relevant publications in a verifiable process, oriented along established best practices and guidelines [Kit07]. In contrast to a systematic literature review, a SMS addresses a broader scope resulting in a more general search string with many hits which can imply a slightly vague data extraction process due to the diversity of found publications [Kit07]. In return, its broader scope enables a SMS to identify further research topics which demand detailed attention in the future.

To holistically answer the outlined research questions for a large and heterogeneous research field, we perform a SMS as depicted in Figure 3.1. To this end, we first search

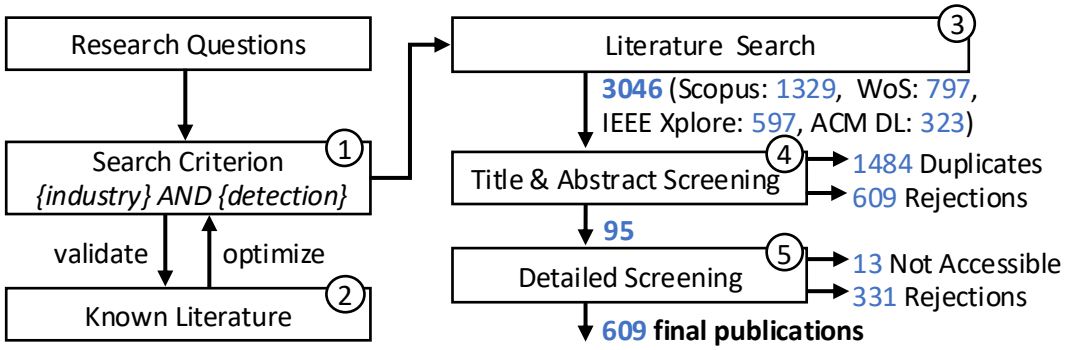


Figure 3.1 To conduct the SMS, we follow a two-staged approach which results in extracting a total of 609 relevant publications proposing novel IIDS as of December 2022.

relevant papers for a broad subject (here IIDS proposals) from the scientific literature with a systematic process. Afterward, publications are analyzed and classified based on the subjects of our analysis, i.e., what the current state of IIDS research looks like. Our SMS focuses specifically on publications that propose IIDSs for ICSs. In contrast to related work, such as by Balla et al. [BHIM22], we only consider publications that leverage at least one *industrial*-specific dataset, i.e., obtained from an ICS or a related testbed and including specific protocols such as Modbus, physical process data, or ICS-specific cyberattacks.

To conduct our SMS, we leverage Parsifal [Fre21] to organize and comprehensibly document our screening process. First, we transformed the research questions into a search string ①, which we successively optimized through validation with an initial set of known and representative literature ②. As depicted in Figure 3.1, the search string combines collections of keywords for the phrases *industrial* and *detection*. After validating the outcome of several search strings and keywords against known literature of that research landscape, we derived the following search string:

(“Industrial Control Systems” OR “Process Control Systems” OR “Supervisory Control and Data Acquisition” OR “PCS” OR “ICS” OR “SCADA”)
AND
(“Attack Detection” OR ”Anomaly Detection” OR “Intrusion Detection”)

The search string was applied to the titles, abstracts, and keywords. Note that search engines do not care about capitalization or singular vs plural.

We then queried four search engines (IEEE Xplore, ACM DL, Scopus, and Web of Science) on December 2022 and found a total of 3046 hits ③. We include publications that were published until the end of 2021. From this initial set of publications, we discarded duplicates (1484 publications) and performed a first screening of all remaining publications’ titles and abstracts ④. In this initial screening, we mostly focused on removing publications from other research domains that still matched our search string and such publications that clearly do not propose an IIDS approach. After this first screening phase, 953 unique publications remained. Note that we did not filter for any specific detection techniques nor industrial domains.

In a final step, we conducted a detailed screening of the remaining publications to extract those that build the foundation for our further analysis ⑤. When accessing the full text of all papers, only 13 publications were not accessible to us and thus omitted. We performed a detailed second screening of all remaining publications, resulting in 331 further rejections of those that do not match our requirements for proposing IIDSs, e.g., belonged to fault detection rather than intrusion detection (cf. Section 2.2.1). From the resulting set of 609 accepted publications, we extracted the data to answer our research questions, such as the datasets and metrics they utilize for their evaluations or the number of publications they compare against. To ensure consistency, one person performed the detailed screening and data extraction while the workload for initial title/abstract screening was shared across multiple persons.

Through our systematic approach, to the best of our knowledge, we are the first to analyze the IIDS landscape in such detail. With 609 analyzed publications, our work bases on a larger knowledge base compared to related work (cf. Section 3.1). This enables us to analyze the evolution of the broad IIDS research landscape.

3.3 Overview of the IIDS Research Landscape

With the collection of publications gathered in our SMS (cf. Section 3.2), we are now able to assess how the overall research landscape on IIDSs has evolved over time and how it connects to each other. As a systematic representation of the IIDS research landscape has been missing so far (cf. Section 3.1), we begin with a high-level overview of its evolution over time in Section 3.3.1. Next, in Section 3.3.2, we are interested in the connectivity, i.e., whether IIDS research is made coherently or progresses parallel to each other. Following up on connectivity, we take a look at how publications compare new findings to each other in Section 3.3.3 and whether reproducing or building upon other works is made easy to advance the research landscape as a whole in Section 3.3.4. We conclude our findings in Section 3.3.5 with the insights gathered so far and lay out the path for subsequent deeper analyses.

3.3.1 Evolution Over Time

To understand the evolution of the IIDS research domain, we focus on the number of published papers over time (cf. Figure 3.2), which we enrich with timestamps of notable cyber incidents and the releases of commonly used evaluation datasets. While the first publications within the IIDS domain date back to 2003, the domain initially received little attention, with only 28 publications until 2012. From 2013 onward, research took off exponentially, with an average increase of 40.9 % in yearly publications. In comparison, the Top 10 cyber security conferences experienced a lower average yearly increase in accepted publications from 7.2 % for Crypto up to only 25.5 % for USENIX Security during the same timespan [Zho09]. In 2021, the last year considered in our SMS, we identified 130 new publications, which is higher than in any previous year.

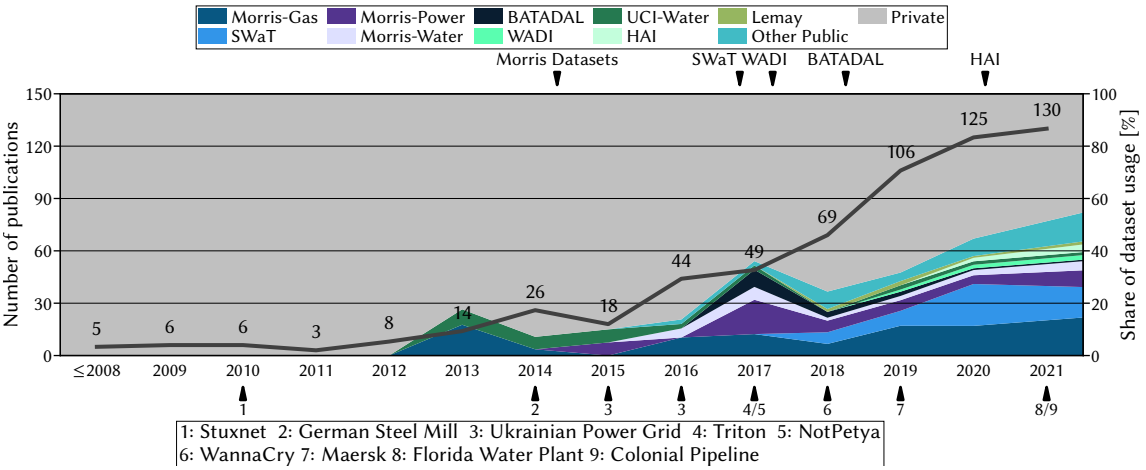


Figure 3.2 Publications on IIDSs took off around 2013 and kept increasing as more cyberattacks occurred. Simultaneously a trend fosters to evaluate on public datasets. The black line shows the number of yearly publications and the colored areas show the distribution of the used datasets over time.

We presume that the key driver for the interest in this research domain is caused by the raised public awareness following the Stuxnet cyberattack and subsequent ones like the two incidents with the Ukrainian power grid [ACZ20]. Apart from such targeted attacks, industries were equally affected by more widespread malware, such as NotPetya or WannaCry [ACZ20], due to their increasing digitalization and Internet-facing deployments (cf. Section 1.1). With attacks still continuing [MSM+21], the peak in 2021 with 130 proposals comes as no surprise — underlining the growing relevance of IIDS research.

A first look at the (publicly) utilized datasets' in Figure 3.2 also allows us to deduce which industrial domains are tackled in research. Unsurprisingly, we find that research indeed takes place for various industrial domains, such as water purification, gas distribution, and electrical power generation, among many others. This conclusion aligns with recent results identifying a growing number of public datasets emerging across many industrial domains by Conti et al. [CDT21].

Takeaway. From this initial assessment, we conclude that IIDS research is a field with a vivid community. IIDS proposals tackle the diverseness of industrial domains based on variously utilized datasets and research experiences steady growth that does not seem to have reached its peak yet.

3.3.2 Coherence

For such a growing research landscape in a diverse industrial environment, we further want to understand how coherent research is performed, i.e., whether the domain is split into loosely connected topics or which directions receive the most attention. Therefore, we visualize the connections among publications by their citation relationships in Figure 3.3.

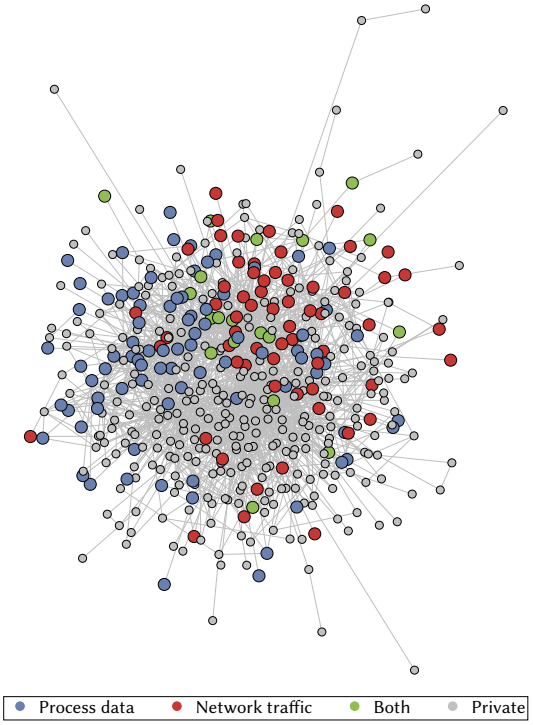


Figure 3.3 Publications arranged in a citation graph reveal two directions which are roughly disjunct into approaches considering network traffic datasets and ones evaluating process data.

Citation data was retrieved and aggregated from OpenAlex and Semantic Scholar for all 609 publications, and we draw a connection between two publications if one cites another. In Figure 3.3, publications are arranged by the Kamada-Kawai force-directed placement algorithm [KK89]. Moreover, for publications utilizing publicly accessible datasets, we colored their vertices belonging to process data datasets, network traffic, or both (cf. Section 2.2.3.1). Note, however, that our analysis omits 125 publications for which no connection to other publications could be found, either because the citation data for the respective publications was incomplete or because the IIDSs were indeed presented without relating to the vast body of existing works.

On average, a publication is cited by 2.9 other IIDS publications, while the Top 5 cited publications [CCG+11, UGC+16, GW13, HSZH14, KS18] (not in order) are cited on average by 46.6 papers as of the 1st March 2023. These numbers provide a first glance at the connectivity in IIDS research. However, note that this observation neither implies that a publication builds on top of prior works from the cited publication nor that they compare to that publication within their evaluations.

While inspecting the citation structure, we observe that the IIDS research domain is divided into two basic directions based on the evaluated dataset types. A first group of 102 papers (blue) resembles the larger class that focuses on process data datasets. A second slightly smaller class of 81 publications (red) corresponds to intrusion detection methodologies detecting attacks in network data. Only rarely (19 times) does a publication contain evaluations for both classes (green). Interestingly, both research areas show little connectivity, indicating a limited exchange of knowledge across these fields. This is backed by the fact that the clustering coefficient for the

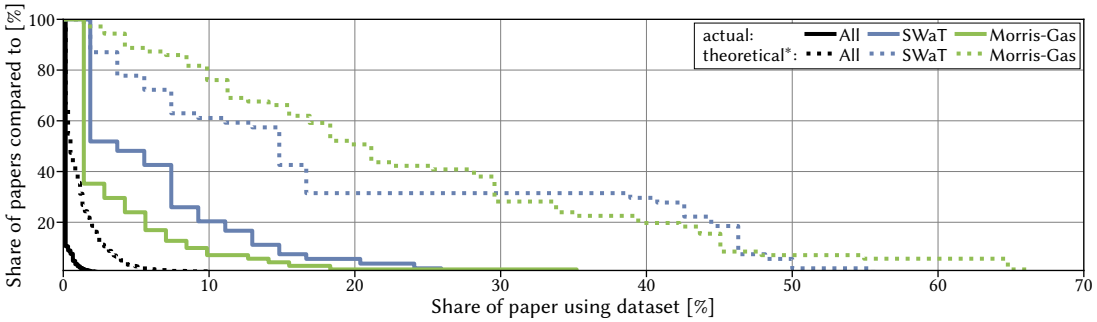


Figure 3.4 Authors compare IIDSs to 0.5 approaches on average, while theoretically, they could compare to at least 6.0 (black). This gap increases for papers evaluating the top two frequently used datasets SWaT or Morris-Gas. We define a paper as comparable to all papers that share one dataset and metric and were published at least one year earlier.

sub-domains (process data 0.15 and network traffic 0.13) is slightly higher than for the entire IIDS landscape (0.11).

Takeaway. Publications are more likely to cite each other if they stem from the same type, which promises a high number of comparisons among them. Still, the low clustering hints at an incoherence problem in the overall research domain.

3.3.3 Comparability

Given our observation that the degree of connectivity is low in the IIDS research domain with just 2.9 citations among each other on average, we now analyze more specifically whether this also affects comparability between IIDSs. Comparability allows authors to determine how well an approach performs, i.e., to highlight the impact of newly proposed contributions over previous work or which approaches might be suitable for real-world deployments. Optimally, this requires that two works under comparison have been evaluated on at least one shared dataset. Likewise, to objectively judge their detection performance, both publications must employ a minimum of one identical evaluation metric.

To judge the degree of comparability across the research landscape for each publication, we extracted the *actual* number of comparisons made by the authors and calculated the number of *theoretically* possible comparisons. Therefore, while conducting our SMS, we gathered how many publications each author uses as comparison references and additionally extracted the exact metrics used in each publication's evaluation. We estimate the minimum amount of theoretically possible comparisons by counting a publication as comparable if it shares at least one common dataset and metric and was published in an earlier year. Note that while not every two publications assume the same attack model, comparability can still be justifiable in the cases where the dataset matches since authors should select a dataset that best fits their approach. This methodology provides a great opportunity to assess actual and theoretical possible comparability, as depicted in Figure 3.4.

Overall, the number of *actual* comparisons performed by researchers is low, with 0.5 publications on average. For the two most-common datasets, cf. Figure 3.2, we observe higher values (SWaT 2.4 and Morris-Gas 1.7). Still, there exists the *theoretical* opportunity for authors to compare a proposed IIDS to an average of 6.0 alternatives. On the one hand, this proves that many works are indeed comparable in terms of datasets and metrics. On the other hand, prominent datasets help in that regard since their theoretical comparability is higher (SWaT 10.0 and Morris-Gas 16.1). Note that it should not be the ultimate goal to compare against as many publications as possible since quality is preferential before quantity.

Looking closer into the details of Figure 3.4, it is interesting that 10 % of the publications evaluating the Morris-Gas dataset (green) actually compare only against 7 % of different Morris-Gas publications. However, for SWaT (blue), 10 % of publications are actually compared to about 20 % of existing works. Meanwhile, theoretical comparability for Morris-Gas publications is even higher than for SWaT (dotted lines). Regarding all publications (black), a total of 95.4 % of publications are not compared to a single IIDS likely resulting from the use of many private datasets.

Takeaway. The state of comparability in IIDS research is low with great opportunities for improvement as many publications share common datasets and metrics.

3.3.4 Reproducibility

In the cases where comparability is not possible or insufficient, reproducing existing works can become necessary, especially if implementations are not publicly available. Generally, reproducing existing works is not uncommon within IIDS research, e.g., to concisely analyze the prospects and limitations of individual approaches [ET22, KWP+22a], prove the feasibility of new ideas upon reproduced implementations, or solely for scientific profoundness [PB07]. Yet, successfully reproducing approaches is not guaranteed [ET22]. To enable the process of reproducing IIDS research, the availability of artifacts, such as datasets or code, is optimally needed.

In our survey, we observe that 33.3 % of the publications utilize public datasets with an improving trend (54.7 % of the utilized datasets in 2021 are public; cf. Figure 3.2). However, successfully reproducing older publications is less likely. While the availability of code is not strictly required, as the relevant details should be part of the publication, it eases the reproducibility process. Unfortunately, it is difficult to ascertain the availability of source code in a systematic way as it is not always clear where to find availability statements or corresponding pointers in publications. Still, we only encountered 21 publications with obvious references, e.g., clearly highlighted repositories. We subjectively deduce an overall low availability of source code across IIDS research. Thus, researchers often have to rely on the descriptions and evaluation results provided by the paper to verify their code.

Takeaway. Reproducibility is challenging as optimally both criteria (public dataset and source code) have to be met. The increasing use of public datasets promises improvements in at least one direction, while publicly available artifacts accompanying publications remain the exception.

3.3.5 Conclusion

From our first overview of the IIDS research landscape, we observe that this field is gaining in importance, with a peak of 130 yearly publications in 2021, the last year of our study. This highlights the relevance of cybersecurity for ICS. At the same time, the proposals address different fields of the industrial domain, as we deduced from the diversity of the leveraged datasets. But digging deeper, we find first signs of insufficiency worth improving.

First, the low connectivity within the research domain is suspicious, with only 2.9 citations on average to other works of that domain. This is reflected also in the comparability of new approaches to existing ones as publications compare on average to just 0.5 papers. While comparability is being perceived as a good scientific standard [PB07], reproducibility also lags behind expectations, yet also in domains beyond intrusion detection research [Bak16].

These observations are interesting, given the growing availability and use of public datasets. Consequently, investigating the way how IIDSs are evaluated might be worth investigating to understand the root causes of these issues.

3.4 Benchmarking Datasets and Evaluation Metrics

With a basic understanding of the IIDS research domain, we now assess how evaluations are conducted in more detail. We conduct the following analyses with the goal of identifying circumstances that inhibit the further consolidation of IIDS research. In this context, the benchmarking datasets and the evaluation metrics resemble essential elements of scientific evaluations (cf. Figure 1.2).

First, the chosen benchmarking datasets are a crucial building block as they serve as the basis for nearly all subsequent performance calculations. While related work has assessed which datasets are readily available (cf. Section 3.1), their exact usage and distribution remain unknown as of now, which we are going to address in Section 3.4.1. The same holds for the evaluation metrics, where related work from other research fields conducted broad studies (cf. Section 3.1), yet not specifically targeted for IIDS purposes, which we tackle in Section 3.4.2. Because of reports about flaws of established metrics [GS14, AQP+22, HYKM22], we additionally conduct a short practical experiment on the expressiveness of different evaluation metrics in Section 3.4.3. Finally, we conclude our findings and try to answer whether these new insights can explain the current insufficiencies of IIDS research in Section 3.4.4.

3.4.1 Datasets

In contrast to other research fields, industries variety implies a similar diversity in benchmarking datasets or testbeds as only then an IIDS can be tested under realistic conditions for its designed use case or its feasibility across multiple scenarios. After an initial overview over the usage of the existing datasets in Section 3.4.1.1,

we specifically highlight which domains and types the relevant datasets represent (Section 3.4.1.2) and whether they are representative for the distributions found in the real world. Lastly, Section 3.4.1.3 concerns their embedding into publications.

3.4.1.1 Evolution of Datasets

As can be derived from the previous Figure 3.2, over the entire timespan of the survey, the majority of used datasets are private, and only 33.3 % of the publications evaluate at least one public dataset. Note that we counted datasets as private if there existed no obvious procedure to retrieve the dataset. While private datasets may represent unique use cases, e.g., real-world data of industrial facilities, they hinder reproducibility and comparisons to related works since they deny access to outsiders. In our survey, we refrained from investigating private datasets in more depth because of the varying degrees of descriptions throughout the publications. Hence, needed details cannot be fully captured or verified. Nonetheless, we observe a trend starting around 2013 toward increased utilization of public datasets, which accounts for 54.7 % of the evaluated datasets in 2021. Therefore, it is more likely that an IIDS uses public datasets if published recently.

This trend follows the publication of high-quality datasets that are still widely used today. When looking at peak usage of public datasets, the SWaT [GAJM16] and Morris-Gas Pipeline [MTT15] datasets jointly occur in 20.4 % of the publications, which is the majority of the publications utilizing a public dataset at all (33.3 %). Other public datasets are thus used much less frequently. As a consequence, a portion of research activities seems to be biased toward these two datasets.

Regarding dataset diversity, we identified 35 unique public datasets, which exceeds previous reports of 23 datasets by Conti et al. [CDT21]. In contrast to Balla et al. [BHIM22] and by the design of our SMS (cf. Section 3.2), we dominantly encounter *industrial* datasets contradicting their observed research bias toward non-industrial datasets. But 16 of the datasets are used once, and 14 occur at least three times (the Top nine public datasets are depicted in Figure 3.2). Thus, availability alone is not decisive for a widespread use and other factors such as covered domain and attacks, or the overall quality of the data play an essential role too.

3.4.1.2 Dataset Types

Next, we examine the Top nine datasets and highlight their different directions in Table 3.2. Among the Top nine datasets, the *data type* is either a network capture, required for network-based IIDSs or a (preprocessed) sample of physical system data, e.g., a time series of water levels (cf. Section 2.2.3.1). IIDSs working with host data (cf. Section 2.2.3.1) play a subordinate role according to our SMS. For the other two types, we identify one major origin that accounts for most of the utilization across research, with iTRUST for process-based datasets and Morris et al. primarily for network-based ones. Considering the type of the Top nine utilized datasets, we observe a strong focus on process-based datasets with 20.3 % compared to 15.6 % for network-based, which is in line with the observations from Section 3.3.2.

Origin	Name	Type	Domain	Protocol	Usage
iTRUST[a]	SWaT [GAJM16]	P*	Water	–	9.0 %
	BATADAL [TGT+18]	P	Water	–	1.6 %
	WADI [APM17]	P	Water	–	1.0 %
Morris et al.[b]	Morris-Gas [MTT15]	N	Gas	Modbus	11.8 %
	Morris-Power [APM13]	P	Electricity	–	5.6 %
	Morris-Water [MTT15]	N	Water	Modbus	2.8 %
Misc	UCI-Water [Poc93]	P	Water	–	2.0 %
	HAI [SLYK20]	P	Diverse	–	1.1 %
	Lemay [LF16]	N	Electricity	Modbus	1.0 %

N: Network captures P: Process data

* Network captures for SWaT exist, but are rarely used in research.

[a] `https://itrust.sutd.edu.sg/itrust-labs_datasets`

[b] `https://sites.google.com/a/uah.edu/tommy-morris-uah/ics-data-sets`

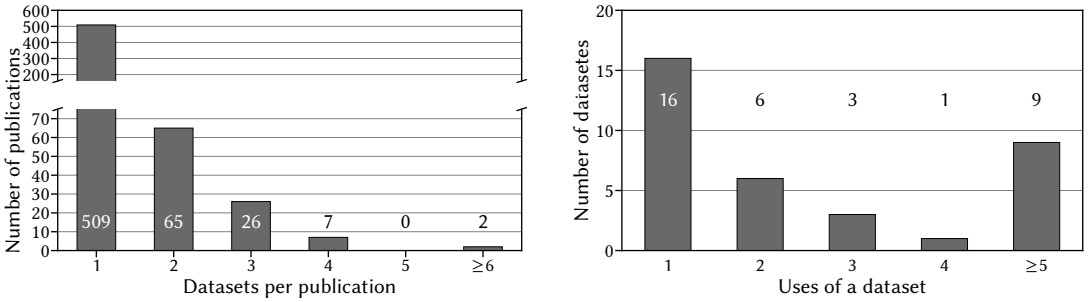
Table 3.2 Across the top nine public datasets, two account for the majority of uses. Despite ICSs' diversity, the top datasets focus on a few domains and protocol combinations.

Since industrial domains are diverse, we expect coverage of them across the datasets as well. However, the covered industrial domains are driven by the water and gas facilities, indicating an underrepresentation of other domains, such as power generation, electricity distribution, or manufacturing. Yet, considering the large numbers private datasets, for which (high-quality) public alternatives do not exist, we cannot conclude that other domains receive few attention nor that those industries show no interest in IIDS research.

Lastly, industries are well known for their diverse and incompatible pooling of network protocols, mostly for legacy reasons [DLP+22] (cf. also Section 1.1). Despite market-share studies identifying 11 dominant network technologies [HMS21], research dominantly focuses on Modbus (having 10 % market share [HMS21]). While we discovered IIDSs for further industrial protocols such as S7 [LNTA17], IEC 60870-5-104 [LNT19], or DNP3 [RSE+20], their representation is marginal and mostly confined to private datasets. Therefore, the distributions of utilized datasets with respect to their type, industrial domain, and network protocol reveal a significant drift between peer-reviewed literature and actual production systems.

3.4.1.3 Research Embedding

Lastly, we assess how the datasets are embedded into research. We begin with the number of datasets that are used within a single publication, as shown in Figure 3.5a. Many publications (509) evaluates a single dataset, and a minority (100) more than one. One publication uses 1.3 datasets on average while 45.7 % of the datasets are used in a single publication only (cf. Figure 3.5b). This observation is in line with the clustering observed in Section 3.3.2, which is more coherent with respect to the most-used datasets, suggesting that researchers primarily focus on a single dataset. Given that we found 35 publicly available datasets, researchers can consider additional datasets when claiming that their IIDS is applicable to several ICS domains. This claim is backed by the fact that two publications evaluated as many as six



(a) Number of datasets used in one paper. **(b)** Amount a dataset is used in papers.

Figure 3.5 Publications usually utilize a single dataset, and only 16.4 % of the papers leverage multiple datasets at all (Figure 3.5a). At the same time, a large number of 45.7 % of the datasets are only used once (Figure 3.5b).

Combination	Count	Origins
Morris-Gas & Morris-Water	12	1
Morris-Gas & Morris-Power	8	1
Morris-Power & Morris-Water	7	1
SWaT & WADI	4	1
Morris-Gas & UCI-Water	4	2
Morris-Gas & SWaT	3	2
Electra Modbus & S7Comm	3	1
Morris-Gas, Power & Water	5	1

No private datasets were considered

Table 3.3 If multiple datasets are used, they mostly stem from the same class or origin, attributing little to richer evaluations.

datasets [GBP14, BJF21]. However, our results also suggest a discrepancy between datasets with respect to ease of use, documentation, and completeness, motivating the limited use of the available datasets.

Looking into the preferred datasets, Table 3.3 enumerates the top dataset combinations. While we observe prominent combinations, the corresponding datasets usually originate from the same source and thus represent similar domains and protocols. Only seven publications evaluate datasets that stem from two origins. Thus, potentially widely applicable IIDSs are evaluated for specific (research) deployments from a single industrial domain, most likely not representative of an entire domain.

Outside the industrial-IDS domain, similar observations have been made, e.g., for network-flow based intrusion detection [MBJ+25]. There, a few established datasets are also commonly used hindering a broad view on the capabilities of detection methodologies in diverse settings. Mondragon et al. [MBJ+25] provide pre-processed and unified versions of several datasets to ease evaluations in the future.

Takeaway. IIDS research is still governed by private datasets, with an increasing trend toward public datasets. However, we observe potential for improving the number of datasets used during evaluation as well as their diversity with respect to type, industrial domain, and network protocol.

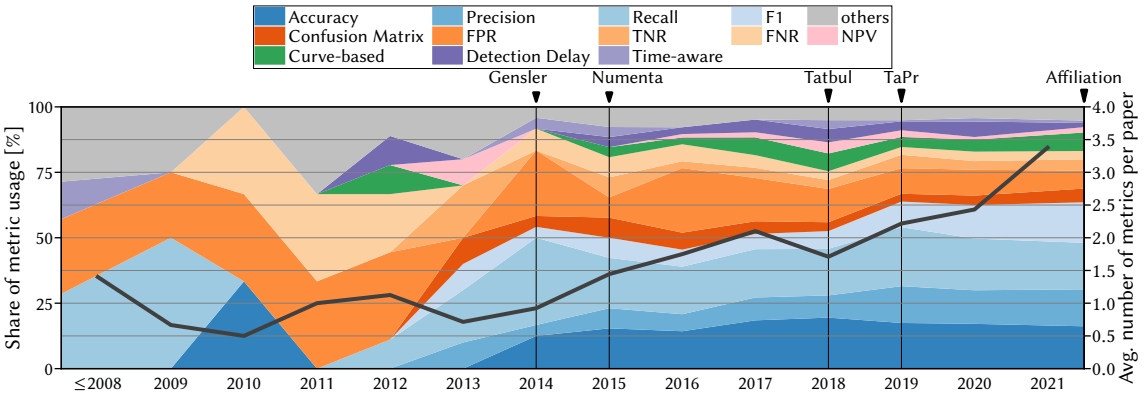


Figure 3.6 Point-based metrics dominate IIDS evaluations, with accuracy, precision, recall, and F1 being the top most used metrics. Over time, the number of metrics in a publication increased to currently 3.4 on average in 2021 (cf. black line).

3.4.2 Evaluation Metrics

While benchmarking datasets are an important aspect of evaluations, they are little useful if not combined with standardized and expressive evaluation metrics. In that regard, one can differentiate between metrics that measure, e.g., the detection performance or computational resources (cf. Figure 1.2).

Since *computational* resources are stated only occasionally throughout the SMS, we shortly introduce which aspects were evaluated. The most prominent aspect, still in 136 publications, refers to the time to train a model or classify a given datapoint/dataset. More infrequently are statistics about CPU/GPU usage (13), RAM utilization (12), or model size (16). However, a sound comparison without equivalent hardware or implementations is challenging. Therefore, those metrics are out of scope in the following.

Rather, we focus on metrics measuring the *detection* performance of an IIDS where a wide variety of metrics exists (cf. Section 2.4). More precisely, we ask how often and when these metrics are used. Overall in our SMSs, we found 186 different metrics and flavors, including subtle deviations such as multi-class or weighted variants. To handle this amount of metrics and flavors in the following, we aggregated them into similar classes. Lastly, since a majority of the metrics are used infrequently, i.e., only 12 occur at least ten times, we bundle rarer metrics into a single class (others).

With this aggregation, we now take a broader view on the utilization of metrics across IIDS research in Section 3.4.2.1. Additionally, we also investigate whether the selection of a given dataset also implies a different choice in metrics in Section 3.4.2.2.

3.4.2.1 Metrics Over Time

To obtain a first overview of the utilization of frequent metrics, we depict their use over time in Figure 3.6. First of all, the number of different metrics used in a single publication on average (2.5 overall) kept increasing since 2013, and nowadays, publications use 3.4 metrics on average. This coincides with the previous observation

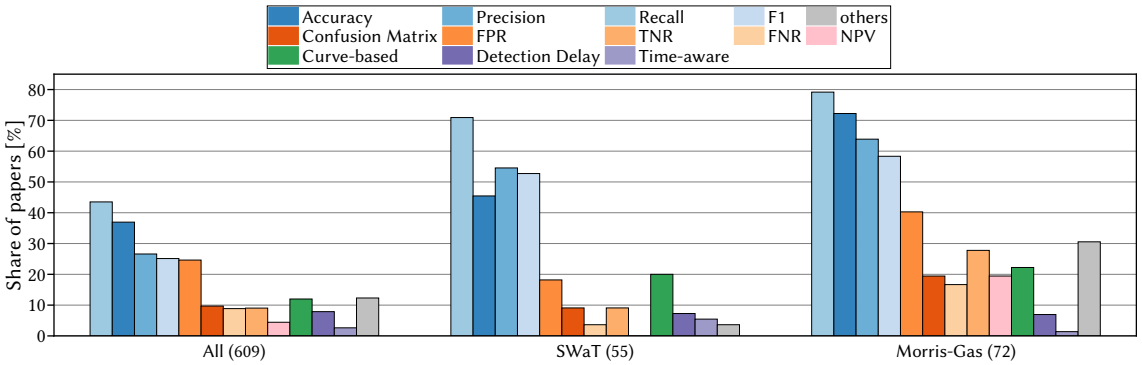


Figure 3.7 Papers utilizing SWaT and especially the Morris-Gas dataset are dominated by point-based metrics. Time-aware metrics are slightly more frequent for SWaT.

in Figure 3.2, where the year 2013 marked the turning point when IIDS research took off. This trend toward more metrics contributes to higher comparability in the research domain and hints at in-depth evaluations. However, there also exist 157 publications that evaluate without any quantitative metrics and instead rely only on textual descriptions, e.g., elaborating which attack scenarios were detected or discussing results visually along graphs. Note that textual descriptions cannot be aggregated into a unified class as they differ significantly, i.e., two publications using textual descriptions hardly describe the same feature.

In contrast to dataset utilization (cf. Figure 3.2), the metric utilization fluctuates less over time. One notable trend, again starting around 2013, is that accuracy, precision, recall, and F1, i.e., the classical point-based metrics [Pow11], have established themselves as metrics with high usage by representing 59.1 % of all used metrics. At the same time, out of the 354 publications utilizing one of these four metrics, only 82 state all four. Thus their usage is inconsistent, and most publications only focus on certain aspects of their expressiveness.

Concerning all point-based metrics, which account for 89.8 % of all metrics, the confusion matrix resembles an important metric as it builds the foundation to calculate all point-based metrics (cf. Section 2.4.1.1). However, out of the 59 papers that publish the confusion matrix, just 20 fully state or discuss all four common metrics (accuracy, precision, recall, and F1), even though this would be easily doable. In the 9.7 % of publications where the confusion matrix is published, at least missing metrics can be calculated, which is not possible the other way round, i.e., the confusion matrix cannot be computed if, e.g., only the F1 score is indicated. It thus remains questionable why publications omit frequently used metrics when all data to compute them has to be available anyway.

Even though it has been known since 2014 that for industrial IDSs, point-based metrics may be flawed [GS14], they make up 89.8 % of all metrics. Among the time-aware metrics, detection delay receives constant but infrequent use by 48 publications overall. Still, detection delay alone does not quantify the portion of detected attacks and thus likely serves to enhance point-based metrics. Newer promising time-aware metrics yet have to gain traction (only 16 publications use them), despite their added value in interpreting IIDS results [HYKM22].

3.4.2.2 Metric Distribution on Datasets

Taking a closer look at the metric utilization, it is interesting to investigate the differences between industrial domains or datasets since we already observed the formation of obvious clusters in Section 3.3.2. However, as not all IIDS publications clearly state which ICS domain their approach is designed for, we opted to concentrate on the selected evaluation dataset. We pick the two most commonly used datasets, SWaT and Morris-Gas, representing not only two dataset types (process data and network captures) but also different domains (Water and Gas), as apparent from Table 3.2. Figure 3.7 depicts the dataset's influence on the chosen metrics by comparing their metrics distribution against the set of all publications.

The Top four metrics (accuracy, precision, recall, and F1) play a major role for the SWaT and Morris-Gas datasets too, even more than across all publications. For example, recall is used in 43.5 % of all publications but indicated for 79.2 % of IIDSs evaluated on the Morris-Gas dataset. The order of usage between them is also similar, i.e., recall is used the most and the F1 score the least. One exception is accuracy, which is indicated less often for the SWaT dataset. This difference might be caused by SWaT featuring fewer attack instances. Another exception is that other point-based metrics (confusion matrix, FPR, TNR, FNR, and NPV) receive greater attention in the Morris-Gas dataset. Contrary, time-aware metrics are slightly more common for SWaT. We suspect the favor for time-aware metrics in SWaT to stem from the dataset nature consisting of longer attacks (cf. Section 2.3.2) compared to individual malicious network packets in the Morris-Gas dataset.

Takeaway. IIDS research leverages a broad spectrum of standard evaluation metrics known from related branches such as machine learning. However, evaluations dominantly build upon point-based metrics, which are known to have flaws, especially on time-series datasets as used in IIDS research [GS14, HYKK19].

3.4.3 Expressiveness of Metrics

Until now, our analyses are based on theoretical observations from literature, e.g., which datasets and metrics are used. In the following, we extend our analysis beyond a SMS with *practical* experiments to understand the quantitative impact of metric choices on the evaluation outcomes and to identify metrics that offer high expressiveness. To this end, we conduct a comparison study across four IIDSs from research on two datasets to compare various metrics. We consider especially newer time-aware metrics [HYKM22] to compare them against their point-based variants.

3.4.3.1 Experiment Design

As we observed in Section 3.4.1, the IIDS research community is governed by two major directions of datasets: network-based datasets such as the Morris-Gas [MTT15] and process data datasets such as SWaT [GAJM16] containing physical time series data. We aim to cover both types within our experiment and thereby also cover two

important IIDS types from research, namely misuse- and behavior-based IIDSs (cf. Section 2.2.3.2). For misuse-based IIDSs, we examine two standard machine learning approaches, Support Vector Machine (SVM) and Random Forest (RF), that were examined on the Morris-Gas dataset in related work [PASE18, WAM20]. Regarding process data, we leverage two behavior-based IIDSs, i.e., Seq2SeqNN utilizing neural networks [KYK20], and TABOR fusing multiple detection methodologies such as timed-automata and Bayesian networks [LAVM18]. Contrary to the misuse-based machine learning approaches, these IIDSs are evaluated on the temporally *ordered* SWaT dataset, which provides dedicated attack-free training data and testing data, including anomalies. Section 2.3.1 contains more information about the datasets.

3.4.3.2 Metrics Under Study

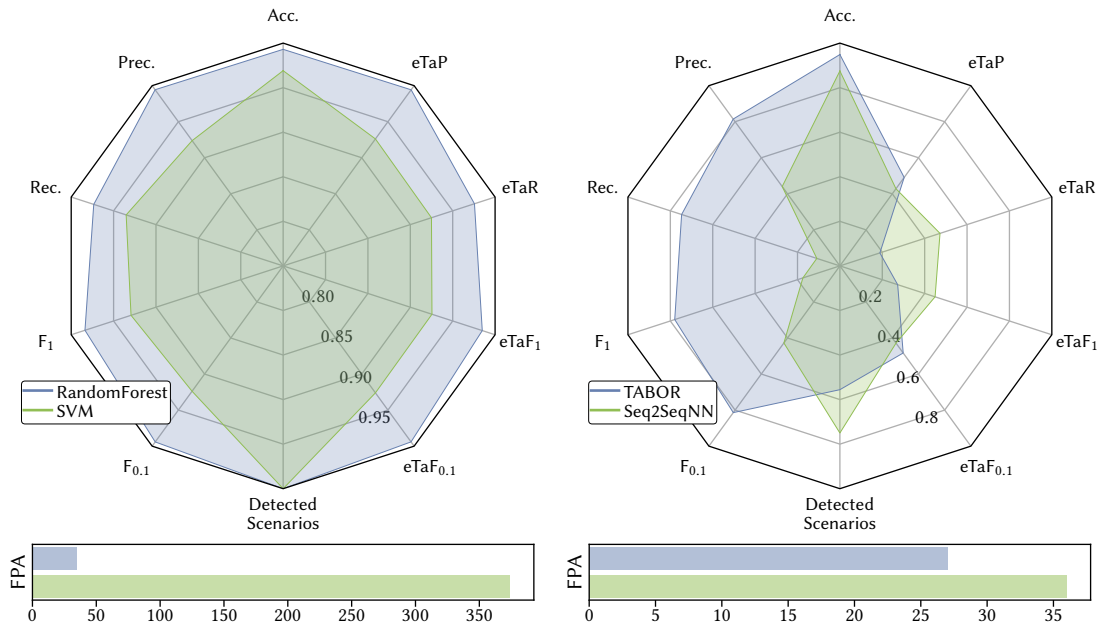
We focus on the four common point-based metrics accuracy, precision, recall, and the F1 score and modern time-aware variants of them called enhanced time-aware recall precision and recall (eTaPR) [HYKM22] (cf. Section 2.4). We opted for the time-aware metrics proposed by Hwang et al. as they provided an implementation for their new metrics. More precisely, eTaP for precision, eTaR for recall, and eTaF1 for the F1 score (there is no time-aware accuracy equivalent). These metrics, like their point-based counterparts, favor high detection rates but diminish consecutive alarms if they start too early or overhang beyond the duration of an attack.

Furthermore, we examine a variant of the F1 score weighting precision and recall differently. This may be crucial in industries since cyberattacks are rare compared to normal behavior, preferring a high precision over false alarms. The datasets in our study already incorporate this class imbalance, with Morris-Gas containing 22 % malicious data, SWaT 12 %, and real deployments likely observing even fewer attacks. Thus, we examine $F_{0.1}$, weighting precision ten times more than recall.

As additional metric, and since there is only one repetition for each attack type in SWaT, we discuss the percentage of detected scenarios (unique attacks). Lastly, we consider the number of continuous False-Positive Alarms (FPAs) as a time-aware variant of the point-based metric false positives. For example, if there is an alarm for 5 minutes and the dataset was sampled in an interval of 1 second, this would count as a single FPA whereas the point-based metric would count 300 individual false positives [LAVM18]. Note that, similar to TABOR [LAVM18], we count an alert only as a false positive if it is at least 60 seconds away from an attack since an attack's effect may persist even shortly after the attack's end, resulting from inaccurate labeling for some datasets [TETC20]. Other authors even went as far as ignoring an entire hour after each attack to avoid the phase during which the physical process might stabilize after an attack [EMKJ20].

3.4.3.3 Results

We begin with analyzing the misuse-based IIDSs on the Morris-Gas dataset in Figure 3.8a. Here, all metrics coherently judge the IIDSs' performance, i.e., RF is



(a) Morris-Gas **(b)** SWaT

Figure 3.8 While all metrics rate IIDSs' performance consistently on the Morris-Gas dataset (Figure 3.8a), they provide a more imprecise pictures on the SWaT dataset (Figure 3.8b).

perceived as strictly better than the SVM. Note that a lower score in FPA is better. Interestingly, even the $F_{0.1}$ score is similar to the original F_1 score, likely due to the high amount of malicious samples (22 %) in this dataset. The time-aware variants draw an identical picture here, but since the attack instances of the Morris-Gas dataset correspond to manipulations of individual network packets, temporal effects are minimal. Overall, the considered metrics coherently judge the IIDSs' performance on the Morris-Gas dataset.

The picture changes for the SWaT dataset comprising of time series of physical states. We additionally depict the raw alert emitted by the IIDSs over time in Figure 3.9. First of all, as depicted in Figure 3.8b, accuracy gives all IIDSs the best performance (well above 0.8). Yet, in comparison to all other metrics, accuracy seems to overestimate their capabilities. We attribute this to SWaT's composition comprising to 12 % of attacks (which is more realistic than Morris-Gas with 22 % as attacks are rare in practice). An IIDS that emits no alarms at all would score an accuracy of 0.88 already. Thus, accuracy may not be best suited in this scenario.

Regarding precision and recall, we observe ambiguity. Seq2SeqNN falls far behind in recall, which we attribute to a single long attack in SWaT (accounting for 63 % of all attack samples) being missed by the approach (cf. Figure 3.9). Besides this attack, Seq2SeqNN achieves decent scores as it correctly detects most of the other attacks (cf. Detected Scenarios). Therefore, point-based metrics overvalue this single attack, i.e., no obvious relation exists between the attack's duration and its severity that would justify this effect. In contrast to the Morris-Gas dataset, the $F_{0.1}$ score clearly favors IIDSs with higher precision, and thus Seq2SeqNN performs slightly better here than in F1.

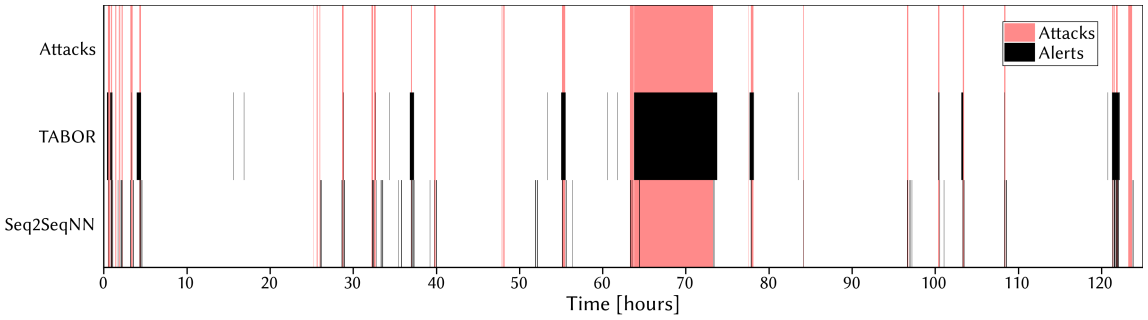


Figure 3.9 Visualizing alerts side-by-side provides an in-depth view of IIDSs' distinct alerting behavior. For example, the performance of Seq2SeqNN in recall (cf. Figure 3.8) can be attributed to a single undetected and prolonged attack of the SWaT dataset. Note that alerts have been extended to a minimum width of 1 minute for visibility.

Time-aware metrics promise to solve the issues of point-based metrics [HYKM22]. In our practical study, the weakness of Seq2SeqNN in recall missing a single long attack is accounted for resulting in a similar score across eTaP and eTaR. Contradicting the point-based metrics, TABOR is now perceived as a much worse approach across all time-aware metrics probably because detected scenarios gains in importance as the metric normalizes the length of the individual attacks. These results are in line with Fung et al. [FSL+22], finding that time-aware metrics are preferable for time-aware IIDSs and point-based scores may be misleading.

Lastly, and relevant for practical deployment, IIDSs with many false positives are unsuitable [Eta17], and thus identifying those in evaluations is crucial. In our experiment, the best IIDS on the Morris-Gas dataset (RF) has also the lowest number of FPA. Yet this picture changes for SWaT where Seq2SeqNN, despite detecting the most attacks, features the most FPA. Since FPA resembles a absolute number, it can hardly be set in relation to the other metrics.

Takeaway. Given our experiment, it is unlikely that a single metric exists that catches all industrial operators' goals, e.g., preferring few false alarms over detected attacks. IIDSs should, therefore, be evaluated with diverse metrics to highlight their capabilities, as cherry-picking metrics may yield misleading results. Lastly, visual comparisons exhibit an added value to evaluations too.

3.4.4 Conclusion

Set out to further investigate potential reasons that inhibit IIDS research's tight consolidation, we analyzed the usage of datasets and metrics in this domain. One positive aspect regarding the datasets is that this research domain nowadays has access to a growing variety of public datasets and authors steadily make more and more use of them. Yet, publications restrict themselves by considering just 1.3 datasets on average, which might be a reason for their limited scope as perceived before (cf. Section 3.3). Consequently, research fails to effectively widen the scope to the available evaluation tools and rather introduces biases by focusing on a few specific niches. At the same time, IIDS research dominantly considers established point-based metrics.

However, their expressiveness is limited, and thus, multiple metrics should be used in parallel and optimally in conjunction with visual comparisons.

3.5 Understanding the Inefficiencies in IIDS Research

While we observe potential to improve the expressiveness of evaluations (cf. Section 3.4), we suspect that there exist deeper issues prohibiting a unified research landscape that are more rooted in the nature and particularities of the industrial domain itself. One key attribute of ICSs is their inherent diversity regarding domains, vendors, technologies, actors, and protocols (cf. Section 1.1). This is also reflected in the publications that stem from a mix of industrial experts, authors of the machine learning domain, and many more (cf. Section 3.1).

To understand which implications this mix has on the research landscape, we perform a case-study on a smaller, selected set of the literature. By reducing the number of considered publications, we are therefore able to look deeper into the proposed approaches. To this end, we begin with an overview of the proposed detection methodologies in Section 3.5.1.

After having established an understanding of the composition of the underlying detection algorithms, we focus on understanding the issues that arise from these observations. First, in Section 3.5.2, we suspect the complexity of research to be a factor limiting the effective use of IIDSs in research and practice. Finally, we look at the domains and protocols these IIDSs are designed for in Section 3.5.3. Thereby, it becomes apparent that research takes place in isolated silos with little exchange, which makes it unnecessarily inefficient and likely prohibits a tighter consolidation.

3.5.1 Overview on Detection Methodologies

Following our analyses of the overall IIDS research landscape, datasets, and metrics, we now focus on the detection methodologies. To this end, we consider a smaller set of 61 publications for which we extract additional information. We are especially concerned with the detection methodology, but also now the domain an IIDS was proposed for and the industrial protocols it was tested against.

After investigating the individual publications in Table 3.4, it became apparent that the types of publications can be divided into two broad categories. *Communication-based* approaches, discussed in Section 3.5.1.1, incorporate characteristics of network traffic while *process state-aware* approaches (cf. Section 3.5.1.2) leverage the (physical) state of an industrial process for their detection, which coincides with our previous observation on the two major dataset types (cf. also Section 2.2.3).

3.5.1.1 Communication-Based Intrusion Detection

Communication-based IIDSs take a view from a network perspective by analyzing (industrial) network packets one-by-one. Since ICS's communication is largely

		Publication(s)	Year	Detection Methodology	Open Source	Public Dataset	Eval. Scenarios	Compares to	Domain	Protocol(s)
Communication-based Approaches	Timings (5)	Valdes et al. [VC09]	2009	Flow Periodicity	○	○	1	0	Synthetic Setup	Modbus
		Barbosa et al. [BSP12]	2012	Periodicity	○	○	1	0	Water Treatment	–
		Ponomarev et al. [PA16]	2016	Telemetry	○	○	1	0	Synthetic Setup	Modbus
		Lin et al. [LNTA17]	2017	Inter-arrival Time	○	◐	3	0	Power Grid	Modbus, IEC-104, S7
		Lin et al. [LNT19]	2019	Inter-arrival Time	○	◐	1	0	Power Grid	IEC-104
	Sequences (6)	Goldenberg et al. [GW13]	2013	DFA	○	○	1	0	Manufacturing	Modbus
		Yoon et al. [YC14]	2014	PST	○	○	2	0	Synthetic Setup	Modbus
		Caselli et al. [CZK15, CZPK15]	2015	DTMC	○	○	1	0	Water Treatment	Modbus
		Ferling et al. [FCCR18]	2018	DTMC	●	○	1	0	Gas	IEC-104
		Lin et al. [LNT18]	2018	PST	◐	○	2	0	Power Grid	IEC-104
		Yun et al. [YHL+18]	2018	Nearest Neighbor	○	○	1	0	–	–
	Classifiers (7)	Shang et al. [SCW+16]	2016	SVM	○	○	1	0	Chemical	Modbus
		Feng et al. [FLC17]	2017	LSTM	○	$\bullet^{M}$	1	1	Gas	Modbus
		Perez et al. [PASE18]	2018	SVM, RF, BLSTM	●	$\bullet^{M}$	1	0	Gas	Modbus
		Anton et al. [ASS19]	2019	RF, SVM	○	$\bullet^{M}$	1	0	Gas	Modbus
		Chu et al. [CLL19]	2019	NN	○	$\bullet^{M}$	1	3	Gas	Modbus
		Maglaras et al. [MJ14]	2019	OCVM	○	○	1	0	–	Modbus
		DIDEROT [RSE+20]	2020	Decision Tree, DNN	○	○	1	0	Power Grid	DNP3
Process State-aware Approaches	Critical States (10)	Carcano et al. [CCG+11]	2011	Language	○	○	1	0	Power Grid	Modbus
		Almalawi et al. [AYT+14]	2014	Outlier detection	○	◐	3	0	Water Treatment	Modbus
		Kiss et al. [KGH15]	2015	GMM	○	$\bullet^{T}$	1	0	Chemical	–
		Pan et al. [PMA15]	2015	Common Paths	○	○	1	0	Power Grid	Modbus, IEEE C37.118
		SysDetect [KS15]	2015	Frequent Itemsets	○	○	1	0	Manufacturing	–
		Kong et al. [KJB17, JKB14, KJM+14]	2016	Temporal logic	◐	○	2	0	Logistics	–
		Adepu et al. [AM16a]	2016	Invariants	○	$\bullet^{S}$	1	0	Water Treatment	EtherNet/IP
		Feng et al. [FPMC19]	2019	Invariants	◐	$\bullet^{S}_{W}$	2	2	Water Treatment, Water Distribution	EtherNet/IP
		Monzer et al. [MBF19]	2019	Rules	○	○	1	0	Water Treatment	Modbus
		Das et al. [DAZ20]	2020	Data Analysis	○	$\bullet^{S}$	1	0	Water Treatment	EtherNet/IP
	Behavior Prediction (11)	Hadžiosmanović et al. [HSZH14]	2014	Autoregression	◐	○	2	0	Water Treatment	Modbus
		Caselli et al. [CZK15]	2015	DTMC	○	○	1	0	Water Treatment	EtherNet/IP
		Ahmed et al. [AMR17]	2017	Kalman Filter	○	○	1	0	Water Distribution	–
		PASAD [AIA18, AA20]	2018	PCA	●	◐$^{S}_{T}$	3	1	Water Treatment, Chemical	Modbus, EtherNet/IP
		Choi et al. [CLA+18]	2018	Control Invariants	●	○	1	0	Logistics	–
		Myers et al. [MSRF18]	2018	Petri-nets	○	●	1	0	Manufacturing	S7
		TABOR [LAVM18]	2018	TA, BN	○	$\bullet^{S}$	1	2	Water Treatment	EtherNet/IP
		Anton et al. [ALGS19]	2019	Matrix Profiles	○	$\bullet^{S}$	1	1	Water Treatment	EtherNet/IP
		HybTester [CZ19]	2019	Hybrid-Automata	○	$\bullet^{S}$	1	0	Water Treatment	EtherNet/IP
		Denque Anton [AS20]	2020	Matrix Profiles	○	$\bullet^{S}$	1	0	Water Treatment	EtherNet/IP
		SAVIOR [QGS+20]	2020	Physical Invariants	●	○	2	1	Logistics	MAVLink
	Correlations (6)	Krotofil et al. [KLG15]	2015	Entropy Analysis	○	◐^{T}	1	0	Chemical	–
		Alippi et al. [ANR17]	2016	Hidden Markov Model	◐	○	3	0	Synthetic Setup	–
		Aggarwal et al. [AKPI18]	2018	Hidden Markov Model	○	○	2	0	Medical, Logistics	–
		Hau et al. [HL19]	2019	Statistics	○	◐	1	0	Medical	–
		NoiSense [AMO20]	2020	Noise Fingerprinting	○	$\bullet^{S}_{W}$	2	0	Water Treatment, Water Distribution	EtherNet/IP
		ProcessSkew [APQ+20, AZM18]	2020	Noise Fingerprinting	○	$\bullet^{S}$	1	0	Water Treatment	EtherNet/IP
	Neural Networks (9)	Kravchik et al. [KS18]	2018	Neural Networks	○	●	1	3	Water Treatment	EtherNet/IP
		Shalyga et al. [SFL18]	2018	Neural Networks	○	$\bullet^{S}$	1	1	Water Treatment	EtherNet/IP
		AADS [ADS20]	2019	DNN	○	$\bullet^{S}$	1	3	Water Treatment	EtherNet/IP
		Seq2SeqNN [KYK20]	2019	Neural Networks	●	$\bullet^{S}$	1	1	Water Treatment	EtherNet/IP
		Elnour et al. [EMK20]	2020	NN+Isolation Forest	○	$\bullet^{S}$	1	5	Water Treatment	EtherNet/IP
		MADICS [PFH+20]	2020	DNN	○	$\bullet^{S}$	1	7	Water Treatment	EtherNet/IP
		Wang et al. [WWLQ20]	2020	Autoencoder	○	$\bullet^{S}$	1	2	Water Treatment	EtherNet/IP
		Xie et al. [XWWT20]	2020	DNN	○	$\bullet^{S}$	1	3	Water Treatment	EtherNet/IP
		Neshenko et al. [NBHF21]	2021	Neural Network	○	$\bullet^{S}$	1	4	Water Treatment	EtherNet/IP
	Classifiers (7)	Nader et al. [NHB14]	2014	SVDD, KPCA	◐	○	2	0	Gas, Water Treatment	–
		Junejo et al. [JG16]	2016	Machine learning	○	$\bullet^{S}$	1	0	Water Treatment	EtherNet/IP
		Inoue et al. [IYC+17]	2017	SVM, DNN	○	$\bullet^{S}$	1	0	Water Treatment	EtherNet/IP
		Chen et al. [CPS18]	2018	SVM	◐	$\bullet^{S}$	1	0	Water Treatment	EtherNet/IP
		Anton et al. [ALGS19]	2019	OCVM, Isolation Forest	○	$\bullet^{S}$	1	0	Water Treatment	EtherNet/IP
		FALCON [SMRM20]	2020	LSTM+ML	○	$\bullet^{S}$	1	2	Water Treatment	EtherNet/IP
		Elnour et al. [EMKJ20]	2020	Isolation Forest	○	$\bullet^{S}_{W}$	2	6	Water Treatment	EtherNet/IP

Power Grid · Water Treatment · Gas · Manufacturing · Medical · Chemical · Water Distribution · Logistics · Synthetic Setup

$\bullet^{S}$: SWaT [GAJM16] $\bullet^{W}$: WADI [APM17] $\bullet^{T}$: TEP [DV93] $\bullet^{M}$: Morris-Gas [MTT15]

Table 3.4 Our survey of 61 intrusion detection approaches confirms the heterogeneity across the industrial research landscape. Overall, IIDSs are mostly developed in isolated silos and seldomly compare to existing research. We even observe similar detection methodologies across different protocols and domains, indicating an enormous potential to transfer achievements to a broader scale of industries.

based on repetitive and predictable machine-to-machine communication, different streams of research propose to leverage the inherent patterns found across industrial protocols. We identify three categories of how regularities caused by the periodic distribution of, e.g., sensor measurements, can be utilized for anomaly detection.

First, irregularities in *timings* can indicate potential malicious activity. For example, if a measurement is sent in regular intervals (every second), any deviation from that pattern may hint at an anomaly due to an attacker dropping or ingesting additional packets. Secondly, the order of message *sequences* exposes repeating patterns that can be learned to identify expected normal communication. If a PLC is programmed to first query different sensor values, then calculate the next actions, and finally command the actuators, this behavior can be monitored on the network. An attacker ingesting additional commands violates this predictable cycle. Finally, a class of *classifier*-based approaches categorize each packet as benign or anomalous based on a set of predefined features. Such classifiers are usually trained on a dataset of known benign and malicious behavior to tell the different traffic classes apart.

3.5.1.2 Process-based Intrusion Detection

In contrast to the communication-based approaches, process state-aware IIDSs are based on (time-)series of industrial system's physical states. A state comprises the combined process information of sensors and actuators aggregated over multiple packets to assess whether a process' physical state indicates an anomaly over time. To identify attacks, process state-aware approaches leverage the repetitiveness and predictability of physical processes, e.g., PLCs controlling a pump to keep a water reservoir's fill level within certain bounds [FPMC19].

While process state-aware IIDSs operate on a more restricted set of information than communication-based IIDSs, their detection methodologies show more diversity, which we broadly classify along five categories. The first category of approaches defines *critical states* of an industrial system, e.g., through externally provided system states by domain experts, and raises an alarm if such a state is reached. Complementary, the second category attempts to *predict* future states based on past observations, raising an alarm if the behavior significantly deviates from the prediction. The third category takes a more local view by investigating *correlations* between an individual or a small set of sensor readings over time, searching for indicators of anomalous activities, such as outliers from learned clusters. Research likewise considers various forms of *neural networks* often tasked to predict the next observed process data similar to behavior prediction. Finally, as for network-based approaches, a category of approaches leverages *classifiers*.

3.5.2 The Complexity Issue of IIDS Research

Given the promise of IIDSs to offer an retrofittable solution to secure industrial networks, the research landscape around industrial intrusion detection has experienced attention across all industrial domains (cf. Section 3.3.1). Surprisingly, the

current state-of-the-art is governed by all kinds of machine learning, which comes at the cost of several complexity issues: (i) The proposed detection methodologies are usually complex with little justification, which then leads to (ii) computational complexity during training or execution of the IIDS and (iii) incomprehensive alarms for ICS operators to investigate the suspected attack. Additionally, difficulties in (iv) deployments can be expected if experts fail to setup an IIDS. Lastly, research's evaluations fail to work out the need of the inherent complexity as (v) the attacks in common evaluation datasets are often non-stealthy and supposedly easy to detect. Based on the detailed data gathered from the narrower set of publications in Table 3.4, in the following, we lay out the reasons for each of the five complexity issues in detail from Section 3.5.2.1 to Section 3.5.2.5.

3.5.2.1 Complex Detection Methodologies

Table 3.4 structured the considered approaches alongside their detection methodologies into different classes. Neural networks and classifiers make up 38% of publications, which usually rely on complex methods for detection. Adding to this, a class of behavior prediction approaches (18%) combines IIDSs with novel and specialized methods and shows that the research community has not settled on a preferred direction in this confined domain. Also, leveraging automata theory, such as Hybrid-automata, or various statistical methods attribute to about 25% of works in this field.

While we occasionally observed related work with more straightforward methods, e.g., inter-arrival times for the communication-based category or complex IIDSs complemented with simpler out-of-bound checks [LAVM18], to the best of our knowledge, such simple approaches have not yet been evaluated in isolation. Thus, especially for the most used datasets, like SWaT, it is unknown whether the contained attacks are of such difficulty that these complex detection algorithms are justified. Consequently, in the following, we detail our survey's findings by focusing on related issues resulting from focusing on complexity rather than simplicity.

3.5.2.2 Computational Complexity

The implementation of any detection methodology should be quick enough to be deployable in real-time environments. More precisely, detection should not be significantly delayed by processing overhead. Even if adequate hardware is available, requirements such as GPUs in several publications [KS18, ADS20, KYK20, WWLQ20, XWWT20, SMRM20] can challenge deployability. These IIDSs leverage clusters of up to twelve CPUs [KS18] or six GPUs [KYK20, SMRM20]. Although training neural networks may be more resource-intensive than executing them, they still have to be lightweight enough to be processed by industrial hardware, e.g., on programmable network switches (as commonly done for traditional IDSs) or resource-constrained and specialized PLCs.

3.5.2.3 Incomprehensible Alarms

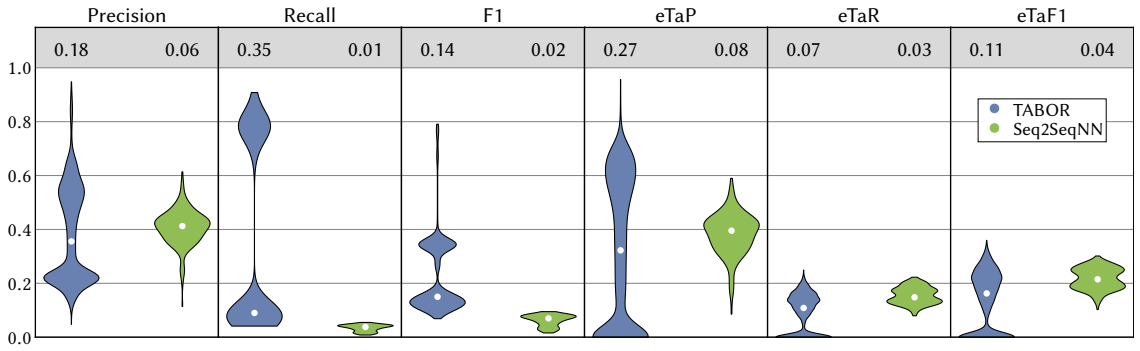
After an IIDS has indicated a potential threat by emitting an alarm, further (manual) investigation is necessary to find and mitigate its cause. This could include determining the affected part of the process and isolating it from the rest of the network. While a few IIDSs' alarms are descriptive [LAVM18, FPMC19] and can ease in-depth investigation, such works are rare in our survey. In most cases, the decisions of machine learning classifiers are incomprehensible or only accessible to highly-trained specialists. For instance, when feeding a normalized vector of process values into an artificial neural network [KYK20], it is not clear why the vector would be classified as benign or malicious, preventing a timely tracing of an alarm back to its source. In case of an attack, actions must be undertaken quickly, which is prohibited if the source of the alarm is unclear. Thus, IIDSs with accurate detection performance are impractical if the alert reason can not be traced back in time.

3.5.2.4 Difficult Deployments

Training an IIDS usually requires configuring hyperparameters, which relies on experts knowing the details of a model. As scientific reproduction studies indicate [ET22], even IDS experts fail when configuring an already published approach (with source code available) to match the original publication's results.

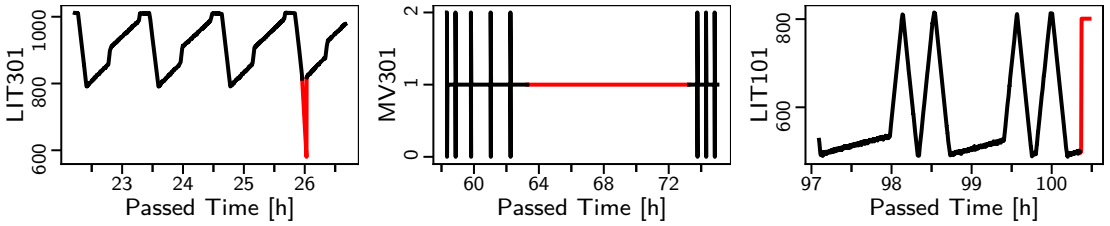
To highlight this issue, we conduct a small practical experiment along the previously analyzed IIDSs TABOR [LAVM18] and Seq2SeqNN [KYK20] again on the SWaT dataset (cf. Section 3.4.3). In this experiment, we repetitively train and evaluate the IIDSs each time with new hyperparameters. For nominal and ordinal hyperparameters, we enumerated all possible values. For rational numbers, we had to define a custom range based on our understanding of the proposed system. During their definition, we took special care that the values proposed in the original publications are contained in our analyzed ranges. Note that for Seq2SeqNN, we examine only one of its six neural networks to reduce the training overhead. Therefore, the results of this experiment are not comparable to the maximum detection performance of the entire model as shown in Figure 3.8b.

At first glance, we observe that hyperparameters affect the performance of IIDSs (cf. Figure 3.10). For example, considering the F1 score of TABOR, the performance varies between 0.79 at best and 0.07 at worst, which implies that, depending on the chosen configuration, the approach performs close to optimal or is inapplicable. But looking at the entire distribution, it becomes apparent that high values are rather outliers as the median performance (white dot) is still low at 0.15. Thus, performance penalties can be expected for TABOR if not parameterized correctly. In contrast, the distribution of Seq2SeqNN is much flatter. There, unlike for TABOR, it is more likely to (randomly) hit a good-performing configuration. Therefore, these effects can explain previously reported problems from related work, e.g., failed reproducibility studies or deploying such approaches in practice [SP10, Eta17, AMM20]. I.e., Erba et al. [ET22] tried to reproduce an IIDS and had trouble finding the hyperparameters to match the publication's results.



The white dot represents the median and the numbers on top are the standard deviation of each distribution.

Figure 3.10 An IIDS's performance depends on an optimal choice of hyperparameter. E.g., obtaining a good configuration for TABOR is more challenging compared to Seq2SeqNN. Thus, judging the expectable performance of an IIDS by a single number can be misleading.



(a) Over- or undershooting **(b)** Flat line **(c)** Steep in- or declines

Figure 3.11 Manual investigations of the SWaT dataset reveal that attacks (red) commonly used for evaluating IIDSs in research are often not stealthy and thus potentially easy to detect.

Publishing IIDSs with fine-tuned hyperparameters clearly showcases the maximal performance a proposed approach can achieve. However, such results carry the risk of misrepresenting how good a system may perform in a real deployment and may prevent fair comparisons of approaches. Consequently, we warn that judging an IIDS's performance by a single configuration, as done currently throughout the literature, can be misleading.

3.5.2.5 Non-Stealthy Attacks

One argument to justify complexity is the goal to unveil stealthy attacks [UGC+16]. While approaches evaluated on specifically crafted datasets exist [AIA18], a closer inspection of the commonly used datasets reveals that many attacks are not difficult to detect. As shown in Figure 3.11 for SWaT, overshooting or undershooting regular process values, remaining for too long in a single state (flat line), or unusual steep inclines or declines do not necessarily require complex detection mechanisms. Still, this is not a flaw of the datasets, as recent real-world incidents prove. For example, the sodium hydroxide concentration of a water treatment plant in Florida was set to hazardous levels (about 100 times increased) [Vas23], constituting a potentially easily detectable real anomaly. Thus, the inherent complexity of IIDSs in research does not match the complexity of attacks in respective datasets and real-world inci-

dents. Consequentially, it is still unknown whether these are detectable with simple detection methods too.

Takeaway. Contradicting their initial promise of an "easily" retrofittable solution, research on industrial intrusion detection is driven by complex approaches. At the same time attacks in evaluation datasets and examples from real-world incidents seem to be detectable straightforward. Given further drawbacks, such as incomprehensible alarms, computational complexity, and difficult deployment, it is unclear whether this complexity is necessary without comparing it to a proper baseline or having tried simpler approaches first.

3.5.3 Silos of Protocol and Domain-dependent IIDS Research

IIDS research is characterized by an inherent heterogeneity across deployment domains, industrial protocols, and architectures (cf. Section 1.1), which we suspect to be the leading cause for the observed incoherence, cf. Section 3.3.2. While it would make sense to transfer the achievements of IIDS research conducted for one domain or protocol to another, we observed that most published approaches are considered for a single use case, i.e., evaluate just one dataset (cf. Section 3.4.1). Authors even state that their "experiment is limited to the SWaT testbed. [... Thus], it could be possible that [their] results do not generalize to other CPSs" [IYC+17]. At the same time, existing surveys identify the need for generalizing IIDSs [ET22], but they do not investigate whether existing work shows potential in this regard. This problem is already known for traditional IDSs [PB18] but is even more severe in industrial settings, where systems expose narrow yet domain-specific behavior [TETC20]. To grasp this phenomenon in greater detail, we take a look at the publications from Table 3.4, especially with respect to the domains these IIDSs are proposed for.

Our analysis of 61 IIDSs confirms our assumption that the IIDS research landscape suffers from heterogeneity, and the overall progress in this research field is thus slowed down. One publication considers a median of 1.0 domains (1.05 on average) for their approach, limiting the impact of their results. At the same time, the underlying fundamental principles remain similar enough to be worth considering elsewhere. As an example, detecting anomalies with a Discrete-time Markov Chain (DTMC) has been proposed individually by Caselli et al. [CZPK15, CZK15] for the water domain and Ferling et al. [FCCR18] for a gas distribution scenario. Such examples also appear in the class of process state-aware IIDSs where, e.g., hidden Markov models were analyzed in synthetic setups, medical, and logistical applications by independent researchers [ANR17, AKPI18]. At the same time, we often read claims such as the following: "In the future, the present implementation of [the] LAD [IIDS] may be extended to design IDSs/ADSs for other application domains of ICSs like power, and water distribution systems" [DAZ20]. Consequently, limiting the applicability of an IIDS to a single domain risks that research is made twice instead of building on top of each other affecting the overall effectiveness of research.

One cause for this situation could be the dependence on just a few of the many industrial protocols. As different industries and vendors each leverage specific protocols

(cf. Section 1.1.2), it is no surprise that most IIDSs are implemented and designed for just a single protocol, cf. Table 3.4. This problem even persists when the underlying process remains the same: For example, in electric power distribution, IIDSs developed for IEC-104 protocol [LNTA17, FCCR18, LNT18, LNT19] (dominant in Europe) cannot be applied to networks relying on DNP3 [FCD+10, RSE+20] (the de-facto standard in the US), and vice versa. While implementing another protocol adds little research value, this part would be required to bring IIDSs into practice. As a rare exception, Lin et al. evaluate their IIDS [LNTA17], checking for irregular timings between similar messages, for three industrial protocols. Still, they focus on a single domain (power grids) and thus do not examine transferability across domains. Again, plans "to extend the analysis capabilities to the other field communication protocols" [FCD+10] are discussed. However they are not reflected in the evaluations of the publications.

While authors claim applicability of their IIDS (without proof) [FCD+10, IYC+17, KYK20, DAZ20], the generalizability of proposed IIDSs to new protocols or transferability across domains remains mostly unexplored [ET22]. Optimally, IIDSs are designed to function across diverse industrial domains with diverging communication protocols to facilitate the protection for several use cases simultaneously. Despite these challenges, we deem generalizability to be theoretically achievable as visible by other projects like PLC4X [Apa20] or StreamPipes [WZR20, ZWSR20] trying to mediate between different industrial protocols already. However, such solutions have not been considered for intrusion detection, and thus, research resides inside its narrow silos lacking essential exchange between them.

Takeaway. The IIDS research landscape suffers from the heterogeneity of ICS in domains and protocols. Consequently, the various research streams in the area of IIDSs are disjoint. As a result, the overall research field is slowed down, as new IIDSs are designed for niches without improving on previous research and are biased to datasets from specific scenarios. It thus remains unknown to which extent the advancements in the detection of cyberattacks on industrial systems can or cannot be generalized and transferred from their isolated niches to a broader scale. Thus, they could sustainably increase security for many industrial networks.

3.6 Research Issues and Contributions

The potential of IIDSs to combat rising threats from cyberattacks against industrial networks is indisputable. Unsurprisingly, our systematic analysis of literature shows an unbroken interest in this research field (40.9 % average yearly increase between 2013 and 2021). Overall, at least 609 publications invest efforts in proposing IIDSs, which are complemented by further works on creating datasets, designing evaluation metrics as well as surveys and meta-analyses (cf. Section 3.1). However, our SMS also unveils and quantifies flaws in this field that hamper scientific progress.

The goal of this dissertation is to address and make progress on these persisting flaws ultimately making IIDSs more accessible for ICSs. To this end, the following

Section 3.6.1 synthesizes the insights obtained from our SMS into five issues persisting in the IIDS research landscape. These issues I1–I5 are then addressed by the subsequent contributions of this dissertation. Section 3.6.2 presents an overview of the contributions and how these interact with the issues identified previously.

3.6.1 Summarizing the Issues Inhibiting Accessible IIDS Research

Our systematic analysis of the IIDS research field reveals that the current state-of-the-art has inefficiencies, eventually slowing down the overall progress in securing industrial deployments. Our SMS, covering the body of literature until 2021, enables quantifying these inefficiencies in contrast to previous (meta) surveys and experiments on a usually narrower scale (cf. Section 3.1). More precisely, we identify five issues (I1–I5) prevalent in IIDS research and summarize them along the results from our SMS in the following.

► I1: IIDS research takes place in isolated silos due to its inherent protocol- and domain-dependence, which slows down the collective achievements.

One limiting factor of IIDS research is its isolated view of niche parts of the ICS landscape. Instead of proposing and evaluating generic detection methodologies applicable to a wide range of industrial scenarios, authors rather consider usually just a single use case. This observation is also reflected in the low number of considered domains in evaluations, with 1.05 domains on average (cf. Section 3.5.3). Similarly, the number of considered industrial protocols per IIDS is at most one, with only rare exceptions. For the few publications that evaluate multiple datasets (16.4 %), these datasets usually stem from the same origin (cf. Table 3.3) and thus are still confined to a single scenario (dataset).

This narrow focus of research inevitably led to work being done twice for different ICS domains (cf. Section 3.5.3). However, this demonstrates at the same time that IIDSs do not have to be protocol- and domain-dependent per se. Summarizing these factors, it becomes apparent that research neither covers the diversity of industrial domains nor communication protocols. Consequently, it remains unclear whether IIDSs are applicable outside the scenario they have been evaluated in, making deployments risky and requiring repeated research efforts for different scenarios.

► I2: Evaluations could be improved since datasets are leveraged scarcely and prominent metrics risk being inexpressive, leading to a low level of comparability.

We identify a lack of diversity in the datasets used for evaluations. Regarding the utilization of datasets, we find that IIDSs are evaluated on 1.3 datasets on average (cf. Figure 3.5). Notably, the majority (509 publications) evaluate their IIDSs on one dataset despite a significant selection of datasets being publicly available (we identified 35 public datasets in our survey). Since there exists this gap between available and utilized datasets, this raises the question of why many datasets are only used rarely. Possible reasons include datasets being too difficult to use, e.g., requiring in-depth knowledge of a specific industrial protocol, adaptions to a different data format, or simply not widely known among researchers.

At the same time, metrics used in evaluations and comparisons pose ambiguity regarding the actual detection performance. As our practical experiments show, not a single metric can describe all aspects of an IIDS, and visual comparisons can disprove, e.g., seeming promising IIDSs. Additionally, the Top four point-based metrics, accuracy, precision, recall, and F1 score, making up 59.1 % of the metric usage, do not accurately capture the detection performance in time-aware scenarios [FSL+22]. They are biased towards the detection of long-lasting attacks (cf. Section 3.4.3). While new metrics [GS14, LA15, TLZ+18, HYKK19, HNR22, HYKM22] are designed that address these issues, these metrics are rarely used in evaluations (only 16 publications).

Lastly, evaluations do not capitalize on the potential for comparisons among the body of existing research, which was already criticized by Giraldo et al. [GUC+18]. On average, an IIDS is compared with 0.5 other proposals, while in theory, authors could compare an IIDS to an average of 6.0 other approaches sharing at least one common dataset and metric (cf. Figure 3.4). This situation is aggravated by the sparse commitment to publish artifacts (cf. Section 3.3.4). This situation leaves researchers no choice other than to reproduce others' works, e.g., to conduct comparability studies — a non-trivial task that is prone to failure [ET22]. Nonetheless, proper comparisons are essential to better understand if and how a (novel) IIDS improves upon existing work and thus collectively move the research field forward.

▶ I3: Existing approaches seem overly complex without proper justification.

The powerful IIDS methodologies proposed by researchers yield promising detection performances, however, at the cost of complexity, requiring resource-intensive operations or complicating their use and comprehensibility for ICS operators (cf. Section 3.5.2). Furthermore, the alarms raised, e.g., by artificial neural networks, are often not explainable, making it challenging to derive concrete actions for mitigating attacks [Eta17]. Meanwhile, we observe that attacks considered in prominent datasets (cf. Figure 3.11) lead to apparent deviations from normal operations. Therefore, we question the need for complexity and ask whether IIDSs have to be complex to reliably detect attacks on industrial systems.

▶ I4: Transferability of IIDSs does not take place on a large scale.

Despite often being claimed as future work that given approaches would be transferred to other industrial domains as well (cf. Section 3.5.3), this property is not analyzed in particular until now. For the reasons laid out in I1, examining transferability to other domains is even more complicated for researchers. Thus, research has to start from scratch for a new domain or use case, leading to work being done twice. Instead, first transferring existing IIDSs to the new use case first and examining their effectiveness before ultimately developing novel detection methods should be a more resource-saving approach by researchers. Only then can remaining ineffectiveness, e.g., an inability to detect a certain type of attack, become evident and addressed with custom solutions to fill just the gaps.

▶ I5: Research does not consider which IIDSs should be deployed.

As the main research body is concerned with publishing newer and better IIDSs, in the end, the operators of an ICS have to select a specific IIDS from the ones

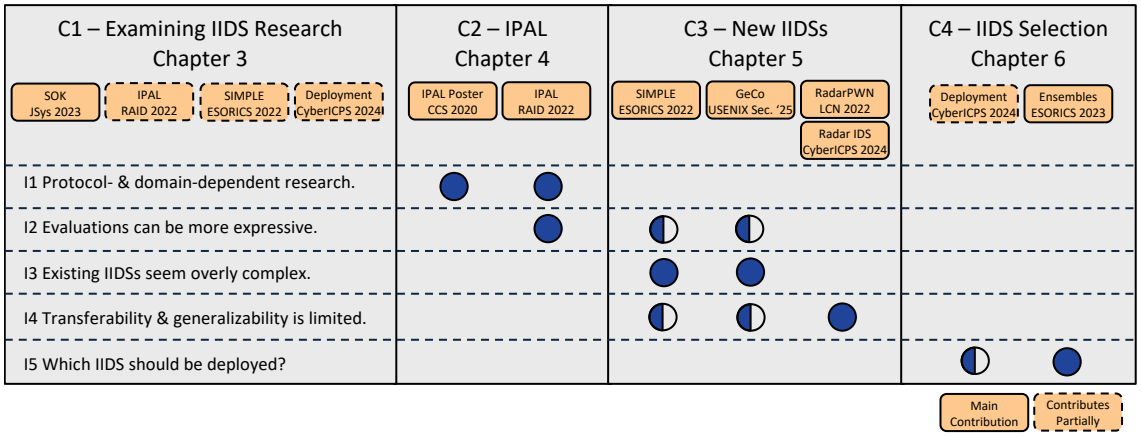


Figure 3.12 During the examination of the literature on IIDS research, the first contribution of this dissertation (C1), we identified five issues (I1–I5) that could be optimized. Setting out to address these issues, this dissertation makes three further contributions (C2–C4) in the following Chapter 4 to Chapter 6.

available that best fits their use case. Given the huge number of yearly publications and different types of detection methodologies, this choice can be challenging as selecting a suboptimal IIDS could diminish the overall cybersecurity of an ICS if it does not perform up to its claimed performance. Thus, the question of which IIDS should be deployed in a given scenario is of importance and remains currently unanswered in the literature.

3.6.2 Overview on the Contributions of this Dissertation

Having derived five issues inhibiting the efficient progress of IIDS research (I1–I5), we now present an overview of the contributions of this dissertation (C1–C4) and how we address these issues. Figure 3.12 provides an overview of the next three chapters and how each publication maps to the individual contributions. The blue circles mark to which extent a publication makes progress with respect to a given issue. A summary of each contribution is provided below.

▶ C1: Unraveling the current state of IIDS research.

This Chapter 3 has analyzed the IIDS research landscape with a SMS and worked out its inefficiencies resembling the first contribution (C1) to this dissertation. Moreover, this chapter examines the current state of IIDS research and unveils the factors that limit its efficiency. Our analysis results in five issues I1–I5, as summarized in Section 3.6.1. To do justice to these issues and thereby advance the current workflows of IIDS research and help making IIDSs more accessible for ICSs, we address them in three subsequent contributions C2–C4 in the remainder of the dissertation.

▶ C2: Inventing the Industrial Protocol Abstraction Layer (IPAL).

As we observed in our SMS, current efforts by the research community on intrusion detection methods mostly focus on specific domains and protocols, leading to a research landscape that is broken up into isolated silos (I1). Thus, existing approaches

cannot be applied to other industries that would equally benefit from their powerful detection capabilities. To better understand this issue, Chapter 4 first analyzes a selection of 61 IIDSs in more detail and we find no fundamental reason for their narrow focus. Although they are often coupled to specific industrial protocols in practice, many approaches could transfer to new industrial scenarios in theory. To unlock this potential, we propose the idea of protocol-independence realized through IPAL, our Industrial Protocol Abstraction Layer, to decouple intrusion detection from domain-specific industrial protocols. To prove IPAL's correctness and feasibility, we conduct a reproducibility study of IIDSs from related work showing that they can operate equivalently well with this new data format compared to operating directly on network data. Afterward, we showcase IPAL's unique benefits. To this end, we study the generalizability of existing approaches to new protocols and conclude that they are not restricted to specific protocols and can perform outside their restricted silos. The idea of protocol-independent intrusion detection realized with IPAL, our evaluation tooling, and provided datasets, therefore, contribute to unifying the future research landscape on intrusion detection. Furthermore, it proves valuable in facilitating further research (I2) as IPAL became the driving research platform to enable the subsequent contributions of this dissertation.

▶ C3: New IIDS concepts to tackle complexity and transferability.

Besides generalizability, various streams of IIDS research rely on advanced and complex approaches, e.g., artificial neural networks, thus achieving allegedly high detection rates. However, as we observed in Section 3.5.2, their inherent complexity comes with undesired properties such as complex detection methodologies, computational overhead, incomprehensive alarms, or difficulties finding appropriate hyperparameters. These issues can limit their broad and easy applicability to other domains. On the other hand, the transferability of IIDSs to other domains not initially thought of remains underexplored as of now. Consequentially, we ask whether industrial intrusion detection indeed has to be that complex or can be more simpler instead, thereby even transferring better to other domains. To answer these questions, we propose our designs of three novel IIDSs in Chapter 5. First, an assessment of minimalist detection methodologies, designed with simplicity in mind (I3), reveals that they have natural benefits and can perform on par with existing complex approaches from related work. Second, GeCo, an IIDS designed by us, is even able to minimize the efforts required by human operators setting up and training an IIDS while achieving even better detection performance and remaining comprehensible. Third, we observe great opportunities for transferring the concepts of intrusion detection to novel domains. In a case-study to detect anomalies in a Marine Radar System (MRS), an entirely new domain, we practically prove that existing IIDSs also have benefits beyond their initial use cases (I4). Complementing existing intrusion detection methods with novel IIDSs, to fill the remaining gaps in their detection capabilities, we achieve initial results for securing MRS with little additional effort and no redundant work.

▶ C4: Examining selection criteria to identify IIDSs suitable for a deployment.

Industrial intrusion detection is effective in detecting even novel cyberattacks and across various domains and use cases. Still, given the tremendous amount of pub-

lications and solutions from the literature, it is not apparent which IIDS should be deployed in an actual system (I5). Chapter 6 takes a look at the obstacles and opportunities for deploying IIDSs. The first decision ICS operators face is the type of detection methodology, namely whether to deploy a misuse-based IIDS or an anomaly detection-based IIDS. Due to their different requirements for the training process, it is initially unclear which direction is the best option. Second, while offering already adequate attack detection, these systems, however, still suffer from false positives that operators need to investigate, eventually leading to alarm fatigue [AAM22]. To address this issue, in this last contribution, we challenge the notion of relying on a single IIDS and explore the benefits of combining multiple IIDSs. To this end, we examine the concept of ensemble learning, where a collection of classifiers (IIDSs in our case) are combined to optimize attack detection and reduce false positives. While training ensembles for supervised classifiers is relatively straightforward, as one can use attack samples in training, retaining the anomaly detection nature of IIDSs proves challenging. In that regard, novel time-aware ensemble methods that incorporate temporal correlations between alerts and transfer-learning to best utilize the scarce training data constitute viable solutions. By combining diverse IIDSs, the detection performance can be improved beyond the individual approaches with close to no FPAs, resulting in an promising paths for strengthening ICS cybersecurity.

Takeaway. This chapter developed an overview on the current state of the IIDS research landscape by means of a SMS. In principle, IIDS research is an active and widespread field of research. However, there is also a need for optimization, especially in the way research is conducted, which we summarize in five issues. These issues form the motivation for the subsequent contributions of this dissertation, which we will address in the next three chapters.

4

IPAL – An Industrial Protocol Abstraction Layer

As a non-intrusive and retrofittable deployable security solution, Industrial Intrusion Detection Systems (IIDSs) offer a great additional layer of defense. However, although different industrial domains exhibit similar predictable communication and process patterns, research on IIDSs and the proposed designs of detection methodologies is highly tailored to specific domains and communication protocols (cf. Section 3.5.3). Consequently, the advancements in the detection of cyberattacks on Industrial Control System (ICS) can often not be transferred from their isolated niches to a broader scale, limiting real-world deployability and slowing down research advancements.

While technical and engineering reasons for such tight coupling might exist, we find this strong interdependence rather surprising and contrary to intuition. From an IIDS perspective, industrial protocols essentially exhibit the same distinct characteristics: They primarily exchange sensed data and commands using a small set of well-defined communication patterns. Additionally, while the specifics of benign and malicious behavior change between different ICS scenarios, these can be trained prior to the deployment.

Although there exists no fundamental conceptual reason for the dependence of IIDSs on specific industrial protocols, IIDSs still specialize to a few protocols, limiting their applicability to single scenarios. Moreover, besides contrary claims [FCD+10, IYC+17, KYK20, DAZ20], it is seldom tested whether a specific IIDS even works outside the precise scenario and industrial protocol it was developed for. Consequently, the overall progress in research on IIDSs is slowed down due to niche solutions that do not improve on prior work from other industrial fields, reducing advancements in detecting cyberattacks on industrial systems (I1).

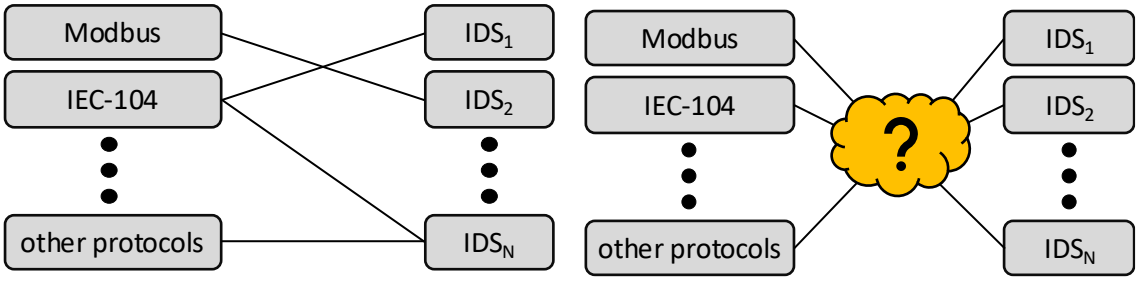
Our goal with this contribution C2 is to break up the prevalent research silos resulting from the interdependence between IIDS approaches and the specific industrial protocol and domain they operate in. To increase the potential of IIDSs, we propose the concept of protocol-independent IIDSs realized through IPAL, our Industrial Protocol Abstraction Layer, in this chapter. The idea of IPAL is to invent a unified data representation for IIDSs to abstract from the different protocols used in ICSs. Thereby, the promise of IPAL is that IIDSs become applicable regardless of the underlying domain-specific industrial protocols. With the additional tooling created around IPAL, such as access to a collection of pre-processed datasets or various evaluation metrics, we, at the same time, provide the technical foundation to enhance evaluations in the future (I2) and improve research's workflows as a whole.

As a first step in that direction, we analyze and motivate the potential benefits of and current need for protocol-independent IIDSs in Section 4.1. To turn this vision into practice, we first assess related work in Section 4.2 and then introduce our own solution, IPAL, an industrial protocol abstraction layer, in Section 4.3. IPAL sits between industrial protocols and IIDSs, captures all aspects relevant to intrusion detection in a unified representation, and thus facilitates protocol-independent and widely applicable IIDSs. We showcase the practical applicability and correctness of IPAL in a reproducibility study of nine IIDSs from related work in Section 4.4. We show that IPAL provides all required information, and we simultaneously reproduce scientific results as an independent party, an important but often neglected step in research. Lastly, we illustrate the various benefits of IPAL for IIDS research in two case studies in Section 4.5. With IPAL, we are able to compare IIDSs, even from opposing research branches, study to what extent they generalize to new protocols, and, for the first time, transfer their advancements to new domains.

4.1 The Case for Protocol-Independent IIDSs

As the first step, we motivate the concept of protocol-independence for IIDSs in general. Currently, the IIDS research landscape is scattered across heterogeneous industrial domains (cf. Chapter 3), leading to a situation as depicted in Figure 4.1a, where IIDSs are developed in isolation for specific industrial protocols and domains. Consequently, research on IIDSs is limited with respect to generalizability, evaluation scenarios, widespread use, and coherence between research streams.

To overcome these limitations, we make a case for *protocol-independent* IIDSs, as shown in Figure 4.1b, where IIDSs do not have to care about the underlying protocol anymore while simultaneously being transferable to various scenarios with little to no additional effort. Our proposal for protocol-independence in IIDSs is motivated by two observations: First, for the purpose of intrusion detection, industrial protocols show similar functionality, i.e., the exchange of sensor readings and actuator commands. Second, in principle, the underlying detection approaches of IIDSs work similarly as they each know or learn a model of a specific industrial process's behavior and detect anomalies by comparing their models to the monitored physical process (cf. Section 3.5.1).



(a) The current state-of-the-art requires a tailored implementation of an IIDS for each industrial protocol.

(b) Instead, protocol-independent IIDSs would allow separating protocol parsing and IIDSs's detection mechanisms.

Figure 4.1 To overcome the limitations of current IIDS research, protocol-independent IIDSs promise to break up the strong interdependence between IIDS approaches and the specific industrial protocol and domain they operate in.

Given these observations, we postulate that a large portion of existing work is not necessarily bound to the specifics of the industrial protocol they are designed for and may generalize well to other protocols. Thus, we are convinced that tackling the diversity in industrial protocols is key to breaking up the isolated silos in the IIDS research landscape and hence allows the wide-ranging application of IIDSs across domains.

To validate the theoretical feasibility of this idea, we first highlight the benefits of protocol-independent IIDSs for the research community in Section 4.1.1. Afterward, we analyze to which extent industrial protocols already share similar functionality in Section 4.1.2 and examine IIDSs for their potential to transfer and generalize across industrial protocols and domains in Section 4.1.3.

4.1.1 Benefits of Protocol-Independent IIDSs

Although IIDSs are tailored to a single industrial protocol and domain nowadays, this does not imply that their benefits are restricted to a single combination. Protocol-independent IIDSs (cf. Figure 4.1b) instead promise to operate detached from specific industrial protocols or domains and thus exhibit several advantages.

First, this paradigm would allow applying IIDSs to multiple industrial protocols within the same domain. For example, power grids that historically rely on different protocols such as IEC-104 (covered by IIDSs [LNTA17, FCCR18, LNT18, LNT19]) or DNP3 (covered by [FCD+10, RSE+20]) would no longer require separate IIDSs. Moreover, if the underlying industrial protocol could be exchanged, the potential to transfer detection methodologies across domains not initially thought of would become directly accessible.

Also, by designing IIDSs with protocol-independence in mind, developers no longer have to extract relevant data from a specific industrial protocol in a time-consuming and error-prone process. For example, Modbus does not define how a value is interpreted (float or integer) and this changes from one deployment to another [Mod18]. Instead, this process can be left to domain experts who know the ins and outs of

specific protocols, which has already proved beneficial for prominent datasets. For example, SWaT [GAJM16], one of the most used datasets according to our survey from Chapter 3, provides pre-processed process data from network traffic.

Finally, it becomes easier to reproduce results, validate findings in different scenarios, and fairly compare IIDSs based on common metrics, thus constituting to sound scientific experiments [GUC+18, UHH+21]. Protocol-independence allows us to efficiently validate results from prior work and expand upon them, a step that is often neglected in research [BBF+19, UHH+21]. Especially, when the original scenario is unavailable, flawed, or too narrow in scope.

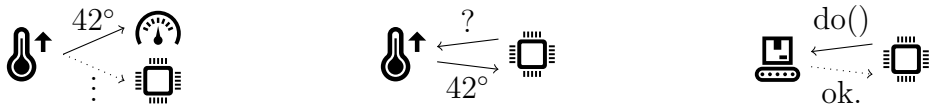
Decoupling IIDSs from the underlying industrial protocol promises to enable the generalizability of IIDSs and their transferability to new scenarios. It can also streamline future research efforts on IIDSs by enabling extensive evaluations, validations, or removing potential biases with regard to specific protocols or domains.

4.1.2 Similarities in ICS Communication Protocols

Given these potential benefits, we ask under which circumstances such a concept would work and what currently prevents it. At the moment, we suspect the different industrial protocols leveraged in ICSs to be one main culprit for this current situation, as depicted in Figure 4.1a. Historically, industrial protocols were designed for unique application requirements, resulting in a variety of special-purpose protocols today [ORZ+24] (cf. also Section 1.1.2). We shortly highlight three protocols in the following that serve as our examples throughout this chapter.

First, the maritime domain uses the NMEA 0183 protocol or its successor based on IP/UDP IEC 61162-450 [IEC18a] to transfer navigational data such as position updates from a Global Navigation Satellite System (GNSS) to an Electronic Chart Display and Information System (ECDIS). It encodes the data as human-readable ASCII characters and simply broadcasts new values to the entire network in regular intervals, e.g., every ten seconds. In contrast, Modbus is a general-purpose protocol intended to request or set values within the memory of Programmable Logic Controllers (PLCs) [Mod18] (cf. Section 2.1.4.1). It is entirely based on a request and response communication pattern where a primary issues requests to read or write data to potentially multiple replicas. As another protocol, European power grids and distribution networks have invented their own protocol, IEC 60870-5-104 (IEC 104) [IEC18b], better suited to communicate changes in values just in time. Because of real-time critical operations, such as shortages, devices can here autonomously decide to publish an updated value if necessary.

This general diversity currently hinders applying IIDS techniques to different protocols. Optimally, a universal and widely deployed industrial protocol would obviate the need to adapt IIDSs to each industrial protocol independently. But, since ICS communication requirements are largely diverse (due to varying constraints on devices, real-time requirements, or the need for authenticated or encrypted communications), such a general industrial protocol seems utopistic. Even if unifying the



(a) *Push* an update of a value to the network.

(b) Use a *request/response* pattern to query specific data.

(c) *Write* data to a device or *call* a remote function.

Figure 4.2 The behavior of industrial protocols can be abstracted into these three basic communication paradigms.

industrial protocol landscape becomes feasible in the future, IIDSs are retrofitted for existing ICSs where specific protocols are already in use and irreplaceable.

Still, despite their diversity, industrial protocols share essential commonalities like supervision and control of physical processes. First, the majority of industrial protocols experienced a shift towards Ethernet- or IP-based communication, which enables many novel applications and attack vectors, but likewise easing passive network monitoring by IIDSs. Thereby, a solution for unifying IIDS solutions that builds upon monitoring Ethernet-based traffic can already be applied to many of today's industrial protocols. Furthermore, as shown in Figure 4.2, industrial protocols' communication behavior can be abstracted into three distinctive patterns: (a) pushing messages to single or multiple devices, (b) request and response, or (c) calls with an optional responses or acknowledgments. This reduces the dissimilarities between industrial protocols that need to be handled by IIDSs. Finally, while protocol specifications dedicate great efforts to encoding, the actual underlying data types can be condensed to a minimal, unified set from an IIDS perspective, as they do not care much about, e.g., encoding, endianness, or bit-length of integers.

Given these circumstances, protocol-independence seems achievable from a protocol-perspective. But so far, it remains open how significant the potential for realizing protocol-independent IIDSs is.

4.1.3 Analyzing IIDSs' Potential for Protocol-independence

To analyze the feasibility of protocol-independence for intrusion detection, we take a look at the IIDSs themselves and especially the data-types (features) they leverage for the detection. In Section 3.5.3, we already identified the need and potential for generalizability beyond the initially considered protocols in publications for a collection of 61 selected and relevant publications (cf. Table 3.4). To analyze to which extent IIDSs really depend on protocol-specific features, we again take a look at these 61 publications and extract the data types required for their detection.

For each approach, we first identify the primary input format, i.e., whether the IIDS works with network packets (P) or aggregated process states (S) that contain solely (physical) measurements of the ICS at a given time. We complement this view with the information an IIDS operates on, i.e., *(i)* timing information of exchanged packets or data, *(ii)* knowledge about communication partners and type of communication, as well as *(iii)* which values are communicated, also including the process state. We identify whether additional packet features are required (e.g.,

packet lengths), whether an externally provided process model is required, as well as whether it trains on anomalous (A) or benign-only (B) traffic.

Table 4.1 depicts the results of this extended analysis and we discuss the results in the following for the classes of communication-based and process state-aware IIDSs.

4.1.3.1 Communication-based Intrusion Detection

Analyzing the type of information communication-based IIDSs operate on, we observe that timing- and sequence-based approaches mostly rely on just four data types: timing information, communicating entities, message types, and often detailed information about transmitted data. In addition, classifier-based approaches often incorporate the exchanged values. Rarely do such IIDSs use further packet features, e.g., packet sizes or checksums, for their classification. Still, none of the approaches take advantage of protocol-specific information, such that, in theory, they could be employed for other protocols and scenarios (for Chu et al. [CLL19], this claim cannot be validated as their automatic feature extraction approach could potentially result in the use of arbitrary features).

Overall, communication-based approaches are not based on a particular structure or behavior of the underlying industrial protocol. Instead, despite IIDSs being tailored to specific protocols, they nearly exclusively use protocol-agnostic information to detect anomalies. Hence, in theory, virtually all considered communication-based IIDSs work protocol-independently, and thus, show potential for deployment in a wide range of industrial domains.

4.1.3.2 Process State-aware Intrusion Detection

In contrast, approaches from the category of process state-aware IIDSs do not rely on packet-specific information and naturally abstract from many industrial protocols' characteristics. More precisely, they operate on the combined process state of all sensors and actuators within the industrial network. I.e., they consider the physical processes, e.g., the measured water levels or state of a pump, rather than which PLC is communicating with another device. Additionally, a large portion of the techniques leverages timing information, e.g., to predict newly observed states based on past observations and compare their predictions to the actual future state. Thus, for process sate-aware IIDSs, the similarity in data requirements theoretically eases the transferability and indicate that IIDSs require little protocol information that can be extracted regardless of the underlying industrial protocol.

4.1.4 Conclusion

Given the current situation, we suspect protocol-dependence to be a significant obstacle that slows down and divides the IIDS research landscape. Thus, we introduce the concept of protocol-independence as a viable solution in that situation. When

		Publication(s)	Year	Detection Methodology	Representation	Timing	Comm. Entities	Message Type	Comm. Values	Process State	Packet Features	External Model	Trains on
Communication-based Approaches	Timings (5)	Valdes et al. [VC09]	2009	Flow Periodicity	P	●	●	○	○	○	●	○	B
		Barbosa et al. [BSP12]	2012	Periodicity	P	●	●	○	○	○	○	○	B
		Ponomarev et al. [PA16]	2016	Telemetry	P	●	●	○	○	○	●	○	A/B
		Lin et al. [LNTA17]	2017	Inter-arrival Time	P	●	●	●	●	○	○	○	B
		Lin et al. [LNT19]	2019	Inter-arrival Time	P	●	●	●	○	○	○	○	B
	Sequences (6)	Goldenberg et al. [GW13]	2013	DFA	P	●	●	●	◐	○	○	○	B
		Yoon et al. [YC14]	2014	PST	P	◐	●	●	●	○	○	○	B
		Caselli et al. [CZK15, CZPK15]	2015	DTMC	P	●	●	●	●	○	○	○	B
		Ferling et al. [FCCR18]	2018	DTMC	P	●	●	●	●	○	○	○	B
		Lin et al. [LNT18]	2018	PST	P	●	●	●	●	○	○	○	B
		Yun et al. [YHL+18]	2018	Nearest Neighbor	P	●	●	○	○	○	○	○	B
	Classifiers (7)	Shang et al. [SCW+16]	2016	SVM	P	●	●	●	●	○	●	○	A/B
		Feng et al. [FLC17]	2017	LSTM	P	●	●	●	○	●	●	○	B
		Perez et al. [PASE18]	2018	SVM, RF, BLSTM	P	●	●	●	○	●	●	○	A/B
		Anton et al. [ASS19]	2019	RF, SVM	P	●	●	●	○	●	●	○	A/B
		Chu et al. [CLL19]	2019	NN	P	○	○	●	○	○	●	○	A/B
		Maglaras et al. [MJ14]	2019	OCVM	P	●	◐	○	○	○	●	○	B
		DIDEROT [RSE+20]	2020	Decision Tree, DNN	P	●	●	○	○	○	●	○	A/B
Process State-aware Approaches	Critical States (10)	Carcano et al. [CCG+11]	2011	Language	S	○	○	○	○	●	○	●	A
		Almalawi et al. [AYT+14]	2014	Outlier detection	S	○	○	○	○	●	○	○	A/B
		Kiss et al. [KGH15]	2015	GMM	S	○	○	○	○	●	○	○	B
		Pan et al. [PMA15]	2015	Common Paths	S	●	○	○	○	●	○	○	A/B
		SysDetect [KS15]	2015	Frequent Itemsets	S	○	○	○	○	●	○	●	A/B
		Kong et al. [KJB17, JKB14, KJM+14]	2016	Temporal logic	S	●	○	○	○	●	○	○	B
		Adepu et al. [AM16a]	2016	Invariants	S	●	○	○	○	●	○	●	-
		Feng et al. [FPMC19]	2019	Invariants	S	○	○	○	○	●	○	○	B
		Monzer et al. [MBF19]	2019	Rules	S	●	○	○	○	●	○	●	B
		Das et al. [DAZ20]	2020	Data Analysis	S	◐	○	○	○	●	○	○	A/B
	Behavior Prediction (11)	Hadžiosmanović et al. [HSZH14]	2014	Autoregression	S	●	○	○	○	●	○	○	B
		Caselli et al. [CZK15]	2015	DTMC	S	●	○	○	○	●	○	○	B
		Ahmed et al. [AMR17]	2017	Kalman Filter	S	●	○	○	○	●	○	○	B
		PASAD [AIA18, AA20]	2018	PCA	S	●	○	○	○	●	○	○	B
		Choi et al. [CLA+18]	2018	Control Invariants	S	●	○	○	○	●	○	○	B
		Myers et al. [MSRF18]	2018	Petri-nets	S	●	○	○	○	●	○	○	B
		TABOR [LAVM18]	2018	TA, BN	S	●	○	○	○	●	○	○	B
		Anton et al. [ALGS19]	2019	Matrix Profiles	S	●	○	○	○	●	○	○	B
		HybTester [CZ19]	2019	Hybrid-Automata	S	●	○	○	○	●	○	○	B
		Denque Anton [AS20]	2020	Matrix Profiles	S	●	○	○	○	●	○	○	B
		SAVIOR [QGS+20]	2020	Physical Invariants	S	●	○	○	○	●	○	◐	B
	Correlations (6)	Krotofil et al. [KLG15]	2015	Entropy Analysis	S	●	○	○	○	●	○	○	B
		Alippi et al. [ANR17]	2016	Hidden Markov Model	S	●	○	○	○	●	○	○	B
		Aggarwal et al. [AKPI18]	2018	Hidden Markov Model	S	●	○	○	○	●	○	○	B
		Hau et al. [HL19]	2019	Statistics	S	●	○	○	○	●	○	○	B
		NoiSense [AMO20]	2020	Noise Fingerprinting	S	●	○	○	○	●	○	◐	B
		ProcessSkew [APQ+20, AZM18]	2020	Noise Fingerprinting	S	●	○	○	○	●	○	●	B
	Neural Networks (9)	Kravchik et al. [KS18]	2018	Neural Networks	S	●	○	○	○	●	○	○	B
		Shalyga et al. [SFL18]	2018	Neural Networks	S	◐	○	○	○	●	○	○	B
		AADS [ADS20]	2019	DNN	S	◐	○	○	○	●	○	○	B
		Seq2SeqNN [KYK20]	2019	Neural Networks	S	●	○	○	○	●	○	○	B
		Elnour et al. [EMK20]	2020	NN+Isolation Forest	S	●	○	○	○	●	○	○	B
		MADICS [PFH+20]	2020	DNN	S	●	○	○	○	●	○	○	B
		Wang et al. [WWLQ20]	2020	Autoencoder	S	◐	○	○	○	●	○	○	B
		Xie et al. [XWWT20]	2020	DNN	S	○	○	○	○	●	○	○	B
		Neshenko et al. [NBHF21]	2021	Neural Network	S	○	○	○	○	●	○	○	B
	Classifiers (7)	Nader et al. [NHB14]	2014	SVDD, KPCA	S	●	○	○	○	●	○	○	B
		Junejo et al. [JG16]	2016	Machine learning	S	◐	○	○	○	●	○	○	A/B
		Inoue et al. [IYC+17]	2017	SVM, DNN	S	◐	○	○	○	●	○	○	B
		Chen et al. [CPS18]	2018	SVM	S	○	○	○	○	●	○	○	A/B
		Anton et al. [ALGS19]	2019	OCVM, Isolation Forest	S	○	○	○	○	●	○	○	B
		FALCON [SMRM20]	2020	LSTM+ML	S	●	○	○	○	●	○	○	A/B
		Elnour et al. [EMKJ20]	2020	Isolation Forest	S	○	○	○	○	●	○	○	B

S: State, P: Packet ●: yes, ◐: partial, ○: no A: Anomalous, B: Benign

Table 4.1 Analyzing 61 IIDSs for the data types used by their detection algorithms confirms the theoretical potential for protocol-independence as most features are not bound to a dedicated industrial protocol. Instead, they focus on a set of generic features that can be derived for many industrial protocols.

looking at industrial protocols, we observe great similarities between them, which provides the foundation for our idea. Next, when considering the features current IIDSs operate on, these likewise show little dependence on specific protocols. While packet-based IIDSs (P) have diverse requirements for the input data, they show great similarities, even across several domains and protocols. Moreover, process state-aware IIDSs (S) also rely on largely common and minimalistic data-types. Therefore, we see no fundamental reasons that prevent applying IIDSs to multiple domains or protocols.

4.2 Related Work

After we examined the potential and theoretical feasibility of protocol-independent intrusion detection, we assess whether such a solution has already been proposed by related work. Besides research on IIDSs covered extensively in Chapter 3, our idea of protocol-independent intrusion detection draws inspiration from different streams of related research and software tools, which we discuss in the following.

Past analyses of IIDSs [UGC+16, GUC+18, ORL20, AMM20], partially with a focus on specific domains [DHX+18, HYL+18, RAMH18, KFPG19, LKP+19], were also able to unveil the problems identified by us such as data heterogeneity and evaluation bias in IIDS research. But analyzing the capabilities of existing IIDSs, the only reproducibility study (considering only model-free state process-based IIDSs) known to us [ET22] shows significant differences between claimed generalizability and reality. These works, in addition to reports on issues with widely used datasets [TETC20], motivated us to quantify these problems and ultimately mitigate them through protocol independence.

Other works [ORZ+24, SK25] also tackles the diversity of protocols and architectures in industrial deployments. By examining multiple network traffic captures of ICS facilities, they can compare the underlying usage of protocols or data rates. The authors perceive that ICS architectures and their properties can significantly vary between deployments. This is why they proclaim that research should indeed consider multiple use cases instead of considering a single scenario to avoid a too narrow focus of the final results.

One solution to tackle protocol heterogeneity in ICS was already developed with the tool SePanner proposed by Meng et al. [MYZ+23]. The authors likewise acknowledge the problem of diverse vendors and protocols yet tackle their transmission and understanding in the network. SePanner automatically derives the semantic meaning of process variables transmitted in industrial protocols. This tool might be helpful for realizing process-state-aware IIDSs but lacks information about the underlying communications required by network-based IIDSs (cf. first part of Table 4.1).

Ryšavý et al. [RM21] propose a library to ease industrial network data preprocessing for intrusion detection. Since their approach requires IIDS developers to implement custom data extractors for each protocol, this solution does not tackle protocol-independence, nor does it facilitate generalizability. Still, to motivate the library,

the authors similarly acknowledge the problem of heterogeneity within the IIDS research landscape [RM21] and likewise strive to move towards an interconnected IIDS community.

The problem of protocol heterogeneity in industrial communication is not exclusive to intrusion detection. However, existing tools from traditional network monitoring, such as NetFlow [Cla04], lack essential features used by IIDSs, such as the process values of the ICS, and are thus not powerful enough for IIDSs. The topic of adapting traditional, rule-based Intrusion Detection System (IDS) approaches such as Snort or Zeek [Roe99, SAH16, Zeek] to industrial domains [YXG+17, GT16, CDF+07, FCD+10, CFMT10, GM14] is also closely related to our work. These rule-based IDSs, however, only detect traditional (known) attacks and thus only complement process-aware IIDSs that are able to detect even (stealthy) attacks exploiting physical processes to cause harm [UGC+16, GUC+18]. Technical proposals to address the issue of protocol pluralism in ICS (e.g., PLC4X [Apa20] and StreamPipes [ZWSR20, WZR20]) do, however, extract insufficient information for IIDSs and consume valuable bandwidth through polling, a limited good in many industrial scenarios [AWT+20, WBH22]. Other work in this context is concerned with the interoperability of devices [ZLZS16, KK18, KSK+19, GCP21] and proposes direct translation between different data representations. Yet translating protocols from one to another seems overly complicated for the purpose of intrusion detection. Moreover, it is not possible or a potential source for error if communication patterns between two protocols do not map to each other. For example, translating information between push- and request/response-based communication (cf. Figure 4.2).

Overall, related work acknowledges many of the issues we aim to solve and has already made advancements to tackle the protocol heterogeneity in ICS, yet usually outside the scope of intrusion detection. Tools explicitly designed for the needs of protocol-independent intrusion detection and assistance to the research community are still missing as of now.

4.3 Design and Implementation of IPAL

Despite their vast diversity, industrial protocols share similar functionality, i.e., communicating measurements and commands (cf. Section 4.1). At the same time, tools such as PLC4X [Apa20] already prove the potential for a unified API for actively querying different industrial protocols, which are, however, not applicable for intrusion detection (cf. Section 4.2). Thus, in theory, it should be possible to transfer IIDS functionality across protocols. Given the potential for realizing protocol-independent IIDSs (cf. Section 3.5.3) and capitalizing on their manifold benefits (cf. Section 4.1.1), we aim to turn this vision into reality.

To this end, we propose IPAL, our design of an *Industrial Protocol Abstraction Layer* to decouple intrusion detection from domain-specific industrial communication protocols in Section 4.3.1. Our core idea is to extract messages from network flows, transfer them into unified information objects, and use them as input for an IIDS. As shown in Figure 4.3, we derive IPAL from the knowledge gathered in our previous

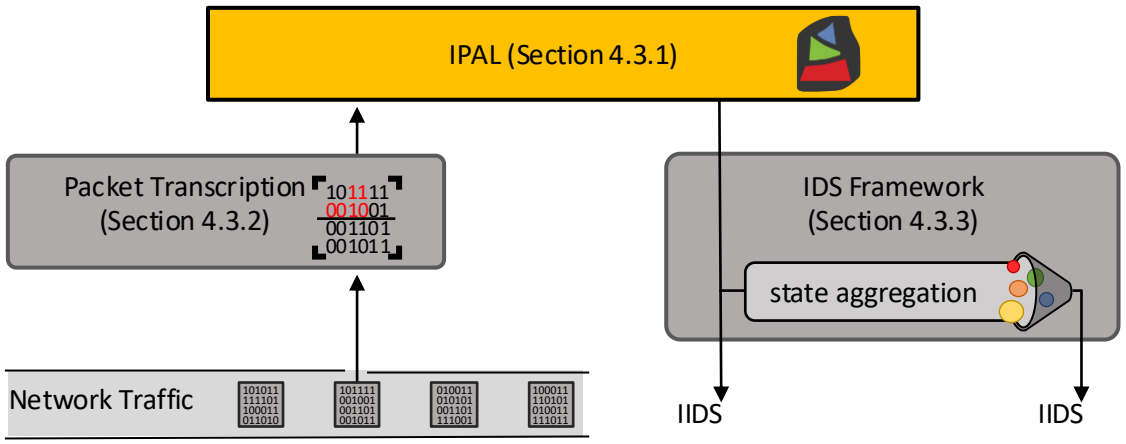


Figure 4.3 To achieve protocol independence in IIDS research, IPAL separates the detection methods from the underlying industrial network protocol with an abstract representation.

survey. We discuss how the landscape of industrial protocols can be transcribed into IPAL messages in Section 4.3.2 and how IIDSs implemented on top of IPAL help transfer their potential to new scenarios in Section 4.3.3. Finally, we discuss the limitations of introducing an abstraction layer in Section 4.3.4.

4.3.1 Designing IPAL

IPAL, our design of an *abstract representation* for industrial protocols, lays the foundation for designing and studying protocol-independent IIDSs by unifying industrial communication. During our comprehensive survey of IIDSs across various industries and communication protocols, we identified inherent similarities with regard to the required data (cf. Table 4.1). Leveraging this knowledge, we deduce the information captured by IPAL, as shown in Table 4.2.

Identifying the optimal trade-off between the protocol abstraction-level and information loss is not trivial. Including every single networking artifact (e.g., TCP retransmissions or IP fragmentation) might blow up the abstraction's complexity with only marginal information gains. As an example, acknowledgments of messages may be of interest to an IIDS. However, the IEC-60870-5-104 protocol acknowledges on three different layers of the communication stack. Consequently, we omit non-application layer acknowledgments and transcribe only semantic confirmations to, e.g., commands. Nonetheless, if deemed necessary, it is possible to include further information in the future.

The construction of our representation is based on two fundamental observations: First, process state-aware IIDSs (cf. Section 4.1.3.2) operate on a restricted set of information, encompassing timings and information about exchanged state information. Meanwhile, communication-based IIDSs (cf. Section 4.1.3.1) operate on a broader set of information. But, they still only rely on packets with (direct) influence on the physical process, such that other packets (e.g., diagnostic data or TCP handshakes) can be ignored. To nonetheless support a wide range of (future) protocol abstractions, we propose the following representation. In total, eleven features

IPAL	Description
`id`	Unique identifier for IPAL messages within a dataset
`timestamp`	Time corresponding to this IPAL message
`length`	Length of the industrial protocol's layer
`protocol`	Name of the underlying industrial protocol
`malicious`	Label for training and evaluation (benign or anomalous)
`source`	Sender of the network packet (e.g., IP and port)
`destination`	Receiver of the network packet (e.g., IP and port)
`message type`	Fine-granular industrial packet's message type
`activity`	Abstracted message type (e.g., request or response)
`responds to`	List of related IPAL messages this message responds to
`process data`	Arbitrary number of process variables and their values

Table 4.2 IPAL captures eleven features in an abstract representation to enable protocol-independence for current IIDSs across multiple domains.

suffice to represent industrial network packets in a common format while preserving the information required by process-aware IIDSs (cf. Table 4.2).

Meta Data. The first observation in our survey regarding the abstraction is the incorporation of metadata across all IIDS categories. Thus, IPAL includes a packet's `timestamp` and its `length`. In addition, a unique identifier (`id`) is included to disambiguate packets. Since multiple industrial protocols can be established in parallel in an ICS, we explicitly encode the underlying `protocol`'s name as well. This can be useful for IIDSs to differentiate between protocol's communication patterns. For example, while IIDSs make use of IEC-104's or S7's repetitive timing patterns [LNTA17], other protocols, such as IEC 61850-GOOSE, have non-linear timings due to exponential back-off timers. Moreover, determining which entity represents a server or a client in a communication can depend on the given protocol. Lastly, since several IIDSs require labeled training data differentiating benign and malicious packets, IPAL includes the `malicious` field. It states whether a packet is considered as benign or malicious for training and evaluating an IIDS.

Addressing Information. Besides meta data, communication-based IIDSs require features to identify the *communicating entities* (cf. Table 4.1). Thus, IPAL includes the `source` and the `destination` of a single packet. These fields are arbitrary strings that, in most cases, represent an IP-port combination but can be extended, e.g., by adding Modbus's unit identifier field, to further disambiguate devices.

Message Identification. One relevant aspect observed during our survey is that communication-based IIDSs leverage the different *message types* of industrial packets on different levels of abstraction (cf. Table 4.1). Some IIDSs [CZPK15, CZK15] analyze the sequence of specific, reoccurring messages, and further IIDSs [LNTA17] only differentiate between requests or responses. Thus, IPAL captures a fine-granular `message type` that captures the precise activity of a protocol. For example, the function code of a Modbus packet or the TypeID of an IEC-104 packet. Second, IPAL also encodes a more generic and protocol-agnostic representation of the `activity`. Here, four `activities` (requests, commands, and their respective answers) suffice (cf. Figure 4.2). Lastly, the `responds to` field lists the `id`s of all IPAL packets a

given message is a response to. For example, Modbus uses requests and responses. Knowing their mapping is crucial to determine the meaning of a response, as only the request contains information about which values are queried.

Process Data. The most important feature used by both types of IIDSs is the process state, i.e., the physical values of the ICS, such as the fill level of a water tank at a given time (cf. Figure 2.1). In IPAL, the field `process data` collects all process variables as well as their current values as communicated within a single packet. For example, the data encoded in a single Modbus response. Depending on the transcribed packet, `process data` can range, e.g., from a single temperature to an array of sensor readings and actuator commands. As process state-aware IIDSs demand a snapshot of the entire system's state, we support state aggregation over several IPAL messages (cf. state aggregation in Figure 4.3). Thereby, we realize a similar extraction as, e.g., through active polling [Apa20]. Such an aggregated state then contains all recent physical values of the ICS in a single message.

A few rare cases exist where additional features (e.g., checksums [CLL19]) are included in the decision-making process of otherwise process-aware IIDSs. These checks are, however, already covered by traditional IDSs (e.g., Zeek [Zeek]) and do not contribute to information about the physical process monitored by process-aware IIDSs. We thus can skip them to avoid redundant work.

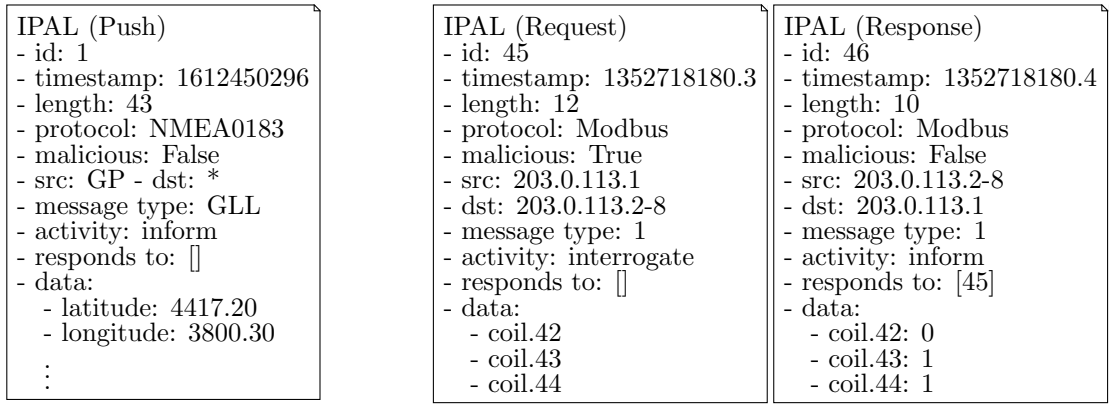
This design of our abstract representation satisfies the requirements found during the literature survey and reasonably abstracts industrial network communication for industrial intrusion detection. Consequently, the eleven features of IPAL suffice to preserve the information required by the process-aware IIDSs and communication-based approaches surveyed in Table 4.1.

4.3.2 Transcribing Industrial Protocols into IPAL Format

IPAL, our abstract representation of industrial network traffic, promises to decouple intrusion detection from the underlying industrial protocols. Yet, to realize protocol-independent IIDSs, real-world network traffic must be transferred into this abstract representation. As shown in Figure 4.3, we implemented a *transcriber* [FC24] that automates this conversion. While integrating specific industrial protocols into IPAL requires a deep protocol understanding, this work needs to be performed only once.

We have already incorporated twelve industrial protocols (Modbus, MQTT, IEC-104, DNP3, S7, IEC 61850-GOOSE, IEC 61162-450, EtherCat, CIP, NMEA 0183, Navico BR24, and MavLink 2.0) used, among others, in popular datasets. Thus, future IIDS research can focus on modeling industrial processes. In the following, we discuss challenges faced while implementing our transcriber.

Our implementation of the transcriber bases on the Python PyShark library [Gre21]. Here, most features of IPAL can be directly extracted from network packets. However, certain protocols require special attention. For example, in ModbusTCP, multiple sub-devices may operate behind a single TCP connection. Thus, `source` and `destination` need to include not only the IP and port but also the device's unit



(a) NMEA position update **(b) Modbus read coil (request and response)**

Figure 4.4 Our concept of transcribing industrial protocols allows us to transfer arbitrary industrial network traffic (bottom) into unified information objects defined by IPAL (top).

identifier. Also, while Modbus's function code directly corresponds to the `message type`, the `activity` has to be derived from the traffic direction, as function codes for requests and responses are identical.

A more challenging aspect is the extraction of the actual process values from industrial communication because of their encoding. To this end, transcribers can contain custom rules to extract and post-process process values from observed communication, similar to StreamPipes [ZWSR20] or SePanner [MYZ+23]. Such rules are small code snippets that allow, e.g., to interpret two 16-bit registers (the biggest data type in Modbus) as a single 32-bit float. Thus, we can integrate the necessary flexibility to define the interpretation of potentially multiple related variables and even directly annotate the interpreted value with a more descriptive and human-readable name.

4.3.2.1 Exemplary Transcription of Protocols

To showcase how IPAL works, we exemplary demonstrate how protocols are transcribed into IPAL messages in Figure 4.4. We consider the two of the three exemplary industrial protocols (NMEA 0183 and Modbus) described before in Section 4.1.2.

For *push* communication depicted with NMEA 0183 in Figure 4.4a, we use IPAL messages of type `inform` to encode the pushed measurements. Besides the encoded communication information, IPAL also bundles the multiple measurement points or control commands of a single network packet. For example, NMEA encodes multiple data points in one GNSS position update message. IPAL models this as a dictionary where each data point comprises a name and a value. In contrast, *request and response*-based protocols (depicted with Modbus in Figure 4.4b) require two independent request and response IPAL messages. Thus, we introduce identifiers (`id`) to link requests with subsequent responses in the responds to field. Moreover, the source and destination fields can contain additional information besides Ethernet

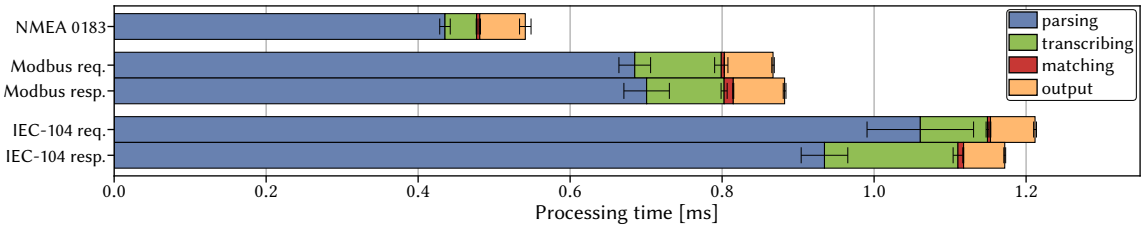


Figure 4.5 The Packet processing time of the transcriber to convert network packets into IPAL format is reasonably fast. We measured 10000 packets per message type each for three different protocols. The confidence interval corresponds to 0.95.

or IP addresses. For example, for Modbus, as shown here, we additionally encode the unit ID as several Modbus devices can be placed behind a single address.

The resulting unified IPAL format can be used to realize protocol-independent detection mechanisms for many industrial protocols since all relevant information types for IIDSs are preserved.

4.3.2.2 Computational Performance Evaluation

Besides functionality, the computational performance of the transcriber is also relevant. Consequently, we evaluated the computational performance of the transcriber to convert network traffic into our IPAL format. To this end, we leverage a single physical core of an Intel i7-9850H CPU. From the results in Figure 4.5, we have already concluded that the performance for many industrial use cases [HHS+18].

For NMEA 0183, the transcriber requires less than 0.6 milliseconds per packet to convert it into IPAL format. This performance should suffice as NMEA 0183 originates from rather slow serial connections. Furthermore, as we measured in the research vessel Deneb [HBW21], packets are sent at a rate of about every 59 milliseconds on average in practice. Next, despite our IEC-104 implementation being the slowest among all three examined protocols, the packet rate in a scientific testbed for power grids is about 116 milliseconds on average [BSL+23]. In both scenarios, the transcriber is sufficiently fast even for potential real applications. In contrast, while the packet interval of the Morris-Gas dataset [MTT15], based on Modbus, can be as low as 30.6 milliseconds between two packets, Modbus packets are sent as fast as every 0.45 milliseconds in other testbeds [FFGS21]. While our transcriber would be too slow in the later setting, with a processing time of about 0.8 milliseconds, our implementation is also not yet optimized for performance.

As most of the time is spent parsing incoming packets, increasing the performance further is possible through implementation improvements or parallelization of the parser and transcriber. With the increasing use of more than one IIDS in a single network [VVKK04], our approach can even improve current IIDSs' performance as the expensive parsing of captured traffic only has to be executed once. Still, performance is currently not our main priority. We rather focus on validating that IIDSs can be efficiently implemented with IPAL.

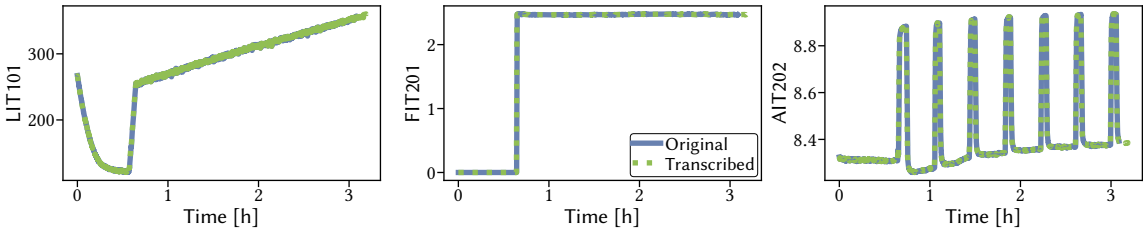


Figure 4.6 When comparing the data provided by a dataset to the data transcribed and extracted in IPAL format from network traces, these greatly coincide. Thus IPAL correctly interprets the data as shown here exemplary for thee process values of the SWaT dataset.

4.3.3 Developing IIDSs with IPAL

To leverage the benefits of our abstract representation, we provide a framework to implement protocol-independent IIDSs on top of IPAL [FC24]. In our analysis in Table 4.1, we observed that IIDSs are either based on communication traffic or on the global process state. As shown in Figure 4.3, for communication-based IIDSs, our framework passes each packet to the IIDS after abstracting it into an IPAL message. But, for process state-aware IIDSs, we need to aggregate multiple packets over time to obtain the current state of the entire process.

To achieve this, our framework caches the process values included within the most recent IPAL messages. We considered different state aggregation methods, but outputting the most recent process values in regular intervals (e.g., each second) yields satisfying results. The process state generated with this method coincides with a newer run of the SWaT testbed closely matching the pre-processed state information provided by the dataset authors, cf. Figure 4.6. Furthermore, as some common datasets (e.g., SWaT [GAJM16]) already provide a state representation, our framework optionally allows restoring these pre-computed states directly.

The IPAL IDS framework enables the realization of communication-based and process state-aware IIDSs on IPAL messages. IPAL already supports twelve industrial protocols from various domains. Thus, IIDSs developed for or adjusted to IPAL can be evaluated for various scenarios based on current and future datasets. IPAL promises to accelerate research on IIDSs, make the research landscape more coherent and move them closer to deployment.

4.3.4 Limitations of Protocol Abstraction

Besides the benefits of IPAL, there are also limitations and disadvantages that we need to discuss. First, any form of abstraction involves the risk of introducing an information loss, as crucial features may be discarded. In our case, IIDSs taking advantage of a feature not covered by our abstraction would likely not be compatible with IPAL. However, considering that IPAL's features are derived from a survey of 61 scientific publications (cf. Table 4.1), we expect the number of currently developed IIDSs that cannot be realized on top of IPAL to be low.

Still, our choice of features captured in IPAL could potentially influence the scope of future research. I.e., limit research to detection methodologies that only consider

those features. However, for dominantly used datasets, we likewise observe a restriction in available information to pre-processed data, e.g., only providing timing and process-state information (cf. SWaT, WADI, HAI, or BATADAL), presumably to facilitate easy utilization of the dataset. Contrary to intuition, the adoption of IPAL could break this trend and even lead to the availability of more features by encouraging the release of unprocessed packet captures, as IPAL provides the tooling to easily process raw data according to researchers' needs. In this vein, IPAL is publicly available [FC24] so that the extracted feature set can be extended in the future if deemed necessary. A similar trend has been observed for Netflow [Cla04], whose abstraction revolutionized network monitoring and introduced newer versions as the need for more features arose.

From a different perspective, protocol abstraction comes with the potential risk of introducing transcription or abstraction inaccuracies affecting the final detection performance of IIDSs realized on top of it. In the following Section 4.4, we tackle this topic by analyzing to what extent the use of our abstraction potentially diminishes the detection performance of IIDSs. Overall, the value of IPAL homogenizing a split-up research landscape outweighs the potential drawbacks arising from introducing an abstraction layer for the growing industrial intrusion detection research field.

4.3.5 Conclusion

While today's IIDS approaches could be capable of detecting cyberattacks for various ICSs, they are often tailored to individual industrial protocols. To obviate the need to apply IIDSs for each industrial protocol individually, we propose the concept of an Industrial Protocol Abstraction Layer (IPAL). By transcribing network flows into protocol-independent and standardized IPAL objects and retaining relevant communication relationships, we provide a basis for widely applicable and protocol-independent IIDSs. Our approach can thus serve as a decent foundation to realize protocol-independent industrial IIDSs as it transparently maintains important communication features such as communication patterns and message relationships.

4.4 Reproducing IIDS Research with IPAL

With IPAL, we propose a novel abstract representation for industrial network traffic to overcome the challenges resulting from protocol-dependence and domain-specific industrial intrusion detection. After having discussed its theoretical feasibility (cf. Section 4.1), we now complement this viewpoint by showing IPAL's *practical applicability and correctness* by performing a reproducibility study of nine IIDS approaches from previous work and reproducing their evaluation results on top of IPAL. Thereby, we also contribute to reproducing scientific research results as an independent party (cf. Section 3.3.4). While deemed extremely important [BBF+19], it is often a challenging and tedious task with little reward and thus rarely performed [ET22, Raj20].

IIDS selection criteria. To conduct a comprehensive reproducibility study, we set out to reproduce at least four representative approaches for each of the two categories

Category	IIDS	Type	Code	Dataset	Result
Communication	Inter-arrival Time [LNTA17]	A	○	◐	✓
	DTMC [FCCR18]	A	●	◑	(✓)
	RF [PASE18]	M	●	●	✓
	SVM [PASE18]	M	●	●	✓
	BLSTM [PASE18]	M	●	●	✓
Process state	PASAD [AIA18]	A	●	●	✓
	Seq2Seq-NN [KYK20]	A	●	●	✓
	TABOR [LAVM18]	A	●	●	✓
	Invariant [FPMC19]	A	●	●	∼

A: Anomaly Detection M: Misuse Detection
●: available ◐: dataset ◑: attacks ○: no
✓: Results reproduced (✓) no comparison to original paper ∼: similar

Table 4.3 We prove the applicability and correctness of IPAL by successfully reproducing nine IIDSs on top of IPAL. These cover all categories from the survey and were selected by the availability of code, evaluation datasets, or tooling to generate the evaluated attacks.

in our survey (cf. Table 4.1), covering all features provided by IPAL. Since we re-evaluate existing IIDSs, we rely on the availability of the original evaluation dataset or tools for attack generation. Also, as re-implementing IIDSs from scratch can be challenging, we focus on open source and only implement IIDSs ourselves if necessary. Thus, our selection was driven by the availability of code or datasets.

Note that we reproduce evaluation results and thus do not invent new attack or detection methodologies. This implies that the reproducibility results are not necessarily comparable, as the authors used fairly different evaluation procedures. Still, we will show in Section 4.5 how IPAL enables conducting a detailed side-by-side evaluation of different IIDSs on new datasets.

In the following, we briefly argue why we selected each IIDS, summarize its core idea, provide re-implementation and evaluation details, and compare our results on top of IPAL to the original publication. Table 4.3 summarizes our IIDS selection and reproduction results.

4.4.1 Inter-arrival Time (Communication-based)

IIDSs from the communication-based category utilize data on a per-packet basis and consider features such as communicating entities and message types. One representative for this category is the Inter-arrival Time (IaT) approach by Lin et al. [LNTA17] implementing anomaly detection. It utilizes periodic traffic patterns in industrial protocols such as Modbus, S7, and IEC-104 to detect, e.g., packet-injection attacks. To this end, the approach measures the *mean* inter-arrival time, i.e., the elapsed time between two packets of the same type (e.g., requests and responses), as well as the maximum temporal deviation (*range*) between packets with the same content and checks for timing violations.

Attack	Model	Metric	Original		IPAL	
			Request	Response	Request	Response
Flooding	Mean	TPR	99.90	99.90	99.98	99.97
		FPR	0.01	0.20	0.01	0.01
	Range	TPR	59.10	56.40	68.21	65.48
		FPR	0.80	1.10	0.98	1.20
		ODR	100.00	100.00	100.00	100.00
Injection	Mean	TPR	96.20	96.60	98.95	98.95
		FPR	0.01	0.20	0.01	0.01
	Range	TPR	100.00	99.50	99.47	99.47
		FPR	0.80	1.10	1.01	1.21
		ODR	100.00	100.00	97.37	97.37
Prediction	Mean	TPR	0.10	0.20	4.60	4.92
		FPR	0.00	0.20	0.01	0.01
	Range	TPR	90.60	91.00	93.69	93.59
		FPR	0.80	1.10	0.97	1.28
		ODR	99.80	99.50	95.21	95.23

TPR: True Positive Rate, FPR: False Positive Rate, ODR: Overall Detection Rate (in %)

Table 4.4 Our re-implemented version of the Inter-arrival Time (IaT) IIDS on top of IPAL closely resembles the original detection rates (cf. Table 2 in [LNTA17]). Slight deviations result from minor differences in the attack generation.

As no implementation of the IaT approach was available, we re-implemented it based on the corresponding publication [LNTA17]. To reproduce the results, we use the only publicly available dataset from the original work [LNTA17], which contains S7 traffic [Smi16]. As only the attack-free dataset was available, we performed an identical 1/10 train-test dataset split and injected "malicious" packets according to the authors' description. Our reproduced results on top of IPAL (cf. Table 4.4) closely match those of the original paper. We attribute the remaining differences to randomness and minor implementation uncertainties with respect to attack generation. Overall, our results show that IPAL provides everything needed to re-implement and reproduce this IIDS.

4.4.2 DTMC (Communication-based)

As an additional communication-based IIDS, we consider an anomaly detection approach relying on Discrete-time Markov Chains (DTMCs) modeling the sequence of network packets for a single connection [FCCR18]. This approach leverages a core property of industrial networks. Devices such as PLCs perform several communication steps in series, e.g., reading values from remote devices and adjusting setpoints later on, resulting in periodic packet sequences repeating in similar order.

Reduction Type	Anomaly Type	Copy [%]			Remove [%]			Swap [%]		
		0.1	1	10	0.1	1	10	0.1	1	10
none	state	0	0	0	0	0	0	0	0	0
	transition	0	15.5	38.5	0	2	14	0	20.5	46.6
overlapping	state	0	0	0	0	0	0	0	0	0
	transition	0	8.5	19	0	1.5	6.5	0	11.5	22
all	state	1	1	1	1	1	1	1	1	1
	transition	0	9.5	22	0	1.5	6.5	0	13	27

Table 4.5 Adapting the DTMC approach to IPAL yields identical detection results in the number of state and transition validations. We do not show the identical values twice here. The definition of the attack definition can be looked up in the respective publication [FCCR18].

	Paper [PASE18]	**Original** [Roc21]	**IPAL**
SVM	94.36 %	94.36 %	96.10 %
RF	99.58 %	99.52 %	99.73 %
BLSTM	98.40 %	98.12 %	96.78 %

Table 4.6 Measuring the accuracy of the public source code (Original) resembles the original paper results. Also, on top of IPAL, we achieve equivalent accuracy most of the time.

We adapted the DTMC approach's publicly available source code [Chr19] to operate on top of IPAL. While a full reproducibility study is infeasible without the original dataset, we can still validate whether the original implementation and the version on top of IPAL produce identical results when presented with the same input. To this end, we generated our own IEC-104 network trace using the simulation framework Wattson for power distribution grids [BSL+23] and the attack tool by the authors to add equivalent attacks to our trace [Chr17]. We performed the same procedure as in the original paper [FCCR18], i.e., a 50/50 train/test split, applying the attack tool ten times for each attack type and counting the number of state and transition violations compared to the trained DTMC. While not comparable to the original evaluation, IPAL correctly preserves all information resulting in identical attack coverage compared to the original DTMC implementation, as shown in Table 4.5.

4.4.3 Classifiers (Communication-based)

Complementing the IaT and the DTMC approach, IIDSs using machine learning classifiers may operate on additional information such as the packet length. Furthermore, they additionally require traces of past attacks for training. Thus we selected the work of Perez et al. [PASE18] as a representative for this category. In their work, they compare three different misuse-based machine learning algorithms against each other, namely Random Forest (RF), Support Vector Machine (SVM), and Bidirectional Long Short Term Memory (BLSTM).

For our reproducibility study, we adapted the implementations of the machine learning classifiers published by the authors [Roc21] to operate on top of IPAL. Fur-

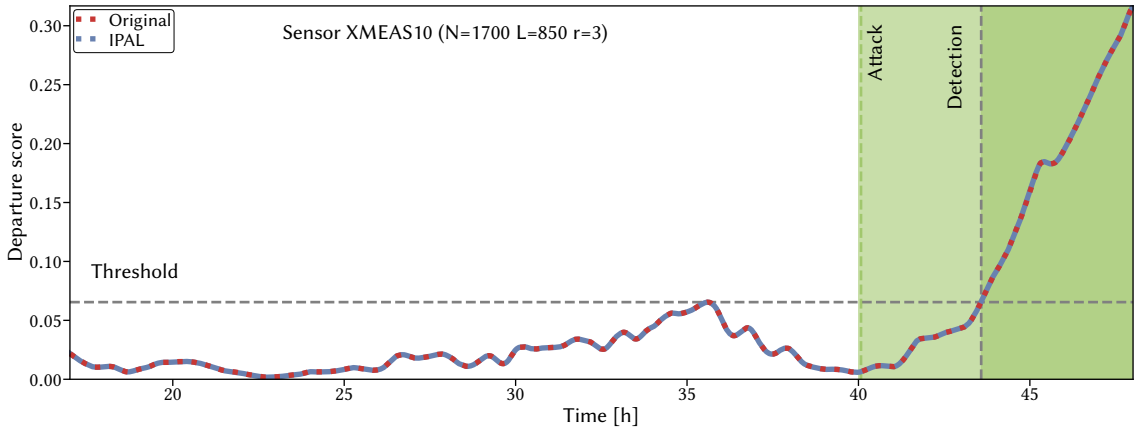


Figure 4.7 PASAD's original departure scores (red, cf. Figure 4 in [AIA18]) match our reproduced implementation (blue). Both implementations have learned identical thresholds of ~ 0.065 and detect the attack simultaneously after $\sim 3.58h$.

thermore, we use the same evaluation dataset [MTT15], hyperparameters, and preprocessing steps as in the original paper [PASE18]. In Table 4.6, we compare the original results of the paper with our replication of the original evaluation as well as the same evaluation on top of IPAL. All three approaches perform equivalently, and the minor deviations can be inferred from randomly selecting a different train and test dataset. Thus, for these three approaches and the previous two communication-based IIDSs, we showed that IPAL correctly preserves all relevant information to achieve equivalent attack detection performance.

4.4.4 PASAD (Process State)

In contrast to the thus far considered IIDSs, the Process-Aware Stealthy Attack Detector (PASAD) [AIA18] detects structural changes in data series and thus represents the process state-aware category utilizing anomaly detection. PASAD's core idea is that legit data series span a vector space, and the distance to the mean of all training vectors indicates an anomaly. An alarm is raised if the distance (departure score) exceeds the maximum observed distance during training (threshold).

Two implementations of PASAD are available, the authors' version in Matlab [Itu18] and a re-implementation in Python [Raj20]. We realized PASAD for IPAL based on the Python version and used the original Matlab implementation to generate reference data. PASAD was trained and evaluated visually on output logs of the Tennessee Eastman Process (TEP) [DV93], parts of the SWaT datasets [GAJM16], and private data of a real water treatment facility. We verify our implementation on the public TEP datasets and discuss one example in Figure 4.7. The results obtained on top of IPAL are identical to the original implementation [Itu18], and both detect the attack simultaneously. By reproducing these results, we show that IPAL is able to preserve all the information required by a process state-aware IIDSs such as PASAD.

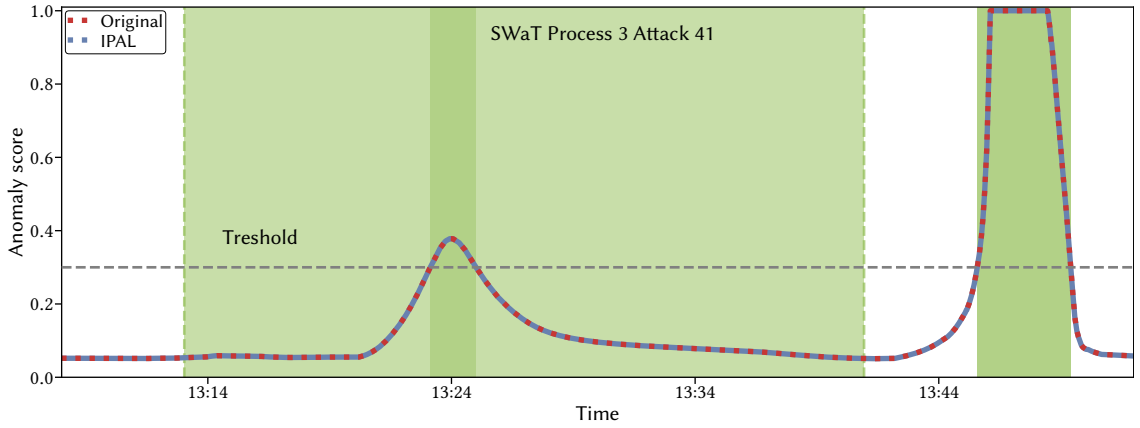


Figure 4.8 Seq2Seq-NN's original anomaly scores (red, cf. Figure 3 in [KYK20]) match our reproduced implementation (blue). Both implementations emit alarms simultaneously.

4.4.5 Seq2Seq-NN (Process State)

Extending on the idea of PASAD, the approach by Kim et al. [KYK20], also implementing anomaly detection, requires knowledge about *multiple* actuators and sensors to predict the expected future states of an industrial system. More specifically, sequence-to-sequence Neural Networks (Seq2Seq-NN), known from language translation, are provided with the recent history of system states (e.g., the last 99 seconds) and predict the next state. Then, the distance between the predicted and actual state is calculated, and the error aggregated over time, resulting in an anomaly score. If the anomaly score exceeds a threshold, the IDS raises an alarm.

For our Seq2Seq-NN implementation on top of IPAL, we adapted its publicly available version [Kim19], extracted the original anomaly scores as a comparison baseline, and used the authors' pre-trained models for the evaluation on IPAL. We exemplarily compare the original implementation and the one on top of IPAL with respect to their ability to detect an attack on stage 3 of SWaT [GAJM16] in Figure 4.8. There the anomaly score is identical, and the attack is detected simultaneously, showing IPAL correctly providing all information for Seq2Seq-NN.

4.4.6 TABOR (Process State)

Complementing PASAD and Seq2Seq-NN, Timed Automata and Bayesian netwORk (TABOR) focuses on one sensor and models the impact of actuators on this sensor [LAVM18]. To this end, TABOR combines three anomaly detection approaches: An out-of-alphabet check alerts if unseen data or data out of regular boundaries is observed, a timed automaton captures the sequence and duration of linear sensor value segments to enable anomaly detection, and a Bayesian network couples sensor segments with actuator states to identify unexpected combinations.

We were able to retrieve the implementation of TABOR from the authors for our re-implementation. TABOR was evaluated on the SWaT dataset [GAJM16], which was divided into smaller logically coherent units (models). Here, we concentrate on

Model 1	Original				IPAL			
	FP	TP	CP	PS	FP	TP	CP	PS
TABOR	0	9	7.85	1889	0	9	7.09	1765
TA	7	26	87.48	189516	7	26	86.3	188829
BN	0	5	1.95	629	0	5	1.27	504
OOA	0	5	5.90	1260	0	5	5.82	1261

Table 4.7 TABOR, on top of IPAL, detects the identical nine attack scenarios (TP) with slightly different coverage percentage (CP) and penalty scores (PS) (cf. Table 2 in [LAVM18]).

SWaT	TPR (Recall)	FPR (Fall-out)
Paper [FPMC19]	0.709	0.0003
IPAL	0.695	0.0027

Table 4.8 The Invariant IIDS achieves similar detection scores on IPAL compared to the publication. The remaining difference is attributed to the fact that a different mining algorithm is used in the author's open source version than evaluated in the publication [Fen19].

reproducing model 1 as its intermediate results in the original paper [LAVM18] ease verifying the correctness of our implementation. We achieved equivalent results for our version of TABOR on top of IPAL (cf. Table 4.7). Our re-implementation scores are identical in true positives (TP) and false positives and only deviates marginally within the coverage percentage (CP) and penalty scores (PS), probably due to minor temporal deviations in the alarms. We confirmed manually that the alerts overlap between the original and our implementation. Concluding, IPAL provides the necessary information to realize even complex, combined process state-aware IIDSs.

4.4.7 Invariant (Process State)

The last anomaly detection approach of our reproducibility study is representative of the category of critical states. Feng et al. [FPMC19] present an IIDS that derives so-called invariants, hence called Invariant IIDS in the following, that must be fulfilled by the ICS at all times (cf. Section 2.2.4.2). An invariant is a boolean implication that, e.g., states if the water level of a tank is rising, its inlet valve must be open. A violation of such a rule would indicate an anomaly. In contrast to other works that define such critical states or equations manually, this IIDS automatically derives these invariants through a mining process from historic ICS data.

Our implementation in IPAL is based on the authors' provided source code [Fen19]. In contrast to the publication, their public code leverages a different algorithm for the mining of the invariants [Fen19], which may result in slight deviations in comparison to the publication's results. As shown in Table 4.8, our reproduced results on top of IPAL for the SWaT dataset are still quite close to the publication, with only a higher False Positive Rate (FPR) value for our implementation. This demonstrates IPAL's ability to serve as a solid framework for a variety of IIDSs.

4.4.8 Summary and Lessons Learned

To showcase the practical applicability and correctness of IPAL, we performed a reproducibility study of nine IIDSs covering the two main IIDS categories with at least four representatives each. By (re-)implementing a diverse set of IIDSs on top of IPAL, we demonstrated that IPAL provides all the information required by various types of IIDSs. By reproducing previous works' results, we have shown that IPAL operates correctly and neither reduces functionality nor detection rates (cf. Section 4.3.4). Consequently, IPAL provides a reliable foundation for protocol-independent IIDSs, thus capitalizing on the promised benefits of an abstract representation (cf. Section 4.1.1).

As indicated in Table 4.3, successfully reproducing others' research requires the availability of the original implementation. Only in rare cases does an evaluation dataset suffice. We were able to successfully reproduce seven IIDS approaches (and even achieve fully identical results for two of them). But, we could not reproduce the evaluation results of the DTMC approach [FCCR18] since the evaluation dataset was not available to us. Still, as the original source code and attack tool were available, we could use another dataset to show that our re-implementation on top of IPAL produces exactly the same results as the original implementation.

While detection performance is of utmost interest, adding abstraction layers, such as IPAL, introduces overheads. For a fast IIDS such as DTMC, we exemplary compared the execution time against the original implementation. IPAL yields an overhead of 10.1% for the benefits of increased applicability. This overhead is expected to shrink for more complex IIDSs.

Overall, the considered nine IIDSs not only rely on the requirements derived from our survey but also cover four distinct protocols (Modbus, EtherNet/IP, S7, and IEC-104) from four datasets [DV93, MTT15, Smi16, GAJM16]. Besides contributing to the important task of reproducing scientific research results as an independent party [BBF+19, ET22], we lay the foundation for *protocol-independent* development, testing, and evaluation of IIDSs by providing a rich foundation for broad evaluations.

4.5 IIDS Generalizability and Transferability Study

Our reproducibility study showed that existing IIDSs can be implemented on top of IPAL without loss of functionality or detection capabilities (cf. Section 4.4). What remains to be shown is that IPAL is actually useful, i.e., it addresses the pressing problems of current IIDS research. More precisely, we demonstrate how IPAL helps to generalize IIDSs to a new industrial protocol, how IIDSs can now transfer to new scenarios with little effort, and how existing IIDSs compare against each other. To this end, we perform two case studies: (i) for communication-based IIDSs in Section 4.5.1, and (ii) for process state-aware IIDSs in Section 4.5.2.

Scenario	**DoS [BTZ+19]**			**Advanced [BTZ+19]**			**MitM [SCPA21]**		
Protocol	GOOSE			GOOSE			Modbus		
Metrics	Precision	Recall	F1	Precision	Recall	F1	Precision	Recall	F1
IAT (mean) [%]	**99**	**100**	**99**	0	0	-	**100**	**98**	**99**
IAT (range) [%]	**90**	**100**	**95**	0	62	1	99	31	47
DTMC [%]	**100**	**100**	**100**	**100**	**95**	**97**	0	0	-

Table 4.9 IPAL enables the comparison of communication-based IIDSs in realistic scenarios. We see that each IIDS detects certain attacks well in practice, while other attacks that should be detectable in theory are not caught.

4.5.1 Generalizing Communication-based IIDSs

The first case study demonstrates IPAL's ability to generalize communication-based IIDSs to new and realistic scenarios. Therefore, we study the two approaches IaT [LNTA17] and DTMC [FCCR18] from our reproduction study (cf. Section 4.4). Both were previously not compared against each other and were evaluated on artificial attacks retroactively integrated into network traces. Thus, their effectiveness in realistic scenarios is currently unknown.

To this end, we utilize novel datasets from realistic industrial systems not yet covered by both IIDSs. The first dataset resembles a Denial of Service (DoS) attack against a power grid communicating with the IEC 61850-GOOSE protocol [BTZ+19]. In the same setting, more advanced attacks are performed based on a combination of message modification, injection, and replaying, comprising our second dataset [BTZ+19]. Finally, we look at a Machine-in-the-Middle (MitM) attack dataset based on ARP poisoning to manipulate the data within Modbus packets [SCPA21]. Note that these datasets already contained these attacks and were specifically designed for evaluating the performance of security solutions. Our transcriber (cf. Section 4.3.2) transparently translates the GOOSE and Modbus network traces into IPAL, obviating the need to modify the existing IIDS implementations to apply them to the new protocols. Table 4.9 summarizes the results on these three new datasets.

DoS. All IIDSs detect the DoS attacks to a high degree. For the IAT's mean model, these results are expected, as this attack is similar to the artificial flooding attack evaluated in the original publication [LNTA17]. Surprisingly, the IAT's range model achieves good detection rates, even though it performed worse in the artificial scenario (cf. Table 4.4). The DTMC achieved the best performance, classifying all packets correctly despite no similar attack being discussed in the original paper outperforming an IIDS specifically designed to detect such attacks.

Advanced. The second scenario considers advanced data manipulation attacks against a power grid similar to the artificial attacks evaluated in both original publications [LNTA17, FCCR18]. Still, neither of the IAT models can sufficiently detect these attacks, despite performing well in the artificial attacks originally evaluated. This behavior can be explained by the low number of data manipulations and variable retransmission timings of GOOSE, leading to a larger variety of inter-arrival

times. On the other hand, DTMC achieves great detection rates, confirming the results from the original publication in a new industrial domain and protocol.

MitM. Finally, we consider a MitM attack manipulating data received by industrial controllers [SCPA21]. While such an attack is a realistic threat in practice, it has neither been considered by IAT nor by DTMC thus far. Still, our results show that the IAT's mean model performs especially well in this scenario without issuing any false alarm. Contrary, DTMC is not able to detect a single malicious packet since this attack does not change the sequence of observed messages.

These results indicate that artificial attacks, retroactively integrated into evaluation datasets [LNTA17, FCCR18], as considered in the original publications [LNTA17, FCCR18], do not necessarily carry over to more realistic attack scenarios, even when considering a similar attack behavior. Yet, these new attack types can be detected well in some instances. This observation highlights that IIDSs have to be studied more extensively to understand which attacks they can detect and which not. Additionally, our case study highlights that one IIDS does not necessarily have to be superior in every attack scenario. Thus, cooperative IIDSs, i.e., several detecting specific attack classes, may improve overall performance. We discuss this idea later in more detail in Chapter 6. Finally, we showed the adaption from one domain or protocol to another scenario, confirming the enormous potential for protocol-independent and domain-agnostic IIDSs as previously identified in Section 4.1.

4.5.2 Transferring Process State-aware IIDSs

Our second case study focuses on process state-aware IIDSs. Since the transferability of IIDSs has not been evaluated sufficiently before in literature, we show how IPAL eases the adaption of IIDSs to new domains and thus understand how IIDSs transfer to new scenarios. Furthermore, as we and Urbina et al. [GUC+18] observed, approaches are insufficiently compared in the literature. For example, PASAD is evaluated only visually [AIA18], while TABOR defines its own detection performance metrics [LAVM18], and Seq2Seq-NN counts an attack as detected even 15 minutes after the attack ended [KYK20].

From our reproducibility study, we have four implementations of process state-aware and anomaly detection-based IIDSs (TABOR [LAVM18], PASAD [AIA18], Seq2Seq-NN [KYK20], and Invariant [FPMC19]) at hand, which were all evaluated at least on the SWaT dataset [GAJM16]. With IPAL, we can now apply these to three new datasets (WADI [APM17], HAI [SLYK20], and BATADAL [TGT+18]), as these stand out as relevant and qualitative datasets in our previous survey from Chapter 3. These datasets were all especially designed for evaluating IIDSs and represent different types of industrial processes. WADI and BATADAL are modeled after a water distribution process, SWaT considers chemical treatments of water, among others, and HAI contains electrical power generation as another interesting domain (cf. Section 2.3.1). During training, we tried to find good parameters for each IIDS and for each dataset with reasonable effort.

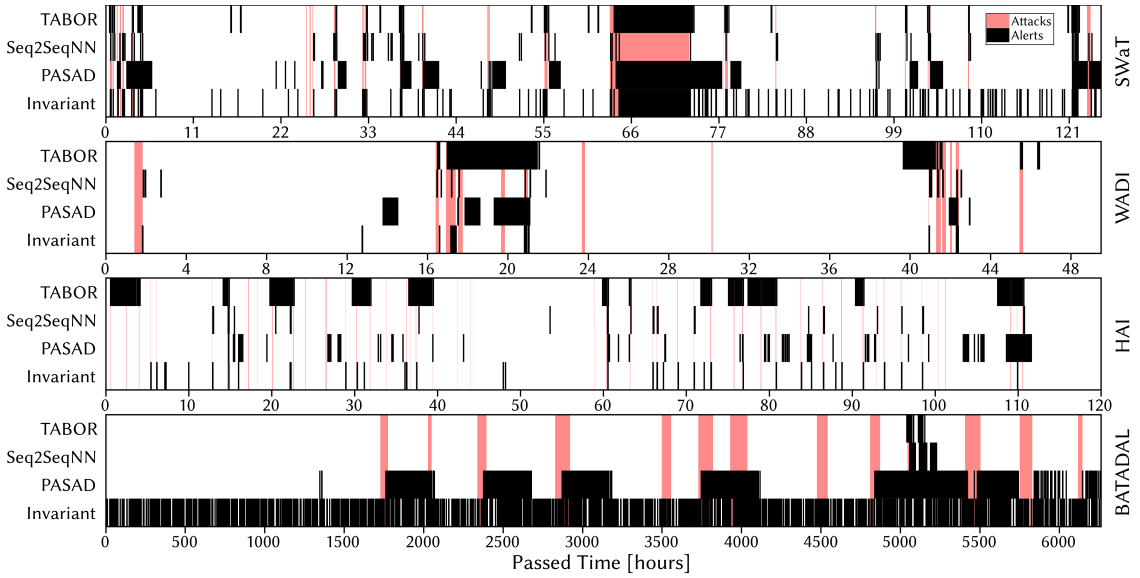
Regarding evaluation metrics, as these can be of varying expressiveness (cf. Section 3.4.2), we decided for a broad collection of eight diverse metrics. First, we

	IDS	Prec.	Rec.	F1	eTaP	eTaR	eTaF1	FPA	Scen.
SWaT	Seq2SeqNN	44.0	10.9	17.5	42.8	47.2	44.9	36	75.0
	TABOR	81.5	74.7	77.9	49.1	18.9	27.3	27	55.6
	PASAD	32.4	71.5	44.6	16.0	4.9	7.5	14	44.4
	Invariant	97.3	69.5	81.1	55.4	32.7	41.1	185	86.1
WADI	Seq2SeqNN	44.4	13.4	20.5	45.4	31.3	37.1	7	64.3
	TABOR	19.1	43.7	26.6	14.9	13.0	13.9	3	57.1
	PASAD	16.4	23.9	19.5	5.4	4.3	4.8	3	35.7
	Invariant	90.0	21.9	35.2	92.3	32.6	48.1	2	42.9
HAI	Seq2SeqNN	8.5	4.6	6.0	7.7	2.9	4.2	14	26.0
	TABOR	4.8	45.1	8.7	0.0	0.0	0	4	46.0
	PASAD	3.3	12.7	5.3	1.0	2.0	1.3	51	16.0
	Invariant	77.2	9.2	16.4	79.2	25.3	38.3	14	50.0
BATADAL	Seq2SeqNN	34.2	5.6	9.6	27.0	6.9	11.0	1	14.3
	TABOR	78.5	6.9	12.7	77.7	14.3	24.1	2	14.3
	PASAD	20.1	52.1	29.1	10.5	21.5	14.1	32	78.6
	Invariant	27.2	45.5	34.0	18.2	74.9	29.3	865	100.0

Table 4.10 IPAL enables the objective comparison of different process state-aware IIDSs on the SWaT dataset and in new scenarios on WADI, HAI, and BATADAL with the help of plenty and standardized metrics.

consider the three traditional point-based precision, recall, and the F1 score, as well as, their newer pendants enhanced time-aware precision (eTaP), recall (eTaR), and F1 (eTaF1) [HYKM22]. Additionally, we state the fraction of detected scenarios (Scen.), i.e., detected attacks, and the number of continuous False-Positive Alarms (FPA). For the same argument as in Section 3.4.3, we again only count alarms as false positive, if they are more than 60 seconds away from an attack. To this end, Table 4.10 summarizes the in-depth results based on these metrics for the IIDSs and datasets under consideration. In addition, we depict the alerts visually in Figure 4.9 to provide a more subjective insight into the underlying alerts.

SWaT. For this dataset, on which the IIDSs were initially evaluated, we see significant differences between the operation of each IIDS. TABOR (55.6) and PASAD (44.4) detect the least amount of scenarios (Scen.) on this dataset while Invariant achieves a total of 86.1 percent detected attacks. Nonetheless, Invariant features also the most FPA with 185 which are seemingly randomly distributed over the entire duration of the dataset (cf. Figure 4.9). While Invariant also yields the highest F1 score among all IIDSs, the number of FPA makes it a bad candidate for practical applications. Contrary, Seq2SeqNN, detecting also many attacks on SWaT (75.0), has much fewer FPA. However, Seq2SeqNN receives the worst score among all IIDSs in F1 since it misses SWaT's longest attack – a phenomenon already discussed in Section 3.4.3. The novel time-aware metrics (eTaP, eTaR, and eTaF1) keep up for this disadvantage and indeed attribute Seq2SeqNN the highest score. Besides detection performance, we also notice differences in how these IIDSs emit alerts. PASAD buffers recent states internally, and Seq2Seq-NN accumulates deviations over time, delaying the alerts for both approaches. Note that the metric detected scenarios does not account for delayed alerts. Contrary, TABOR has great precision with



Short alarms are slightly enlarged to become visible in this figure.

Figure 4.9 IPAL allows the side-by-side comparison of four process state-aware IIDSs on industrial datasets and simultaneously enables to understand how IIDSs generalize to similar (WADI, and BATADL) and new domains (HAI).

respect to the attack time, yet alarms are announced retrospectively due to internal segmentation. Such differentiation only show in an in-depth side-by-side analysis as enabled by IPAL.

WADI. While the WADI dataset [APM17] is not unknown to the research community (cf. Table 3.2), only the authors of the Invariant IIDS evaluated it originally. WADI is similar to SWaT, and we can show how simple transferring these IIDS to a similar domain with IPAL is. Here, TABOR detects 57.1 scenarios, PASAD detects only 35.7 attacks and Seq2Seq-NN 64.3 scenarios. Interestingly, Invariant is now one of the best IIDSs on WADI detecting reasonably many attacks with just 2 FPA. Likely because it is the only IIDS of our collection that was originally evaluated on WADI. Note that during the training of PASAD, computational limits were reached due to the larger dataset size, which might contribute to the lower detection quality. Moreover, precise tuning of the IIDSs hyperparameters could result in slightly better detection results (cf. Section 3.5.2). Still, the initial results of transferring an IIDS to a new dataset with IPAL are promising without the need to design new IIDSs from scratch.

HAI. After transferring these four IIDSs to a similar domain, we now study how they perform in other industrial settings. Thus, we consider the HAI dataset [SLYK20] based on a hardware-in-the-loop simulator where one stage contains a boiler process used for power generation. This dataset is much noisier than SWaT or WADI and less seldomly exhibits clear repetitive patterns challenging all IIDSs. For example, HAI's turbine process does not exhibit obvious repetitive patterns. TABOR does not transfer well, as it flags large parts of the dataset as an anomaly, while Seq2Seq-NN detects just 26.0 percent of the scenarios with 14 false alarms. PASAD also detects 16.0 scenarios but with far more false positives (51). Transferring these IIDSs to

HAI with IPAL reveals their unexplored potential to transfer to new scenarios but at the same time also highlights their limitations.

BATADAL. The final dataset, BATADAL, again resembles a water distribution scenario, but for a much lager municipal water system. The dataset also covers a long timespan over multiple months however with a much lower sampling rate (hourly samples) compared to the other datasets sampling rate of one second. On this dataset, the Invariant IIDS nearly constantly emits an alert making it useless. While PASAD seems to perform bad on first sight, its alerts usually begin within an attack but last too long. In contrast, Seq2SeqNN and TABOR perform much better with few false positives (up to two) but likewise detect just one attack. We presume that this diminished performance results from the low sampling rate of the dataset and consequently from the higher fluctuations in the data making it harder for the IIDSs to identify repeating patterns.

In conclusion, IPAL enables a detailed look at how different IIDSs compare against each other, as promised. We even observed how differently IIDSs emit alerts, motivating the use of well-chosen metrics once again (cf. Section 3.4.3). Besides this in-depth comparison, IPAL allows the application of IIDSs to new datasets quickly without the need to redesign novel approaches from scratch. However, we also saw that their performance lags behind on certain datasets, hinting at problems with the underlying detection methodologies of these new datasets.

4.6 Conclusion

Intrusion detection is regarded as a complementary protective measure to detect such attacks in a timely manner and thus mitigate their consequences. Consequently, a research community gathered around industrial intrusion detection (cf. Chapter 3). The produced research advancements are, however, surprisingly scattered, as many solutions are proposed for distinct communication protocols in specific industrial domains, thus hindering their transfer of research achievements to other industrial domains (I1).

To better understand this phenomenon, we examined 61 scientific IIDSs in detail and identified an unexplored potential. While practically operating in protocol and domain-dependent silos, theoretically, neither do IIDSs operate on the information of certain communication protocols, nor are their fundamental detection methodologies specific to individual domains. Consequently, there is a potential for *protocol-independent* IIDSs to protect industrial networks across scenarios, not only to improve the research landscape but also to contribute towards widespread real-world deployments.

To unleash this potential, we propose IPAL, our Industrial Protocol Abstraction Layer that decouples intrusion detection from domain-specific industrial communication protocols. To this end, IPAL transcribes protocol features relevant to intrusion detection into a common abstract representation. To show the applicability and correctness of our approach, we conducted a reproducibility study of nine IIDSs

from related work, proving that IIDSs can indeed be implemented on top of IPAL. Finally, with the ability to transfer IIDSs seamlessly across different industrial protocols and domains, we studied how existing IIDSs generalize to new scenarios. We find that existing works are *(i)* indeed not confined to specific domains or protocols, but also that *(ii)* the type of attacks a given system can detect is not sufficiently studied, and *(iii)* that there is potential for improving on the detection methods such that they more seamlessly apply to all considered datasets. Still, our work lays the foundation to break up the protocol-dependence of IIDS research and enable further studies regarding the generalizability and transferability of IIDSs in new research or real-world scenarios.

To make IPAL a helpful instrument for the IIDS research community and foster wider and more expressive evaluations in the future (I2), we published everything as an open-source project [FC24]. As of June 2025, this project includes support for twelve (industrial) protocols to transcriber network traffic into IPAL representation, 20 implementations of IIDSs, 19 datasets accessible to researchers, and tools for visual and objective evaluations to analyze IIDSs' detection performance with up to 32 common and novel metrics. After publishing IPAL, another framework with similar intentions as IPAL emerged in the research community called Lumen [SSA+22]. While intended to assist in machine learning-based anomaly detection for Internet of Things (IoT) networks, their motivations and methodologies, i.e., unifying common features and operations of such IDSs or providing easy access to plenty of datasets, remain fairly similar to IPAL. Thus, tackling the current issues of research as done with IPAL is a promising solution and resembles the first step to generalizable and transferable IIDS research (cf. Section 1.4). Last but not least, IPAL sets the foundation for the subsequent results and contributions of this dissertation in the following chapters.

5

New IIDS Designs

While Industrial Intrusion Detection System (IIDS) research has proposed various detection methodologies (cf. Section 3.5.1), two questions remain unanswered. First, we want to understand the believed necessity of complexity in IIDS research (I3). Secondly, another goal is to transfer IIDS solutions to a novel domain (I4). We tackle these issues by proposing novel detection methodologies, measuring their capabilities, and comparing them to related work. To this end, the Industrial Protocol Abstraction Layer (IPAL) provides us with the necessary tooling, e.g., datasets and re-implementations of IIDSs, to conduct efficient and effective analyses (cf. Chapter 4). Note that this chapter exclusively concentrates on anomaly detection.

As the first part of this chapter, Section 5.1 challenges the current notion that IIDSs have to be complex. To disprove this misconception, we design four rudimentary and simple IIDSs, such as testing whether process values exceed or undercut the minimum or maximum values seen during training. Comparing these minimalist algorithms against complex related work reveals that complexity is indeed not necessary in all the cases for the given datasets.

Next, since our simple methods offer room for improvement and could be mitigated with likewise simple attacks, we tackle these flaws by inventing another IIDS named GeCo in Section 5.2. During the design of GeCo, we take special care not only to remain simple and comprehensible but also to make it usable for Industrial Control System (ICS) operators in the end.

As the final part of this chapter, Section 5.3 investigates whether the results and insights achieved so far are transferable to another novel domain, as promised to be possible with IPAL. For this excursion, we select the maritime domain due to its under-representation in the ICS domains (cf. Table 3.4). More precisely, we tackle Marine Radar Systems (MRSs) by first proving their vulnerability, then establishing a simulation environment to evaluate existing IIDSs in this scenario and fill the identified gaps in detection capabilities with novel approaches.

5.1 SIMPLE Industrial Intrusion Detection

Besides the issues tackled by IPAL in Chapter 4 centered around how the current workflows of IIDS research can be improved, we still have not addressed issue I3. I.e., whether detection methodologies indeed have to be as complicated as considered in research. While the encountered complexity is associated with disadvantages, such as reduced comprehensibility or computation overheads, as laid out in Section 3.5.2, it might be necessary to yield good detection results. Thereby, complexity, in general, can be justifiable. However, there is no proof in the literature that IIDSs need to be constructed as complex as proposed by the current state-of-the-art.

To tackle this research gap, we study to what extent IIDSs can be simple and to which extent they are comparable to related work. For example, merely keeping track of the minimum and maximum of observed process values. Surprisingly, such IIDSs have received no attention from the research community so far. Likely as they have never been considered suitable in traditional networks, e.g., data centers. However, as we show in this contribution, this conclusion is not necessarily true for ICSs due to the repetitive and predictable nature of their underlying processes.

We begin with defining six properties that make an IIDS S.I.M.P.L.E. in Section 5.1.1 and, based on these requirements, design four exemplary IIDSs that satisfy these properties in Section 5.1.2. Leveraging IPAL as our evaluation platform (cf. Section 5.1.3), we then compare the detection performance of our SIMPLE IIDSs against state-of-the-art complex related work in Section 5.1.4. Our results in Section 5.1.5 show that SIMPLE IIDSs detection performance is on par with complex related work and can be ported across industrial domains. Simultaneously, our further investigations with respect to computational performance and hyperparameters in Section 5.1.6 highlight the benefits of simplicity for later deployments. Discussing our findings in Section 5.1.7, we come to the conclusion that complexity is not necessary per se (Section 5.1.8).

5.1.1 Properties of S.I.M.P.L.E. Intrusion Detection Systems

The focus of research on complex IIDSs, as laid out in Section 3.5.2, leads to inherent drawbacks such as incomprehensive alerts, difficult deployments, computational complexity, or missing justification for complexity as attacks from current datasets seem easily detectable. To find out whether this inherent complexity is justifiable, we propose six properties for Sufficient, Independent, Meaningful, Portable, Local, and Efficient (SIMPLE) IIDSs instead. These properties promise to alleviate the current issues if satisfied by an IIDS. Note that we use *S.I.M.P.L.E.* to refer to these properties and *SIMPLE* for an actual IIDS that satisfies these requirements.

Sufficient. Although simpler in design, an IIDS should be sufficient to detect most attacks while emitting few false alarms (compared to complex approaches).

Independent. Training an IIDS to a scenario is complicated due to hyperparameters influencing the training process (cf. Section 3.5.2.4). In contrast, SIMPLE models

should be independent of hyperparameters and specialized personnel required to find such hyperparameters or re-evaluate a trained model after any modification.

Meaningful. As IIDSs protect physical processes, providing operators with meaningful alerts is essential (cf. Section 3.5.2.3). They allow determining which sensors/actuators behave anomalously and thus enable to initiate appropriate countermeasures in a timely manner.

Portable. Since the industrial domain is inherently heterogeneous, an IIDS should be portable to various industrial scenarios. I.e., it needs to be adaptable to different ICS processes and all kinds of sensors/actuators types.

Local. Detection methodologies should be local to individual sensors/actuators so that they can be adjusted to their distinct behavior. Furthermore, locality enables partially adjusting an IIDS when the ICS is modified, and sensors/actuators are added or removed without obsoleting other models.

Efficient. The detection methodology should be computationally efficient during training and live detection. Since an ICS's process may change, quick re-training avoids extensive periods in which an obsolete model is used. Efficiency during live detection enlarges hardware and deployment choices and eases timely responses.

Besides our definition for S.I.M.P.L.E., related work already postulated a similar set of requirements [Eta17]. Overall, our six properties address the challenges of complex detection approaches widely found in literature and thus provide the foundation for easily understandable, lightweight, transferable, and effective intrusion detection.

5.1.2 Designing SIMPLE IIDSs

To turn the postulated properties into reality and thus lay the foundation for answering whether industrial intrusion detection indeed has to be inherently complex, we design four SIMPLE IIDSs. Our approaches are inspired by typical attack patterns found in scientific datasets (cf. Figure 3.11), natural ICS behaviors, and share a set of common characteristics as explained in the following.

At the core, a SIMPLE IIDS learns a single model per sensor/actuator of the ICS and is trained in a single pass. Not only do separate models fulfill locality, but they are necessary as process variables exhibit different value ranges, i.e., sensors (float) and actuators (discrete), obviating the need for additional normalization known from complex related work (e.g., [IYC+17, ADS21, AS21]). Simultaneously, we avoid introducing process dependencies into the model, which would increase complexity. By iterating over the data just once, we significantly reduce the computational complexity of the training process.

All our detection models train a lower (min) and an upper (max) threshold of a certain, easily computable property and emit an alarm if one of these thresholds is exceeded. To account for variability in physical values and between process cycles

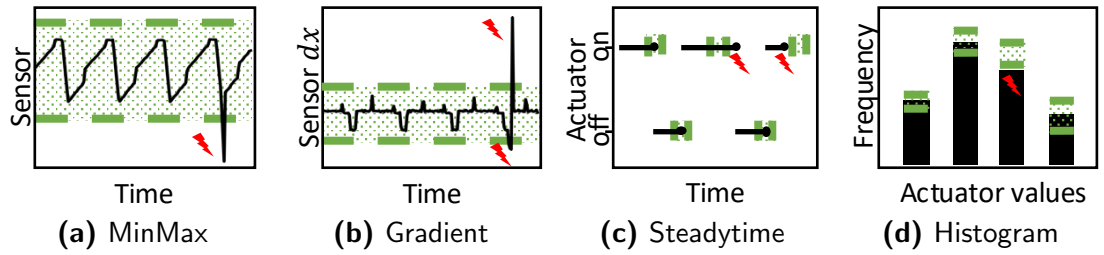


Figure 5.1 We introduce four SIMPLE IIDS ideas detecting over- or undershooting with a MinMax approach, steep in- or declines with Gradient, flat lines with Steadytime, and unnatural process fluctuations with a Histogram. Each IIDS trains an allowed range (green area). If the threshold (green line) is surpassed, an alarm (red arrow) is emitted.

due to noise or the fact that training data might not cover all expected data ranges (cf. Section 2.2.1), we introduce an error margin to the learned thresholds as follows:

$$min_{\text{err}} := min - \frac{max - min}{2} \qquad max_{\text{err}} := max + \frac{max - min}{2} \tag{5.1}$$

The resulting thresholds min_{err} and max_{err}, which are then used for emitting alarms, effectively double the trained range. While this approach is highly opportunistic, it is universally applicable and could be theoretically adjusted easily. Yet, we refrain from doing so in the spirit of simplicity.

In the following, we present the design of our four SIMPLE IIDSs (MinMax, Gradient, Steadytime, and Histogram) based on these common characteristics. As visualized in Figure 5.1, each approach is inspired by natural ICS behaviors or typical attack patterns. We do not claim that our set of SIMPLE IIDSs is exhaustive; theoretically, many others exist. Nevertheless, the following four approaches are adequate to study whether the inherent complexity of IIDSs observed in related work is required. Still, it is essential to note that not every approach is equally well suited for each type of sensor/actuator.

MinMax. The minimum and maximum (MinMax) approach (cf. Figure 5.1a) detects whether a sensor's/actuator's current value exceeds the range observed in the training data and raises an alarm if any observation falls outside that range (± error margin). This approach is motivated by the intuition that process values of industrial systems relate to physical measurements or setpoints and thus usually obey certain limits. For example, temperatures below the freezing point of a liquid are not desirable for pumping it through pipes. Even if the physical setup does not limit the value range, operational requirements may impose restrictions on the allowed data range. For example, the pH value of a liquid may not exceed a specific range to be non-hazardous. Thus, we assume that an industrial system exhibits a class of values inside well-defined minimum and maximum limits.

Gradient. Following a similar intuition, the Gradient approach (cf. Figure 5.1b) detects whether a sensor's/actuator's slope exceeds the minimum and maximum observed during training (± error margin). While MinMax observes global changes, more subtle attacks occurring within these limits may remain unnoticed. For example, as shown in Figure 3.11c, the sensor is set to a high value within the operational

limits, yet far too abrupt, thus introducing a noticeable discontinuity. Hence, the Gradient approach assumes that ICSs have continual character, i.e., physical values such as temperatures cannot change at arbitrary speed.

Steadytime. Focusing on another temporal aspect, the Steadytime approach (cf. Figure 5.1c) detects whether a sensor/actuator remains static, i.e., does not change its value, for a shorter or longer time than seen during training (± error margin). This approach is motivated by the observation that an attack, e.g., freezing a sensor/actuator (cf. Figure 3.11b) such as a pressure relief valve, cannot be detected by checking whether a value or the velocity of a value change remains within certain boundaries (MinMax/Gradient). Since a steady state is difficult to define for noisy sensor data, Steadytime takes only process values into account if the number of distinct values during training is sufficiently small (≤ 10 in our case).

Histogram. Specifically targeting the occurrence of values, the Histogram approach (cf. Figure 5.1d) tracks their distribution within a fixed-sized window and tests whether it is in line with a histogram seen during training (± error margin). The underlying intuition expects a similar distribution of reoccurring values between process cycles. This approach can detect the existence and absence of frequent value changes, which the other three approaches cannot detect. The histograms are created by counting the number of times each distinct value appears in a sliding window. We merge them into a single histogram that covers each value's minimum and maximum occurrences across all distinct fixed-sized windows. The window size should match the duration of a process cycle, which could be automatically determined in an additional run over the dataset prior to training the histograms. Like Steadytime, Histogram only applies for process values with a few distinct values (≤ 10 in our case), as comparing two histograms value-by-value is unfeasible for noisy sensor data.

These four proposals stand in stark contrast to related work, which focuses on inherently complex approaches such as leveraging multiple Autoencoders [CLRM21] or fusing two IIDS directions into one solution [EMK20]. While further refinements to our IIDSs are possible, we explicitly focused on fundamental and minimalistic approaches to understand their effectiveness and assess whether industrial intrusion detection really needs to be complex or can be more S.I.M.P.L.E. instead.

5.1.3 Evaluation Setup and Methodology

Using our four SIMPLE IIDSs, we can now study the question of whether industrial intrusion detection needs to be complex or whether and to which extent SIMPLE approaches provide a viable alternative.

To this end, we implemented our four SIMPLE IIDSs (cf. Section 5.1.2) on top of the IPAL IDS framework (cf. Chapter 4), which allows comparing them to our reproduced IIDS implementations. To objectively quantify the performance of both SIMPLE and complex IIDSs, we refer to the same set of performance metrics as determined in Section 4.5.2. We utilize precision, recall, the F1 score, eTaP, eTaR, eTaF1, False-Positive Alarm (FPA), and the fraction of detected scenarios (Scen.).

Then, we evaluate our IIDSs on four industrial datasets based on physical testbeds and including attacks against the industrial process. First we leverage the SWaT dataset [GAJM16], representing a multi-staged water treatment system, as it is the most widely-used dataset. This experiment allows us to determine whether SIMPLE IIDSs are sufficient to detect most attacks and are competitive to complex approaches from related work in Section 5.1.4. Next, to evaluate the portability across industrial scenarios in Section 5.1.5, we additionally consider the WADI dataset [APM17], serving as an example of transferability to a water distribution scenario, the HAI dataset [SLYK20] modeling a power generation and storage facility which is an entirely different industrial domain, and the BATADAL dataset [TGT+18]. In Section 5.1.6, we then additionally examine the IIDSs' hyperparameters and computational efficiency to estimate the difficulty setting up our SIMPLE IIDSs. This was considered as another relevant property in Section 3.5.2.4. Lastly, we discuss and conclude whether IIDSs can indeed be S.I.M.P.L.E. or whether we find justification for complex approaches as from the literature in Section 5.1.7.

5.1.4 Sufficiency: SIMPLE on Par With Complex Approaches

First, we study whether SIMPLE IIDSs are *sufficient* to detect most attacks (cf. Section 5.1.1) and whether industrial intrusion detection must be inherently complex. To this end, we compare our approaches' detection performance to related work in an in-depth evaluation based on SWaT [GAJM16], as it is the most widely-used dataset in literature for this IIDS type (cf. Section 3.4.1).

SWaT consists of a training part of normal ICS behavior and a test part containing 36 attacks (cf. Section 2.3.1). We trained our four IIDSs on the training data omitting the first ∼22 hours during which the system stabilizes after activation. As seven out of SWaT's 51 sensors and actuators do not maintain the regular patterns observed during training, we excluded those from further evaluation. Skipping the stabilization phase [KS18, LCGN18, MW19, NJMP20, PFH+20, WWLQ20, XAY21] and omitting process values [PFH+20, WWLQ20, KS21, NTNK21, KKL22] in SWaT are common practices in related work. Notably, instead of excluding the process values, a process expert could manually adapt a pre-trained SIMPLE model, which is difficult for complex approaches such as the weights of a Neural Network.

High-level Ability to Detect Attacks

We use SWaT to gain a first assessment of the ability of our SIMPLE IIDSs to detect attacks and to compare them to complex approaches from related work. To this end, we analyzed all publications evaluating the SWaT dataset according to Scopus and Semantic Scholar as of April 20, 2022. Out of the 63 publications evaluating on SWaT, we consider those that provide sufficiently detailed information on which specific attacks they detect, resulting in eleven publications covering twelve complex IIDSs. Table 5.1 visualizes which SIMPLE (green and blue) and complex (red) IIDSs can detect which of the 36 attacks in SWaT.

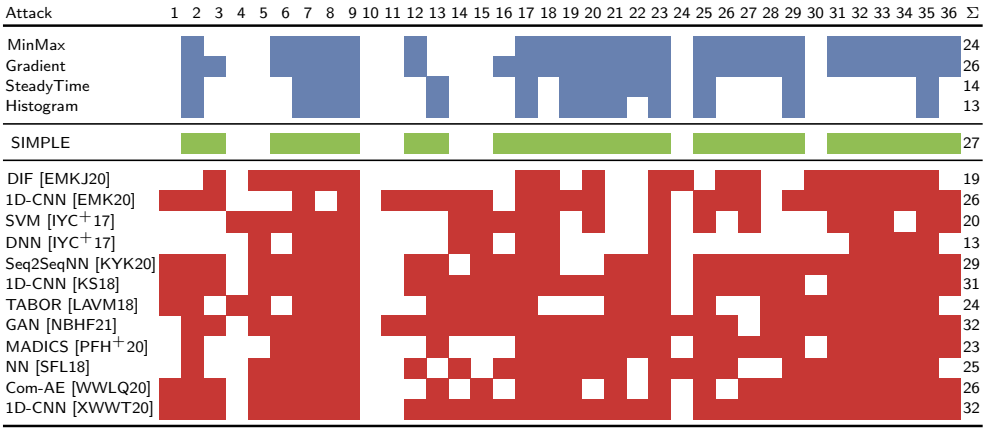


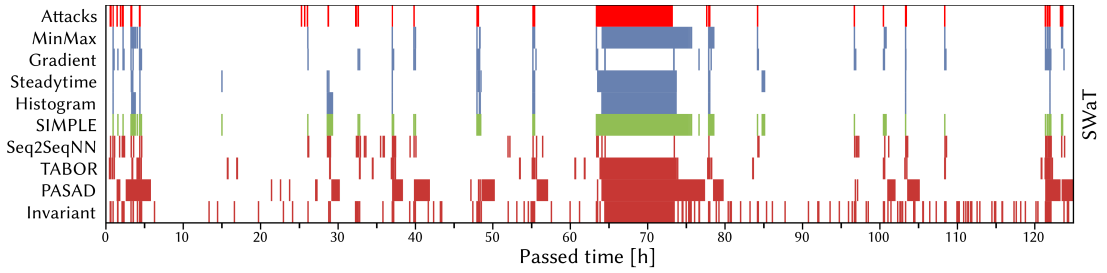
Table 5.1 Already a high-level analysis reveals that our SIMPLE IIDSs in combination (green) can detect 75% of attacks from the SWaT dataset, thus performing comparably to complex approaches from related work (red).

Our four combined approaches (denoted with "SIMPLE") detect a 75% of the attacks (27 of the 36 attacks contained in the dataset). Our arguably most simple approach (MinMax) alone can detect 24 attacks, and Gradient performs best by detecting 26 attacks. For comparison, the average number of detected attacks by related work is 25.0. Aggregating related work's capabilities, they detect all except a single scenario. However, in the twelve scenarios that *all* SIMPLE approaches detect, seven of the twelve complex IIDSs do not fully cover these seemingly easy-to-detect attacks. Regarding the nine attacks that are not detected by any SIMPLE approach, four (4, 10, 11, 14) have repeatedly been reported as non-detectable [SFL18, KYK20, XWWT20], and we observe inefficiencies in complex approaches too. Notably, not a single attack is detected by all complex approaches but not by our SIMPLE methods.

Takeaway. SIMPLE IIDSs seem on par with their complex counterparts. They detect up to 75% of the attacks on SWaT, but less compared to the best complex IIDSs <for the benefits of increased simplicity.

In-depth Comparison

Besides high detection rates, IIDSs should have a low false positive rate [Eta17]. To study how SIMPLE IIDSs fare against selected complex related work, we study their alert behavior in-depth visually in Figure 5.2 and alongside metrics in Table 5.2. We again provide combined results for "SIMPLE", where an alarm is emitted whenever any IIDS emits an alarm. Notably, this may fuse alerts into larger ones, which can result in fewer overall true- and false positive alarms. Since this evaluation requires access to complex IIDSs, we select those provided as re-implementations from the IPAL IDS framework, i.e., Seq2SeqNN [KYK20], TABOR [LAVM18], PASAD [AIA18], and Invariant [FPMC19]. These four IIDSs are a representative selection of the different detection methodology classes (cf. Section 4.1). Furthermore, reproducing more results is a time-consuming especially if no source code is available as it is the case for all other publications from Table 5.1.



Short attacks and alarms are enlarged to a minimum width of 0.15% for visibility.

Figure 5.2 A visual inspection of the alerts of SIMPLE IIDSs (green) shows thorough coverage with respect to the attacks for the SWaT dataset (red). The alerts of the four representative complex approaches (dark red) are less expressive and contain more false positives.

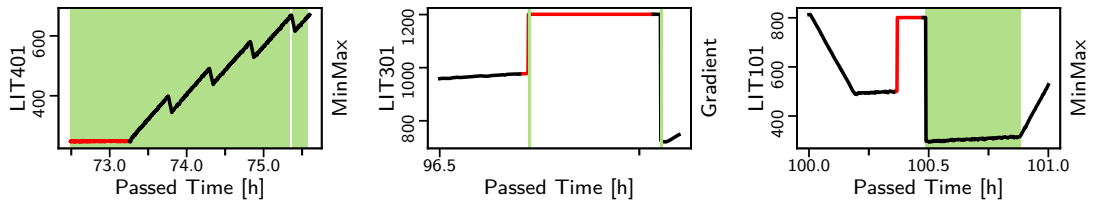
	IDS	Prec.	Rec.	F1	eTaP	eTaR	eTaF1	FPA	Scen.
SWaT	MinMax	75.0	80.6	77.7	67.8	47.1	55.6	7	66.7
	Gradient	29.7	0.3	0.5	20.5	6.0	9.2	27	72.2
	Steadytime	89.0	75.0	81.4	81.6	30.1	44.0	4	38.9
	Histogram	85.2	71.7	77.9	70.9	23.2	34.9	0	36.1
	SIMPLE	70.7	86.7	77.9	58.7	47.2	52.3	18	75.0
	Seq2SeqNN	44.0	10.9	17.5	42.8	47.2	44.9	36	75.0
	TABOR	81.5	74.7	77.9	49.1	18.9	27.3	27	55.6
	PASAD	32.4	71.5	44.6	16.0	4.9	7.5	14	44.4
	Invariant	97.3	69.5	81.1	55.4	32.7	41.1	185	86.1

Table 5.2 Across all relevant evaluation metrics, SIMPLE IIDSs (especially in combination) are competitive to the studied complex approaches from related work.

Upon visual inspection based on Figure 5.2, the alerts emitted by the SIMPLE IIDSs coincide with the attacks to be detected to a large extent. Furthermore, during non-attack periods, they do not emit large amounts of false alarms. Overall, the four complex approaches contain visually more false alarms and detect fewer attacks. Thus, while from a high-level view, the complex IIDSs under study appeared to detect slightly more attack scenarios (cf. Table 5.1), they come at the price of more false positives and consequently reduced utility.

Moreover, most related complex approaches' alarms are incomprehensible as they often exhibit multiple alarms around an attack (Seq2SeqNN) or alert over a long time range covering many attacks (PASAD). Our IIDSs, on the other hand, precisely overlap with the attacks and additionally allow to determine the potential malicious sensors and actuators through their locality property (cf. Section 5.1.1). For example, for 21 of the attacks detected by Gradient, the alerts stem from the process value indicated as the attack point in the SWaT dataset, thus providing a reliable starting point for subsequent incident response.

The individual metrics summarized in Table 5.2 confirm our previous observation that SIMPLE approaches detect large amounts of attacks (Scen.), emit few false alarms (FPA), and perform on par with related work. Notably, while Steadytime and Histogram detect fewer attacks, they simultaneously have the lowest FPA score of all



(a) Alerts may continue during process stabilization.

(b) Gradient does not raise continuous alerts.

(c) Attacks may have a delayed impact.

Figure 5.3 IIDS performance metrics can show a skewed picture when to be detected physical anomalies (green) are misaligned with the actual attack timing (red).

IIDSs under study. In terms of precision, recall, and F1 score, the MinMax, Steady-time, and Histogram IIDS outperform Seq2SeqNN and PASAD and perform roughly equivalent to TABOR. In the novel eTaF1 metric, SIMPLE even outperforms all four complex approaches by more than seven percent. These metrics are surprisingly good, considering the simplicity of the detection methods, which are not optimized to any metrics (a problem common for machine learning approaches [KWP+22a]).

Alert Behavior

Interestingly, while Gradient showed good detection performance in the visual comparison, it fares poorly for the individual metrics. The main reason for this phenomenon is that these metrics favor long attack coverage. To better understand the SIMPLE IIDSs mechanics, we take a detailed look at their detection phase.

We occasionally see alerts stretching significantly further (with interruptions) than the actual attack. In Figure 5.3a, the MinMax IIDS raises an alarm throughout the ICS's recovery phase since the process values still deviate from their normal values and fluctuate until stabilizing. The Gradient IIDS reveals another phenomenon in Figure 5.3b, leading to supposedly false alerts inherent to its design. As it indicates in- or declines, its alerts are short, which results in a poor performance with respect to metrics evaluating the attack coverage. While this method is precise in finding the actual beginnings and endings of attacks, it often raises an alarm shortly after an attack when the process quickly returns to normal operation. Finally, in Figure 5.3c, we observe effects that can occur after the actual attack ended (or where datasets are not precisely labeled). All of these effects result in insufficient attack coverage and false alarms, such that the good performance of IIDSs is not captured well by the available metrics.

Takeaway. All four SIMPLE IIDSs detect a sufficient number of attacks in the SWaT dataset. Combining the SIMPLE approaches allows detecting 75% of all attacks while visually emitting only a few false alerts. Moreover, raised (false) alarms are meaningful due to their local design and comprehensible models. Compared to related work, SIMPLE IIDSs can keep up with complex approaches in terms of the number of detected alarms, and especially false positives.

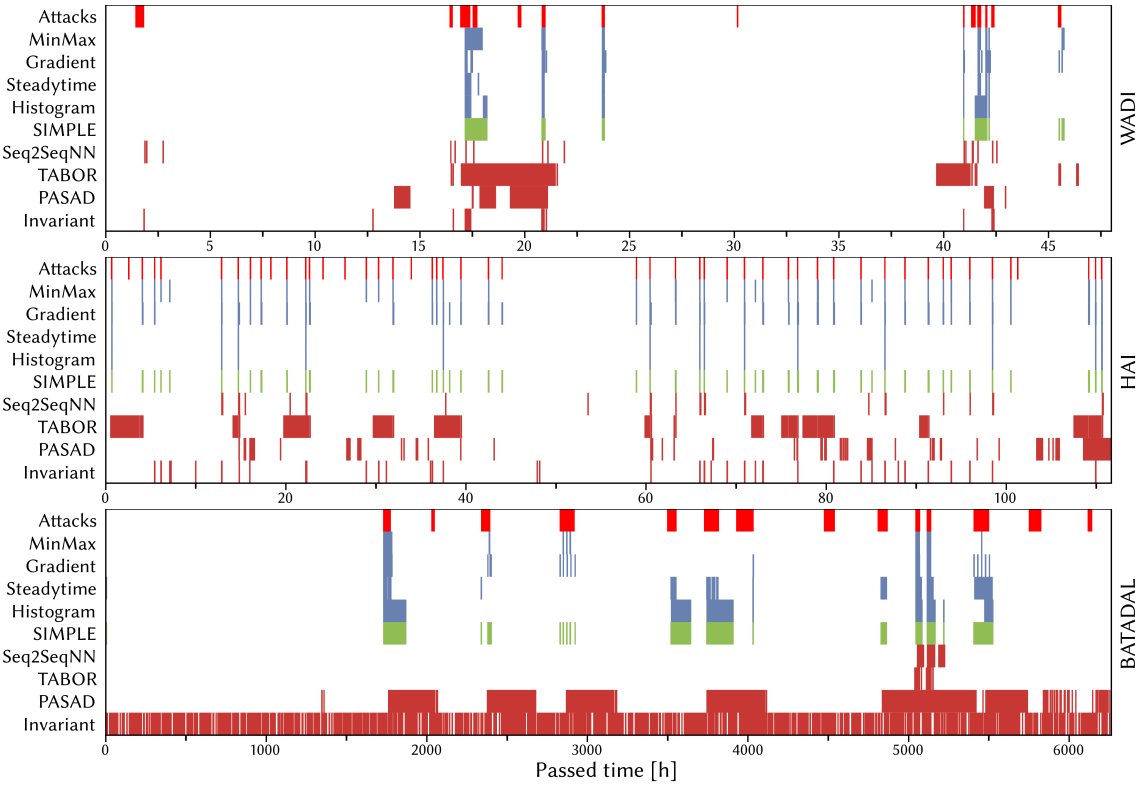


Figure 5.4 Porting SIMPLE IIDSs (blue and green) to three additional datasets shows their transferability to various industrial scenarios, while complex IIDSs (dark red) perform worse. Note that the results on the SWaT dataset have previously been discussed in Figure 5.2.

5.1.5 Portability: SIMPLE Works Effortlessly in New Settings

To ensure that IIDSs are widely applicable, they must be portable to various industrial scenarios and processes with a short training phase and without requiring (re-)inventions (cf. Section 5.1.1). Consequently, we show the portability of our IIDSs by applying them to three additional industrial datasets (WADI [APM17], and HAI [SLYK20], and BATADAL [TGT+18], cf. Section 5.1.3). We again compare our IIDSs to the four complex representatives from related work (Seq2SeqNN, TABOR, PASAD, and Invariant, cf. Section 5.1.4). Unlike ours, porting the complex IIDSs to the new datasets required extensive manual work to find suitable models and hyperparameters (cf. Section 4.5.2). Once more, we analyze the results both visually in Figure 5.4 and for various metrics in Table 5.3.

WADI. At first glance, with 64.3% of detected attacks overall, the SIMPLE IIDSs do not perform as strongly on WADI as on SWaT. However, even our worst-performing IIDSs, Steadytime, Gradient, and Histogram (50.0%), still outperforms PASAD (35.7%) and Invariant (42.9%). More importantly, we visually observe only two false alarms not closely related to an attack (at around 18h and 42h). While the complex IIDSs exhibit similarly few false alarms, their alarms are usually too imprecise to match a single attack.

	IDS	Prec.	Rec.	F1	eTaP	eTaR	eTaF1	FPA	Scen.
WADI	MinMax	70.2	41.4	52.0	74.8	47.4	58.1	3	57.1
	Gradient	78.6	0.4	0.9	69.6	18.1	28.8	6	50.0
	Steadytime	87.7	29.6	44.2	87.0	38.7	53.5	1	50.0
	Histogram	64.1	31.7	42.4	63.7	43.2	51.5	6	50.0
	SIMPLE	58.2	43.6	49.8	57.0	52.1	54.4	8	64.3
	Seq2SeqNN	44.4	13.4	20.5	45.4	31.3	37.1	7	64.3
	TABOR	19.1	43.7	26.6	14.9	13.0	13.9	3	57.1
	PASAD	16.4	23.9	19.5	5.4	4.3	4.8	3	35.7
	Invariant	90.0	21.9	35.2	92.3	32.6	48.1	2	42.9
HAI	MinMax	87.3	38.1	53.0	87.8	60.5	71.6	3	86.0
	Gradient	89.1	8.8	16.0	82.7	40.1	54.0	1	80.0
	Steadytime	85.7	10.8	19.2	82.5	17.2	28.4	0	28.0
	Histogram	85.7	10.8	19.2	82.5	17.2	28.4	0	28.0
	SIMPLE	87.0	39.8	54.7	86.0	61.5	71.7	4	88.0
	Seq2SeqNN	8.5	4.6	6.0	7.7	2.9	4.2	14	26.0
	TABOR	4.8	45.1	8.7	0.0	0.0	0	4	46.0
	PASAD	3.3	12.7	5.3	1.0	2.0	1.3	51	16.0
	Invariant	77.2	9.2	16.4	79.2	25.3	38.3	14	50.0
BATADAL	MinMax	100.0	10.7	19.3	100.0	30.5	46.7	0	42.9
	Gradient	96.7	9.9	18.0	89.8	30.1	45.1	3	50.0
	Steadytime	86.5	39.0	53.8	91.0	49.3	63.9	1	64.3
	Histogram	42.0	28.5	34.0	33.3	26.5	29.5	1	50.0
	SIMPLE	52.0	43.3	47.2	49.0	42.8	45.7	4	71.4
	Seq2SeqNN	34.2	5.6	9.6	27.0	6.9	11.0	1	14.3
	TABOR	78.5	6.9	12.7	77.7	14.3	24.1	2	14.3
	PASAD	20.1	52.1	29.1	10.5	21.5	14.1	32	78.6
	Invariant	27.2	45.5	34.0	18.2	74.9	29.3	865	100.0

Table 5.3 For all relevant metrics, e.g., detected attacks, our SIMPLE IIDSs transfer better to new industrial settings than complex related work. Note that the results on the SWaT dataset have previously been discussed in Table 5.2.

HAI. The SIMPLE IIDSs perform well for the HAI dataset with 88.0% of detected attacks, and nearly no FPA for Gradient, Steadytime, and Histogram. Notably, MinMax and Gradient perform especially well on HAI, thus showing that attacks can be detected reliably even with the simplest approaches. Complex related work, in contrast, falls far behind, and TABOR is even largely inapplicable, likely due to HAI's less pronounced regular patterns.

BATADAL. Our last dataset under consideration turned out to be challenging for related work (cf. Section 4.5.2). However, our SIMPLE IIDSs achieve excellent results on BATADAL with up to 71.4% of detected attacks and with few false positives. MinMax, Steadytime, and Histogram each feature just up to one FPA. Thus, SIMPLE IIDSs are even superior over their complex counterparts on this dataset.

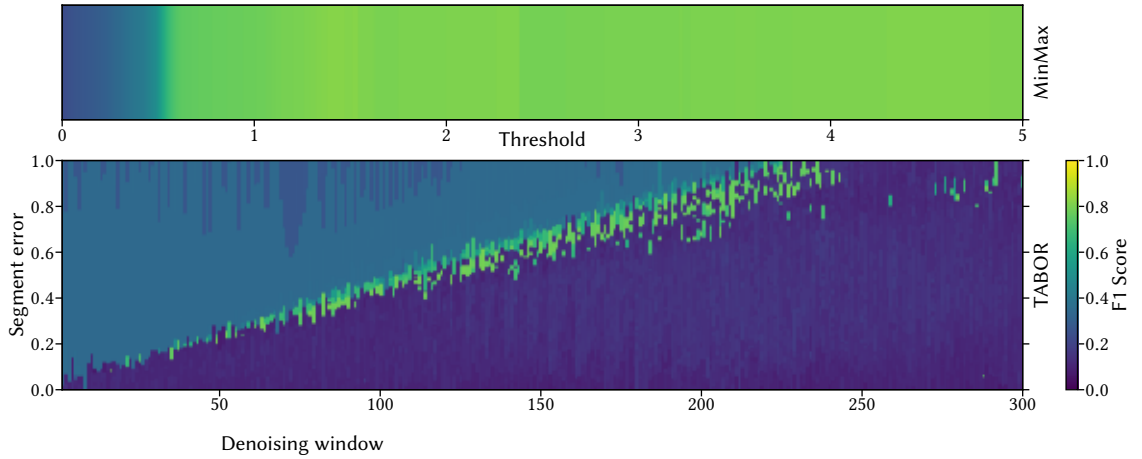


Figure 5.5 The impact of hyperparameters can vary significantly between approaches. While for MinMax on the SWaT dataset (upper plot), one hyperparameter is decisive for the entire performance, suitable configurations for TABOR (lower plot) are more challenging to obtain as several hyperparameters influence each other.

Takeaway. SIMPLE IIDSs, unlike their complex counterparts, can be ported to new datasets without major manual effort. Furthermore, considering that our approaches, in contrast to complex approaches, performed best on HAI (representing an entirely new industrial domain), this perfectly proves their portability.

5.1.6 Hyperparameters and Computational Efficiency

Since hyperparameters play a crucial role in the practical performance of an IIDS, we also consider this aspect here in a case-study for the MinMax IIDS. While we did not define explicit hyperparameters during the design of our SIMPLE IIDSs, the detection thresholds min_{err} and max_{err} can be adjusted to likely yield even better detection in other scenarios. In our extended version of the detection mechanism shown in Equation 5.2, we make the detection range variable by a *threshold* factor (t). If t is set to 1, this corresponds to the results evaluated before.

$$min_{\text{err}} := min - t \cdot \frac{max - min}{2} \qquad max_{\text{err}} := max + t \cdot \frac{max - min}{2} \tag{5.2}$$

To examine the influence of the hyperparameter on the MinMax IIDS, we measured its performance for 10.000 randomly sampled values of t between zero and five. In Figure 5.5, we compare MinMax and TABOR on the SWaT dataset and F1 score. We visualize the detection performance as a heatmap in dependence of relevant hyperparameters of these IIDSs. We selected TABOR, as it is an interesting candidate with respect to its hyperparameters, cf. Section 3.5.2.4. In contrast, MinMax was selected as it is likely the simplest detection methodology and we are interested in whether its results are largely determined by a good threshold or not.

For MinMax, we identify that its detection algorithm is largely independent from hyperparameters. For small values (below 0.6), there is a significant drop in detection

Dataset	MinMax		Gradient		Steadytime		Histogram	
(training size)	Train [s]	Live [ms]	Train [s]	Live [ms]	Train [s]	Live [ms]	Train [s]	Live [ms]
SWaT (415.0k)	11.1	0.03	14.5	0.03	10.9	0.02	18.5	0.05
WADI (784.6k)	65.5	0.07	73.6	0.09	60.6	0.06	89.8	0.11
HAI (216.0k)	9.8	0.04	12.8	0.06	9.1	0.04	12.1	0.06
BATADAL (8.8k)	0.3	0.09	0.3	0.10	0.3	0.10	0.3	0.10

Table 5.4 Measuring the speed during training and live detection of our four SIMPLE IIDSs reveals that they are reasonably fast compared to related work using, e.g., up to two hours on a GPU cluster [KYK20].

performance, but afterward, there are only subtle changes and the threshold has no significant impact anymore. Still, MinMax's maximal performance can even be slightly better (0.82) than with the default threshold if the hyperparameter is tuned perfectly indicating that the results presented before could turn out even better. In contrast, TABOR turns out to be difficult to configure (cf. Section 3.5.2.4), which is caused by interdependencies in two of its seven hyperparameters. Both hyperparameters influence the performance, and changes to one hyperparameter alter the optimal value of the other hyperparameter. Thus, only a combination of correctly set hyperparameters yields good configurations, which complicates setting up TABOR and explains our previous observation in Section 3.5.2.4 where only a few configurations yielded good performance. However, this difficulty can be avoided with SIMPLE methods as MinMax.

As last aspect in our evaluation, we consider the *efficiency* of our proposed IIDSs. More precisely, we measured the duration of the training and live detection parts on a single CPU core of an Apple M2 Pro laptop. As visible from Table 5.4, our approaches train a model in under 90 seconds for the largest dataset, cf. Histogram and WADI. During live performance, a single prediction, i.e., one line of each dataset, takes less than a fraction of one millisecond. Overall, our SIMPLE IIDSs are not only independent of the hyperparameter choice but also computationally efficient during training and live detection.

5.1.7 Discussion: Industrial Intrusion Detection Can Be SIMPLE

Wrapping up our evaluation, we recapitulate the promised properties of S.I.M.P.L.E. (sufficient, independent, meaningful, portable, local, and efficient) and discuss to which extent our proposed IIDSs capitalize on them.

Although we relied on straightforward detection methods and chose an opportunistic error threshold, our approaches proved to be on par with significantly more complex detection methods. Most attacks are detected for the SWaT and HAI datasets, and across all four examined datasets, our IIDSs are *sufficient* compared to related work (cf. Section 5.1.4 and 5.1.5).

Contrary to related work, which, e.g., requires unique parameterization [LAVM18], our SIMPLE approaches are *independent* of hyperparameters (cf. Section 5.1.6).

While theoretically, the margin of error or the Histogram's window size or MinMax's threshold could be additionally fine-tuned for better performance, even their default values, as evaluated by us, yield adequate performance.

The alerts emitted by our approaches largely coincide with the attacks (cf. Figure 5.2 and Figure 5.4). Furthermore, these carry *meaningful* insights for incident response, e.g., to which extent the trained threshold is exceeded (MinMax and Gradient).

As our SIMPLE approaches transfer to four diverse datasets (cf. Section 5.1.5), they have successfully proved to be *portable* across various industrial settings.

Already by design (cf. Section 5.1.2), all our SIMPLE approaches are *local*, i.e., operate on a per-sensor basis. As such, they are inherently able to identify the triggering value directly. To exemplary illustrate the resulting advantages for the SWaT dataset, which provides precise information on the attack location, MinMax and Gradient, e.g., could easily identify 18 respectively 21 attack locations correctly, easing attack identification and hence incident response.

Finally, the IIDSs are *efficient* with respect to computing resources (cf. Section 5.1.6) as they rely on elementary computational operations both during model creation and detection. For example, MinMax and Steadytime only perform an interval test, Gradient requires an additional subtraction for the slope computation, and Histogram counts and compares the last recently occurring process values. Besides computational efficiency, they are also optimized for a low memory footprint, thus easing their applicability in resource-limited industrial settings. MinMax and Gradient only store the minimal and maximal bounds on a per-sensor basis, while Steadytime and Histogram only require the thresholds per occurring value for each sensor.

Takeaway. The four exemplary approaches presented in this section satisfy the properties of a sufficient, independent, meaningful, portable, local, and efficient IIDS. Thus, we show that industrial intrusion detection can indeed be S.I.M.P.L.E., challenging the necessity of inherent complexity found across related work (I3).

5.1.8 Conclusion

Striving to achieve optimal detection of attacks, the research community proposed a wide variety of approaches to detect anomalies in the process state across different industrial domains. However, these approaches show an inherent complexity where high detection performance is accompanied by disadvantages such as a lack of model and alert comprehensibility or a high demand for computing resources. Considering that IIDSs leverage the repetitive nature of physical processes, we wonder why simpler detection methods have not been considered so far.

To overcome this gap, we studied whether IIDSs can be S.I.M.P.L.E. (Sufficient, Independent, Meaningful, Portable, Local, and Efficient) instead of having to rely on complex models with all their disadvantages. Thus, we designed four exemplary minimalistic approaches, such as straightforward range checks. Surprisingly, as we show across four distinct industrial datasets, simplicity does not result unusable detection capabilities, as simple methods are on par with significantly more complex

related work. Simultaneously, simple approaches offer highly beneficial properties such as eased configuration, model and alert comprehensibility, and reduced computational overhead. Thus, these IIDSs provide a viable alternative to complex approaches, raising the question of whether slight increases in detection capabilities justify computational overheads and reduced utility.

Still, it remains open whether our results are constrained by the studied datasets (i.e., the included attacks that are too "easy" to detect) or whether SIMPLE IIDSs are inherently sufficient to detect cyberattacks. Regarding the usefulness of the datasets, such analyses were already conducted in related work [TETC20, LSAA23, SK25] (cf. also Section 3.1.2). Similar discussions take place in other domains, such as for network-flow based intrusion detection [MBJ+25]. Thus, further studies on the quality of utilized datasets and especially their representatives regarding the complexity of real-world attacks are necessary. To this end, evaluations on physical hardware or hardware-in-the-loop setups can provide valuable insights [SHK+24].

Next, when publishing a new IIDS leveraging, e.g., the datasets considered by us, its evaluation should at least consider the results of this work as a baseline. When looking into practice, we already see the philosophy of S.I.M.P.L.E. being taken up by other research projects, be it to invent new IIDSs [SAP24] or to critically review existing benchmarking datasets [LSW24].

In summary, we showed that IIDSs can be S.I.M.P.L.E., challenging the current necessity of complexity found across related work. Thus, future research has to investigate the raison d'être for a complex IIDS with respect to advanced and stealthy attacks, beyond limiting their evaluations to the datasets currently in widespread use, for which simple approaches suffice.

5.2 GeCo – A Generalizable and Comprehensible IIDS

Even though the four SIMPLE IIDSs, as demonstrated in Section 5.1, function well on the discussed datasets, they have one inherent weakness per design. Since they consider just a single process value at a time without correlations to its dependent values, sophisticated attacks could leverage this property and undergo their detection by attacking multiple process values at once. At the same time, the BATADAL dataset does not contain multi-point attacks, and here, SIMPLE IIDSs still miss four attacks. This raises the question of whether an acceptable increase in complexity can be justifiable if an IIDS enhances the detection performance while remaining usable and comprehensive for ICS operators. To this end, we aim to develop a novel IIDS that is targeted for *practical use* whilst tackling the issues of existing approaches as discovered by SIMPLE.

Concerning IIDSs from related work that consider correlations between process values, these can be roughly divided into two areas: First, data-driven IIDSs that learn purely on historical data samples and, therefore, usually base their detection algorithm on machine learning. On the other end of the spectrum, there are IIDSs that depend on external expert- or system-information provided e.g., in the form of

mathematical models of the ICS process. Thus, the latter require significant manual engagement by operators.

However, both directions exhibit natural and opposing advantages and disadvantages. Data-driven IIDSs, on the one hand, facilitate applicability to different ICS applications as long as the required training data is available (cf. Chapter 4). On the other hand, their machine learning background hinders the transparency required to comprehend alerts [FZB24]. In the end, a simple alert from the entire system does not suffice to identify false positives or localize anomalies or attacks [LGH+23]. In contrast, IIDSs relying on expert knowledge are limited with respect to transferability to new domains. These systems demand manual configuration based on scarce expert knowledge and thus risk operating with incomplete models. Still, as experts are an integral part of the training process, alerts promise to be more comprehensible, potentially leading to faster mitigation of attacks.

Large cooperations may tackle some of these disadvantages through e.g., additional personnel, but smaller facilities lack the resources for such measures. However, also smaller operators of ICSs, especially critical infrastructure, regularly become the target of cyberattacks. The attack on a Texas water facility leading to overflowing tanks for up to 45 minutes before detection, is only one of the recent cyberattacks on critical infrastructure [Lyn24]. To address these concerns, the upcoming European NIS-2 regulation even requires the deployment of appropriate intrusion detection for a broad scope of relatively small businesses [NIS].

With this contribution, we want to combine the advantages of explainability and automation into an IIDS. Thus, we want to offload the hard manual work of experts to computers, thereby avoiding human error or incompleteness in modeling. Meanwhile, alerts should remain easily explainable for quick reactions to anomalies. As ICS operators likely have a background in control theory, we consider state-space models that describe the physical behavior and dependence of each sensor and actuator individually [Che84], as a viable modeling strategy for, both, automatic learning and interpretability.

Concretely, we design, implement, and evaluate GeCo – a *ge*neralizable and *co*mprehensible IIDS that lowers the burden for ICS operators to deploy IIDSs. In contrast to prior works that rely on labor-intensive manual specification of state-space models [CLA+18, APQ+20, QGS+20], GeCo automatically derives appropriate models expressed as mathematical correlations from historical process data based on *function templates* suitable for various ICSs and domains. An alarm is thus always associated with one or multiple sensor readings, such that it can be easily verified by operators who know the control logic.

Based on related work, we justify in Section 5.2.1 why such an IIDS currently does not exist. We then design (Section 5.2.2) and implement (Section 5.2.3) GeCo, an IIDS automating the work of ICS experts, providing comprehensible alarms, and transferring to various industrial domains. Section 5.2.4 demonstrates GeCo's effectiveness in a detailed evaluation against five IIDSs, surpassing not only their detection performance in most metrics but also keeping up with expert-derived rules. Moreover, we prove its effectiveness, understandability, and ease of training across

four datasets and a user study. Ultimately, we discuss and conclude the achievements of GeCo in Section 5.2.5 and whether GeCo can still be considered a S.I.M.P.L.E. approach.

5.2.1 Related Work

The task of an IIDS is to notify ICS operators in case of a cyberattack without emitting too many false positives [AAM22]. Thus, detection performance is the primary subject of most scientific evaluations [GUC+18, LWW+23]. However, two other factors are also relevant, namely generalizability and comprehensibility. First, in the ICS sector, two facilities are rarely built identically, featuring heterogeneous hardware and a wide variety of industrial protocols [ORZ+24]. Therefore, the generalizability and transferability of an IIDS, i.e., its ability to apply to multiple ICS applications or even domains, is crucial (cf. Section 4.1). Secondly, IIDSs should provide guidance for operators to understand the alert, trace its cause, or discard false alarms [Eta17, WTvS+22, FZB24].

To understand whether existing IIDSs can satisfy the often ignored aspects, we investigate relevant publications published at major security conferences (cf. Table 5.5). Assessing these, it becomes apparent that existing IIDSs roughly fall into two categories: Mechanisms that are purely data-driven and those that make use of external system-knowledge provided by experts during the IIDS's training phase.

Data-driven IIDSs employ vastly distinct detection methodologies ranging from classical machine learning such as one-class SVMs [IYC+17], Neural Networks [KYK20], or hidden Markov models [AKPI18] to various ICS-specific approaches [AIA18, LAVM18, CZ19, FPMC19, WTvS+22, YHC+20]. One common advantage is the minimal amount of input required for training. They merely leverage historical data samples of the physical process, making them applicable in various use cases (cf. Chapter 4). Additionally, through their automation, they usually learn a complete model of the entire ICS, making it harder for attackers to circumvent detection. On the downside, as machine learning and especially custom detection methods are heavily used, understandability, e.g., through attribution methods, is difficult to achieve [FZB24]. At the same time, this opaque operation risks model overfitting as the trained model can hardly be humanly validated [KWP+22a, WWB+24].

Knowledge-based IIDSs are enhanced with expert knowledge, of which a large portion requires input in the form of mathematical descriptions (cf. Section 2.1.2) either as a state-space model or a physical model. Given that the foundation is an expert-provided model, the IIDS becomes comprehensible for the operator and even allows them to perform corrections to the model. On the contrary, the IIDS is limited to those ICSs with the necessary resources to precisely model their processes, resulting in IIDSs that are often specifically designed for one scenario [CLA+18, QGS+20]. Moreover, the models might be incomplete due to the amount of manual effort required by humans designing them. For example, modeling just the water flows in a scenario similar to our example [GAB+18, APQ+20], gives attackers the potential to hide their actions. Hence, the completeness and transferability of these IIDSs are severely limited.

	Paper	Venue	External Information	Comp. Model
Data-driven	Aggarwal et al. [AKPI18]	CPS-SPC	✗	✓
	Aoudi et al. [AIA18]	CCS	✗	✗
	Cardenas et al. [CAL+11]	AsiaCCS	✗	✓
	Castellanos et al. [CZ19]	ACNS	✗	✓
	Feng et al. [FPMC19]	NDSS	✗	✓
	Hau et al. [HL19]	CPSS	✗	✗
	Inoue et al. [IYC+17]	IEEE ICDM	✗	✓
	Kim et al. [KYK20]	CyberICPS	✗	✗
	Krotofil et al. [KLG15]	AsiaCCS	✗	✓
	Lin et al. [LAVM18]	AsiaCCS	✗	✗
	SIMPLE (Section 5.1)	ESORICS	✗	✓
	Yang et al. [YHC+20]	RAID	✗	✓
Knowledge-based	Adepu et al. [AM16a]	AsiaCCS	Process Invariants	✗
	Adepu et al. [AM16b]	IFIP	Process Invariants	✗
	Ahmed et al. [AOZ+18]	AsiaCCS	State-Space Model	✗
	Ahmed et al. [APQ+20]	WiSec	State-Space Model	✗
	Palleti et al. [PTS18]	J. Process Control	State-Space Model	✗
	Choi et al. [CLA+18]	CCS	Physical Model	✗
	Ghaeini et al. [GAB+18]	SAC	Physical Model	✗
	Quinonez et al. [QGS+20]	USENIX Sec	Physical Model	✗

Table 5.5 Research is split into IIDSs using data-driven learning methods and those depending on (manual) input, which usually leads to incomplete models due to human efforts.

It is difficult to judge whether data-driven or knowledge-based IIDSs yield better results due to inconsistent evaluation methodologies [GUC+18, FSL+22, LWW+23]. Yet, their competing advantages motivate us to combine their strengths in a system that can be configured automatically based on historical data, but is also generally applicable and delivers comprehensible alarms.

5.2.2 Design of GeCo

Manual approaches for constructing expert-based IIDSs do not scale with an increasing number of ICS components. Hence, the focus of this work is the development of a data-driven procedure that does not rely on domain experts but, at the same time, mitigates the current shortcomings of data-driven IIDSs (cf. Section 5.2.1). Capable of a precise prediction of ICS behavior, state-space models have long been considered a reliable method for system analysis [Che84, YB09, IM11]. In the context of industrial intrusion detection, they have been considered in proof-of-concept deployments [CAL+11], use case-specific scenarios still with human input [QGS+20], or only with linear models [CAL+11, AOZ+18]. However, their broader scalability, transferability, and explainability for IIDSs remain unexplored.

To close this research gap, we design GeCo relying on the automatic data-driven derivation of state-space representations. Their precise modeling of system behavior combined with clear comprehensibility of its alert decisions, automated construction,

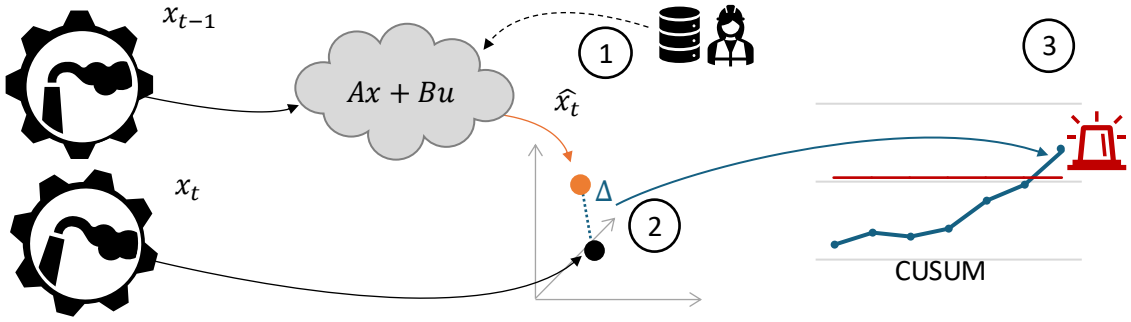


Figure 5.6 GeCo combines expert-knowledge with data-driven techniques to find state-space models. Then, it analyses the difference between predicted $\hat{x}_t$ and observed system state x_t.

and quick transferability among different ICS domains thus promises to alleviate the urgent need for flexible and easily deployable cybersecurity solutions [NIS, Lyn24].

5.2.2.1 High-Level Design Overview

As sensors and actuators of ICSs measure values belonging to correlated physical processes, data by different components is frequently coupled to each other. Assuming that one component is compromised, i.e., its data deviates from normal patterns, this can be identified by calculating the expected data based on a subset of correlated components. The core idea of this work thus consists of expressing the benign behavior of every component in a given ICS as a linear or non-linear combination of suitable sensor measurements and actuator states. Based on this mathematical model, the future behavior can be predicted and compared to currently observed measurements, thereby detecting compromises of one or more components. As all industrial processes abide to the laws of physics, this high-level approach promises to mitigate limitations with respect to deployments in different scenarios.

Concretely, we propose GeCo, which is based on automatic learning of a so-called *state-space model* [Che84, IM11] describing the temporal behavior of an ICS's underlying physics. Our model is based on a set of generic *function templates*, which are fitted against historic ICS data to find the equation best describing the observed ICS behavior (cf. ① in Figure 5.6).

Once learned, the state-space model can be employed to power GeCo. First, the sensor state x_t of the ICS at time t is measured. Then, ② GeCo predicts the state $\hat{x}_t$ based on the trained model and previously extracted sensor state x_{t-1} and actuator state u_{t-1}. GeCo ③ keeps track of the cumulative distance Δ between the predicted and measured states in the weighted past. If this cumulative distance reaches a predefined threshold, an alarm is raised as the system has abnormally different behavior compared previous observations.

5.2.2.2 Modeling the Behavior of an ICS

GeCo models the behavior of ICSs leveraging state-space representations. Generally, feedback flows within control loops determine its future state and can be mathematically modeled with state-space models [Che84, IM11]. This model describes how the state of a system component evolves over time $t \in \mathbb{R}$. It depends on the observed state $x_t \in \mathbb{R}^n$ and the values of the control input $u_t \in \mathbb{R}^p$ of an ICS containing n measured physical values and controlling the process with p actuators (cf. Figure 2.2). Data of sensors and actuators $y_t \in \mathbb{R}^n$ can be modeled as a combination of the system state x_t and the control input u_t. State-space models are defined by the following equation [Che84] that describes the state based on its current observation and the control input with two matrix functions $\mathbf{A} : \mathbb{R} \to \mathbb{R}^{n \times n}$, and $\mathbf{B} : \mathbb{R} \to \mathbb{R}^{n \times p}$:

$$x_t = \mathbf{A}x_{t-1} + \mathbf{B}u_{t-1} \tag{5.3}$$

The $\mathbf{A}$ matrix models the behavior of the physical process under undisturbed conditions. Depending on the dynamics of the physical system, such undisturbed conditions exhibit different results. For example, while the water level of a tank remains constant without water inflow or outflow, the temperature loss of a heated system over time represents a system that changes without control input. A simplification is to model the change in undisturbed conditions by only depending on the current state. Then, the $\mathbf{A}$ matrix takes the form of a diagonal matrix:

$$\mathbf{A} = \begin{bmatrix} a_1 & & \\ & \ddots & \\ & & a_n \end{bmatrix} \in \mathbb{R}^{n \times n} \tag{5.4}$$

Here, a value of 1 e.g., describes the uncontrolled water level staying constant, while a value < 1, e.g., describes the decreasing temperature without external heating.

The $\mathbf{B}$ matrix, on the other hand, describes the influence of control inputs on the system. Accurately describing $\mathbf{A}$ and $\mathbf{B}$ enables the precise prediction of an ICS's behavior at any time. Consequently, any substantial deviation (besides noise or inaccuracies) from this prediction can be assumed to be an anomaly we aim to detect. We thus intend to automatically learn these state-space models on historical data logs without any manual engagement by domain experts.

5.2.2.3 Learning the State-Space Model

State-space models can be complex differential equations that may be generated by domain experts for IIDSs. Such manual effort quickly runs into cost and scaling limitations, both for small companies that lack expertise as well as larger companies with complex processes. The failure of a group of experts from Singapore's Public Utility Board to derive a system model for the SWaT testbed despite months of work shows how challenging this process is [WSJ+18]. Conveniently, GeCo only needs to predict changes over small time steps.

From now on, we consider the state-space model for each process value individually and refer to the *state-transition function* of the i^{th} process value as F^i. The main idea

of GeCo is thus to approximate these state-transition functions. The approximation of the **A** matrix in Equation 5.4 gives us a state-transition function of the form:

$$F^i(x_{t-1}, u_{t-1}) \equiv \hat{x}_t^i = a_i \cdot x_{t-1}^i + B(u_{t-1}) \tag{5.5}$$

$B(u_{t-1})$ is the change due to control inputs. Since interactions from control inputs can underlie various physical effects, we model these effects more precisely compared to our one-fits-all approach for the A matrix. Thus, we refer to a flexible set of *function templates* that are multivariant functions with any amount of control inputs u_{t-1} and parameters b.

To determine $B(u_{t-1})$ for a given process, GeCo exhaustively tests which function template best describes the state transition from x_{t-1} to x_t. Therefore, each potential function is fitted with parameters (a and bs) by the least squares optimization algorithm (we use SciPy's `curve_fit` function) and keeping track of the best fit according to the lowest mean squared error for the training data. The best function for each process value gives us the state-space model that we use during the attack detection phase.

For now, we only consider the following two additive B^+ and multiplicative B^* function templates of the form:

$$B^+ \equiv B(u_{t-1}) = b_0 + \sum_{0<i<l\leq p} b_i \cdot u_{t-1}^{a_i} \tag{5.6}$$

$$B^* \equiv B(u_{t-1}) = b_0 + \prod_{0<i<l\leq p} b_i \cdot u_{t-1}^{a_i} \tag{5.7}$$

These two sets of function templates consider up to their length l many control inputs. We observe promising results even when limiting l to reduce the search space. Yet, the list of function templates can be expanded by domain-experts to more accurately model specific physical phenomena. They could even be pooled together by different experts. New function templates can, after all, only improve GeCo's performance at the cost of slightly prolonged training.

5.2.2.4 Two Illustrative Examples

GeCo learns the physical processes of an ICS based on function templates. To better understand how GeCo works, we review two illustrative examples: (i) the water level in the processes introduced in Figure 2.1 and (ii) the water temperature in a tank with hot inflow as seen in the HAI [SLYK20] process.

Water Level. In particular, we look at the learned state-transition function for the `LIT101` sensor of the SWaT dataset [GAJM16] with a function length of 2 for the B matrix (cf. Section 2.1.1):

$$\hat{x}_t^{\texttt{LIT101}} = \underbrace{1.0 \cdot x_{t-1}^{\texttt{LIT101}}}_{A} + \underbrace{0.192 \cdot u_{t-1}^{\texttt{FIT101}} - 0.197 \cdot u_{t-1}^{\texttt{FIT201}} + 0.009}_{B} \tag{5.8}$$

The water level `LIT101` is described by the inflow and outflow meters, `FIT101` and `FIT201` respectively. The A-coefficient matrix of the mined function describes how

the system behaves if no control command is issued, i.e., the water level remains constant if no flow is recorded at `FIT101` or `FIT201`. The B matrix describes how the water level increases or drops if the in- or outflow meters measure movement. The factors 0.192 and -0.197 are automatically derived by the mining algorithm and the unit measured by the flow meters is converted into a change of water level for the given tank size. Finally, we see a mostly negligible corrective summand b_0 that likely stems from inaccuracies and noise in training data.

Water Temperature. To see how additional function templates can improve a model's accuracy, we look at the approximated temperature of a tank with an inflow of heated water. Currently, GeCo approximates processes linearly. However, we can extend the set of function templates with a template that describes the change in temperature of two mixed fluids (assuming a constant specific heat capacity):

$$B(u_{t-1}) = b_0 \cdot \frac{b_1 \cdot u_{t-1}^{a_0}}{b_2 \cdot u_{t-1}^{a_1} + b_1 \cdot u_{t-1}^{a_0}} \cdot (u_{t-1}^{a_2} - u_{t-1}^{a_3}) \tag{5.9}$$

Here, $u_{t-1}^{a_0}$ is the volume of the inflowing fluid, $u_{t-1}^{a_1}$ is the volume of fluid in the tank, and $u_{t-1}^{a_3}$ and $u_{t-1}^{a_2}$ are the respective temperatures. Indeed, when adding this function template and training on the HAI, GeCo identifies the correct control inputs and finds fitting parameters to predict the temperature `TWIT04` of the tank `TK03` in the HAI testbed, where `TIT01` is the temperature and `FT01Z` is the flow rate of the incoming warm water, and `LIT01` is its fill level of `TK03`:

$$\hat{x}_t^{\texttt{TWIT04}} = \underbrace{1.0 \cdot x_{t-1}^{\texttt{TWIT04}}}_{A} + 0.00217 \cdot \underbrace{\frac{9.70 \cdot u_{t-1}^{\texttt{FT01Z}}}{22.6 \cdot u_{t-1}^{\texttt{LIT01}} + 9.70 \cdot u_{t-1}^{\texttt{FT01Z}}} \cdot (u_{t-1}^{\texttt{TIT01}} - u_{t-1}^{\texttt{TWIT04}})}_{B} \tag{5.10}$$

Among all the possible combinations, the state-transition functions for `LIT101` and `TWIT04` learned by GeCo explain the physical process similarly to how an expert may have described the system [APQ+20]. Importantly, GeCo does not require an expert to describe the system dynamics. The automatic learning of state-space representations can uncover complex correlations across the ICS that even domain experts may not be aware of, making anomaly detection more robust.

5.2.2.5 Online Attack Detection with GeCo

The trained model is used to perform anomaly detection by analyzing the deviation of real-time system data from the predictions. To this end, GeCo periodically extracts the current state of the system x_t, and the control inputs u_t. Together with the learned state-space model, GeCo computes a prediction for the next system state $\hat{x}_t$ as described in Section 5.2.2.3.

For anomaly detection, GeCo compares the predicted state $\hat{x}_t$ and the later measured actual physical state x_t by calculating the absolute residue $\|x_t - \hat{x}_t\|$. Aiming to make the detection more robust, we not only consider the step-wise residuals, but take into account their development over time as recommended in related work [QGS+20,

UGC+16]. To this end, we apply the change detection algorithm CUmulative SUM (CUSUM), which accumulates the deviations relative to a specific target mean over time and warns whenever the accumulated sum exceeds a predefined threshold. This way, small gradual deviations are better picked up and errors within our predictions introduced by noise or inaccuracies in the modeling and mining step are compensated for. We again consider state-transition functions F^i individually for the CUSUM calculation as they may have different scalings and sensitivities to influences from the outside, resulting in the following definition:

$$\begin{aligned} &CUSUM_0^i := 0 \\ &CUSUM_t^i := max(0, CUSUM_{t-1}^i + \|x_t^i - \hat{x}_t^i\| - \delta^i) \end{aligned} \tag{5.11}$$

$CUSUM_t^i$ iteratively computes the cumulative absolute prediction error of F^i up to the time t. In each step, this value is corrected by the drift δ^i, which expresses a constant but expected drift tendency. This drift δ^i is set during the training phase as the mean residue plus one standard deviation. Next, T^i is the *detection threshold* for the state variable x^i that determines when an alert should be raised. For T^i, we use the maximum CUSUM value observed during training and consider a safety margin expressed by a scaling factor S.

$$T^i = S * \max(CUSUM_t^i) \tag{5.12}$$

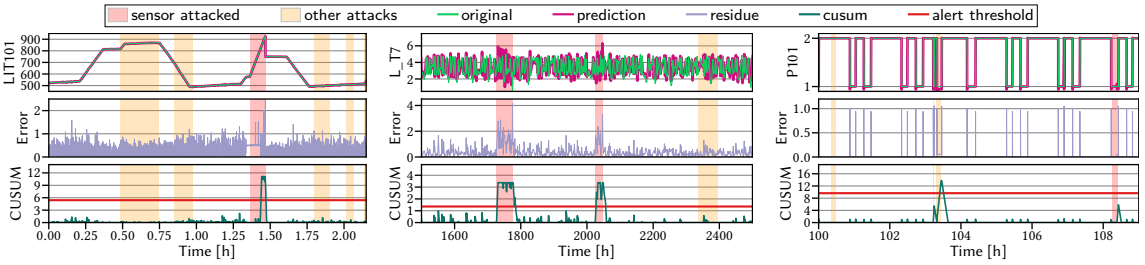
Lastly, and in line with common practice [UGC+16], we limit the absolute growth of the CUSUM score to prevent massive overshooting of the threshold and long recovery periods to the area below the threshold, especially if δ^i is small. Thus, a second hyperparameter G (growth factor) restricts the maximum $CUSUM$ value to $min(CUSUM_t^i, T^i + G \cdot \delta^i)$.

In summary, GeCo computes the $CUSUM^i$ of the deviations between the predicted $\hat{x}^i$ and the measured value x^i at each time step for every process value. If the $CUSUM^i$ value of any state variable surpasses the predefined threshold T^i, GeCo assumes anomalous behavior and raises an alert until the value has fallen below the threshold. These alerts not only inform the operator of anomalous activities but also provide context information to help locate the source of the disturbance as each sensor is monitored individually.

5.2.2.6 GeCo in Practice

To summarize the construction procedure, we demonstrate GeCo's operations as a well-functional IIDS. To this end, we show the alert behavior for three examples from diverse data types (linear, noisy, and binary data) to provide evidence that GeCo can function in all these situations in contrast to related work usually focusing on a single, linear use case [CAL+11].

Starting with Figure 5.7a, GeCo is applied to the level meter `LIT101` (cf. Section 5.2.2.4), while the testbed is under attack. This includes one direct attack against `LIT101`, steadily increasing its measurements by 1mm per second (cf. red area in Figure 5.7a). GeCo registers this compromise and correctly alerts during



(a) Example of a detected attack (red) by GeCo against the level meter LIT101 increasing its measurements by 1 mm every second.

(b) Example for the alert decisions of GeCo monitoring sensor L-T7 of the BATADAL dataset displaying noisy behavior.

(c) Example of GeCo monitoring the discrete values of the pump P101 which was forced to remain turned off by the attacker.

Figure 5.7 Applying GeCo to three exemplary scenarios, we find that it correctly alerts not only attacks within the example discussed in Section 2.1.1, cf. Figure 5.7a, but also demonstrates high detection confidence for ICS with volatile behavior and noisy residues, cf. Figure 5.7b, as well as components working only with discrete values, cf. Figure 5.7c.

the ongoing attack. This is initiated by higher than normal residues between predictions and real data (cf. middle plot). The final alert is launched by the corresponding CUSUM values distinctly overshooting the predefined threshold (cf. red line). The alert overlapping with the relevant attack shows that GeCo unveils the `LIT101` sensor's abnormal behavior. Meanwhile, no alert is raised while other system components are under attack (cf. yellow areas), such that GeCo also helps in understanding and localizing the source of anomalies.

To demonstrate that GeCo correctly alerts if the behavior of a component is less foreseeable, we examine the sensor `L-T7` from the BATADAL dataset [TGT+18]. This sensor monitors the water level in a tank of a municipal water distribution network with just a single measurement per hour. Thus, the residues between predictions and real data are much noisier (cf. Figure 5.7b). Still, both attacks targeting `L-T7` are identified by GeCo as both the residues and the CUSUM values are deviating clearly from their norm. Even though the value dispersion among the residues is much higher making it more difficult to detect attack. This confirms the assumption from Section 5.2.2.3 that even a model with a lower level of precision in the predictions suffices for accurate attack detection.

Lastly, we consider a discrete data example with the pump `P101` in Figure 5.7c. This example showcases the need for the CUSUM algorithm instead of deriving alert decision directly from the calculated residues. While the momentary deviations between predictions and real data plotted in the middle subplot of Figure 5.7c do not allow a definite revelation of anomalous behavior produced by the attack against `P101`, the CUSUM mechanism results in a successful attack detection. Therefore, we conclude that CUSUM, as a stateful change detection algorithm, is a necessary design element of GeCo.

5.2.2.7 Data Input Required by GeCo

To showcase the transferability of GeCo to various domain and use case, we further refine the input GeCo needs to train a model and perform online attack detection. Similar to other data-driven IIDSs, GeCo works fully automatically. The only required information that must be included in training and live datasets are historical time series of process values with regular time intervals. No attacks are required for training (anomaly detection). The amount of data varies across processes, as we later show in Section 5.2.4.3. If GeCo is supplied with these time series of process values and suitable hyperparameters, GeCo automatically infers the relationships between the process values during training without outside assistance. This sets GeCo apart from knowledge-based IIDSs that require manual and time-intensive labour by ICS operators to derive meaningful invariants [AM16a, AM16b] or state-space models [PTS18, APQ+20, QGS+20] (cf. Table 5.5). During live operation, GeCo only requires state snapshots, i.e., a list process values, in regular time intervals.

5.2.3 Implementation Details and Evaluation Setup

Bringing the theoretical design of GeCo into practice, we sketch our implementation in Section 5.2.3.1 and our evaluation setup in Section 5.2.3.2.

5.2.3.1 Implementation Details

As our implementation platform, we select the IPAL IDS framework from Chapter 4. Since IPAL is based on Python, we leverage the `curve_fit` function from the SciPy library to find suitable parameters for the functions defined in Section 5.2.2.3.

One aspect assumed in Section 5.2.2.3 is that it is well-defined which process values belong to the observed state x_t and which to the control input u_t. To keep the input from external experts minimal, we relax from that assumption and simply test all combinations. Also, during training of the state-space model and the CUSUM detection threshold, we use the first 80 % of the training data to learn the state-space model and use all 100 % of the data to calculate the CUSUM's drift and detection threshold. Thereby, we avoid overfitting the CUSUM detection method to data on which the state-space model was trained. The pseudocode in Listing 5.1 summarizes the implementation of GeCo.

5.2.3.2 Evaluation Setup

To obtain meaningful results [CDT21], we apply GeCo to the four datasets used before, namely SWaT, WADI, HAI, and BATADAL. Again, we compare GeCo against the previously reproduced IIDSs (TABOR, Seq2SeqNN, PASAD, and Invariant) and additionally our combined SIMPLE IIDS from Section 5.1. As performance metrics, we again use Precision, Recall, and the F1 score, their newer time-aware pendants eTaP (Precision), eTaR (Recall), and eTaF1 (F1) [HYKM22], the fraction of detected

```
def train(x, y):
  for target in process_values:
    for func in function_templates:
      for length = 0 To max_function_length:
        forall subsets of process_values with length:
          # Fit function to first 80% of the data
          xt = x[:80%], yt = y[:80%]
          parameter = fit(xt, yt, func, target, subset)

          # Test if new fit is the current best fit
          errors = func(x, parameter) - y
          if MSE(error) > current best model for target:
            continue

          # Calculate drift and threshold
          drift = mean(errors) + stddev(errors)
          cusum = threshold = 0
          for error in errors:
            cusum = max(cusum + abs(error) - drift, 0)
            threshold = max(threshold, cusum)

def test(state):
  for function in functions:
    diff = cur_state - function(prev_state, parameter)

    # Update CUSUM and limit infinite growth
    cusum = max(cusum + abs(diff) - drift)
    cusum = min(cusum, threshold + drift * growth_factor)

    if cusum >= threshold * scale_factor
      raise alert
  prev_state = state
```

Listing 5.1 Pseudocode of GeCo's for training and live testing.

Dataset	Function length k	Scale factor S	Drift factor G
SWaT	3	1.42	5.98
WADI	3	1.32	9.74
HAI	3	8.02	1.44
BATADAL	3	1.39	2.16

Table 5.6 Selected hyperparameters of GeCo for each dataset.

scenarios (Scen.) and the number of continuous FPA. Note that we still count an alert only as a false positive if it occurs at least 60s later than the attack ending since an attack's effect may persist even shortly after the attack's end, resulting from inaccurate labeling for some datasets (cf. Section 2.3.2).

Lastly, we describe how we set the three hyperparameters of GeCo. The maximum function length (k) is set to $k \leq 3$ for all datasets. The other two hyperparameters for CUSUM, the scale factor S and drift factor G, are set individually for each dataset as depicted in Table 5.6.

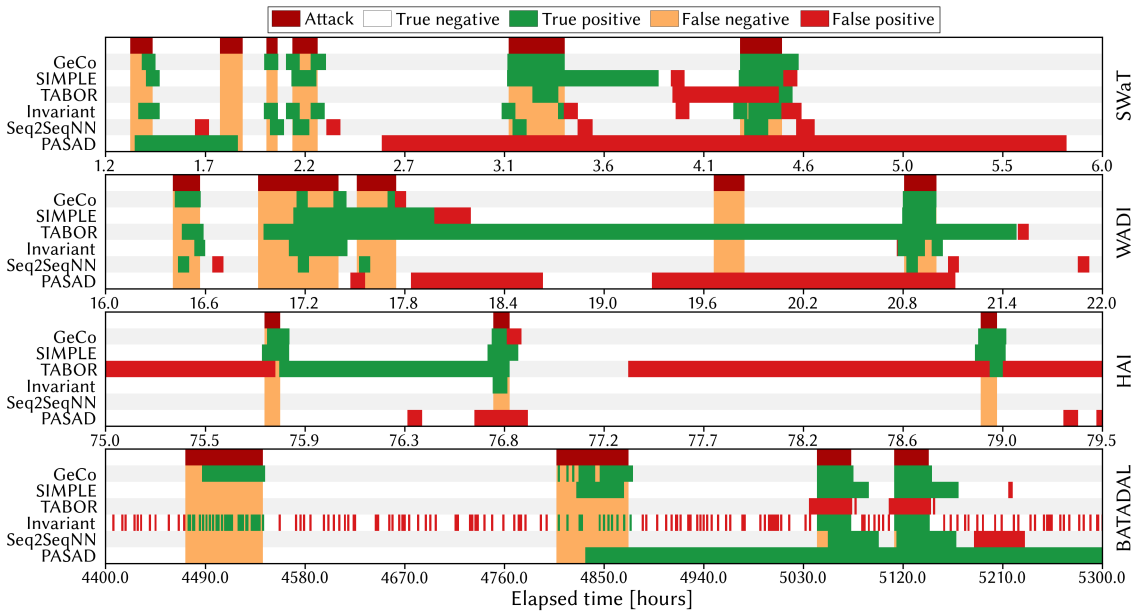


Figure 5.8 GeCo precisely indicates the attacks which can be difficult to infer from other IIDSs as their alerts do not overlap as consistently with the attacks as GeCo. The whole image of all alerts is contained in Figure 5.9.

5.2.4 Evaluation of GeCo

We have shown that GeCo detects cyberattacks in a wide variety of data types in Figure 5.7. Additionally, we now assess on a larger scale whether GeCo holds up to its claims to achieve a middle ground between required expert knowledge, transferability, comprehensibility, and automation.

First, we consider its detection performance in comparison to related work Section 5.2.4.1. Next, Section 5.2.4.2 concerns the comprehensibility of GeCo and its alerts. Section 5.2.4.3 measures the computational performance to judge whether the tradeoff between time invested by the computer or the expert is reasonable. As discussed before, the role and influence of hyperparameters of a new IIDS is worth investigating, which we do for GeCo in Section 5.2.4.4. Lastly, since training data may not be as clean as under real conditions, we consider how GeCo behaves if the training data is tainted in Section 5.2.4.5.

This extensive, five-fold evaluation provides great insights into the properties of GeCo and helps arguing why the added complexity compared to the S.I.M.P.L.E. properties is justified in this case.

5.2.4.1 Detection Performance

We now analyze GeCo's detection performance in a three-fold analysis. First, we consider its performance compared to related work and finish with a comparison to knowledge-based approaches.

	IDS	Prec.	Rec.	F1	eTaP	eTaR	eTaF1	FPA	Scen.
SWaT	*GeCo*	94.8	79.0	86.2	83.1	60.7	70.2	4	86.1
	SIMPLE	70.7	86.7	77.9	58.7	47.2	52.3	18	75.0
	TABOR	81.5	74.7	77.9	49.1	18.9	27.3	27	55.6
	Invariant	97.3	69.5	81.1	55.4	32.7	41.1	185	86.1
	Seq2SeqNN	44.0	10.9	17.5	42.8	47.2	44.9	36	75.0
	PASAD	32.4	71.5	44.6	16.0	4.9	7.5	14	44.4
WADI	*GeCo*	92.6	32.1	47.7	91.3	56.3	69.7	0	78.6
	SIMPLE	58.2	43.6	49.8	57.0	52.1	54.4	8	64.3
	TABOR	19.1	43.7	26.6	14.9	13.0	13.9	3	57.1
	Invariant	90.0	21.9	35.2	92.3	32.6	48.1	2	42.9
	Seq2SeqNN	44.4	13.4	20.5	45.4	31.3	37.1	7	64.3
	PASAD	16.4	23.9	19.5	5.4	4.3	4.8	3	35.7
HAI	*GeCo*	75.4	55.5	63.9	73.8	65.4	69.4	8	90.0
	SIMPLE	87.0	39.8	54.7	86.0	61.5	71.7	4	88.0
	TABOR	4.8	45.1	8.7	0.0	0.0	0	4	46.0
	Invariant	77.2	9.2	16.4	79.2	25.3	38.3	14	50.0
	Seq2SeqNN	8.5	4.6	6.0	7.7	2.9	4.2	14	26.0
	PASAD	3.3	12.7	5.3	1.0	2.0	1.3	51	16.0
BATADAL	*GeCo*	93.8	73.4	82.3	97.0	88.1	92.4	0	100.0
	SIMPLE	52.0	43.3	47.2	49.0	42.8	45.7	4	71.4
	TABOR	78.5	6.9	12.7	77.7	14.3	24.1	2	14.3
	Invariant	27.2	45.5	34.0	18.2	74.9	29.3	865	100.0
	Seq2SeqNN	34.2	5.6	9.6	27.0	6.9	11.0	1	14.3
	PASAD	20.1	52.1	29.1	10.5	21.5	14.1	32	78.6

Table 5.7 GeCo performs above average compared to relevant related work and across many metrics and datasets. Cells in grey resemble the best-performing IIDS for a given dataset and metric. The performance of related work was measured with the re-produced implementations from the IPAL IDS framework to obtain numbers for all metrics and datasets.

GeCo Compared to Related Work

One of the most important aspects defining the significance of an IIDS is its detection performance [ALS23], i.e., its capability to detect the majority of cyberattacks whilst emitting few false positives [AAM22]. To this end, we now discuss the results objectively with metrics in Table 5.7 and visually along their alerts displayed in Figure 5.8. As the algorithm of GeCo is deterministic, a single snapshot of its performance suffices.

While GeCo is not perfect, it outperforms existing IIDSs in 23 of 32 metrics across all datasets (cf. grey cells of Table 5.7). Most notably, on BATADAL, our approach performs optimally detecting all attack scenarios with zero false positives, a currently unmatched performance. Also, on SWaT, the most prominent evaluation dataset [LWW+23], the results are outstanding, especially since it detects the most attacks, together with Invariant, but with the fewest false positives. GeCo only falls behind in precision and recall on SWaT, which are contradictory, valuing either the most detected attacks (recall) or correct alerts (precision). Thus, it is unsurprising

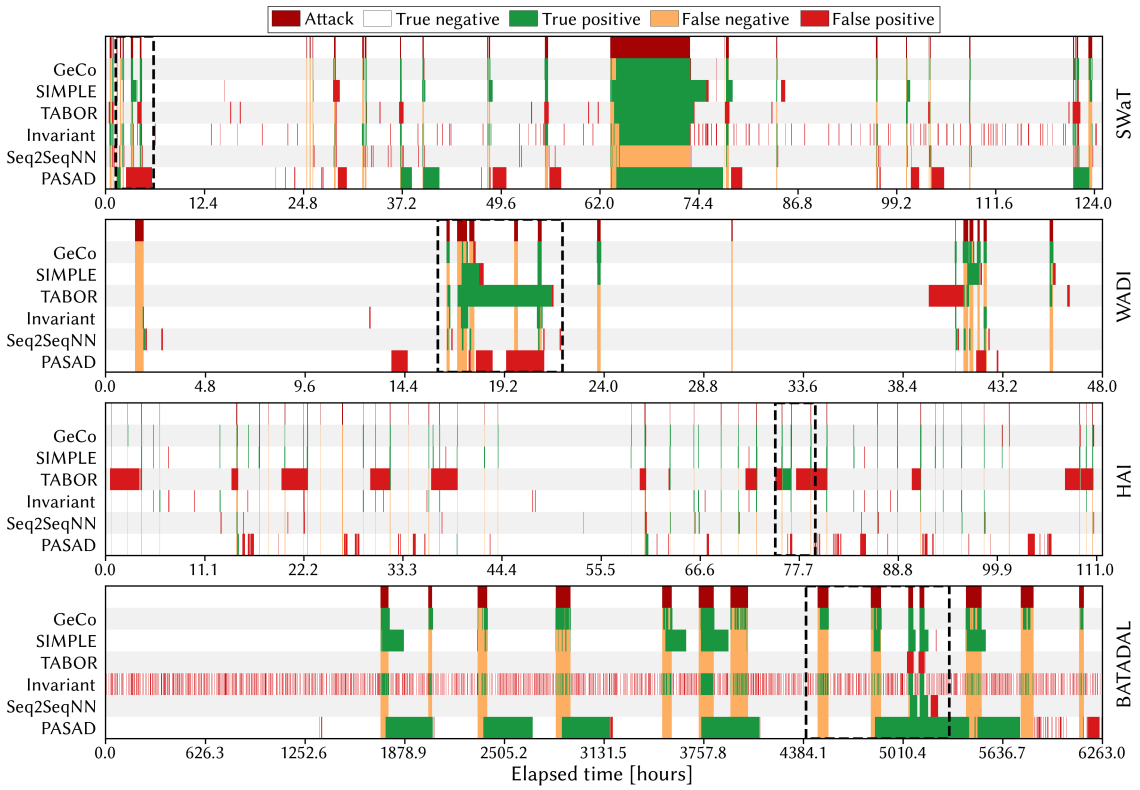


Figure 5.9 In our visual analysis of GeCo in Figure 5.8, we depicted a partial view on the alerting behavior due to better visibility. This figure shows the entire alerting behavior of GeCo and related work for all analyzed datasets. The areas marked with dotted lines correspond to the regions shown before in Figure 5.8.

that one IIDS from related work is better for each metric. Noteworthy, GeCo beats all IIDSs in the F1 score on SWaT, which mediates between these two extremes. On WADI, we exceed in eTaF1 and again have zero FPA whilst detecting the most attacks. On HAI, GeCo performs over average compared to related work. Here, GeCo again detects the most scenarios with few FPA while being dominant in the F1 score and being just 2.4 percent points behind SIMPLE in eTaF1. In the end, ICS operators have to decide which characteristic of an IIDS expressed through the different metrics best fits their needs [LWW+23], of which GeCo can satisfy a broad range with its detection performance.

For this reason, we also take a visual approach by looking at the individual alerting behavior of GeCo and related work. Figure 5.8 depicts an excerpt of each dataset (the complete figure can be found in Figure 5.9). GeCo precisely indicates the ranges of the attacks in the datasets with only little deviation and, especially during benign behavior of the ICS, no apparent false positives are raised. The two false positives of GeCo visible in WADI at around 17.8 and HAI at about 76.8 occur shortly after the attacks finished and are thus likely still related to fluctuations in the data from the attack. In contrast, the alerts from related work are not as consistent throughout all datasets, which highlights GeCo's ability to function well in all these scenarios.

Lastly, we take a look at attacks that are not detected by GeCo (false-negatives). On SWaT, six attacks are missed. SIMPLE also misses all these attacks, showing

SWaT dataset	Prec.	Rec.	F1	eTaP	eTaR	eTaF1	FPA	Scen.
GeCo	94.8	79.0	86.2	83.1	60.7	70.2	4	86.1
Knowledge-based	92.1	69.2	79.0	75.7	28.7	41.6	3	52.8

Table 5.8 Compared to a knowledge-based IIDS leveraging invariant rules, GeCo's performance is not inferior on SWaT.

that GeCo does not fail to identify easily detectable attacks [WTvS+22]. Furthermore, five of GeCo's six false-negatives on SWaT are neither detected by TABOR, Seq2SeqNN, nor PASAD. Across all datasets, only five false negatives of GeCo are detected by another IIDS. But, while Invariant detects two false negatives of GeCo on SWaT, Invariant also has large amounts of false positives. Additionally, while Invariant and Seq2SeqNN detect the first attack in WADI, this only happens at the end of the attack when the system is stabilizing again (cf. Figure 5.9). Thus, judging whether GeCo should also be able to detect all false-negatives is difficult as (i) other IIDSs might detect those only by chance and (ii) datasets contain attacks that are reported as non-detectable [KYK20]. Still, GeCo shows no systematic deficiency compared to related work with respect to false-negatives.

GeCo shows competitive performance, often better than related work. Notably, on two datasets, GeCo has zero false positives and just 4 on SWaT, which is important as false positives and alert fatigue are issues in practice [AAM22].

Comparison to an Experts' Knowledge-based IIDS

GeCo compares highly competitively to data-driven IIDSs. However, given that GeCo targets to supersede the hard manual work of experts, it is equally essential that this high level of automation does not results in a penalty of detection capabilities compared to manually created knowledge-based IIDSs. In contrast to data-driven IIDSs, where IPAL provides implementations and detailed evaluation results (cf. Chapter 4), obtaining similar input for a comparison with knowledge-based IIDSs is challenging. Either related work does not state any metrics [AM16a, AM16b, GAB+18], evaluates their IIDSs in practical experiments [CLA+18, QGS+20], or considers selected subparts of datasets [AOZ+18, APQ+20].

To compare GeCo to knowledge-based IIDSs, we consider publications that propose expert-created invariants (boolean equations) that must always be satisfied for the SWaT dataset. Knowing, for example, that `FIT101` measures the inflow after the valve `MV101` (cf. Figure 2.1), we can formulate the following invariant:

$$\texttt{MV101}\, open \Rightarrow \texttt{FIT101} > x \tag{5.13}$$

We identified a total of five publications that provide manually crafted rules for SWaT [AM16a, FPMC19, PAG17, PTS18, UMJA20] and a list of invariants by the SWaT dataset creators [iTr]. We collected and implemented 63 invariants, of which we omitted 16 due to a high number of false alerts and 14 because they were duplicated. In total, this leaves us with 33 invariants (listed in Table 5.9), which we

combined into a knowledge-based IIDS. These invariants do not require training and the IIDS raises an alert whenever any invariant is violated.

We measure the performance of the knowledge-based IIDS on the SWaT dataset. We compare the detection performance to GeCo in Table 5.8 and observe similar performance for both approaches. While the knowledge-based IIDS detects fewer attacks (Scen.), it yields one less false positive. Overall, when experts are being replaced with GeCo, we obtain a detection performance that is on par with labor-intensively created knowledge-based IIDSs.

5.2.4.2 Model and Alert Comprehensibility

Besides detection performance, the comprehensibility of emitted alerts are equally crucial in ICS applications [Eta17]. First, a transparent IIDS model simplifies supervision and prevents unintentional overfitting, as the operator can verify whether the fitted model actually resembles the physical process. Second, an IIDS has to provide a degree of transparency that allows ICS operators to comprehend why the IIDS raised an alert [WTvS+22]. Yet, effectively understanding and validating alerts in retrospect can be cumbersome for machine learning models [FZB24]. Comprehensibility also eases incident response as the cause for the fault becomes apparent. We analyze GeCo's comprehensibility on a theoretical level and by conducting a preliminary user study.

Localizing and Understanding GeCo's Alerts

Model comprehensibility has already been discussed earlier in Section 5.2.2.4. There, the IIDS discovered equations that accurately reflect the datasets' physical process. Thereby, operators with knowledge about the concrete ICS can verify the trained model and correct it if necessary.

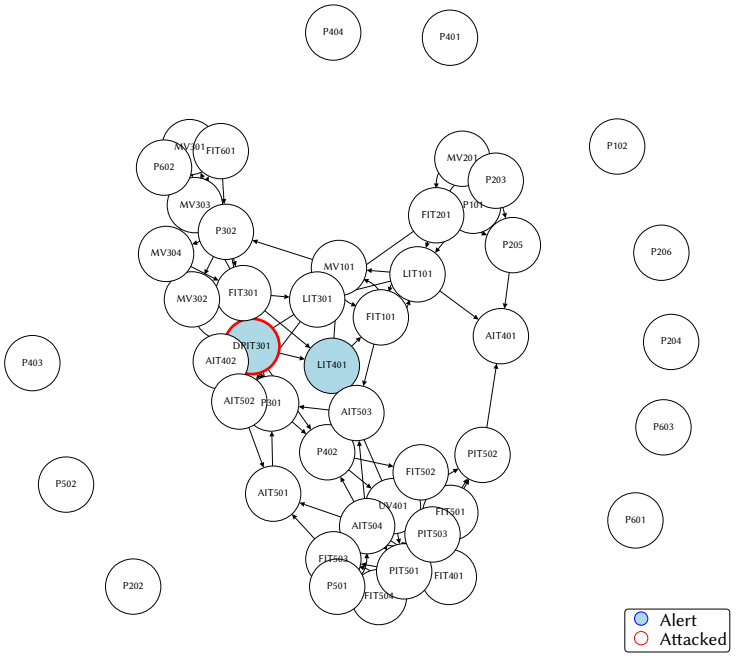
Now, we focus more on understanding the alerts. As our methodology, we show how an operator could trace the cause of an alert along with one exemplary attack. We consider attack number eight of the SWaT dataset (cf. red attack for SWaT in Figure 5.8 at 3.1h). Here, the attacker targets a pressure meter `DPIT301`, setting it to a higher value.

The first step in investigating an alert is determining which part of the ICS is under attack. GeCo makes this simple since we model each process value with exactly one equation. Thus, our intuition is that the attacked location can be narrowed down by the equation(s) that diverge. For example, an attack modifying the `LIT101` value likely violates the corresponding equation (cf. Equation 5.8), as demonstrated in Section 5.2.2.6, or an equation that depends on `LIT101`.

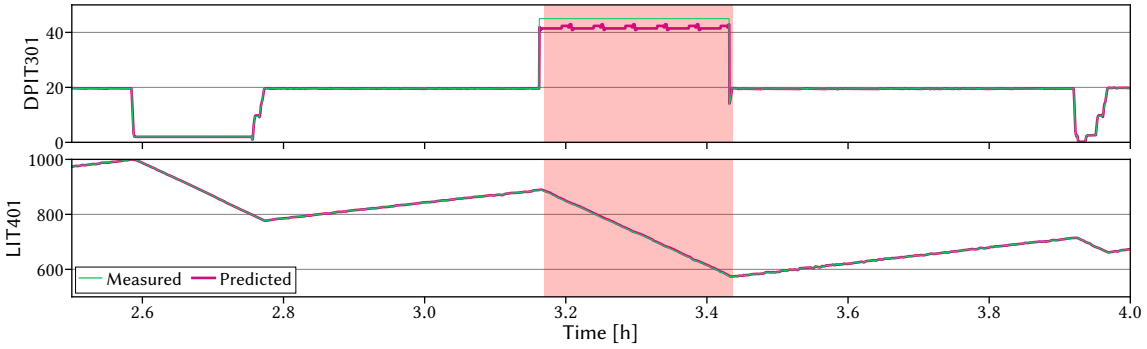
We now validate this intuition along with an example depicted in Figure 5.10. We start by drawing the dependencies between the equations in Figure 5.10a. The nodes represent the different process values. We draw an edge between two nodes x1 and x2, if the state-transition function of x1 depends on x2. For example, the node for `LIT101` is connected to `FIT101` and `FIT201` (cf. Equation 5.8). The graph

ID	Invariants
1	LIT101<490 ⇒ P101 off ∨ P102 off
2	LIT301<790 ⇒ P101 on ∨ P102 on
3	LIT301>1010 ⇒ P101 off ∨ P102 off
4	LIT301<790 ⇒ MV201 open
5	LIT301>1010 ⇒ MV201 closed ∧ P101 off
6	FIT201<2 ⇒ P201 off ∧ P202 off ∧ P204 off ∧ P206 off
7	AIT201>260 ⇒ P201 off ∧ P202 off
8	AIT503>280 ⇒ P201 off ∨ P202 off
9	AIT202<695 ⇒ P203 off ∨ P204 off
10	AIT203>500 ⇒ P205 off ∨ P206 off
11	AIT402>330 ⇒ P205 off ∨ P206 off
12	LIT301<785 ⇒ P301 off ∨ P302 off
13	LIT401>900 ⇒ P301 off ∨ P302 off
14	LIT401<300 ⇒ P301 on ∨ P302 on
15	P401 on ∨ P402 on ⇒ FIT401>0.1
16	FIT401<1 ⇒ UV401 off
17	P401 off ⇒ UV401 on
18	UV401 off ⇒ P501 off
19	UV401 on ⇒ P501 on
20	FIT401<1 ⇒ P501 off
21	LIT101>810 ⇒ P601 on
22	LIT101<490 ⇒ MV101 open ∧ FIT101>0.1
23	LIT101>810 ⇒ MV101 closed ∧ FIT101≤1
24	P101 on ⇒ MV201 open
25	LIT301>1015 ⇒ MV201 closed ∧ P101 off
26	LIT301<785 ⇒ P301 on
27	FIT301>0.1 ⇒ P301 off
28	LIT401<290 ⇒ P301 on ∨ MV302 open
29	FIT401<1 ⇒ P301 off
30	MV101 open ⇒ FIT101>0.1
31	FIT401<1 ⇒ P301 off
32	LIT101<490 ⇒ FIT101>0
33	LIT301<790 ∧ LIT101>810 ∧ MV201 open ⇒ P101 on

Table 5.9 The 33 consolidated invariants from six sources [AM16a, PAG17, PTS18, FPMC19, UMJA20, iTr] that we combine into a knowledge-based IIDS.



(a) GeCo's alerts enable inferring in which part of the ICS an attack took place as nearby dependent process values indicate the presence of an attack in that region. Here, we show attack 8 of SWaT.



(b) Having identified the potential attack points, a closer investigation of the predicted and measured behavior gives further insights into the attack. Here, DPIT301 behaves differently than predicted.

Figure 5.10 GeCo enables operators to analyze attacks quickly without requiring a profound machine learning background. Here, the attack increased the value of DPIT301 to a fixed value above 40 [GAJM16], beyond the standard operating limit.

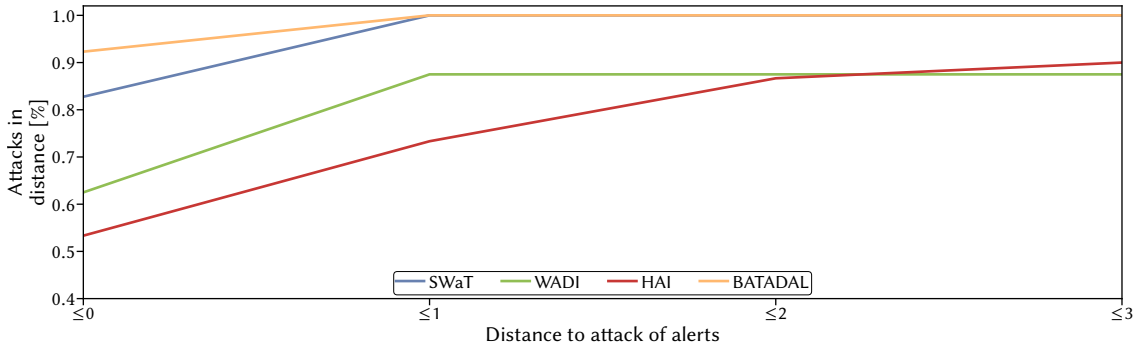


Figure 5.11 GeCo indicates most attacks in short distances to the formulas that break according to the dependency graph, cf. Figure 5.10a. Note that we omit unconnected values here, cf. P102 in Figure 5.10a, as their distance to other nodes is infinity.

is arranged by the Fruchterman-Reingold force-directed algorithm, drawing related variables closer together. Nodes not connected to the graph are not dependent on other values. We also mark where the attack took place (red border), here `DPIT301`. We highlight the process values that deviate from their predictions and lead to an alert by GeCo (blue circle). From this dependency graph, we can derive that the attack likely targeted `LIT401` or `DPIT301` as both predictions deviated from the observed behavior. This narrows the investigation to a small part of the ICS.

This methodology is just the first step to locating an attack. Next, we take a closer look at the measured and predicted values for `DPIT301` and `LIT401` in Figure 5.10b. For `DPIT301`, we observe a clear distinction between its predicted behavior (red) and measured behavior (green) during the attack. First, the predictions are much noisier than prior to or after the attack, which hints at a misbehaving system. Secondly, the predictions presume a lower value for differential pressure `DPIT301` than actually measured. Indeed, this coincides with the attack description, which explains that the attacker sets `DPIT301` to a higher value than normal (> 40). The alerting `LIT401` level meter then belongs to the tank after the pressure measurements and is thus also in close proximity to the attack. Overall, an ICS operator is able to correctly locate and comprehend an ongoing attack with GeCo without requiring profound machine learning knowledge.

Figure 5.11 shows that this methodology also works for other attacks. GeCo's alerts are, on average, over all datasets and attacks 0.5 hops away from the attack point and often identical to the target. Thus, GeCo facilitates comprehensibility as the alert usually coincides with the attacked process value.

Another aspect visible in Figure 5.10a is that most process values form a connected graph. Thus, more complicated or stealthy attacks modifying multiple process values at the same time are likely to be picked up by more than one equation. The SWaT, WADI, and HAI datasets already contain such multi-point attacks that GeCo successfully detects. Lastly, since the graph is mostly connected, an attack would need to spoof nearly all process values at the same time and correctly to each other's dependencies to stay hidden. This is a clear benefit of GeCo in contrast to knowledge-based IIDSs of related work often modeling the ICS just partially (cf. Section 5.2.1).

Participant	Industry Domain	Role
P1	Energy	Testbed Engineer
P2	IT Security	Consulting
P3	Engine Manufacturing	IT/OT Interface Engineer
P4	Maritime Manufacturing	IT/OT Software Developer
P5	Energy	OT Network Planning
P6	Automotive	Secure Product Development

Table 5.10 Backgrounds of interview participants.

GeCo's model and alerts are comprehensible as they closely correlate to the ICS's underlying physical process. In contrast, works such as Seq2SeqNN provide a single suspicion score for the entire process derived from a neuronal network's decision, and the comprehensibility of machine learning IIDSs is generally limited [FZB24].

ICS Engineers' View on GeCo

GeCo promises comprehensible and actionable alerts. To support our theoretical analyses, we perform qualitative interviews with experts from various industrial domains, similar to Fung et al. [FZB24]. We aim to assess the comprehensibility of GeCo's alarms, i.e., to what extent attacks are understood, and resulting actionability from ICS engineers' point of view.

Study Design. At the start of each structured interview, we asked the participant to name the industry domain they are working in and to describe their role. Afterward, we presented the basic industrial process from Figure 2.1, which was chosen as the process can be understood rather quickly. Participants were then introduced to the context-enriched GeCo alarms. During the interviews, participants were tasked to analyze four scenarios from the SWaT dataset, three attacks constrained to the investigated subprocess (3, 21, 30) and one false alarm. A separate attack (33) is used to introduce participants to GeCo. Participants were presented with the resulting alarms of GeCo in randomized order and tasked with explaining the attack and how they would react. We compared their answers to the actual attack descriptions of SWaT [GAJM16]. At the end of each interview, we asked participants about their general satisfaction with the context information provided by GeCo, especially the identification of relevant sensors. Our study design follows the guidelines of our institutional ethics committee and is exempt from ethics review (cf. Ethics Considerations). On average, interviews lasted 35 minutes.

Participants. We contacted potential participants with experience in OT environments from our professional and personal networks. We recruited six participants from various industry domains, as listed in Table 5.10. They were not compensated for their participation. The interviews were conducted virtually and in person.

Results. We start by assessing the comprehensibility of GeCo's alerts. Overall, the participants were able to identify the origin of the attacks with high accuracy as only one of the true positives of 18 scenarios was misinterpreted. The value-specific alerts of GeCo with low detection delays were key to narrow down the

search (P1, P2, P5, P6). This coincides with our theoretical analysis that GeCo identifies those values directly involved in or influenced by the attack (cf. Figure 5.10a). The false positive was correctly identified by three participants (P1, P5, P6) while two others stated that they can not find an attack (P2, P4). Later, P2 acknowledged that "there will always be false positives ... [But] if you are operating such plants, you will likely know whether a certain behavior is expected and could identify this as false positive."

All participants derived suitable mitigation measures or test procedures for further investigation of all attacks, including checking the real water levels or turning pumps or valves on and off to validate that the water levels are tracked as expected. We can thus conclude that GeCo generates actionable alerts.

Regarding the quantity of the presented data, the selection of the relevant process values by GeCo was perceived as helpful as there was "not too much information and [information] was cut away where not necessary"–P3. The way how the information was presented and pre-selected by GeCo helps "increase the reaction speed" since it would otherwise "take longer to obtain an overview of [all] the data"–P5.

Overall, our interviews emphasize GeCo's advantages. Three participants even proactively asked whether we are willing to test GeCo in their OT environments, highlighting the demand for comprehensible intrusion detection for ICSs.

5.2.4.3 Computational Performance and Usability

We are furthermore interested in the computational complexity of GeCo, which encompasses two dimensions. The first dimension concerns the live detection phase. On a server equipped with two AMD EPYC 7452 CPUs with 64 cores and 236G RAM, GeCo classifies state snapshots often in a fraction of 1ms, including the data-parsing overhead of the IPAL IDS framework, as shown in Table 5.11. Thus, GeCo can keep up with real-time critical applications.

The second dimension concerns the training phase. We begin with a theoretical analysis of the search space for the state-space model. For an ICS with n process values, t function templates with lengths of up to k, the number of combinations in the search space is given by (cf. pseudocode in Listing 5.1):

$$\texttt{combinations} := t \cdot n \cdot \sum_{i=0}^{k} \frac{n!}{(n-i)! \cdot i!} \tag{5.14}$$

We see that adding new function templates (t) impacts the number of combinations linearly, whereas the facultative impact of n on the complexity is potentially concerning. The other hyperparameters (scale and drift factor) do not affect the computational performance. To analyze practicality, we measured the training time on each dataset, as shown in Table 5.11.

Training times vary drastically between 3.6m for BATADAL and 46.9d for WADI. Except for WADI, this is comparable to complex IIDSs from related work, such as Invariant or Seq2SeqNN. Seq2SeqNN was trained on six GPUs [KYK20], highlighting

Dataset (Training Length, n, k)	Tests Equation 5.14	Training Total	Training One Fit	Live
SWaT (495k, 48, 3)	1.77M	15.1h	3.92s	0.2ms
WADI (784k, 122, 3)	73.87M	46.9d	7.02s	0.1ms
HAI (216k, 72, 3)	8.97M	1.3d	1.64s	0.9ms
BATADAL (8k, 42, 3)	1.04M	3.6m	0.03s	1.7ms

Table 5.11 Computational performance demanded by GeCo for training and live detection. Training times are normalized to a system with 128 CPU threads.

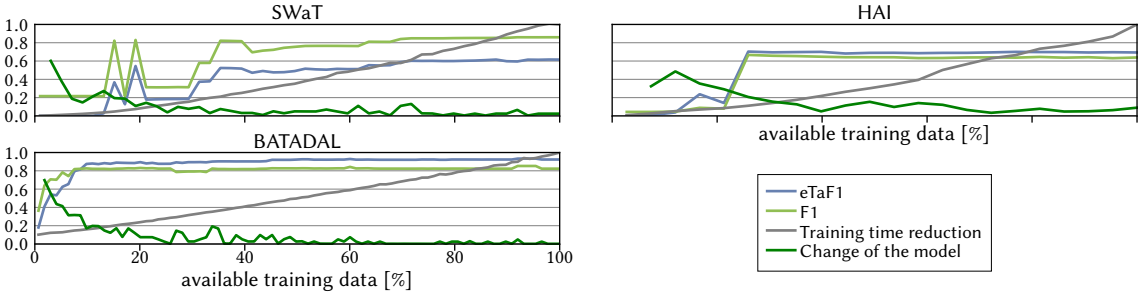


Figure 5.12 Decreasing the training data of GeCo has little impact on the detection performance, but significantly reduces the training time. The change of the learned state-space model can help estimating when the training can be stopped.

the immense hardware requirements. Due to a different architecture, Seq2SeqNN's two hours of training time are difficult to compare to our measurements. On the contrary, training Invariant on the same hardware takes on average 9.5h for SWaT, while GeCo requires 15.1h.

WADI, as the largest dataset with the most process values, represents an upper limit for GeCo's feasibility. Alleviating this situation, training has to be done only once and must not be repeated for hyperparameter tuning since this step is performed afterward (cf. Listing 5.1). Furthermore, it is possible to reduce the search area by restricting the search to components that are in close proximity to each other.

In the following, we analyze the impact of the amount of training data on *(1)* the training time and *(2)* the accuracy of the learned model. Therefore, we gradually increase the amount of training data accessible to GeCo while recording the training time. We then evaluate the learned model against the entire test dataset. We performed this analysis for all datasets except for WADI due to its high training demands (cf. Table 5.11). GeCo usually keeps its good detection performance even if the training data is reduced drastically (cf. Figure 5.12). For example, for BATADAL, the training time increases nearly linearly with the amount of training data such that with 10% of the training data, about 84% of the training time can be saved. Meanwhile, detection performance is only reduced by 0.002 (0.048) in F1 (or eTaF1). We see similar potential to reduce training time without impacting detection performance for SWaT and HAI.

While this property relaxes the issue of prolonged training, operators still need to understand when to stop training. First, we can infer from Figure 5.12 that having

```
def diff(a, b):
  diffs = []
  for target in process_values:
    if a and b use different function_templates: #1
      diffs.append(1)
    elif process variables of a and b differ: #2
      diffs.append(1)
    else: #3
      diffs.append(abs(
        (MSE(a) - MSE(b)) / max(MSE(a), MSE(b))
      ))
  return mean(diffs)
```

Listing 5.2 Pseudocode for calculating the model difference.

more training data does not diminish detection performance. For more insights, we now access how the learned state-space models converge.

To this end, we calculate the relative difference between two trained models as detailed in Listing 5.2. We calculate a score for each function learned by the model. If the models either use different function templates (cf. #1) or depend on different process values for the prediction (cf. #2), we attribute the highest difference (1). Otherwise (cf. #3) we calculate the relative difference between the mean squared error (MSE). The MSE is the difference between the training data and the model's prediction and indicates whether one fit is better than another. The difference can be between 0 and 1, with 1 denoting a maximal change between two models.

This methodology is motivated by scree plots used by PASAD [AIA18] or loss curves from machine learning. When considering the change of the model over the amount of training data (cf. green lines in Figure 5.12), we observe that saturation is reached if enough training data is provided. For example, according to the green curve, GeCo has finished training on SWaT after about 75% of the training data is used. This coincides with the moment when the F1 and eTaF1 curves (blue and red) stabilize. This metric thus provides a good estimate on GeCo's training progress.

5.2.4.4 The Influence of Hyperparameters

Finding hyperparameters in novel settings is a known issue in deploying anomaly detectors [PBB19, AMM20]. Looking toward deploying GeCo, an IIDS that requires complex hyperparameters to be tuned, risks yielding inferior performance in practice [WMV20, FSL+22], especially since ICS operators are not necessarily machine learning experts. While for established approaches such as Random Forests advice on how to tune their hyperparameters exists [PBB19], this is not the case for a novel IIDS such as GeCo. Thus, we aim to understand the influence of GeCo's hyperparameters by i) limiting the number of hyperparameters per design, ii) transparently assessing how hyperparameters affect GeCo's performance, and iii) by providing guidelines on how to choose hyperparameters and explain their effect to prevent exhaustive search in the following. In comparison, related work often disregards this topic entirely [CAL+11, KLG15, LAVM18, CZ19, HL19, YHC+20].

We assess the influence of GeCo's hyperparameters as done in Section 3.5.2.4. We extend our evaluation by examining GeCo under randomly sampled configurations

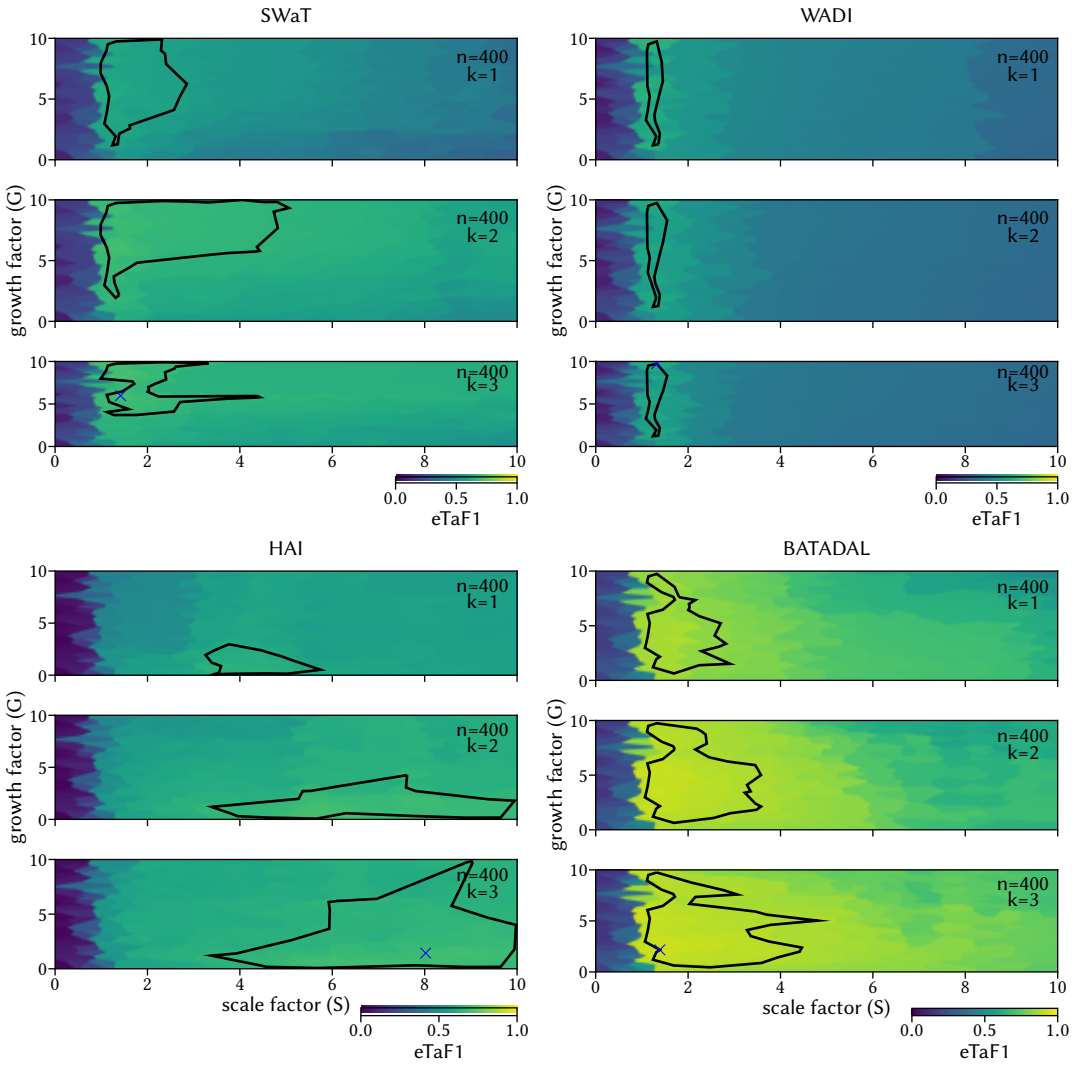


Figure 5.13 Finding hyperparameters for the GeCo promises to be an easy task since the approach shows good performance over wide areas. The hyperparameters used for our evaluation are marked by the × and the black line indicates the area within 5% eTaF1 of the maximum.

over all three hyperparameters, namely the growth factor G, the scale factor S, and the maximal function length k. For G and S, we examined a range between 0 and 10, and for the maximum function length k, we examined values from 1 to 3.

For BATADAL (cf. Figure 5.13), there are large regions with "stable" performance, i.e., where the influence of the hyperparameters is minimal. Only if the scale factor is chosen too small or too high, approximately below one or above four, we observe diminishing performance. A higher function length seems beneficial for increasing the maximum performance since the regions marked with black lines, indicating the area within 5% of the maximum, enlarge from $k = 1$ to $k = 3$. On this dataset, little trends regarding the growth factor are visible. SWaT and WADI show a similar behavior, but for HAI a higher scale factor and function length are beneficial. In contrast, TABOR performs only optimally if multiple hyperparameters depending on each other are set adequately (cf. Figure 5.5). As we demonstrated, operators of

Dataset	Prec.	Rec.	F1	eTaP	eTaR	eTaF1	FPA	Scen.
SWaT	−0.00	−0.02	−0.01	−0.01	−0.04	−0.03	−10	+0.03
WADI	−0.00	−0.03	−0.04	−0.00	−0.01	−0.02	+0	+0.00
HAI	−0.00	−0.13	−0.09	+0.01	−0.10	−0.06	−18	−0.12
BATADAL	−0.00	−0.03	−0.02	−0.00	−0.01	−0.01	+0	+0.00

Table 5.12 GeCo finds a good model even if the training data is tainted, i.e., containing two attacks in this case. The values show the difference between a model trained without attacks and one with two attacks in training. A negative number indicates a performance loss in percent points. Note that a positive number for Scen. is better.

GeCo likely have to spend little time and resources on tuning these hyperparameters and can rather invest the time to validate and optimize the trained model.

Besides this analysis, we advise on how to parameterize GeCo based on our experience. First, we recommend fixing $k = 3$ (if computationally feasible cf. Section 5.2.4.3) as it provides a good trade-off between training duration and detection performance. Once a first model has been trained, we recommend analyzing whether process variables exist for which GeCo cannot find suitable equations. These can often be identified by false positives and could either be initially ignored or tried to improve by adding more suitable function templates (cf. Section 5.2.2.4). Only then do we recommend fine-tuning the other two hyperparameters. Modifying the scale factor (S) affects the sensibility of the alert threshold. Here, a higher value raises this threshold and leads to fewer alerts. From our hyperparameter analysis (cf. Figure 5.13) we recommend initially setting S to about 1.5. The growth factor (G) can affect the length of alerts and smaller values shorten alerts if the CUSUM values decay slowly. Finding an appropriate G can be more complicated as we observed variance across our datasets (cf. Table 5.6). Yet, its impact on detection performance is usually smaller than that of S.

5.2.4.5 Effect of Tainted Training Data

The training data contained in the datasets may be superficially clean. In contrast, for real deployments, it is hard to guarantee clean data since, e.g., it could contain artifacts from faults or even unnoticed attacks. Moreover, due to changes in the network or variations of the process, the training might need to be repeated from time to time. Thus, we want to understand how GeCo performs if the training data is tainted, i.e., some attack data is contained within the training data.

Therefore, we adopt a similar methodology as proposed by Uetz et al. [UHH+24]. Instead of training GeCo solely on the dedicated (benign) training data, we additionally taint the training data with some of the evaluation data containing attack samples. More precisely, we split each evaluation dataset after the second attack. The first split, containing two attacks, is added to the training data while the remaining attacks are used for evaluation as before. In Table 5.12, we compare the detection performance of GeCo being trained on the intended benign dataset part

(baseline) and the tainted data. As we are left with two fewer attacks for the evaluation, we re-evaluated the baseline also on the same evaluation data missing those two attacks to keep the results comparable.

Overall, tainted training data has a comparable effect on all scenarios. The F1 scores decrease on average by 4.0 percent points, and eTaF1 decreases by 3.0 percent points. However, we cannot draw the expected conclusion that GeCo's performance worsens on tainted data, as the number of FPAs decreases significantly for SWaT and HAI. For example, we observe 18 fewer FPA on HAI. Meanwhile, GeCo only detects marginally fewer attacks in HAI while the other datasets see no difference or even experience a slight increasement (SWaT). We suspect that these results stem from potential overfitting in a too clean dataset, yet can not validate these speculations.

We conclude that the overall performance of GeCo is not significantly worsened by tainted training data. Thus, we expect GeCo to work decently outside the lab, as its training does not require to be perfectly clean of any anomalies.

5.2.5 Discussion and Summary

Having analyzed the various prospects of GeCo, we discuss whether GeCo has achieved a middle ground between data-driven and expert systems in Section 5.2.5.1. Finally, we conclude our work and answer whether GeCo can still be considered a S.I.M.P.L.E. IIDS in Section 5.2.5.2.

5.2.5.1 Discussion

After demonstrating the capabilities of GeCo, we now discuss whether we achieved a middle ground between required expert knowledge and automation. Furthermore, we identify potential for further improving this trade-off.

One substantial optimization potential of GeCo, especially for larger ICSs with many process values, results from the current brute-force approach to learn the correlations (cf. Section 5.2.4.3). Here, leveraging experts could further improve this situation, e.g., through guiding the search with further domain knowledge. For example, providing GeCo with a mapping of how certain process values are related to each other can significantly reduce the search space. Nonetheless, reducing the search space too much through human input risks missing essential dependencies between the process values necessary for GeCo's detection performance. This added manual effort is again a tradeoff between computational and human resources.

Another option for improvement is the function templates that only linearly impact the training time (Equation 5.14). Here, GeCo can accommodate complex templates, especially non-linear ones, if needed for niche real-world applications, as proven successful for detecting attacks against autonomous vehicles [QGS+20]. Next, while GeCo is concerned with simple time-invariant state-space models, which do not accurately capture all physical phenomenons, this still suffices for intrusion detection as GeCo discovered attacks from various datasets and domains: water treatments

(SWaT), water distribution (WADI and BATADAL), or from the electrical domain (HAI). Thus, GeCo achieves a detection performance better than many existing IIDSs on average, with further options for experts to improve on.

From this discussion, it becomes apparent that GeCo shifts the optimization possibilities back towards the user of the IIDS, i.e., the operators of an ICS. In contrast, IIDSs from the machine learning domain try to improve results with technical solutions, often increasing complexity, such as varying neural network architectures [KS18, DH21, KS21]. While these improve detection performance on paper, they move further away from a usable and comprehensible IIDS. In contrast, GeCo breaks this cycle and gets the ICS operator back into the loop to retain control over their IIDS and, ultimately, the security of their ICS. Overall, GeCo enables ICS operators flexibly deciding how much expert-knowledge or automation is necessary setting up the IIDS instead of solely relying on either expert knowledge or automation thereby achieving a new kind of middle ground.

5.2.5.2 Conclusion

GeCo revolves around the central idea that once one or more components of an ICS are attacked, the data of dependent components is not correlated to the data of attacked components in a way that was learned during the training phase, resulting in a detectable deviation of system behavior. To make use of this observation, we propose GeCo – a generalizable and comprehensible IIDS that automatically leverages detailed system knowledge and thus minimizes human input. More specifically, using a set of generic state-space model templates, GeCo automatically learns a system model of the ICS.

GeCo mitigates the need to manually design detection models, as is required from ICS operators in previous works, and remains comprehensible for ICS operators who supervise the IIDS. But for the advantages of addressing the previous issue, its detection methodology is much more elaborate than, e.g., MinMax's minimum and maximum checks. Consequently, the question arises whether GeCo can still be considered S.I.M.P.L.E. (cf. Section 5.1.1).

Since GeCo exceeds related work, including our combined SIMPLE IIDS, in most metrics, its detection performance can be considered sufficient. Moreover, we have shown that GeCo is likewise easy to configure and, thereby, mostly independent of hyperparameters. At the same time, its alerts remain meaningful as the cause for an attack can be traced down. Portability has been proven by applying GeCo to the same four datasets that were considered for evaluating SIMPLE. GeCo still acknowledges the locality principle. However, now each process value depends on a minimal set of other related values. Still, adjustments to each formula found by GeCo are possible without making the entire model obsolete, as would be the case for neuronal networks. Finally, while GeCo is efficient in live detection, this property lags behind in the training phase.

Overall, GeCo satisfies most of the S.I.M.P.L.E. properties while retaining high generalizability across various use cases and being easy to comprehend in its alert de-

cision process by human operators with no machine learning background. Consequently, GeCo paves the way towards giving the users of an IIDS control back over their tools to protect valuable physical processes in ICS and critical infrastructure.

5.3 Transferring IDSs to Maritime Radar Systems

The IIDSs considered so far have a strong focus on the water, gas, and electric domains, for which plenty of datasets exist (cf. Table 3.2). Yet, IPAL promises to transfer existing approaches to other domains with little additional effort. In that regard, one sector that has come increasingly into focus is the *maritime sector*.

A modern vessel is effectively a moving ICS that is fully digitized and autonomous shipping is also on the rise [KKG20, KK21]. For example, Electronic Chart Display and Information Systems (ECDISs) have superseded paper charts, communication equipment, Global Navigation Satellite System (GNSS) devices, and various sensors now interface with each other over shipboard networks [TJ19b, CPR+20]. Alongside digitization, the maritime industry has seen increased interconnectivity of its assets, with readily available satellite and mobile communication networks allowing for remote maintenance, chart updates, and crew access to the Internet [TJ19b]. Thus, the previously "water"-gapped shipboard systems become increasingly connected to the outside world, leading to new cyber threats, a serious concern that the nascent field of maritime cybersecurity research seeks to address.

In that regard, the vulnerabilities of fundamental maritime systems on board vessels have been intensively discussed in the literature [AAG19, CPR+20, HBP21, BKS+23] and demonstrated for various maritime systems, such as the Automatic Identification System (AIS) [BPW14, AG22, WRBW22], GNSS [BH17], VSAT [PMS+20], and radar [SRFM20, LMAR23], showing their possible impacts on the economy, ecology, and human lives. However, security research on *Marine Radar Systems (MRSs)* remains limited, especially regarding techniques to mitigate potential cyberattacks.

Radio detection and ranging (radar) enables the detection and localization of objects through the emission of electromagnetic waves and the reception of their reflection. While radar is critical in various applications, MRSs are particularly important for the navigation of aircraft and vessels. In the maritime domain, radar increases the safety of sea traffic by showing landmasses and physical objects on dedicated displays. Additionally, known landmarks or special Aids to Navigation, e.g., reflectors, allow radar to serve as a position fix. Especially in busy areas or low visibility conditions, radar is crucial to prevent collisions.

The trend of digitalization likewise affected MRSs, where digital radar displays have replaced their analog predecessors and their networking with other navigation instruments allows them to display a unified view of the vessel's state and surroundings to navigators [BDW09]. However, these communication channels over which sensitive navigation data is transmitted are rarely secured in practice [HBP21], resulting in serious risks of cyberattacks. Furthermore, as many marine navigation radars rely on non-standard, often vendor-specific, and proprietary communication protocols, establishing comprehensive cybersecurity solutions is challenging [HBW21].

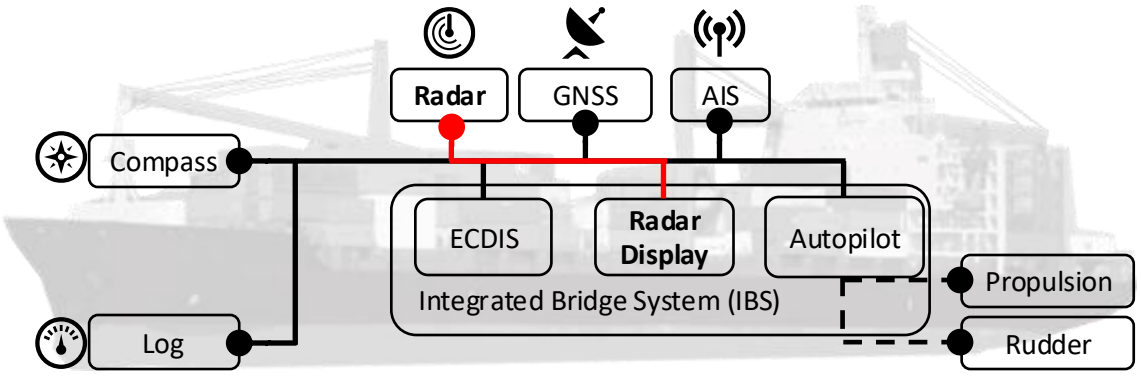


Figure 5.14 Modern vessels comprise a multitude of navigational sensors interconnected with a shared Ethernet network. The use of proprietary yet insecure radar protocols between the radar antenna unit and display, cf. path marked in red, poses a potential cybersecurity risk.

Leveraging our established tools and knowledge so far, we apply intrusion detection methods to MRS in the following. We begin with a brief background on MRS in Section 5.3.1 and then explain the possible attack vectors in Section 5.3.2. Since comprehensive datasets are missing for this research area so far, we develop our own RadarSec-Lab simulation environment in Section 5.3.3 and a Radar Attack Tool (RAT) to conduct cyberattacks in Section 5.3.4. With these tools at hand, we can set up our framework for attack detection in MRS in Section 5.3.5 and analyze the effectiveness of existing Intrusion Detection Systems (IDSs) applied to MRS Section 5.3.6. The identified gaps in detection capabilities motivate the invention of novel MRS-specific detection methods in Section 5.3.6. Summarizing our work in Section 5.3.8, we were able to quickly provide effective IDS solutions MRS.

5.3.1 Background on Digital Bridges and Marine Radar

The primary duty of a vessel's crew is a safe operation, which especially comprises navigation and collision avoidance. To this end, they are assisted by various digital systems that are bundled in the bridge of a vessel similar to a control room of an ICS. This observation show great similarities to ICSs. After introducing the architecture of modern vessels (Section 5.3.1.1), we shift our focus to MRS (Section 5.3.1.2) and the network protocols leveraged by them (Section 5.3.1.3).

5.3.1.1 Digital Systems Onboard Vessels

Safe operation of vessels demands precise and extensive environmental observations, i.e., measuring the heading with a compass or the speed with a log. On modern ships, an Integrated Bridge System (IBS) assists the crew with these tasks by combining navigation equipment, controls over steering and propulsion, as well as communication devices in a unified system [AAG19, HBP21] (cf. Figure 5.14). Typical components include a ECDIS, radar, autopilot, GNSSs, speed logs, gyrocompasses, and AIS which are explained in the following.

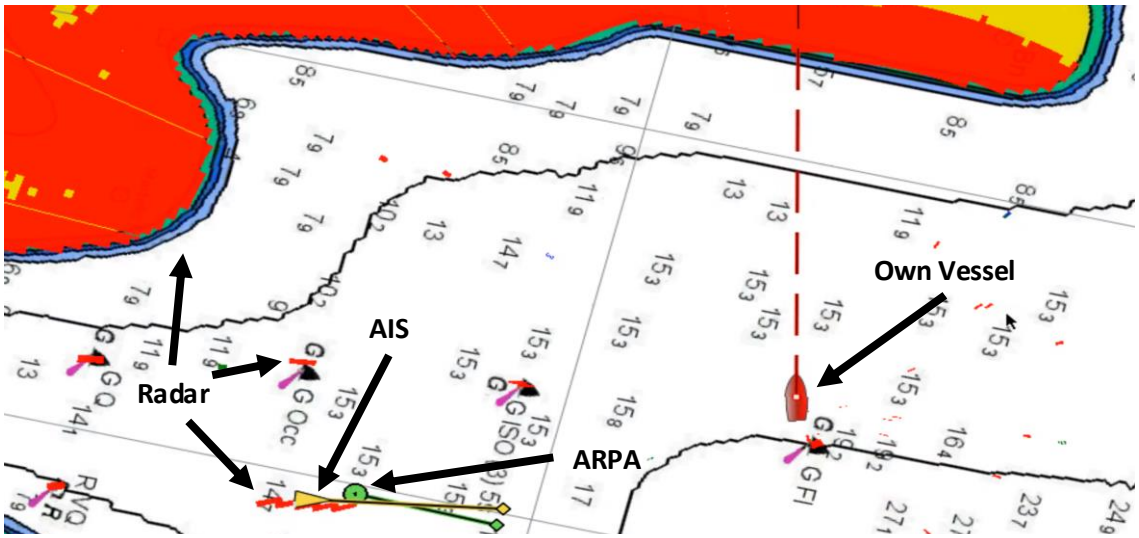


Figure 5.15 Radar visualizes landmasses, buoys, and surrounding vessels (red). Moving targets can be tracked with Automatic Radar Plotting Aid (ARPA) (green circles), which are additionally complemented by AIS broadcasts (yellow triangles).

Necessary sensors for navigation are nowadays fully digitized such as the GNSS, enabling worldwide localization. In addition, AIS transceivers periodically broadcast vessels' positions via radio frequencies to surrounding vessels increasing situational awareness and assisting in collision avoidance, especially in traffic-dense areas or during poor visibility conditions. All, sometimes even redundant, sensors distributed across the entire vessel get aggregated into a central IBS and visualized on nautical displays, e.g., ECDISs, to support the crew's decisions. Based on that data, an autopilot may control the rudder and propulsion, steering along pre-defined waypoints.

The foundation to transmit sensed data to the IBS is usually a regular Ethernet-based network, turning a modern vessel into a tightly integrated ICS [RCL11]. A common solution for the transmission of nautical data, except for radar images, is NMEA 0183 [IEC18a]. Developed initially as a human-readable, serial point-to-point protocol, this standard has been transferred to multicast UDP communication over conventional Ethernet [RCL11, LGH+18]. A similar trend as we observed for the digitization of ICSs in Section 1.1 and again NMEA 0183 over Ethernet is neither encrypted nor integrity protected.

5.3.1.2 Maritime Radar Systems

These systems contribute to either navigation or collision avoidance exclusively, but radar is capable of addressing both. While static landmasses or buoys are visualized, automatic tracking of moving objects, enabled by an Automatic Radar Plotting Aid (ARPA), can reveal another vessel's course (cf. Figure 5.15). Moreover, measuring the bearing and distance to charted objects also enables secondary means for localization even at night or in adverse weather conditions [BDW09]. Therefore, MRSs play a critical role in any IBS and because of their importance for collision avoid-

Vendor	Protocol	UDP	Auth.	Discovery	Scanlines*	Resolution*		
						Echo	Angular	Radial
Garmin	HD	✓	✗	static IP	1	1 bit	720	252
	xHD	✓	✗	static IP	1	8 bit	1440	705
Navico	3G	✓	✗	advertised	32	8 bit	2048	1024
	4G	✓	✗	advertised	32	8 bit	2048	1024
	BR24	✓	✗	static IP	32	8 bit	2048	1024
	HALO	✓	✗	advertised	32	8 bit	2048	1024
Raymarine	E120	✓	✗	advertised	1024	8 bit	2048	1024
	Quantum	✓	✗	advertised	1	8 bit	250	250

Properties marked with * indicate maxima.

Table 5.13 The eight marine radar protocols under study all rely on unprotected UDP multicast connections. Primarily the resolution and datagram encoding differ slightly.

ance, it is mandated by the International Convention for the Safety of Life at Sea (SOLAS) on passenger vessels and any vessel exceeding 300 gross tonnage [SOL02].

Technically, MRSs can be abstracted into three main components, cf. Figure 5.14. First, the transceiver antenna emits short directional electromagnetic pulses and receives the *echo*, i.e., the signal reflection of objects. The measured time difference between the pulse's emission and its echo's reception allows for estimating the distance to the detected object. The echoes of all objects detected by orienting the pulse in a given direction through the rotating antenna constitute a *radar spoke*. Second, a processor unit performs the signal processing. It is typically interconnected to the rest of the IBS to leverage exchanged sensor data, e.g., to keep the image in a north-up orientation using data from the gyrocompass. Lastly, the radar display, also called Plot Position Indicator (PPI), is the interface for the operators. It displays the radar image composed of the individual radar spokes and allows settings such as the scan range to be adjusted.

5.3.1.3 Overview of Radar Protocols

Unlike for other maritime data communication (cf. Section 5.3.1.1), no standardized radar protocol exists. Instead, vendors use proprietary protocols to transmit radar images to the IBS and send back control sequences. To conduct a comprehensive security analysis, we first investigate the inherent (dis)similarities of eight protocol families across three vendors by analyzing their respective implementations in the open source radar_pi plugin [Ope22] for OpenCPN [Ope23],

As shown in Table 5.13, all eight protocols share a similar data structure transmitting echoes' reception strengths to the display in scanlines, likely due to the common underlying measurement principle. Scanlines encode a single bearing or portion of an azimuthal degree as byte arrays where the first value corresponds to the bin closest to the radar, i.e., the first echo returning to the antenna, and subsequent values are progressively farther. The radar image's resolution depends on the scanline array's length (radial) and the number of scanlines (angular) required to make up a 360°

sweep. Due to size limitations in IP/UDP, a datagram usually encodes a fraction of the image (e.g., 32 scanlines per datagram for Navico BR24).

All eight protocols have in common that they rely on unprotected UDP transmissions posing a cybersecurity threat. Furthermore, discovering radar devices within a network is often made simple through the use of statically assigned IP addresses and the broadcasting of UDP packets. Overall, Navico BR24 shares the most common features, as can be seen in Table 5.13. Because its data and control structure are, furthermore, well documented [DBS11], it is chosen for our further research.

5.3.2 Taxonomy of Network-based Radar Attacks

Crucially, the Navico BR24 protocol and other commonly used protocols lack essential security features, thus making marine radar susceptible to cyberattacks. To this end, we first elaborate on the threat model in Section 5.3.2.1 after which we present potential attack vectors from Section 5.3.2.2 to Section 5.3.2.6 and discuss their consequences on nautical decisions.

5.3.2.1 Threat Model

We consider an attacker who intends to manipulate the navigation radar image shown in the IBS by manipulating the transmission of radar network packets. Tampering with the displayed contents may affect nautical decisions, reduces the ability to navigate securely or, in the worst case, endangers the vessel or its surroundings. While maritime networks are supposedly "water"-gapped, i.e., physically isolated, they still expose attack surfaces. Typical attack vectors, e.g., ranging from satellite communication to human factors, can lead to compromised devices and malware implants [TJ19a, CPR+20, HBP21, BUH+22]. Even a direct compromise of the radar unit is possible, e.g., by exploiting software and operating systems [SRFM20].

Still, executing attacks against MRSs is a non-trivial task. Referring to the MaCRA framework [TJ19b], we therefore assume any attacker to at least possess the resources of a $Tier_3$ threat actor. Such an attacker has the resources to exploit vulnerabilities, to gain access to the IBS and the necessary domain knowledge and testing environments to develop specialized attacks. For example, understanding of protocols in use and the victim's system as a whole. More precisely, we specifically focus on an attacker who has already gained control over an arbitrary network device with the ability to read and alter radar communication.

5.3.2.2 Attack Methodology

As proven in related work, MRSs are susceptible to various attack vectors such as hardware- or software-based exploits [SRFM20] or complementary threats from the domain of electronic warfare [LMAR23]. In 2020, Sviličić et al. conducted an extensive security assessment of radar systems on board two oil tankers [SRFM20],

detecting many vulnerabilities in their underlying operating system, some of which were critical enough to enable a complete takeover of the system.

Likewise, due to the typical lack of confidentiality and authentication mechanisms in the employed communication protocols (cf. Section 5.3.1.3), attackers may inject malicious commands or image data. While not demonstrated until today, such attacks could arbitrarily alter the displayed radar image on the PPI and deceive the crew's understanding of the navigational situation. Consequently, our focus is on network-based attacks.

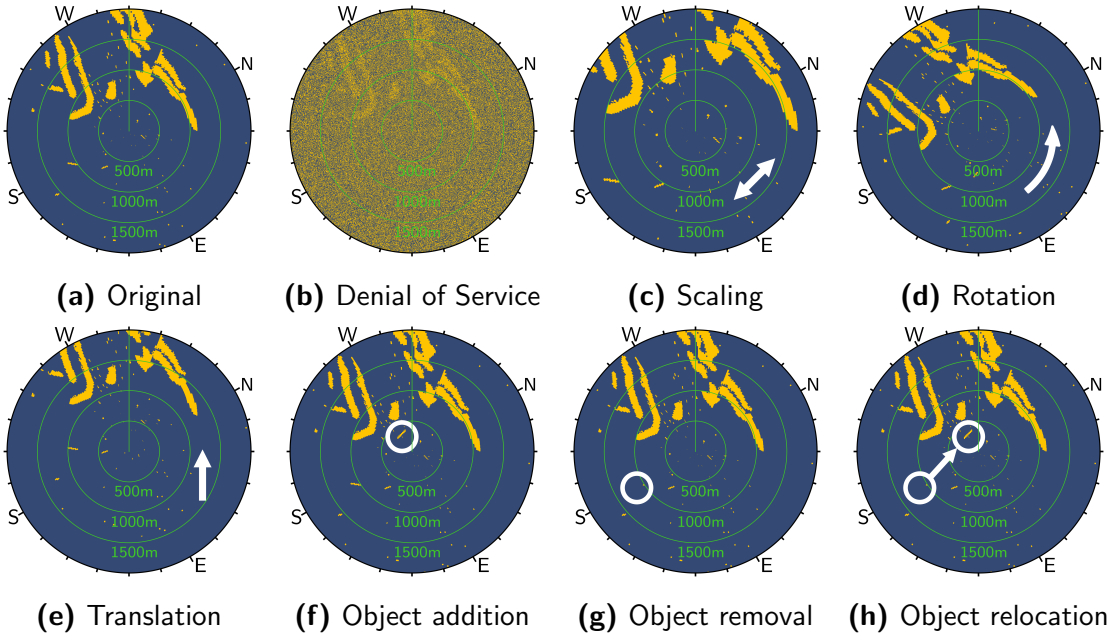
Concerning network-based attacks, one overarching classification is the attacker's position in the network, cf. red path in Figure 5.14. In a Machine-on-the-Side (MotS) position, an attacker does not control the communication channel between processor and PPI, and can merely eavesdrop and inject packets. In a Machine-in-the-Middle (MitM) situation, the attacker has complete control over the communication channel and can prevent messages exchanged between processor and PPI from reaching their destination. The capabilities of MitM depend on the exact position of the attacker in the IBS topology. This publication considers an attacker isolating the PPI from the network with complete control over both the radar system's communication and all the NMEA traffic exchanged with the PPI.

At the same time, MRSs have multiple communication channels, e.g., the radar image stream, a control channel from the PPI to the radar unit, or a return channel for reports about the radar's state as in the case for Navico BR24 [DBS11]. These channels can all be individually attacked. Generally, the implementation details of attacks against any channel heavily depend on the radar communication protocol used, the exact layout of the system, and the installed hardware.

In the following, we elaborate on the different modifications that an attacker can make to the radar image and the consequences for the situational awareness of the crew. Figure 5.16 provides an overview or all attack types.

5.3.2.3 Denial of Service Attacks

First, we consider Denial of Service (DoS) attacks, where the attacker impedes the regular use of the radar, which can take different forms. The simplest but most detectable form is blocking all communication between the radar and display unit. However, due to the lack of integrity protection, DoS attacks can also be carried out by altering the displayed content. In this case, arbitrary modifications are possible that reduce the usefulness of the radar image, such as freezing or progressively blurring it. In both cases, an observer on the bridge might remain unaware of the attack and could attribute the distortions to a hardware failure or environmental conditions. Figure 5.16b illustrates the result of a blurring DoS attack compared to the original radar image depicted in Figure 5.16a. Thus, a paralyzed radar system has detrimental consequences as it reduces vessels' ability to localize and navigate.



(a) Original **(b)** Denial of Service **(c)** Scaling **(d)** Rotation

(e) Translation **(f)** Object addition **(g)** Object removal **(h)** Object relocation

Figure 5.16 We identify seven classes of network-based attacks against marine radar. Manipulating the radar image can lead to severe consequences if misinterpreted by the crew. While DoS attacks (Figure 5.16b) disable the radar or obscure smaller objects through artificial noise, transformation attacks (Figure 5.16c–e) affect derived nautical data, e.g., distances to landmasses. In contrast, manipulating radar reflections of surrounding ships (Figure 5.16f-h) endangers the safety of vessels.

5.3.2.4 Transformation Attacks

Since attackers can manipulate the radar image arbitrarily (cf. Section 5.3.2.3), they can apply general image transformations. Therefore, we consider the three fundamental transformations (scaling, rotation, and translation) that can be used to manipulate seafarers' perception of their current environment and discuss their value for an attacker. To stretch on the consequences of such manipulations, one must consider that radar is a navigational instrument used to determine ranges and bearings, especially under poor visibility conditions. Thus, wrong navigational decisions can ultimately lead to accidents.

Scaling. Scaling the radar image results in a mismatch between the dialed and displayed range. Increasing the zoom level, i.e., displaying a smaller portion of the radar image and thus increasing perceived distances (cf. Figure 5.16c), is trivial once protocol datagrams can be manipulated. As all information of the smaller portion is contained within the original larger image, it can be cropped and re-scaled. If the zoom level is instead decreased, additional information is required to fill in the blank spaces at the edge of the screen.

Rotation. Similarly, rotating the radar image disturbs the perceived bearing of landmarks or other vessels (cf. Figure 5.16d). The consequences of such an attack can be detrimental, as a false bearing to a vessel on a collision course can reduce valuable time to initiate evasion maneuvers or a rotation can induce a course correction resulting in a displacement.

Translation. Third, translating the radar image sideways or along the direction of travel can create the impression of drift or velocity changes. For instance, as shown in Figure 5.16e, translation in the direction of the vessel's heading gives the impression of a slowdown and increases the perceived remaining distance to hazards, thus increasing the risk of head-on collisions. Technically, besides redrawing the transmitted echoes, this effect can equivalently be achieved by progressively increasing the delay between network packets. However, a translation induces a parallax effect, i.e., objects become visible or disappear from the radar display due to the changed perspective, which the attacker cannot easily compensate for.

Overall, image transformations are effective means of performing targeted attacks on nautical decisions with hazardous consequences. Therefore, the position of a vessel could be misjudged, and obstacles may not be adequately avoided.

5.3.2.5 Object Manipulation Attacks

While image transformations are well suited to manipulate static surroundings (e.g., landmasses), an attacker could also selectively manipulate individual (mobile) objects, e.g., aids to navigation or other vessels traveling the seas. By selectively manipulating portions of a radar image, adding, removing, and/or repositioning individual radar echoes is feasible.

Addition. Adding new (moving or stationary) objects can be performed by manipulating the datagram and scanline contents for the specific location to resemble the desired radar echo, as shown in Figure 5.16f. A new echo may cause the navigator to adjust the vessel's course and force it to enter dangerous waters in an attempt to avoid the supposed collision.

Removal. In contrast, an attacker could also manipulate the network traffic to hide the presence of certain radar echoes (cf. Figure 5.16g), e.g., to provoke a collision. While visual surveillance could identify the existence of the removed object, poor visibility can shorten the reaction time for adequate evasion maneuvers, thus making a collision inevitable.

Relocation. Lastly, an attack can relocate objects (cf. Figure 5.16h), which can be seen as a combination of the two previous attacks. Here, the difference in the position between original and relocated objects just becomes progressively erroneous, whereas the (non-)existence of added or removed objects can be verified visually. Hence, such a manipulation is harder to detect, even in good visibility conditions.

While object manipulation requires a detailed analysis of the image's content to identify individual echoes, redrawing the data at network-level remains fairly simple as it is analog to, e.g., adding random noise (cf. Section 5.3.2.3). At the same time, these attacks indicate that even a minimal change in the information displayed by the radar can lead to far-reaching consequences. While seafarers are ordered to keep watch despite the abundant digital assistance their working environment, e.g., surrounding machine noise, facilitates fatigue increasing the chances for human error [KJL24] and such attacks to the radar image remaining unnoticed.

5.3.2.6 Sophisticated Context-Aware Radar Manipulations

All considered attacks enable an attacker to change the displayed radar image arbitrarily. So far, they are conceptualized regardless of the current navigational situation, and the outcome, e.g., a rotation attack placed during a critical course correction, could be much more severe. In addition, since radar is not the only shipboard system used for navigational purposes (cf. Section 5.3.1.1), targets removed from the radar display may still be visible via other means such as AIS.

Conversely, sophisticated, context-aware, and situative radar attacks can realize stealthy and hardly detectable attacks. Due to the nature of modern vessels' network traffic, which primarily consists of broadcasted and unencrypted UDP messages, there are unique opportunities in which specific attacks benefit from additional external information.

First, *passively* overhearing navigational status updates, such as the GNSS position, speed or heading, and AIS [AG22] allows attacks to be triggered remotely upon arrival in a given region or translating the radar image in the vessel's direction during acceleration for further disguise. Also, potential targets for the object removal attacks can be inferred from AIS or ARPA. Second, *actively* manipulating the other network's traffic, as proven successful [HBP21], can enhance radar manipulations, e.g., by dropping AIS-related messages during an object removal. In that case, the MitM attacker needs to be capable of dropping the related information. Unless seafarers can visually confirm the incorrectness of the data provided by the IBS, it is unlikely that such sophisticated attacks will be detected.

Overall, the taxonomy highlights that attacks against marine radar systems' network communication can potentially be performed easily. Also, they have far-reaching consequences, which might not be directly detectable by the vessel's crew.

5.3.3 RadarSec-Lab – A Marine Radar Simulation Environment

Radar attacks pose a severe threat to the safe operation of vessels which we aim to defend against. Being theoretically feasible (Section 5.3.2), it remains unknown whether these can be conducted *practically* against a MRS.

Because attacking real vessels introduces an operational risk, we use simulation for our evaluation to safely conduct radar attacks. However, there exists no holistic maritime simulation framework to adequately perform network-based radar attacks. Therefore, we set out to design our own. Using a simulation approach provides further benefits, including reproducibility and adaptability in the case of upcoming attacks or architectures. Likewise, it can serve as an evaluation platform for radar security solutions by the research community. To this end, we initially define requirements to be fulfilled (Section 5.3.3.1), describe our final design (Section 5.3.3.2), and assess our environment (Section 5.3.3.3).

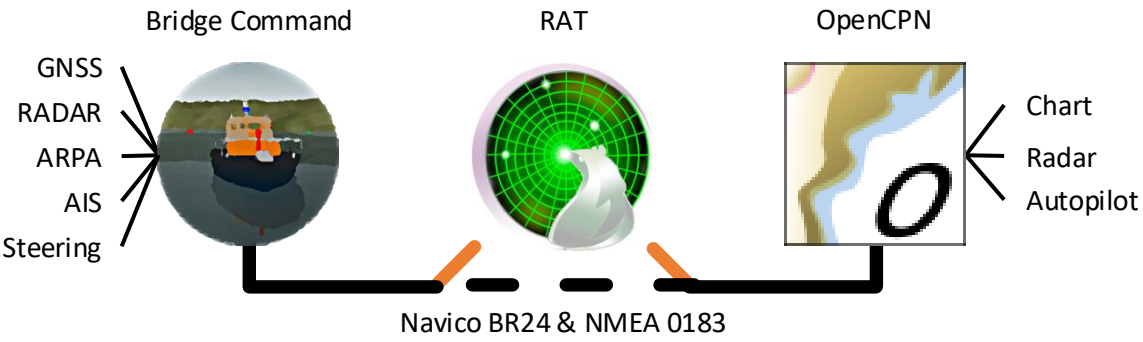


Figure 5.17 The radar simulation environment consists of two components. Bridge Command (BC) simulates a virtual ship and surrounding vessels, while OpenCPN serves as a chart plotter, including a radar display and an autopilot. Finally, RAT (Section 5.3.4) injects radar cyberattacks into the network traffic.

5.3.3.1 Requirements for a Radar Simulation Environment

To develop a simulation environment that also serves as a scientific tool, we refer to common practices and guidelines [UHH+21]. First, details about the simulation must be made *transparent* to avoid false conclusions from conducted experiments. Next, *adaptability* facilitates the future addition of components, e.g., implementing more sophisticated radar attacks or protocols. Ensuring *realism*, i.e., that deduced results are transferable to the real world, is of utmost importance since only then do approaches also tackle the pressing challenges in real systems. Finally, *availability* and *replicability* are important properties, allowing others to reproduce experiments. During the following design of the radar simulation environment, we take special care to fulfill these properties.

5.3.3.2 Design of a Radar Simulation Environment

To design a radar simulation environment with a high degree of *realism*, we model all relevant parts of modern vessels according to the model established in Figure 5.14. This involves sensors measuring the vessel's simulated surroundings, especially radar, including an ARPA, GNSS, and AIS, as well as actuators controlling the vessel in speed and direction. These components should be connected over regular Ethernet and communicate with an IBS via common maritime standards and radar protocols (cf. Section 5.3.1.3), against which cyberattacks will be conducted subsequently. Our simulation environment is designed along this generic model (cf. Figure 5.17).

Radar and Environmental Simulation

As the basis for our marine radar and environmental simulation, we choose Bridge Command (BC) [Bri22], an open source software for training navigational and radar skills, which has also found use in research [SBB19, AOGK22]. It features realistic virtual maps, including terrain, other surrounding vessels, and buoys, which can all affect the radar image, as well as an integrated ARPA. BC allows users to pilot a vessel while corresponding nautical data is relayed over the network via NMEA 0183.

Scenario name	Map	Vessels	Route
Simple Estuary	hilly estuary (fictional)	13 (fictional)*	5.9 nm*
Santa Catalina	hilly island (real)	90 (historical AIS)*	5.5 nm*
Rostock	flat harbor (real)*	76 (historical AIS)*	3.9 nm*

Items marked with * were contributed by us.

Table 5.14 The simulation environment features three scenarios with diverging maps, vessels, and routes for the autopilot.

Since BC does not natively support the transmission of radar images to radar displays over the network, we add this functionality by implementing a radar protocol. As we argued in Section 5.3.1.3, we select the Navico BR24 radar protocol for our research as being a reasonable representative with respect to its features and radar resolution among the available protocols. We further enhance the resolution of BC's radar simulation to increase the radar's *realism*. Finally, we add support for broadcasting AIS Class A reports, signaling the position, heading, and speed of surrounding vessels, as context-aware attacks require this information (cf. Section 5.3.2.6).

Designing Realistic Scenarios

Besides modeling digital systems, designing realistic scenarios is of equal importance. First, the terrain influences the final radar image, and second, potential radar security solutions have to cope with varying environmental conditions in practice, i.e., operate in unknown environments. To this end, we design three scenarios with diverging properties, as shown in Table 5.14.

BC already comes with several maps, all of which include terrain, buoys, and vessels, from which we select *Simple Estuary*, a fictional map, and *Santa Catalina*, modeled like the real island off the coast of California. As both scenarios mainly consist of steep shorelines and mountains, we add the mostly flat *Rostock* scenario, a German port on the Baltic Sea. We generated the map from actual height cartography material and added buoys according to nautical charts. Furthermore, surrounding vessels in the scenarios Santa Catalina and Rostock were modeled using historical AIS data of both regions.

IBS and Autopilot

To model the IBS, we choose OpenCPN [Ope23], an open source chart plotter and navigation aid. OpenCPN receives the maritime network protocol data and simultaneously serves as the radar display and control unit via its radar_pi plugin [Ope22] (cf. Figure 5.17). Finally, to tackle the *replicability* of individual experiments and to ensure that the simulated vessel navigates along the same waypoints within the environment, we leverage OpenCPN's autopilot functionality coupled with our autopilot implementation in BC using the exchange of standard NMEA 0183 autopilot sentences via the network. For each scenario, we mark out a predefined route (cf. Table 5.14) with a duration of roughly 18 minutes as input for the autopilot.

Overall, the radar simulation environment allows piloting vessels in realistic scenarios along repetitive paths and observing the generated network traffic transmitted towards an IBS.

5.3.3.3 Requirement Assessment

After presenting the technical details, we now assess the developed environment with respect to the requirements stated in Section 5.3.3.1.

Beginning with *realism*, we compared the simulation's network traffic to recordings of an actual vessel (Deneb [BSH25]) and found that only a few proprietary NMEA sentences specific to that vessel and ones irrelevant for radar, such as GNSS satellite reception strength, were missing. Additionally, we carefully examined the radar image quality. Landmasses and buoys are located correctly, and other vessels' radar echoes are consistent with AIS and ARPA tracking, as initially shown in the previous Figure 5.15, taken from the simulation of the Simple Estuary scenario. The terrain even masks objects behind line of sight. Some constraints of such a simulation environment are the low resolution of landscapes and BC's simplified radar reflectivity model. Further limitations are discussed in Section 5.3.8.2.

Adaptability is provided by the extensibility of the environment's components. Further radar protocols can be integrated, scenarios interchanged in BC, and attacks added to RAT (cf. Section 5.3.4). Finally, by publishing the testbed together with its documentation, we tackle *availability*, *transparency*, and *replicability*, making it an ideal platform to conduct scientific radar security research.

5.3.4 Implementing a Radar Attack Tool

With the radar simulation environment at hand, we can develop radar-specific network attacks. This is accomplished by our Radar Attack Tool (RAT), which serves as a Proof-of-Concept for all network-based attacks laid out in Section 5.3.2. It interfaces with the communication between the sensor network and the IBS either as MitM, i.e., a malicious device with complete control over traversing traffic, or in a MotS position.

Technically, RAT bases on Scapy, a Python interface for packet manipulation. While RAT is designed with a multitude of radar protocols in mind, based on the observations in Section 5.3.1.3, for this Proof-of-Concept, we focus on the Navico BR24, already supported by the simulation environment. The following Section 5.3.4.1 discusses the challenges and unique opportunities of implementing the different attack types along Navico BR24.

5.3.4.1 Attack Implementation

As established in Section 5.3.2, there are four attack classes: DoS, transformation, object manipulation, and context-aware attacks. In the following, we showcase their

application to a real radar protocol (Navico BR24) and explicitly highlight the latter class of context-aware attacks utilizing knowledge from nautical NMEA 0183 data.

Denial of Service

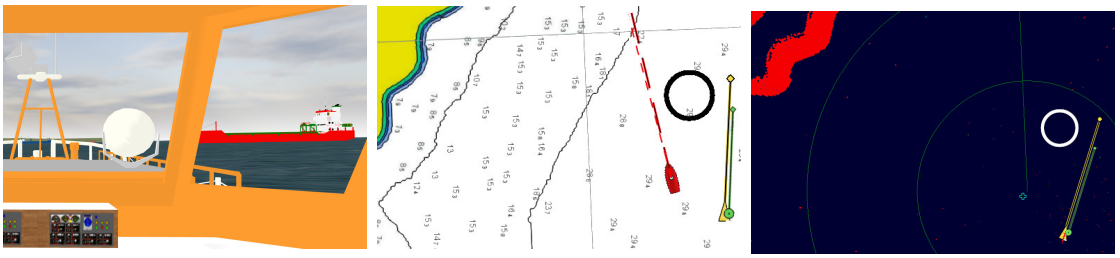
DoS attacks can be executed against many protocols. In a MitM position, it suffices to overwrite the scanline's pixels with arbitrary data, i.e., noise or blank, as no integrity protection is in place. A MotS attacker can send a duplicated and modified packet right after the original one which then overwrites the previous benign radar spoke on the PPI. Given a single scanline, the number of scanlines per radar sweep, and a pixel position, RAT can convert the location of that pixel from polar to Cartesian coordinates and vice versa [DBS11]. Thus, the entire radar image can be redrawn arbitrarily or frozen to the last image, constituting a slightly more sophisticated variant of a DoS attack.

Alternatively, a DoS can also be achieved in Navico BR24 by altering specific control communication registers to instruct the radar unit to turn off continuously. Under some circumstances this can be especially critical since turning the radar back on may take some time. For example, some radar devices require $1-2$ minutes to warm up before being functional again.

Transformation

The transformation attacks are divided into three subcategories and can be implemented quite efficiently for Navico BR24. Regarding the *scaling* attack, the pixels associated within each scanline need to be modified by either zooming in or out. Note that extending the range requires generating artificial pixels not scanned by the radar. To this end, RAT configures the radar to scan a larger area and only forward a rescaled sub-image. In Navico BR24, this can be achieved by manipulating the scale field in the datagram header instead of individual pixels. The same holds for the *rotation* attack, respectively attacking the header's angle field. To rotate an image pixel-wise, the data associated with a specific scanline need to be shifted to another scanline or even across multiple UDP packets. Lastly, *translation* attacks require more effort and are solely based on pixel manipulation accomplished analog to redrawing entire images (cf. DoS). Again this may affect pixels across different UDP packets.

Still, a sudden transformation can introduce a noticeable discontinuity in the radar image. To make transformation attacks more subtle, RAT leverages external sensor data from the network, like the rate of turn announced via NMEA 0183, for further disguise. For example, incrementally rotating or translating the radar image during a vessel's course correction further than intended. This approach is similar to redirected walking, known from virtual reality and used to bend virtual worlds without users noticing to trick them into exploring larger virtual worlds than real space allows for [SBJ+08].



(a) Simulated view from the bridge in BC

(b) RAT removes AIS and ARPA target in OpenCPN

(c) RAT overwrites the vessel's radar echoes

Figure 5.18 RAT overwrites objects' radar echoes shown on the radar display (Figure 5.18c) and simultaneously blocks AIS and ARPA related tracking information displayed on the chart display (Figure 5.18b). Navigators relying solely on insecure digital systems could overlook the hazard depicted in Figure 5.18a.

Object Manipulation

Attacks adding, removing, or relocating radar echoes are enabled by the additional contextual information sensors provide and require selectively redrawing parts of the image by adding or blanking echoes. However, the seamless integration into an IBS with its complementary tracking capabilities, i.e., AIS or ARPA, is more challenging.

Considering *additions* first, these can be performed in two steps. First, a new radar echo needs to be introduced, after which appropriate ARPA and AIS messages need to be generated for that echo. The latter can be injected into existing UDP broadcast traffic. Since AIS refers to absolute GNSS positions instead of ARPA's relative radar vector bearings and distances, the position of the newly added echo needs to be calculated, e.g., derived from overhearing the vessel's actual position in the network. The *removal* of objects is analog to additions yet requires an attacker who can not only intercept the radar image but also suppress the corresponding AIS and ARPA messages towards the IBS. *Relocation* of objects can be assembled by combining both methods.

Concluding, RAT is the first tool to launch a wide range of network-based cyber-attacks against marine radar and IBSs that holistically integrates complementary nautical data. An exemplary attack conducted within the combined simulation environment with RAT is depicted in Figure 5.18. These tools provide the means to execute harmful network-based radar attacks in a safe environment and establish a valuable basis for future security research.

5.3.5 Setup for Attack Detection in Marine Radar Systems

Considering the spectrum of attacks, effective countermeasures are urgently needed. Optimally, preventive measures such as message authentication address the cause of the current threats. But in their absence or as additional security measure for defense-in-depth, we focus on detective measures as a first step.

An IDS alerts the crew about (ongoing) attacks instead of remaining unnoticed. Since past cyberattacks against MRSs are currently either not documented or of

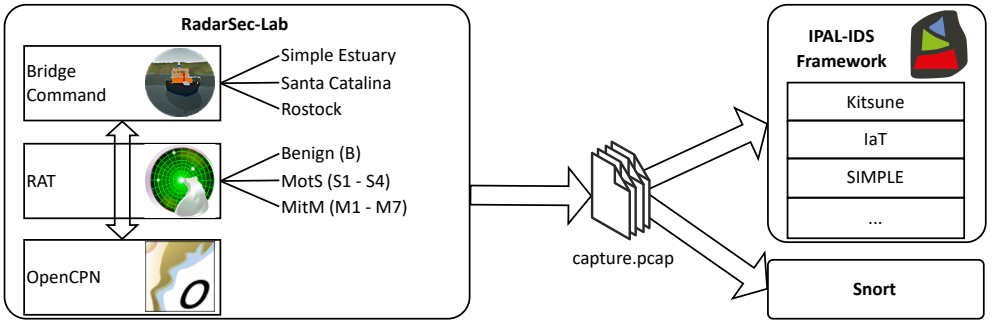


Figure 5.19 We utilize the RadarSec-Lab environment, the IPAL IDS framework, and Snort as another established IDS to examine the effectiveness of IDSs in a marine radar environment.

solely academic nature, we cannot assume that sufficient samples of attacks exist to train an IDS. Hence, we consider anomaly detection-based IDSs trained on benign data to obtain a model of normality (cf. Section 2.2.3.2). Anomalies, i.e., any deviation from these models, are considered an attack.

There exists a body of commercial and academic IDS solutions both from the field of the Information Technology (IT) and industrial domain. Consequently, we do not start developing IDSs from scratch right away and rather assess the capabilities of existing solutions first to then close potential remaining gaps. For detecting anomalies in radar communication, we consider four classes of existing IDSs: i) *rule-based* IDSs matching packets against a set of rules [Roe99], ii) *timing-based* IDSs, e.g., measuring packets' inter-arrival times [LNTA17], iii) IDSs leveraging *machine learning* to learn complex network patterns [MDES18], and iv) IDSs that analyze the *application layer* data, such as the radar image in our context. However, given the broad threat model, we believe that a single IDS is unlikely to adequately capture all attack vectors. Thus, our research question is:

Which existing IDS solutions are suited for protecting MRSs, and where do novel approaches need to be developed to fill in the gaps?

Given the simulation environment, and RAT, we combine these tools into our measurement setup to examine the effectiveness of a selection of IDSs, cf. Figure 5.19.

Data collection. From the RadarSec-Lab, we generate a dataset for the training and evaluation of IDSs. We record one packet capture for each attack type, including a benign scenario without attacks (B). Descriptions of each attack can be found in Table 5.15. Each capture lasts 1100 seconds and contains a randomized number of attacks with randomized duration with at least 70 seconds of benign traffic separating two consecutive attacks. Parameterizable attacks have randomized settings. For example, the angle by which the radar image is rotated by the attacker is selected randomly between 5° and 45°. To additionally increase stealthiness, we introduce the rotation slowly by rotating the radar image by a few degrees more each turn. Finally, to evaluate environment-specific effects, each scenario is run in three simulated scenarios available in RadarSec-Lab (Simple Estuary, Santa Catalina, and Rostock) each with a different type of vessel and with three repetitions. The resulting dataset contains 117 packet captures, covering all 13 attack scenarios (B, S1–S4, and M1–M7) with about 36 hours of simulation.

Scen.	Description
B	MRS operates normally without attacks.
S1	Commands the radar to be turned off.
S2	Modifies the protocol header to rotate the radar image.
S3	Adds fake vessels and removes existing vessels from the radar image.
S4	Distorts the radar image (random, blank screen, shift image).
M1	Commands the radar to be turned off.
M2	Distort the radar image (random, blank screen, static overlay)
M3	Freezes the radar image to the last received image.
M4	Scales, rotates, or shifts the radar image instantaneously.
M5	Scales, rotates, or shifts the radar image slowly over time.
M6	Scales, rotates, or shifts the image disguised in the vessels movements.
M7	Adds fake vessels and removes existing vessels from the radar image.

Table 5.15 Description of our attack scenarios provided by RAT.

Experiment. The packet captures serve as input for the evaluation of various IDSs, including Snort [Roe99] and the IDSs of the IPAL IDS framework for which we added support for the considered Navico BR24 network protocol in the transcriber. To train them, we use one benign scenario of the Simple Estuary environment as the training dataset and the second benign repetition of Simple Estuary to adjust the IDSs' sensitivity. Our configuration goal is to minimize the amount of false positives to avoid disrupting the crew during normal operations.

Metrics. To measure the IDSs' effectiveness, we leverage a different set of three metrics since we now consider individual network packets instead of process states with temporal correlations as before in this dissertation. First, we consider an attack to be detected if any alarm is raised during its execution and state the fraction of successfully detected attacks. Second, the False Positive Rate (FPR) metric counts the number of false positives relative to the number of benign packets. Lastly, an IDS should also ideally detect anomalous behavior in a timely manner. Therefore, we also consider the average time to first detection (TTD). Every metric is calculated individually for each of the twelve attack types but counting all three environments and three repetitions.

5.3.6 Effectiveness of Existing IDSs Applied to Marine Radar

We now assess which existing IDS proves promising in detecting which attack type. To this end, Section 5.3.6.1 introduces the four IDSs used in our analyses, and Section 5.3.6.2 shows their strengths and deficiencies in protecting MRSs.

5.3.6.1 IDS Selection

Research has proposed plenty of IDSs to differentiate benign and malicious behavior (cf. Chapter 3). Given the four directions from in Section 5.3.5, we select one representative each and present their core idea as well as how they transfer to MRS.

```
1 alert udp any any -> any 6680 ( msg:"RadarOps A disabled";
2   content:"|00 C1 00|"; depth:3; sid:6680;)
3 alert udp any any -> any 6680 ( msg:"RadarOps B disabled";
4   content:"|01 C1 00|"; depth:3; sid:6681;)
5 alert udp any any -> any 6678 ( msg:"Duplicate scanline";
6   content:"|00 44 0D 0E 00 00 34 92|";
7   threshold:type both,track by_dst,count 2,seconds 1; sid:6678;)
```

Listing 5.3 Our Snort rules for Navico BR24. Rule 1 and Rule 2 alert on setting the MRS to standby, while Rule 3 detects too frequent occurrences of a scanline.

Rule-Based (Snort). A rule-based IDS, such as the prominent software Snort [Roe99] considered here, monitors network traffic. It inspects individual packets for patterns associated with known malware strains and attacks, matching their header and content against a set of rules. Since Navico BR24 is not supported, a naive application of existing rules, e.g., the default ones included with the Debian Snort package, is ineffective in detecting any attack in our data. Thus, we develop custom rules for Navico BR24 as depicted in Listing 5.3. The first two rules match packets with register commands disabling the radar unit. Rule 3 defines a constraint on benign behavior. It assumes that real-world radar antennas have a maximum rotation speed and thus excepts a scanline with a specific angle to be received not earlier than a certain time threshold (one second here) after the previous one.

Timing-based (IaT). The second IDS, available in the IPAL IDS framework, considers the inter-arrival time between network packets, leveraging the periodicity of network traffic and raising an alarm when deviations exceed pre-determined thresholds [LNTA17] (cf. Section 4.4). To this end, IaT's model estimates the mean μ and standard deviation σ of the underlying IaT distribution for a small window of consecutive packets. These values are the basis of the model's upper and lower thresholds defined as $\mu \pm N\sigma$, where N is a user-defined sensitivity threshold, which we adjust to achieve no false positives on the training data. In the marine radar context, this IDS constructs a model for the radar image stream. Since the rotation speed of the radar antenna is constant, the network packets delivering the image step-by-step are assumed to occur regularly.

Machine learning (Kitsune). From the domain of machine learning, Kitsune is a widely used general-purpose network IDS [MDES18] which we also integrated into the IPAL IDS framework retrospectively. It trains artificial neural networks to perform anomaly detection on network traffic. The IDS's ability to apply to any kind of traffic makes it an interesting candidate. Internally, Kitsune keeps track of multiple features, such as the bandwidth or IaT, and calculates an anomaly score, the Root Mean Square Error (RMSE). This score is accumulated over time with a window mechanism. The user defines a threshold upon which an alert is raised for each packet exceeding it.

Application-layer (Steadytime). The last IDS monitors application data, i.e., the payload of network packets, which in our case contains the radar image and control commands. Due to the large amount of features, each pixel of the radar image can be considered a feature, and since most pixels are either zero (no echo) or 255 (echo), many process-based IIDSs from the IPAL IDS framework are not applicable. Still, Steadytime from Section 5.1 can be leveraged as it assumes that monitored

	Metric	B	S1	S2	S3	S4	M1	M2	M3	M4	M5	M6	M7
Snort	Det. Attacks [%]	–	100	100	100	100	0	2.2	2.3	4.7	2.2	0	7.1
	FPR [%]	0	0	0	0	0	0	0	0	0	0	0	0
	Mean TTD [s]	–	0	1.8	1.2	1.3	–	11.4	1.7	12.3	1.7	–	35.8
IaT	Det. Attacks [%]	–	0	100	100	100	0	0	0	0	0	0	0
	FPR [%]	0	9.0	4.5	4.3	4.3	9.1	0	0	0	0	0	0
	Mean TTD [s]	–	0	3.0	3.1	3.1	–	–	–	–	–	–	–
Kitsune	Det. Attacks [%]	–	100	100	100	100	100	8.9	2.3	0	2.2	0	0
	FPR [%]	0	29.3	6.6	6.2	6.2	27.4	0	0	0	0	0	0
	Mean TTD [s]	–	3.5	0.5	0.5	0.5	4.0	18.4	6.2	–	6.0	–	–
Steadyt.	Det. Attacks [%]	–	0	100	100	100	0	60.0	100	7.0	2.2	9.1	4.8
	FPR [%]	0	0	0	0.1	0.1	0	1.5	1.6	0	0	0	0
	Mean TTD [s]	–	–	0	0.1	0	–	7.4	62.6	20.8	9.4	22.8	19.8

Table 5.16 Performance of existing IDSs in a MRS context. No IDS raises a false alarm in the benign scenario (B), and they reliably detect the MotS attacks (S1–S4) yet struggle to detect MitM attacks, especially those targeting the radar image.

features regularly change, such as the radar angle changing with each transmitted radar spoke. It measures the time a feature remains static and compares it to the minimal and maximal static time seen during training. More concretely, we select the following radar features: i) the scanline counter and angle values, as these should change regularly between each transmitted packet. ii) The sum of each image frame's first scanline's pixels and the sum of pixels in scanlines with angle 0, since given natural noise, perfectly static images can be considered anomalous.

5.3.6.2 Evaluation

Applying the IDSs to the recorded data, we obtain the results depicted in Table 5.16. First, we observe that all IDSs' FPR is 0.00% if no attacks are present, cf. column B. This is interesting as we trained only on the Simple Estuary scenario but evaluated in all three scenarios. We therefore conclude that no IDS is sensitive to environmental changes in terms of false positives. However, regarding the detected attacks, we find a clear distinction between MotS (S1–S4) and MitM (M1–M7).

The MotS attacks are detected well, except for IaT and Steadytime failing to detect S1. Since S1 deactivates the radar, IaT and Steadytime should be able to detect the attack as well due to the missing network traffic. But they emit the alerts too late namely when receiving the first packet after the radar is activated again. Next, the TTD is small with 0 to 3.1 s, which is in the order of one full rotation of the radar (about 2.4 s in RadarSec-Lab). During attack conditions, the FPR is generally higher than during benign behavior. While Snort and Steadytime still feature a FPR of 0 to 0.1 % in MotS attacks, IaT and Kitsune reach levels of up to 29.3 %. Both IDSs make use of window mechanisms, where past packets factor into the classification of the current packet, leading to trailing false positives.

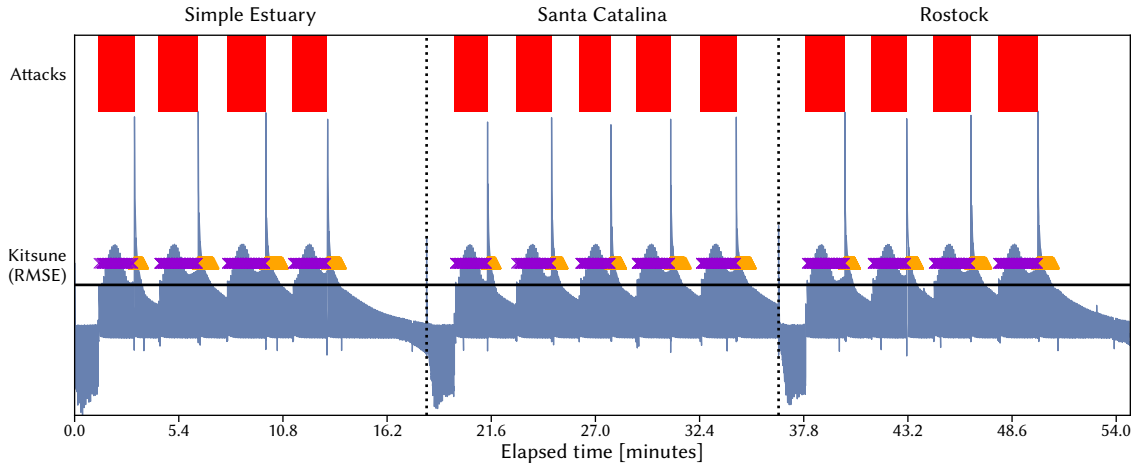


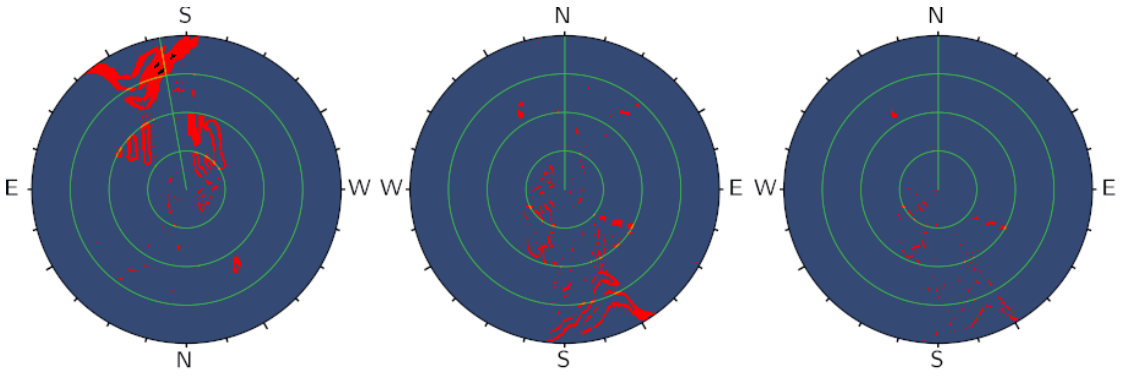
Figure 5.20 Performance of Kitsune on the S1 scenario. Attacks are highlighted in red, whereas alerts are indicated with purple (true alert) and yellow (false alert) markers. The blue curve depicts Kitsune's anomaly score, and the black line is the alert threshold.

This effect is particularly pronounced at the end of attacks in scenario S1, where the resumption of the image stream after a long interruption causes a notable jump in the IDSs' anomaly scores. Figure 5.20 depicts this phenomenon exemplarily for Kitsune. During attacks the anomaly score (blue curve) steadily rises in absence of packets. At the end of each attack when the radar image stream resumes, the jitter on that communication channel causes a notable jump in the anomaly score, leading to trailing false positives (yellow marker).

We deem these false positive rates to be acceptable due to the approaches' excellent FPR results in the absence of attacks (B). Interestingly, the Snort rules outperform the more complex IDSs but require understanding the protocol's packet format, and might fail in a more dynamic or noise-prone environment.

Despite these results, MitM attacks are, for the most part, not reliably detected. The purely network-based IDSs, Snort and IaT, fail to detect any attack sufficiently or do so just by chance. Kitsune can still detect M1, in which the traffic volume drops dramatically in the absence of the transmission of image data, as one of the features Kitsune monitors is the per-host bandwidth. Steadytime, as it also considers the application layer, i.e., the transmitted radar image, detects M2 and M3, where the radar image is static. But, this could trivially be circumvented by attackers by adding noise to the image, as would also be the case for real deployments, making the approach less usable in practice.

Takeaway. Tying back to our research question, existing IDSs are suitable for MRSs as they effectively detect MotS attacks while raising no false alarms during benign operations. However, it turns out that it is not feasible to detect the broad class of MitM attacks, especially those that change the radar image.



(a) Raw difference **(b)** Stabilized orientation **(c)** Movement-adjusted

Figure 5.21 Stabilizing radar images with the vessel's position and orientation from NMEA 0183 enables *Image-Delta* to remove legitimate differences in the images.

5.3.7 Inventing Novel Radar-Specific IDS Solutions

While existing IDSs are indeed transferable to MRS scenarios, we have likewise identified a lack of detection capabilities for MitM attacks. To narrow this remaining gap, we develop two novel IDSs designed to detect anomalies in the displayed radar image. First, we present and evaluate *Image-Delta*, a rather simple approach (Section 5.3.7.1), and then consider the more complex *Chart-Diff*, based on nautical charts (Section 5.3.7.2).

5.3.7.1 Radar Image-Delta IDS

According to the International Maritime Organization (IMO), MRS must scan their environment at a rate of no less than 12 rpm [IMO82]. At typical vessel velocities, movement in the short interval between two full radar scans should yield only small differences in consecutive images. In contrast, our image manipulation attacks introduce sudden changes, especially if they alter the entire image. Hence, *Image-Delta* determines the amount of change, the *delta*, between subsequent images and raises an alarm if the observed change exceeds the acceptable range.

Because most of the difference between two consecutive images is expected to be caused by the vessel's movement (translation and rotation), *Image-Delta* leverages NMEA reports for compensation, raising the sensitivity of the detector as shown in Figure 5.21. The stabilization adds the current heading to each scanline's angle to ensure a north-up orientation, cf. Figure 5.21b. Then, each pixel is offset by the recent movement of the vessel reducing even more legitimate differences between the figures, cf. fewer red echoes in Figure 5.21c. As a final step, we minimize the influence of noise with a Gaussian blur filter. The difference between all pixel values, cf. red parts in Figure 5.21c, is summed up and divided by the total value of pixels occurring in both images, giving a relative change between the images (δ). Scaling the delta in such a manner normalizes the final value, making it independent from radar specifics such as resolution, and reducing the influence of environmental factors such as the prevalence of landmasses.

Metric	B	S1	S2	S3	S4	M1	M2	M3	M4	M5	M6	M7
Det. Attacks [%]	–	0	44.4	0	47.7	0	75.6	61.4	100	31.1	15.9	7.1
FPR [%]	0	0.3	0.6	0	0.8	0.3	1.5	1.9	1.4	1.6	1.1	0.1
Mean TTD [s]	–	–	2.1	–	1.9	–	1.6	7.1	3.0	17.0	14.4	6.9

Table 5.17 *Image-Delta* outperforms existing IDSs regarding MitM but struggles in sophisticated attacks and performs worse in previously unseen environments.

Finally, this delta serves as the basis for the detection mechanism. During training on the Simple Estuary scenario, we measure the maximum δ, which multiplied by a sensitivity parameter serves as the detection threshold. During detection, a δ value exceeding the threshold causes an alert.

Results

We subject *Image-Delta* to the same evaluation as before, cf. Table 5.17. Again, we validate the absence of false positives across all benign scenarios B. The purely image-based IDS performs worse in MotS scenarios S1 and S3, where no or little manipulation of the image occurs. Still, substantial modifications to the image (S2 and S4) are detected. As *Image-Delta's* goal is to address the shortcomings of existing IDSs, we focus on the MitM scenarios below.

Scenarios in which *Image-Delta* performs best are those where the image changes rapidly, i.e., blanking or randomly overwriting the radar image (M2), freezing the image (M3), or suddenly scaling, rotating, or translating it (M4). The attack types less reliably detected are those that implement mechanisms to increase stealthiness, i.e., applying image modifications slowly over time (M5) or scaled with the vessel's measured movements (M6). Note that switching off the radar (M1) is also not detected as no new radar images are generated. Lastly, missing the vast majority of attacks of M7, *Image-Delta* proves ineffective in this scenario. Only the echoes of single ships are manipulated, which is largely indistinguishable to the IDS from noise or inaccuracies in the stabilization and movement compensation process.

Concentrating on the stealthy attacks from M5 in the Simple Estuary scenario, which was used for training, the IDS detects 9 of the 15 attacks. But the detection performance is diminished on the other two scenarios (Santa Catalina and Rostock), cf. Figure 5.22. *Image-Delta*'s performance depends on features of surrounding landmasses. Both benign changes and those caused by the attacker's gradual manipulation of the image are much more pronounced in Simple Estuary, causing the learned detection threshold to be inadequate for the two other other environments. Depending on training conditions, the IDS can therefore be expected to underperform or cause false positives in unseen environments, a serious impediment to real-world applicability of the approach.

Lastly, we examine the computational performance of *Image-Delta* as it has to handle large streams of uncompressed radar images. We measure the processing time for a single run of packet parsing and the detection methodology on a Intel i7-1365U

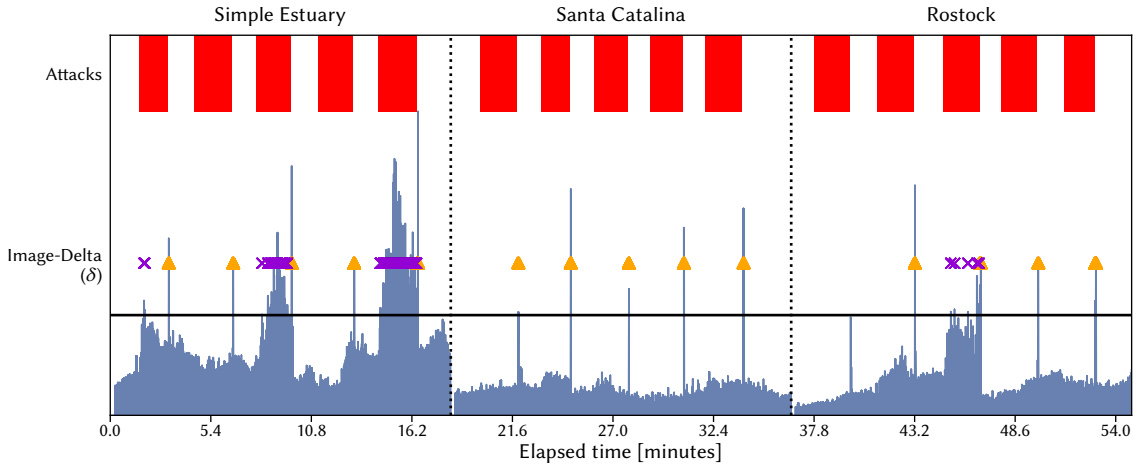


Figure 5.22 Performance of *Image-Delta* on the M5 scenario. Attacks are highlighted in red, whereas alerts are indicated with purple (true alert) and yellow (false alert) markers. The blue curve depicts *Image-Delta*'s anomaly score, and the black line is the alert threshold. Note that the false positives at the end of attacks are caused by the sudden return to the unmodified image, resulting in a large delta in all cases.

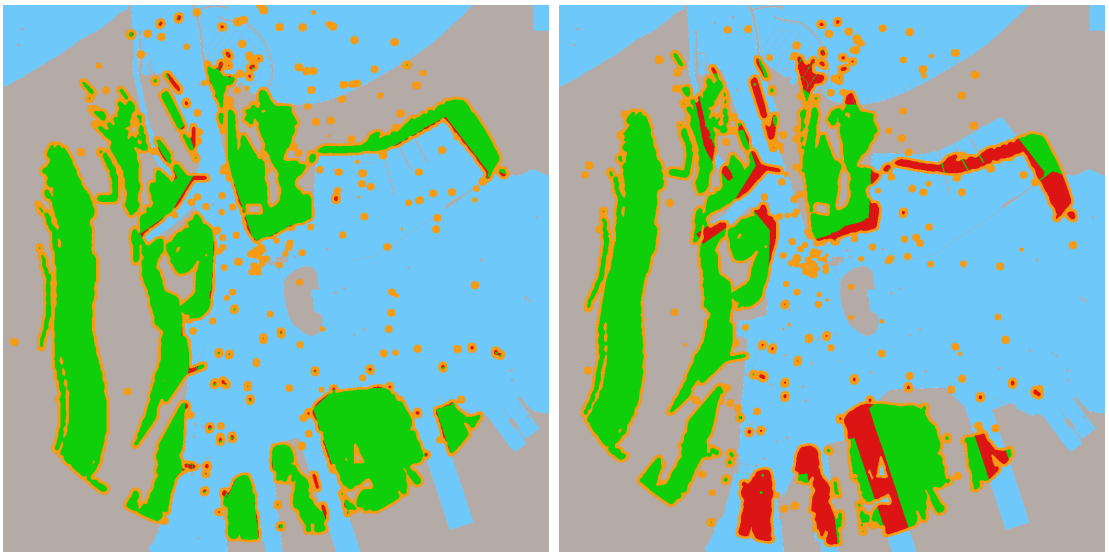
CPU with 16GB of RAM. Parsing an 18.3 minutes long PCAP consisting of 362 724 packets takes about 7.9 minutes, in large part due to the slow pyshark library of IPAL's transcriber. Our unoptimized implementation of *Image-Delta* on top of the IPAL IDS framework processes the parsed PCAP in 6.4 minutes. We therefore consider the approach to be feasible in processing Navico BR24 traffic in real time.

Takeaway. In contrast to previous IDSs, *Image-Delta* detects sudden overlaying and transformation of the radar image even in MitM scenarios. It is less effective in stealthier, gradual variants of the attacks, and inadequate in detecting object manipulation attacks targeting relatively small objects such as ship echoes. One drawback is that its performance can depend on terrain features, e.g., the jagged landscape of Simple Estuary represents more opportunities for both benign inaccuracies and deliberate manipulation of the image to increase the anomaly score than the two other simulated scenarios' smoothly rising land. As a result of training *Image-Delta* on the Simple Estuary world, it is therefore not sufficiently sensitive for the two other environments.

5.3.7.2 Radar Chart-Diff IDS

Our second IDS aims to improve *Image-Delta*'s lack of environmental awareness. The main idea is to obtain some ground truth about the environment, i.e., landmasses or buoys, from a trusted source. Since IBSs are obligated to have regularly updated, accurate digital charts on board for the areas in which the ship operates, these can be correlated with the radar image. Hence, the idea is to overlap the radar image with the charts and measure their difference.

For that purpose, *Chart-Diff* considers each radar echo independently. If an echo's position corresponds to a location that is marked as a landmass on the chart, this pixel is considered valid, cf. green pixels in Figure 5.23a. Note that not every pixel



(a) Normal condition **(b)** Anomalous condition (rotated)

Figure 5.23 *Chart-Diff* counts the amount of invalid radar echoes (red) that do not correspond to the landmasses (gray). Weak echoes are attenuated through the filtering process and discarded as noise (orange), improving the sensitivity of the detector.

Metric	B	S1	S2	S3	S4	M1	M2	M3	M4	M5	M6	M7
Det. Attacks [%]	–	0	100	0	61.4	0	46.7	95.5	81.4	84.4	56.8	2.38
FPR [%]	0	0	0	0	0	0	0	0	0	0	0	0
Mean TTD [s]	–	–	4.6	–	3.9	–	3.2	25.4	7.3	26.2	20.8	9.7

Table 5.18 *Chart-Diff* is capable of detecting sophisticated manipulations such as translation, or rotation, by having more information available about the area from charts.

on the radar image representing landmasses must contain an echo since landmasses may be shadowed, e.g., through mountains. In contrast, invalid (non-verifiable) echoes can fall into four categories for which ground truth is hard to obtain: other vessels, uncharted objects, inaccuracies, or noise. Although information about other vessels can be obtained from AIS, not all vessels are equipped with AIS. Also, AIS signals could be tampered by sophisticated attackers [BPW14], and indeed RAT already manipulates them appropriately. Likewise, uncharted objects or remaining inaccuracies cannot be predicted. Still, we apply a Gaussian blur filter to the image to find and ignore small or weak echoes resulting from noise, cf. orange pixels in Figure 5.23b. For detection, *Chart-Diff* learns the maximum proportion of invalid pixels, i.e., red pixels, during normal conditions. If an attack tampers with the radar image, e.g., rotating it, we expect the number of invalid pixels to increase and exceed the normal maximum, cf. Figure 5.23b.

Results

Like *Image-Delta*, *Chart-Diff* is also unable to detect S1, S3, and M1. Especially, S3 is difficult to detect since adding, or removing vessels on the screen, cf. Table 5.15,

affects just a few pixels of the radar image. But Chart-Diff performs better in S4, M3, M5, and M6, especially in the scenarios we aimed to improve, such as M5, where the detection rate increases from 31.1 % to 84.4 %. *Chart-Diff* even makes great progress on M6, containing the stealthiest image manipulations conducted during vessel maneuvers, e.g., rotating the radar image while the vessel is turning. Lastly, leveraging ground truth from the charts reduces environment-specific influences on the anomaly score. Unlike for *Image-Delta*, surrounding landmasses' features may vary widely as long as the accuracy of the charts remains accurate. In scenario M6 we nonetheless find *Chart-Diff* to underperform in the Santa Catalina scenario. There, the vessel's few course changes and the relatively short duration of attacks do not provide enough opportunity for the attacker to cause a significant rotation of the image. Translation in the direction of travel is also largely undetected as the vessel moves parallel to a relatively straight shoreline of this scenario.

We again measure the computational performance of our newly proposed IDS with the same methodology as for *Image-Delta*, cf. Section 5.3.7.1. *Chart-Diff* takes 5.7 minutes to process the 18.3 minutes long PCAP thereby being slightly faster than *Image-Delta*. Again, this IDS proves promising for time detection in real-time.

Takeaway. As an alternative radar-specific IDS, *Chart-Diff* outperforms *Image-Delta* on stealthier attacks. Additionally, one advantage of *Chart-Diff* is that its alerts can be visualized to the crew by highlighting suspicious echos once an alert is raised, cf. Figure 5.23. For example, while vessels are lighted, landmasses usually are much harder to recognize especially under poor visibility conditions or at night. Thereby, navigators can validate and reflect the ongoing situation with Chart-Diff and assess it themselves if necessary.

5.3.8 Related Work, Limitations, and Summary

Closing our analyses, we analyze related work on anomaly detection for MRS in Section 5.3.8.1. We then discuss limitations in Section 5.3.8.2 and ultimately summarize the state of intrusion detection for MRS in Section 5.3.8.3.

5.3.8.1 Related Work

Maritime cybersecurity has become a subject of interest to research. Consequently, attacks against IBSs [HBP21] and associated components such as GNSS [BUH+22, BH17], the NMEA protocol [THM+22, BJL+21], or AIS [BUH+22, BPW14, KCH18, WRBW22] are well researched. In contrast, the security-related investigation of marine radar systems constitutes a critical gap in the current research landscape, even though existing research is alarming: A vulnerability scan of the shipboard radars of two oil/chemical tankers revealed a broad range of security risks (e.g., missing access control) and attack points [SRFM20]. Outside the maritime sector, Cohen et al. [CGES19] provide a general threat taxonomy and a concerning attack scenario by realistically simulating the manipulation of radar data in air traffic control. Nevertheless, thorough security analyses regarding marine radar systems and specific security solutions are missing to date.

While different countermeasures to thwart maritime attacks exist, they focus on AIS [GK20], NMEA [BJL+21], and GNSS [JTP16, BH17] or offer general security improvements, e.g., by segmenting the network topology into different zones [RT14]. Furthermore, although not specific to maritime settings, multiple security solutions and mitigation approaches target radar systems in general or specific application fields. For example, deep learning-based anomaly detection can detect manipulations of data streams between the radar and the control system, as demonstrated with real radar data from experimental [CLS+22] and aerial radar systems [dZS+22]. Besides detective measures, hash-based integrity checks and encryption were developed for the ASTERIX protocol used for data exchange in air traffic control and evaluated within simulated communication between aircraft and airport control [CBB15]. Finally, algorithmic approaches, including numerical evaluation, aim toward consistent snapshots of distributed radar networks to increase resilience [MPA17] or combat false data injection [YFZ+18, SWL+21].

These research efforts provide valuable insights and directions for improving radar-related security. However, it is unlikely that these can be readily transferred to maritime scenarios. Radar data from *stationary* air traffic control does not necessarily serve for evaluating *mobile* shipboard radar solutions. For instance, landmasses on static radar do not change at all compared to the movements of a vessel. Furthermore, the datasets and simulations used in existing radar-related work, often depending on hardware access [dZS+22], cannot be used to reliably develop and evaluate new maritime-specific security solutions.

In the maritime domain, radar images combined with charts have been utilized for localization and detection of GNSS tampering [DBAG22], similar to *Chart-Diff*, but with a different purpose. The only directly related work by Longo et al. [LRAM23] proposes a policy-based IDS for the ASTERIX CAT-240 radar protocol in a maritime scenario. Their policies enforce a constant rotation speed and a strict monotonicity of message identifiers, which is analogous to detecting the maliciously injected duplicates of image frames in the MotS scenarios S2–S4.

5.3.8.2 Limitations and Future Work

Since related work lacks accessible tooling to conduct our research, we developed the RadarSec-Lab. As a simulation environment, RadarSec-Lab offers the possibility of scientifically reproducible experiments, but falls short in replicating user-induced behavior of real seafarers. Thus, we have not yet considered such behavior, e.g., changing the radar resolution or scanned ranges. Our environment also has two other technical limitations. First, while the network traffic patterns are largely comparable to a real network, RadarSec-Lab supports only one of many network protocols (Navico BR24). Still, MRS protocols from different vendors are relatively similar, cf. Section 5.3.1.3 and thus adding a new protocol would be technically feasible yet require reverse-engineering other protocols as their specifications are closed and proprietary. Second, the radar echoes simulated by BridgeCommand lack realism, since compared to real MRS they are of lower resolution, overly sharp, and only approximate physical effects such as reflectivity and changing weather

IDS	S1	S2	S3	S4	M1	M2	M3	M4	M5	M6	M7
Snort [Roe99]	✓	✓	✓	✓	✗	✗	✗	✗	✗	✗	✗
IaT [LNTA17]	✗	✓	✓	✓	✗	✗	✗	✗	✗	✗	✗
Kitsune [MDES18]	✓	✓	✓	✓	✓	✗	✗	✗	✗	✗	✗
Steadytime (cf. Section 5.1.2)	✗	✓	✓	✓	✗	(✓)	✓	✗	✗	✗	✗
Longo et al. [LRAM23]	✗	✓	✓	✓	✗	✗	✗	✗	✗	✗	✗
Image-Delta	✗	(✓)	✗	(✓)	✗	✓	✓	✓	(✓)	(✓)	✗
Chart-Diff	✗	✓	✗	(✓)	✗	(✓)	✓	✓	✓	(✓)	✗

The results by Longo et al. [LRAM23] are derived from their publication and not measured by us.

Table 5.19 Our evaluation on the effectiveness of IDSs reveals that MotS attacks are reliably detectable by established IDSs while lagging in MitM scenarios. Set out to remedy this situation, *Image-Delta* and *Chart-Diff* make progress in that direction, yet still leave room for improvement in the most sophisticated attacks (M6 and M7).

conditions. To avoid more false positives during worsened weather conditions than encountered during training of an IIDS one could dynamically raise the detection thresholds but likely at the cost of lowered detection performance.

With regard to a deployment of Radar-IDSs, future work still faces challenges. First, we trained and evaluated with little data compared to vessels that operate for weeks. Second, beyond detection performance, giving appropriate advice to the crew about the MRS's state is crucial to be of actual benefit, as proposed in [vRS$^+$+22]. Here, *Chart-Diff* IDS may be a first step in that direction, cf. Figure 5.23. Next, the issue of verifying other vessels and AIS signals persists [WRBW22]. While Katsilieris et al. [KBC13] have validated AIS signals with radar echoes, nowadays both sources have to be considered vulnerable. Finally, the most significant conceptual issue of image-based IDSs is that these are mainly suited for coastal regions. In open waters, where there are virtually no reference signals to determine whether the image is altered, detecting anomalies is much more difficult. This scenario as well as coordinated attacks simultaneously targeting multiple IBS components, i.e., radar, AIS, and NMEA, are a challenging field for future research.

5.3.8.3 Summary

Modern vessels' Integrated Bridge Systems (IBSs) and, thus, their radar systems have experienced a similar increase in digitalization and interconnectivity as ICSs, leading to new cybersecurity challenges in this domain. As our final assessment of how well intrusion detection can detect cyberattacks in this scenario, we summarize all results in Table 5.19 with respect to the effectiveness of each IDS for the individual attack vectors. In addition to our experiments, we take the results of Longo et al. [LRAM23] from related work into account, yet marked as grey since we transferred their results to our attack taxonomy without extra evaluations.

Overall, our results show the potential for applying intrusion detection for MRS to remedy the current situation. Concerning our analyses of existing approaches and related work (upper and middle part of Table 5.19), we conclude that MRS can be

protected against MotS attacks using existing approaches when adapted to radar network traffic, which IPAL achieves. Set out to address the gap in MitM attacks, novel radar-specific, image-based IDSs such as *Image-Delta* or *Chart-Diff* provide a remedy in these situations, especially for attacks M2–M5. But they still struggle with stealthy attacks that are carefully executed by sophisticated adversaries. Attacks modifying individual echoes, such as a nearby vessel (M7), are currently not covered by any approach.

Even if the resilience of MRS network protocols to cyberattacks will be hardened preventively, radar-image-based IDSs could still be beneficial to thwart attacks from the electromagnetic warfare vector [LMAR23]. Therefore, the novel detectors we introduced promise great value for future systems. In conclusion, considering the advantages and disadvantages of each approach demonstrated in our evaluation, currently, only a combination of multiple IDSs promises adequate protection for MRSs against most attacks.

5.4 Conclusion

The third contribution of this dissertation (C3) bundles our inventions of new IIDSs. In the first part, we tackled the observation that intrusion detection in the past was governed by complex methodologies that had inherent disadvantages if eventually brought into practice (I3). But as we showed, this complexity is not necessary per se as decent results can already be achieved by S.I.M.P.L.E. detection methods. Building on this observation, we invent GeCo, an even stronger IIDS with slightly increased but justifiable complexity that makes a significant step toward how IIDSs should be designed and evaluated to be of practical use. GeCo especially aims to improve the handling of the IIDS for ICS operators by being comprehensible and automatizing human tasks.

Besides these achievements, we also want to emphasize that the evaluations of these IIDSs are extensive compared to previous ones observed in related work in our Systematic Mapping Study (SMS) (I2). First, we continually compared against four different and widely used datasets instead of the 1.3 datasets on average by related work (cf. Section 3.4.1.3). Additionally, we compare our SIMPLE IIDSs and GeCo to at least four other IIDSs, whereas related work does so for only 0.5 publications on average. Furthermore, our evaluations use a broad mix of established point-based and modern time-aware metrics in combination with visual examinations of the results. Together with other analyses about the new IIDS designs, such as comprehensibility, user studies, computational performance, or hyperparameter stability, these IIDSs are well examined for their various capabilities [ALS23]. While the obtained results show great performance with respect to to the evaluated datasets in the lab, it still remains unclear how well this performance transfers to the real-world on physical hardware [SHK+24]. Nonetheless, since all IIDSs built by us are published in the IPAL IDS framework [FC24], this allows other researchers to replicate and build upon our findings in the future. Thus, IPAL, together with the new IIDSs, lays a good foundation for quickly implementing and effectively

comparing new IIDSs to a comprehensible baseline, effectively improving the current workflows of IIDSs research.

This research platform has proved to be helpful for transferring intrusion detection into new territories (I4). To detect harmful cyberattacks against MRS onboard vessels, we successfully adopted existing methodologies such as IaT or SIMPLE's Steadytime approach to this domain. While not providing exhaustive protection, filling the remaining gaps in the detection capabilities of IIDSs for this domain with custom-fitted approaches proves efficient rather than starting from scratch.

Overall, this chapter neatly demonstrates that we and other researchers have developed powerful methods to successfully detect various types of cyberattacks in diverging scenarios that are now ready to be combined and brought into practice.

6

Selecting an IIDS for Deployment

As we observed, the Industrial Intrusion Detection System (IIDS) research landscape experiences a steady growth in the number of publications proposing detection methodologies (cf. Chapter 3). We have even contributed additional IIDSs in Chapter 5. However, the issue of selecting an IIDS that one wants to deploy remains unanswered as of now (I5) and can become challenging given the pool of options for IIDS to choose from.

One fundamental decision at the beginning of selecting an IIDS for a deployment use case is the direction of the detection methodology. Broadly speaking, IIDS detection models can be classified into misuse- and anomaly-detection each with diverging philosophies (cf. Section 2.2.3). Of those general directions, misuse-based IIDSs especially require samples of attacks for their training. But, recording large amounts of qualitative training data from the actual Industrial Control System (ICS) can be difficult in practice. Thus, in Section 6.1, we first tackle the question of how complicated the training of misuse-based IIDSs can become.

If we instead concentrate on only a single direction of detection methodologies, the pool of potential IIDS candidates remains large. Currently, the usual conception is that an ICS has to implement the single-best performing IIDS. However, as we already observed in Section 4.5.1 and Section 5.3, this might result in blind spots or attack types that are missed and could be detected better with another second IIDS. To more scientifically and broadly assess how effective deploying a single IIDS is, we reconsider the results achieved so far in Section 6.2 and find that a single IIDS most likely is disadvantageous and several IIDSs working in cooperation might be better.

Putting this idea to the test, the remaining Section 6.3 to Section 6.5 motivate and examine various methods of ensemble learning to leverage the advantages and disadvantages of several IIDSs to achieve optimal protection for ICSs.

6.1 Deployment Challenges of Misuse-based IIDSs

Research demonstrates the alleged effectiveness of hundreds of newly proposed IIDSs by evaluating them on dedicated datasets and publishing achieved detection performances (cf. Chapter 3). In vitro, these IIDSs achieve excellent results [LAVM18, FPMC19, KYK20]. However, when it comes to real-world deployments, these solutions are challenging to configure [ET22] and then cannot perform as promised [SP10, AMM20]. While the scientific literature already identifies various challenges for applying research IIDSs in practice [SP10, AMM20], we proclaim that one aspect impacting *misuse-based* IIDSs' deployability remains unaddressed.

As misuse-based IIDSs train on labeled samples of genuine behavior *and* attack data (cf. Section 2.2.3.2), e.g., Random Forests (RFs) or Support Vector Machines (SVMs) [PASE18], it remains unclear how much training data is required to maximize detection performance. This question is especially critical in the case of misuse-based IIDSs, where collecting attack samples in a testbed might still be relatively easy, but collecting real-world attack samples is much harder [BSL+23]. In contrast, this issue does not exist for anomaly detection as they solely train on genuine data.

Obtaining attack samples from cyberattacks in real systems is more challenging, as their collection would expose, disrupt, and potentially damage sensitive infrastructure or facilities. For artificial datasets and testbeds used in research [CDT21], on the other hand, it is relatively easy to generate such attack samples. Thus far, IIDS proposals do, however, all require custom training phases for a concrete deployment with hardly any model transferability across scenarios [Eta17]. We thus observe a large discrepancy between training data availability for research activities and real-world deployments, which may be one reason for the challenging deployments of current research proposals [AMM20].

6.1.1 Research Question

The deployment of IIDSs in real industrial networks proves challenging since experts already fail to reproduce the detection performance of existing IIDSs in the lab [SP10, Eta17, AMM20]. Besides aspects, such as hyperparameters discussed before in this dissertation in Section 3.5.2.4, we suspect training data availability, especially samples of attacks, to be one potential cause for this situation. The detection algorithms themselves are often applicable to multiple industrial domains (cf. Chapter 4). However, they assume to be trained separately for each deployment to learn the expected behavior. For example, the learned boundaries of a water tank's maximum acceptable fill level differ for each IIDS deployment. Consequently, it is inevitable to train an IIDS for a specific target use case. Yet, this challenge of training an IIDS is not critically reflected in research where simply another (existing) dataset can be leveraged. To verify our suspicion and improve future evaluation methodologies of IIDSs to reflect their actual deployability in real-world scenarios, we ask the following research question.

How many attack samples do misuse-based IIDSs need? The training of misuse-based IIDSs requires samples of benign *and* malicious data samples. For example, the

(a) Regular training dataset with all attacks. **(b)** Start with only a single attack sample. **(c)** Iteratively add attacks for training.

Figure 6.1 Our evaluation methodology to access the number of attack samples required for learning a misuse-based model. We depict only the training data since the test part remains unchanged. Furthermore, benign data (green) remains unaffected.

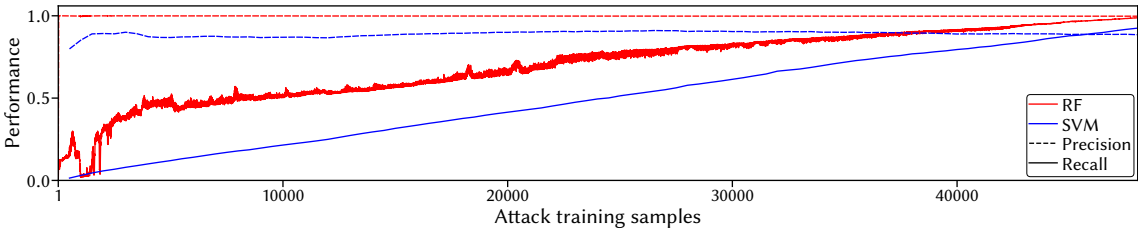
most commonly used dataset in that research area, the Morris Gas dataset [MTT15] (cf. Section 3.4.1), consists of 274.628 samples, of which 22% are attacks. For evaluations, authors usually randomly shuffle and split this dataset, leveraging 80% for training and the rest for evaluations [ASS19, PASE18]. With this split, the training data still contains around 48.000 attack samples. Yet, obtaining this number of attack samples from each ICS an IIDS should be deployed seems unrealistic considering the costs and risks associated with their collection leading to our question.

6.1.2 Experiment Setup

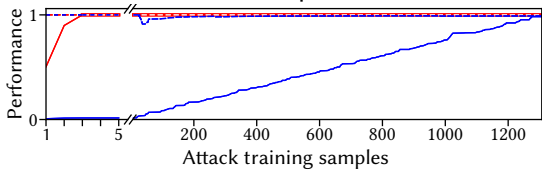
To analyze the effects of training data on misuse-based IIDSs, we consider the RF and a SVM classifier as used in several proposed IIDSs [JG16, PASE18, ASS19]. As independently examined by Perez et al. [PASE18] and Anton et al. [ASS19], these classifiers can be adapted to operate on Modbus network traffic via derived features such as the function code or transmitted process values. The classifiers are trained and evaluated on a set of benign and malicious Modbus packets. Our experiment is based on existing re-implementations of these two IIDSs made available in the IPAL IDS framework from Section 4.4.3. We took care to use the same data preprocessing and hyperparameters as mentioned in the publication [PASE18].

Concerning the datasets, we leverage the same dataset used to evaluate the two IIDSs [PASE18, ASS19], which is also the most commonly used dataset for misuse-based IIDSs Section 3.4.1. This dataset has been recorded in a miniature gas-pipeline ICS environment leveraging Modbus as communication protocol (cf. Section 2.1.4.1 and Section 2.3.1.5). Within this setup, a total of 60048 attack samples across 35 types of attacks with varying complexity, such as reconnaissance or modifying set-points, have been collected.

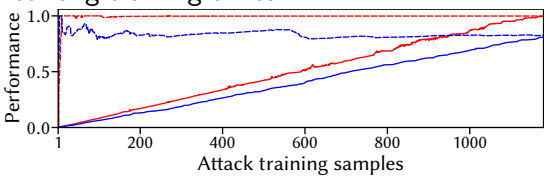
To understand how many attack samples are necessary to train a misuse-based IIDS, we reduce the number of attacks contained in the training dataset while keeping the number of benign training data constant (cf. Figure 6.1). We start with a random 80/20 train/test split as in the original evaluation [PASE18, ASS19] and five folds. We remove all but one attack sample and then gradually increase the number of attacks in the training data. For each new attack samples, we retrain the IIDS model. For each trained model, we calculate the average recall (fraction of identified attacks) and precision (fraction of correct alerts) over all folds. Our methodology allows measuring how the detection performance behaves if an IIDS is trained on a limited number of attack samples and thus enables assessing how many samples are necessary to obtain a good-performing model.



(a) After an initialization phase, the recall increases linearly with more attack samples, while changes in precision are minimal. To yield high detection scores, more than 40.000 malicious packets are required in training for both IIDSs. Note that the data for SVM was sampled in steps of 500 attacks due to long training times.



(b) The RF requires few training samples on attack number 19 (clean registers).



(c) Attack 10 (change physical value) requires many samples to be trained.

Figure 6.2 Gradually increasing the number of attack samples within the training data reveals that both RF and SVM require lots of data to yield satisfying detection performance. For a simple attack, cf. Figure 6.2b, the RF requires only about three samples.

6.1.3 How many attack samples do misuse-based IIDSs need?

Having established our methodology, we can now assess the deployability of misuse-based IIDSs with respect to the number of attack samples. To this end, Figure 6.2 depicts the detection performance of the RF and SVM classifiers. In Figure 6.2b and Figure 6.2c, we show two exemplary attack types in isolation.

We begin with an overview in Figure 6.2a. Both IIDSs achieve the best detection performance when trained on all (those in the original train set) available attack samples. However, as we reduce the number of attack samples, we observe a nearly linear reduction in recall of both IIDSs. For RF, the performance drops from 0.987 to about 0.44 if just 5000 attack samples are provided. Below this threshold, the recall for RF drops even more. The SVM shows a similar trend while showing fewer fluctuations. Interestingly, precision remains largely unaffected in both cases, which we presume results from not changing the amount of benign training data.

To better understand these effects, we now conduct the same experiment while only considering a single attack type during training and testing. For example, we show attack type 19 (as defined by the Morris-Gas dataset [MTT15]) in Figure 6.2b and attack type 10 in Figure 6.2c. For attack type 19, we observe a vastly diverging behavior between RF and SVM. The RF achieves optimal detection rates after just three attack samples. Here, the IIDS has likely learned to identify that this attack uses a Modbus function code not occurring during normal behavior. In contrast, this generalization does not apply to the SVM. Attack type 10, shown in Figure 6.2c, which manipulates reported sensor readings, proves difficult to learn for both IIDSs. Here, the recall continues to grow linearly as more attack samples are available for

training. Overall, we see that only with a high number of malicious training samples can the IIDSs score the excellent detection results reported in the publications.

Looking closer at our results, we see that misuse-based IIDSs can generalize an attack pattern in some cases, as observed, for example, for the RF classifier for attack type 19, which introduces an otherwise unused Modbus function code. This attack should thus also easily be detected by simple rule-based IIDSs [FCD+10]. In general, however, we observe relatively little generalization for both IIDSs. The linearly increasing recall scores with increasing the number of attack samples rather indicate an overfitting behavior of the classifiers. Only the precise misbehavior observed during training is also later classified as such. These results provide further evidence for prior work by Kus et al. [KWP+22a], who already identified a lack of generalization during supervised IIDS training.

6.1.4 Discussion and Conclusion

We find that misuse-based IIDSs are rather unsuited in diverse ICS environments due to only performing well with many attack samples in training, potentially due to overfitting, which aligns with previous research [Eta17, AMM20, MAM21, KWP+22a]. Yet, these issues are hardly discussed in prior work as publications promoting the use of machine learning in ICS mainly focus on the detection performance [JG16, PASE18, ASS19]. Consequently, novel designs for misuse-based IIDSs must be critically reviewed to be considered suitable for real ICS deployments.

From a research perspective, an adopted evaluation methodology that more deeply assesses the capabilities and especially training properties of an IIDS in the lab may be suited to estimate its training demand upfront. In that regard, the evaluation methodology we presented enables, on the one hand, inferring the learning rate from which the amount of required training data for a deployment can be estimated. On the other hand, by visualizing the learning rate of individual attacks, first signs of overfitting can be revealed. In addition, the methodologies proposed in related work [KWP+22a, ALS23] can answer how well a misuse-based approach generalizes to unknown attacks, e.g., found during live operations, which are not part of the training data. Together, such enhanced evaluation methodologies can reveal IIDSs that a) require little training samples and b) generalize to a wide variety of (zero-day) cyberattacks beyond the ones seen in training.

While the sketched concepts may work well for research, it is not directly apparent how their insights transfer to deployments. For example, the issue of operational drifts such as wear and tear, which can invalidate once-trained models over time, has been neglected by us [MAM21]. Answering whether an IIDS is ultimately deployable in an actual system thus likely has to involve the expertise of ICS stakeholders. Regarding our work, we can, therefore, not finally argue how much training data would still be acceptable or how many false positives and false negatives are tolerable without conducting experiments together with ICS experts within an actual ICS.

SWaT	eTaP	eTaR	eTaF1	Scen.	FPAs
TABOR	49.1	18.9	27.3	20	27
Seq2SeqNN	42.8	47.2	44.9	27	36
PASAD	16.0	4.9	7.5	16	14
Invariant	55.4	32.7	41.1	31	185
MinMax	67.8	47.1	55.6	24	7
Gradient	20.5	6.0	9.2	26	27
Steadytime	81.6	30.1	44.0	14	4
Histogram	70.9	23.2	34.9	13	0
GeCo	83.1	60.7	70.2	31	4

WADI	eTaP	eTaR	eTaF1	Scen.	FPAs
TABOR	14.9	13.0	13.9	8	3
Seq2SeqNN	45.4	31.3	37.1	9	7
PASAD	5.4	4.3	4.8	5	3
Invariant	92.3	32.6	48.1	6	2
MinMax	74.8	47.4	58.1	8	3
Gradient	69.6	18.1	28.8	7	6
Steadytime	87.0	38.7	53.5	7	1
Histogram	63.7	43.2	51.5	7	6
GeCo	91.3	56.3	69.7	11	0

HAI	eTaP	eTaR	eTaF1	Scen.	FPAs
TABOR	0.0	0.0	0.0	23	4
Seq2SeqNN	7.7	2.9	4.2	13	14
PASAD	1.0	2.0	1.3	8	51
Invariant	79.2	25.3	38.3	25	14
MinMax	87.8	60.5	71.6	43	3
Gradient	82.7	40.1	54.0	40	1
Steadytime	82.5	17.2	28.4	14	0
Histogram	82.5	17.2	28.4	14	0
GeCo	73.8	65.4	69.4	45	8

BATADAL	eTaP	eTaR	eTaF1	Scen.	FPAs
TABOR	77.7	14.3	24.1	2	2
Seq2SeqNN	27.0	6.9	11.0	2	1
PASAD	10.5	21.5	14.1	11	32
Invariant	18.2	74.9	29.3	14	865
MinMax	100.0	30.5	46.7	6	0
Gradient	89.8	30.1	45.1	7	3
Steadytime	91.0	49.3	63.9	9	1
Histogram	33.3	26.5	29.5	7	1
GeCo	97.0	88.1	92.4	14	0

Detected Scenarios (Scen.) counts the number of absolute detected attacks.

Table 6.1 These nine approaches from relevant literature to detect cyberattacks highlight that relying on a monolithic IIDS introduces risks, as none of them detects all attacks. Moreover, determining the "best" IIDS heavily depends on the chosen metric.

6.2 One IIDS is not Enough

Having determined misuse-based IIDSs to be unsuited for the protection of ICSs, the large class of anomaly detection operators can choose from still remains. These IIDSs are capable of detecting even zero-day attacks. But this advantage comes at the risk of emitting False-Positive Alarms (FPAs) [AMM20, AAM22].

In practice, every alert has to undergo analysis by an operator to decide whether it is reasonable to interrupt operation in case of a suspected cyberattack [SH24]. Consequently, as stated by Etalle et al. [Eta17], FPAs significantly contribute to an IIDS's total cost of ownership. Moreover, since cyberattacks against ICSs' are still rare [ACZ20], minimizing the number of FPAs is equally as important as detecting attacks because they increase the risk of *alarm fatigue* [AAM22] where operators start ignoring the IIDS over time, such that actual attacks can slip through.

To meet these high detection standards, most scientific proposals aim to find yet another single-best, *monolithic* IIDS that outperforms existing work. From this pool of solutions, ICS operators must decide which conceptual approach they adopt by weighing all proposals based on their capabilities. However, this decision can be difficult as prior work raised the suspicion that no optimal IIDS exists that detects sufficiently many attacks [ET22]. To confirm this claimed insufficiency, we analyze nine IIDSs implementing anomaly detection in the following.

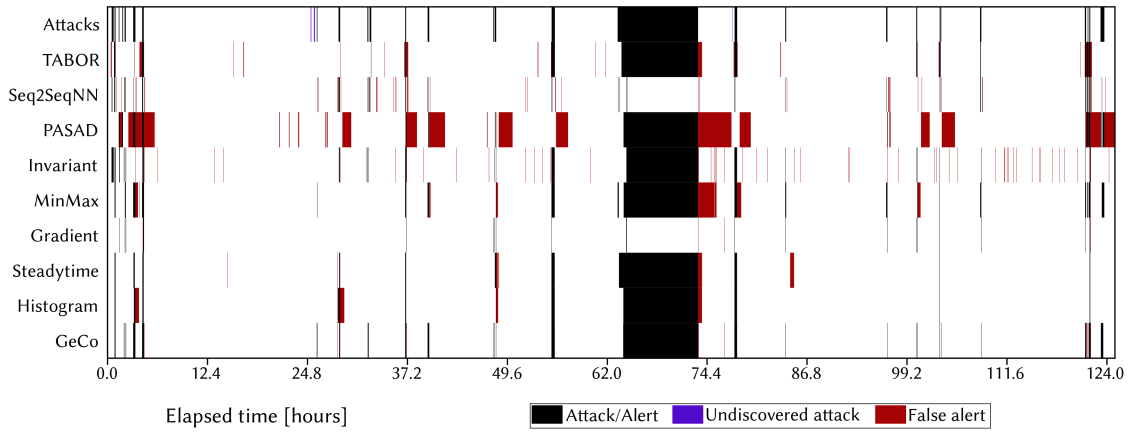


Figure 6.3 The alerts emitted by IIDSs on the SWaT dataset exemplify the challenges operators face during investigations, as they each exhibit distinct alerting behavior.

More precisely, we study the capabilities of the four reproduced IIDSs TABOR, Seq2SeqNN, PASAD, and Invariant, and our five proposals MinMax, Gradient, Steadytime, Histogram, and GeCo from this dissertation (cf. Section 5.1 and Section 5.2). Subsequently, we compare them against all four considered datasets, namely SWaT, WADI, HAI, and BATADAL. Note that we consider the absolute number of detected scenarios (Scen.) in this chapter compared to the relative number of detected scenarios in percent as done before. This absolute count of scenarios eases discussing in the following whether an approach detects one or two more scenarios compared to the number of absolute FPA. As a reminder, SWaT features 36 attacks, WADI and BATADAL 14 attacks and HAI 50 attacks.

Given the results from Table 6.1, we can confirm the IIDSs' insufficiency. No single approach detects all 36 cyberattacks (cf. Scen.) on SWaT, yet 33 would be detectable in combination. Moreover, across the metrics, no monolithic IIDS performs best in all of the metrics (cf. grey cells in Table 6.1), indicating that certain compromises towards one or another metric have to be made by operators selecting an IIDS for their ICS. On WADI, GeCo is nearly optimal except for one metric but no IIDS detects all 14 attacks whereas 13 would be detectable in combination.

While the Invariant IIDS excels in detected scenarios on SWaT (together with GeCo), it fares badly concerning FPAs (185). On the contrary, Histogram detects the fewest cyberattacks on SWaT but also yields no FPAs, resulting in reliable indications that can counteract the risk of alarm fatigue. Both extremes seem undesirable for a deployment, and a trivial combination, e.g., with a logical OR, would enable great detection yet add up all the FPAs from both approaches.

Aside from detection performance, the alerts should be accountable to ICS operators to initiate appropriate countermeasures [SP10, Eta17]. Yet, when visually analyzing the alerts (cf. Figure 6.3), we observe how differently these IIDSs indicate attacks. MinMax produces seven FPAs occurring near the actual attacks, which is in stark contrast to Invariant spreading alarms even across broad regions of benign behavior. Contrarily, PASAD, and Histogram exhibit a different phenomenon of "overhanging" alerts (red) after the attacks, which complicates determining the actual range of an

attack. Given these distinct behaviors, truly understanding an IIDS's alerts requires expert knowledge of the underlying detection mechanism.

Takeaway. IIDSs have complementary strengths and opposing weaknesses. Thus, relying on a single IIDS can yields suboptimal ICS security. This situation increases the burden for operators, who must select an IIDS that best fits their deployment, as a trivial ensemble of IIDSs, where all alerts are simply ORed, would better detect attacks yet likely suffer from an excessive multitude of FPAs.

6.3 Motivation for Ensemble Learning

This disappointing situation from Section 6.2 motivates looking for solutions that cleverly combine the advantages of multiple IIDSs to (i) improve the detection capabilities and (ii) make alarms generally more precise. Fortunately, real-world deployments are not restricted to integrating just the best-performing IIDS. Instead, they can choose from the several available approaches to play off their advantages and disadvantages against each other. These observations raise the question of whether a combination of multiple IIDSs can join forces to optimize detection performance beyond what each individual approach can achieve.

The problem of combining a collection of classifiers (e.g., IIDSs) into a single system is generally known as ensemble learning [TPSJ12, ZM12, SR18], a concept that we introduce in the next Section 6.3.1. However, while the training of an ensemble is relatively straightforward and well studied for misuse-based IIDSs (cf. Section 6.3.2), where simply another round of training suffices [GGB+21, LXW+21, KWP+22b, NLC22], retaining the anomaly detection nature, i.e., requiring no attack knowledge during training, as is desirable for usage in ICSs, proves challenging. We lay out these challenges in Section 6.3.3 after which we explore how *ensembles of IIDSs with anomaly detection* can be built and to which extent they are superior to monolithic deployments, e.g., by detecting more attacks in total or reducing FPA.

6.3.1 Background on Ensemble Learning

Instead of proposing and evaluating new IIDSs, leveraging multiple IIDSs side-by-side can be another path to more secure ICSs. In this regard, ensemble learning represents a subfield of machine learning concerned with the process of training multiple classifiers, i.e., IIDSs, and fusing them into a single model to increase predictive performance [TPSJ12, ZM12, SR18] as proven successful in, e.g., the fields of computer vision [LLL+11] or biometric recognition [SSR19]. The methods in the literature can be divided into homogeneous and heterogeneous ensembles. Since homogeneous ensembles combine the same type of classifier, i.e., only SVMs, they are rather suited to create monolithic IIDSs, as demonstrated before [PA16, MJC16, CZLT18, KSC21, LT21, YKP+23]. In contrast, heterogeneous ensembles combine diverging approaches. Given the diversity of IIDSs proposed in the literature (cf. Section 3.5.1), we focus on heterogeneous ensemble learning.

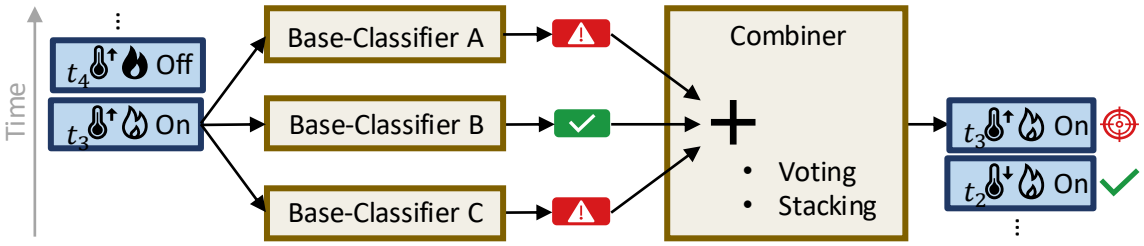


Figure 6.4 With ensemble learning, a combiner joins the output of several base-classifiers into a single decision. In the case of learning-based ensembles, an additional training step on the outputs of the base-classifiers and the expected outcome may be necessary.

The general methodology for heterogeneous ensemble learning is visualized in Figure 6.4. First, a set of *base-classifiers* is pre-trained on a training dataset. Then, their outputs are fed into a *combiner* which fuses them into a single classification. More precisely, the input of the combiner is a judgment of each IIDS, whether it has identified an alert (1) or classifies the situation as benign (0) for the current data point. For a set of n IIDSs, the algorithm receives an alert vector $v \in \{0,1\}^n$. Hereby, the combiner can either be learning-based, i.e., *stacking* machine learning classifiers trained over a dataset, or rule-based, e.g., weighted *voting* to combine binary outputs [MMSJS12, TPSJ12]. According to best practices [Rok10, Zho21], a learning-based combiner must be trained on out-of-sample data for the base-classifiers, which can be achieved, e.g., by splitting the dataset into separate train sets for base-classifiers and combiner. When measuring the combiner performance, the evaluation must be conducted on a set of previously unseen data.

Lastly, although ensembles promise to improve detection performance, they can hinder interpretability [SP10] or accountability [Eta17], i.e., they complicate reconstructing why an alarm is emitted, which is important for attack mitigation.

6.3.2 IIDS Ensemble Learning in Related Work

Ensemble learning is already used to fuse (smaller) detection concepts into an IIDS [VVKK04, LAVM18, AKDP20, KYK20, RSE+20]. For example, TABOR [LAVM18] fuses three detection methods together. Also, Seq2SeqNN [KYK20] trains one neural network for each ICS process stage (six for SWaT), and only a single model has to emit an alert. Al-Abassi et al. [AKDP20] use decision trees to stack the results of multiple neural networks. Lastly, Radoglou-Grammatikis et al. [RSE+20] combine two IIDSs, tasked to detect known and unknown attacks respectively, using a logical OR. While such approaches fall into the category of rule-based ensembles, they aim to establish a monolithic IIDS rather than complex ensembles.

Still, more complex ensemble learning has been examined for IIDS applications. Kus et al. [KWP+22b] analyzed the impact of several methods covering voting and stacking to join seven *misuse-based* IIDSs, such as RFs or SVMs, but only achieved marginal improvements below 1 % in the F1 score. Likewise, Upadhyay et al. [UMZS21] combined six *misuse-based* IIDSs with majority voting achieving similar results. Gao et al. [GGB+21] combined two deep learning models using a stacked

Multi-Layer Perceptron (MLP) with improvements in F1 score of only up to 0.41 %. Nguyen et al. [NLC22] combined three classifiers using stacking with an MLP. They achieved an increase of 0.14 % in F1 score but had little room for improvement in the first place as the best base-classifier achieved an F1 score of 99.58 %. Li et al. [LXW+21] combined three classifiers using majority voting to increase accuracy by up to 1.62 %. Lastly, Balaji et al. [BSAM21], who experimented with Logistic Regression (LR) as ensemble method, conclude that they rather learned attack signatures and leave unsupervised learning to future work. However, to date, comparable research on ensembles for anomaly detection is still missing.

6.3.3 Challenges of IIDS Ensembles for Anomaly Detection

In related work, ensemble learning managed to achieve only slight improvements (around +1 % in F1 score) in the IIDS context, likely due to small margins within the used base-IIDSs (cf. Section 6.3.2). In our baseline results from Table 6.1, however, we observed different behaviors and IIDSs distinctly detecting attacks, indicating much greater potential for ensembles to work with.

Notably, most related work from the IIDS domain considers *misuse-based* ensembles [LXW+21, GGB+21, KWP+22b, NLC22]. This severe limitation introduces the risk of only detecting the trained attacks [KWP+22a], effectively transforming anomaly detection back into a misuse-based IIDS. Moreover, given their reliance on attack data during training which is scarce in the ICS domain [BSL+23, CDT21] (cf. Section 6.1), the standard methodology used to train (and evaluate) such misuse-based ensembles is incompatible with the training *anomaly detection*, which is imperative in the ICS domain. To this end, we explain the three requirements (R1–R3) ensembles for anomaly detection have to fulfill.

R1–Benign training only. IIDSs implementing anomaly detection train exclusively on benign data, eliminating the need for attacks in the training set (cf. Section 2.2.3). While, in the ICS context, the ensemble's base-IIDSs could still be trained on benign data, ensemble learning usually remains dependent on observing the IIDSs under benign *and* attack conditions (cf. Section 6.3.2). This limitation effectively rules out the types of learning-based ensembles discussed in related work.

R2–Sequential data series. One property of IIDSs that is critical for their success is their ability to analyze dependencies in sequential data series. For example, Seq2SeqNN accumulates the drift between predicted and observed behavior, detecting even subtle deviations [KYK20]. Thus, the input data must be ordered chronologically, making the standard methodology from related work to randomly shuffle datasets into training and evaluation parts [GGB+21, KWP+22b, LXW+21, NLC22] infeasible for ensembles with anomaly detection as it alters a data series' order.

R3–Temporal dependencies. Another temporal effect is that cyberattacks persist over a longer period of time, e.g., to thwart detection by inflicting the damage slowly [AIA18]. Since the effects may not happen instantaneously, an IIDS may detect attacks with some delay. For example, one base-IIDS might detect the attack at an early stage while a second IIDS requires more time, such that their alerts do not

overlap. This issue demands time-aware ensembles which are capable of correlating both alerts to one attack. Yet, current ensembles base their decisions on a single data point without considering recent history, and no methods that take advantage of such temporal dependencies are known to us.

Takeaway. Due to their individual inefficiencies, a single IIDS is not enough for strong ICS cybersecurity. However, ensemble learning promises to use precisely this diversity as an advantage, albeit existing ensemble methodologies turn out infeasible because of the unique requirements of anomaly detection.

6.4 The Potential of Ensemble Learning

As learning-based ensemble methods are inapplicable to anomaly detection (cf. Section 6.3.3), novel ways to fuse IIDSs into ensembles are required. Following the principles of simplicity from Section 5.1, we first examine the potential of weight-based voting schemes in Section 6.4.1 to establish a theoretical baseline. We then study to how this potential can actually be leveraged in Section 6.4.2.

6.4.1 Potential Analysis of Weight-based Ensembles

Ensemble methods for anomaly detection have to fulfill a new set of criteria compared to related work relying on misuse-based training (cf. Section 6.3.3). For our first potential analysis, we consider voting mechanisms, an approach that is not dependent on attack data for training and retains the temporal order of the data.

6.4.1.1 Design

As the first step, we train every base-IIDS on benign data for the target ICS such that, when presented with new data, we obtain an alert vector v encoding the judgment of each IIDS. Next, the ensemble assigns an individual weight $w \in \mathbb{R}^n$ to each IIDS. Then, a combined alert is emitted if the weighted sum of IIDS outputs meets a threshold $t \in \mathbb{R}$, i.e., $v_1 \cdot w_1 + \cdots + v_n \cdot w_n \geq t$.

The crucial part is finding suitable weights w and a threshold t which can either be achieved manually to implement strategies such as majority votes, or derived from knowledge of previous deployments. Consequently, this ensemble method does not necessarily require training (R1). Moreover, it leaves the data series intact (R2) yet does not consider temporal dependencies (R3).

6.4.1.2 Selecting a Baseline

Before exploring this ensemble method's potential, we must define a baseline for comparison. However, as apparent from our previous study, an IIDS can be optimized for certain objectives, e.g., to perform well in a particular metric (cf. Section 6.2).

SWaT	eTaP	eTaR	eTaF1	Scen.	FPAs
MinMax	67.8	47.1	55.6	24	7
GeCo	83.1	60.7	70.2	31	4
Opt. Weights	88.2	65.7	75.3	31	4
Any	22.6	39.3	28.7	33	152
≥2-Alerts	53.2	47.7	50.3	31	43
≥3-Alerts	60.3	35.6	44.8	29	41
≥4-Alerts	75.2	37.9	50.4	27	31
Majority	86.8	33.2	48.0	17	14
All	85.5	2.9	5.6	4	0

WADI	eTaP	eTaR	eTaF1	Scen.	FPAs
MinMax	74.8	47.4	58.1	8	3
GeCo	91.3	56.3	69.7	11	0
Opt. Weights	91.9	77.6	84.2	12	2
Any	33.9	51.4	40.9	13	13
≥2-Alerts	43.0	42.3	42.6	13	2
≥3-Alerts	64.7	60.5	62.6	11	9
≥4-Alerts	86.1	42.9	57.3	8	1
Majority	92.2	28.7	43.8	6	2
All	0.0	0.0	0.0	1	0

HAI	eTaP	eTaR	eTaF1	Scen.	FPAs
MinMax	87.8	60.5	71.6	43	3
GeCo	73.8	65.4	69.4	45	8
Opt. Weights	79.5	71.8	75.4	46	0
Any	11.8	44.3	18.6	46	62
≥2-Alerts	49.7	55.6	52.5	44	25
≥3-Alerts	83.0	50.6	62.9	41	6
≥4-Alerts	81.5	29.0	42.8	27	1
Majority	76.6	19.6	31.2	18	1
All	0.0	0.0	0.0	0	0

BATADAL	eTaP	eTaR	eTaF1	Scen.	FPAs
MinMax	100.0	30.5	46.7	6	0
GeCo	97.0	88.1	92.4	14	0
Opt. Weights	97.3	87.9	92.4	14	0
Any	7.7	45.0	13.2	14	564
≥2-Alerts	28.6	59.9	38.7	14	267
≥3-Alerts	69.0	62.7	65.7	11	42
≥4-Alerts	87.7	51.8	65.1	10	8
Majority	90.3	30.2	45.2	5	5
All	100.0	12.7	22.5	2	0

Detected Scenarios (Scen.) counts the number of absolute detected attacks.

Table 6.2 IIDS ensembles have the potential to yield better results than any monolithic approaches (cf. Opt. Weights). But, to find these weights via means of trivial strategies (lower part), a tradeoff between detected scenarios and FPAs has to be made.

This decision is also heavily influenced by how FPAs and the risk of missing an attack are weighted by researchers (or operators), which is why finding one suitable metric is difficult in general [LAVM18, KYK20, HYKM22].

Consequently, we decided to compare our ensembles against two promising IIDSs from the baseline. Given the results from Table 6.1, we select the MinMax and GeCo IIDSs as our primary baseline since they feature the best eTaF1 score on average, which is a tradeoff between good precision and recall.

6.4.1.3 Results

To examine the *theoretical* potential achievable with this methodology in the first step, we leveraged an optimization algorithm to systematically search for optimized weights and thresholds (Opt. Weights). To this end, we leveraged Ray Tune [LLN+18] to optimize these parameters for the eTaF1 metric. Note that weights and the threshold were searched within the interval $[-1, +1]$ instead of the entire $\mathbb{R}$ w.l.o.g. (any weighted vote can simply be scaled such that all the parameters are within any arbitrary non-empty interval in $\mathbb{R}$).

A weight-based ensemble with optimized parameters manages to outperform any base-IIDS in eTaF1 (cf. upper part of Table 6.2). Surprisingly, GeCo's eTaF1 score

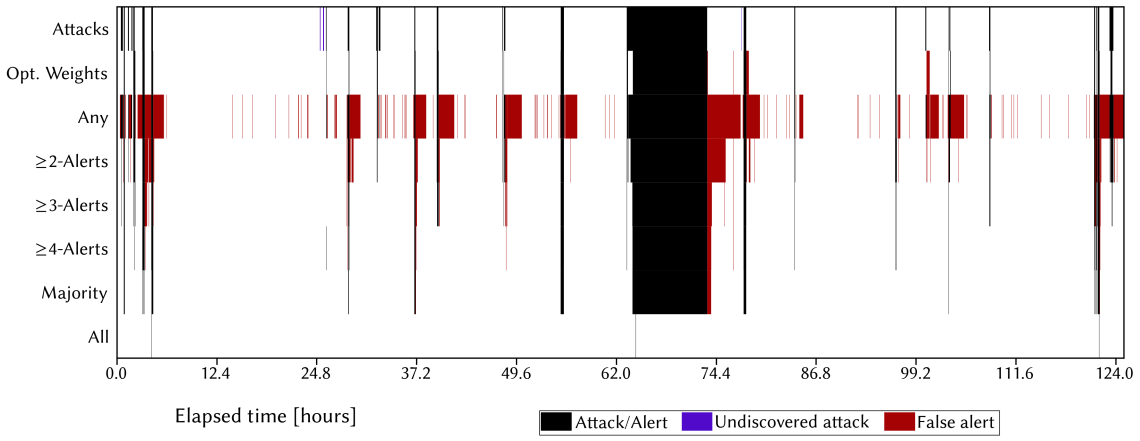


Figure 6.5 Simple ensemble strategies drastically improve the understandability of alerts compared to the patterns shown in Figure 6.3 for SWaT. Even though strategies like ≥2-Alerts measure 43 FPAs on SWaT, these reside within the vicinity of attacks.

on SWaT is exceeded by +5.1% points, which is a major improvement, especially considering related work on ensemble learning yielded improvements of only around +1 % point (cf. Section 6.3.2). Together with Invariant and GeCo, this ensemble detects the same amount of scenarios on SWaT, namely 31 of the 33 attacks alerted by any base-IIDS. While this ensemble still exhibits 4 FPAs, they are confined to the vicinity of actual attacks (cf. upper part of Figure 6.5). More importantly, the ensemble's alerts do not show any phenomena discussed in Section 6.2, i.e., overhanging or randomly distributed alerts are eliminated, drastically improving understandability.

Also on the other datasets, we observe substantial improvements. The eTaF1 score on WADI is even more than +14% points higher than the best monolithic IIDS and the ensembles detects twelve scenarios which is one more than any base IIDS. No performance increase is achieved for BATADAL but here GeCo already yields an identical eTaF1 score of 92.4% points. Still on BATADAL, the ensemble again detects all 14 attacks with zero FPA as the best standalone IIDS GeCo.

When analyzing the optimized weights for the SWaT dataset, it is unsurprising that MinMax and GeCo, both with the highest eTaF1 scores among the base-IIDSs, received the highest weight both with 0.89 and could already yield an alert on its own since $t = 0.53$. The ensemble can even make use of Invariant with a weight of 0.28 despite its FPAs. Interestingly, some IIDSs received negative weights (Histogram −0.44, TABOR −0.10, Seq2SeqNN −0.32, and PASAD with even −0.70). We presume that the ensemble leverages those IIDSs to filter out FPAs, e.g., since PASAD in some cases, does not overlap with any attack at all (cf. Figure 6.3).

6.4.2 Finding Parameters for Weight-based Ensembles

Our theoretical results prove that weight-based ensembles are of practical use given a set of suitable weights and a threshold, but they do not provide ways to find such a configuration in a practical manner. Since performing such an optimization is

infeasible in practice due to a lack of training data (R1), we now investigate six straightforward strategies to choose those parameters *manually*.

6.4.2.1 Manual Strategies

The most basic strategies we can implement with weights are *All* and *Any*, which emit an alert if all/any IIDSs emit an alert at the same time (corresponding to a logical AND or a logical OR). To realize them, setting all weights to 1 and the threshold to $t = 1$ (Any) or $t = n$ (All) is sufficient. Applications of these strategies can already be found in literature (cf. Section 6.3.2).

In between these extremes, further combinations are imaginable. For our evaluation, we consider just a subset of these combinations for the sake of simplicity: We define four more strategies that raise an alert when the *Majority* of IIDSs emit an alert or at least two, three, or four alerts are present. Note that these manual strategies can all be implemented with the proposed method, e.g., again setting all weights to 1 and $t = n/2$ corresponds to the Majority strategy.

6.4.2.2 Results

Overall, the results from such simple configurations cannot keep up with the optimized performance (cf. lower part of Table 6.2). When considering the trend of detected scenarios and FPAs from *Any* to *All*, these strategies suffer from being too conservative in either eTaP (precision) or eTaR (recall). While *Any* indicates all 33 detectable attacks on SWaT and *All* has no FPAs, they perform worse in the respective opposite metrics. Moreover, none of the trivial ensembles manage to improve eTaF1 compared to the single-best base-IIDS on all datasets.

Fortunately, when digging deeper into the ensemble results by visualizing their alert patterns in Figure 6.5 (lower part), all approaches, besides All and Any, score reasonably well on SWaT when evaluated qualitatively. Their alerts visually coincide with the attacks, and the FPAs are in close proximity to the actual attacks, notably eliminating Invariant's many randomly distributed FPAs. Compared to the original base-IIDSs' alerts from Figure 6.3, even these straightforward ensemble strategies yield a usable result that can improve the perception for the ICS operator. However, this observation is not backed by current metrics.

Takeaway. Weight-based ensembles increase the performance by up to +14% points in eTaF1 on the WADI dataset compared to the best monolithic IIDS even detecting more scenario than any base-IIDS. However, finding suitable weights is non-trivial in the absence of training data because simple strategies, such as a majority voting, leave a gap toward the optimum. Still, the advantages of weight-based ensembles lie in their simplicity, e.g., that weights can be tuned manually throughout the operation, and their ability to reduce FPAs effectively.

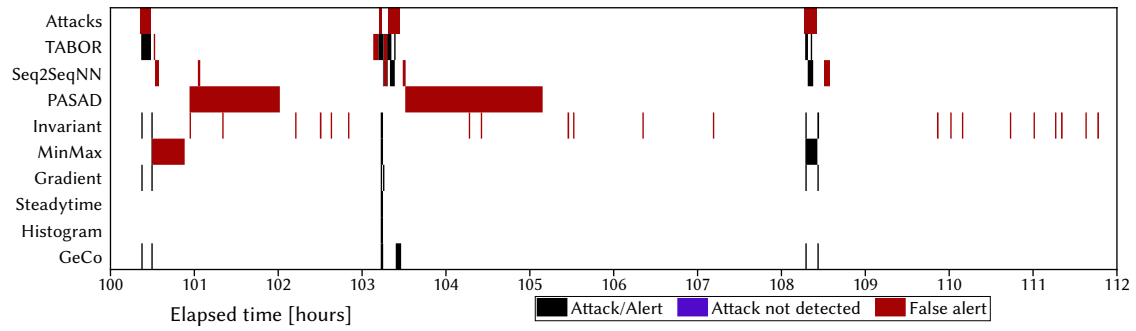


Figure 6.6 This close-up reveals the individual alerting behavior of different IIDSs on the SWaT dataset. However, by ignoring temporal correlations, an ensemble cannot differentiate between, e.g., Invariant's short alarms that usually indicate FPAs and longer true alarms.

6.5 Time-aware Ensemble Learning

One property of IIDSs that could impair the success of actual ensembles may be temporal effects (cf. R3). Recent metrics, such as eTaF1, are already time-aware, i.e., they can deal with IIDSs that alert at different times during a longer attack. But, to the best of our knowledge, ensemble learning focuses solely on single instances without temporal dependence. To shed light on this issue, we first examine the alerting behavior of the IIDSs (cf. Section 6.5.1) and identify the potential for optimizing their alerts by taking temporal correlations into account (cf. Section 6.5.2). We then assess whether novel concepts of time-aware ensembles can improve the current situation (cf. Section 6.5.3).

6.5.1 Individual Alerting Behaviors Complicate Ensembles

Calculating the optimal weights resulted in unexpected behavior (cf. Section 6.4.1). For example, Gradient was given a low weight of 0.08, even though it visually performs well (cf. Figure 6.3). Similarly, Invariant received a high weight of +0.27, despite exhibiting 185 FPAs. To understand these outcomes, Figure 6.6 provides a close-up of a few attacks and corresponding alerts for the SWaT dataset.

Starting with MinMax, whose alerts are occasionally flagged as FPAs, these simply seem delayed such that a trained ensemble or an operator knowing this phenomenon could still use these indications for attack detection. Especially if other IIDSs indicate attacks around the same time. Likewise, Gradient indicates discontinuities at an attack's beginning and end. But, its second alert is often counted as FPA as the attack has ended at that time instance (cf. also Figure 5.3). Next, although hardly recognizable in Figure 6.6, Invariant's randomly occurring FPAs are usually short, i.e., lasting just a few seconds, compared to true alarms. The same effect can be observed for TABOR where FPAs are often just one second short.

Such effects pose issues to ensembles without temporal knowledge, as they can hardly distinguish different types of alarms. Yet, knowing such (often technically conditioned) effects provides unique opportunities for stronger ensembles.

6.5.2 Time-aware Ensemble Learning on Normalized Alarms

Our initial attempt to leverage this time-aware information in ensembles is to introduce a postprocessing step per IIDS right after each IIDS has emitted its alerts. Thereby, we can implement simple strategies that "normalize" the alerts prior to forwarding them to the ensemble method for decision-making.

From our case study in Section 6.5.1, we identified four strategies for MinMax, Gradient, Invariant, and TABOR, which help to clean up their alerts. For MinMax and Gradient, we simply extend their alerts artificially by one minute such that scenarios are detected if the alert is just slightly off. For Invariant, which suffers from randomly placed short alerts, we only consider those alerts where an alarm is emitted for more than ten consecutive seconds. We chose a similar approach for TABOR, where we filter out every alert that lasts just one time instance.

Incorporating temporal information into ensembles may help to obtain better models and ultimately reduce the number of FPAs due to "misinterpretation". In practice, IIDS authors can provide guidance to aid in developing such strategies.

6.5.3 Potential of Time-aware Ensemble Learning

To assess the potential of time-aware ensembles, we applied these strategies in isolation for each of the four IIDSs: Invariant, MinMax, Gradient, and TABOR. The other IIDSs remained unaffected.

As shown in the upper parts of Table 6.3 (Baseline), this approach improves their performance. Compared to the previous baseline, MinMax reduces the total number of FPAs to two for SWaT and WADI. The same observation holds for Gradient, which reduces the FPAs by 26 for SWaT and thereby increases its eTaF1 score by 31.9 % points. These results indicate that most FPAs of MinMax and Gradient were in close proximity to actual attacks or triggered during the recovery of the ICS right after the attack. The strategies for Invariant and TABOR likewise yield a reduction in FPAs but also in the detected scenarios (in the case of Invariant). We assume that the reduction of detected scenarios for Invariant results from the filtered random alerts, which may have (falsely) contributed to the high number in the first place.

This improvement is not restricted to the base-IIDSs as it carries over to the ensembles, which now fare better using these "normalized" alerts (cf. New Results in Table 6.3). For example, the optimized weight-based ensemble on SWaT can increase the eTaF1 score (+2.5), detect one scenario more and reduce the FPAs by two. These improvements also transfer to the simple strategies with manually chosen weights as they also increase their eTaF1 scores on average. Here, the strategy $\geq$*3-Alerts* yields the best eTaF1 score in three out of the four datasets making it consistently the best manual ensemble.

Takeaway. Temporal knowledge inside ensembles improves their performances to a point where manual strategies can become usable, but usually still with fewer detected attacks than the optimum. Besides incorporating IIDS-specific information,

		IIDS/Ensemble	eTaP	eTaR	eTaF1	Scen.	FPAs
SWaT	Baseline	MinMax	$68.5^{+0.7}$	$48.8^{+1.7}$	$57.0^{+1.4}$	24^{+0}	2^{-5}
		Gradient	$45.7^{+25.2}$	$37.4^{+31.5}$	$41.1^{+31.9}$	26^{+0}	1^{-26}
		TABOR	$50.5^{+1.4}$	$18.9^{-0.0}$	$27.5^{+0.2}$	20^{+0}	9^{-18}
		Invariant	$90.5^{+35.1}$	$30.9^{-1.8}$	$46.1^{+5.0}$	14^{-17}	3^{-182}
	New Results	Opt. Weights	$85.4^{-2.7}$	$71.4^{+5.7}$	$77.8^{+2.5}$	32^{+1}	2^{-2}
		Any	$23.4^{+0.8}$	$36.9^{-2.4}$	$28.6^{-0.1}$	33^{+0}	40^{-112}
		≥2-Alerts	$54.2^{+1.1}$	$49.2^{+1.5}$	$51.6^{+1.3}$	31^{+0}	15^{-28}
		≥3-Alerts	$64.6^{+4.4}$	$43.6^{+8.0}$	$52.1^{+7.3}$	28^{-1}	8^{-33}
		≥4-Alerts	$81.4^{+6.2}$	$41.3^{+3.4}$	$54.8^{+4.4}$	22^{-5}	6^{-25}
		Majority	$90.4^{+3.6}$	$35.7^{+2.5}$	$51.1^{+3.2}$	16^{-1}	1^{-13}
		All	$67.0^{-18.5}$	$3.9^{+1.0}$	$7.4^{+1.7}$	4^{+0}	0^{+0}
WADI	Baseline	MinMax	$71.6^{-3.2}$	$48.4^{+1.0}$	$57.7^{-0.3}$	8^{+0}	2^{-1}
		Gradient	$57.6^{-12.0}$	$31.6^{+13.5}$	$40.8^{+12.1}$	7^{+0}	3^{-3}
		TABOR	$14.1^{-0.8}$	$13.0^{-0.0}$	$13.5^{-0.4}$	8^{+0}	1^{-2}
		Invariant	$96.6^{+4.3}$	$32.2^{-0.4}$	$48.3^{+0.1}$	6^{+0}	0^{-2}
	New Results	Opt. Weights	$92.3^{+0.4}$	$77.7^{+0.0}$	$84.4^{+0.2}$	12^{+0}	1^{-1}
		Any	$32.1^{-1.8}$	$47.8^{-3.6}$	$38.4^{-2.5}$	13^{+0}	9^{-4}
		≥2-Alerts	$41.2^{-1.7}$	$42.8^{+0.5}$	$42.0^{-0.6}$	13^{+0}	2^{+0}
		≥3-Alerts	$65.7^{+1.0}$	$65.1^{+4.6}$	$65.4^{+2.9}$	11^{+0}	7^{-2}
		≥4-Alerts	$85.0^{-1.0}$	$48.4^{+5.4}$	$61.7^{+4.4}$	8^{+0}	2^{+1}
		Majority	$82.6^{-9.7}$	$31.5^{+2.8}$	$45.6^{+1.8}$	6^{+0}	2^{+0}
		All	$100.0^{+100.0}$	$4.1^{+4.1}$	$8.0^{+8.0}$	1^{+0}	0^{+0}
HAI	Baseline	MinMax	$68.3^{-19.5}$	$56.9^{-3.6}$	$62.1^{-9.6}$	43^{+0}	3^{+0}
		Gradient	$51.1^{-31.6}$	$45.6^{+5.5}$	$48.2^{-5.8}$	40^{+0}	1^{+0}
		TABOR	$0.0^{+0.0}$	$0.0^{+0.0}$	$0.0^{+0.0}$	23^{+0}	0^{-4}
		Invariant	$71.1^{-8.1}$	$11.3^{-14.0}$	$19.5^{-18.8}$	10^{-15}	0^{-14}
	New Results	Opt. Weights	$77.1^{-2.4}$	$74.7^{+3.0}$	$75.9^{+0.4}$	46^{+0}	0^{+0}
		Any	$4.6^{-7.2}$	$13.7^{-30.5}$	$6.9^{-11.7}$	46^{+0}	52^{-10}
		≥2-Alerts	$42.8^{-6.9}$	$57.5^{+1.9}$	$49.1^{-3.4}$	45^{+1}	21^{-4}
		≥3-Alerts	$74.9^{-8.1}$	$57.2^{+6.6}$	$64.9^{+2.0}$	38^{-3}	2^{-4}
		≥4-Alerts	$81.1^{-0.5}$	$35.4^{+6.4}$	$49.3^{+6.5}$	27^{+0}	1^{+0}
		Majority	$73.0^{-3.6}$	$21.1^{+1.5}$	$32.8^{+1.6}$	16^{-2}	0^{-1}
		All	$0.0^{+0.0}$	$0.0^{+0.0}$	$0.0^{+0.0}$	0^{+0}	0^{+0}
BATADAL	Baseline	MinMax	$49.1^{-50.9}$	$19.1^{-11.4}$	$27.5^{-19.3}$	6^{+0}	0^{+0}
		Gradient	$10.6^{-79.2}$	$5.7^{-24.4}$	$7.5^{-37.6}$	7^{+0}	0^{-3}
		TABOR	$90.3^{+12.5}$	$14.3^{+0.0}$	$24.7^{+0.5}$	2^{+0}	0^{-2}
		Invariant	$100.0^{+81.8}$	$21.2^{-53.7}$	$35.0^{+5.8}$	4^{-10}	0^{-865}
	New Results	Opt. Weights	$97.1^{-0.2}$	$90.0^{+2.1}$	$93.4^{+1.0}$	14^{+0}	0^{+0}
		Any	$21.8^{+14.1}$	$39.6^{-5.5}$	$28.1^{+14.9}$	14^{+0}	29^{-535}
		≥2-Alerts	$37.7^{+9.1}$	$36.2^{-23.7}$	$36.9^{-1.8}$	11^{-3}	1^{-266}
		≥3-Alerts	$66.8^{-2.3}$	$41.5^{-21.2}$	$51.2^{-14.5}$	10^{-1}	2^{-40}
		≥4-Alerts	$73.8^{-13.9}$	$33.1^{-18.7}$	$45.7^{-19.4}$	8^{-2}	0^{-8}
		Majority	$92.5^{+2.2}$	$30.4^{+0.2}$	$45.7^{+0.5}$	5^{+0}	0^{-5}
		All	$100.0^{+0.0}$	$11.5^{-1.2}$	$20.7^{-1.8}$	2^{+0}	0^{+0}

Superscript numbers show the difference between the prior baseline (Table 6.1) and results (Table 6.2). The other base-IIDSs not listed here remained unaffected. Detected Scenarios (Scen.) counts the number of absolute detected attacks.

Table 6.3 Incorporating temporal knowledge about the individual IIDSs' alerting behaviors not only improves their respective detection performance (upper part) but also significantly reduces the FPAs in an ensemble (lower part).

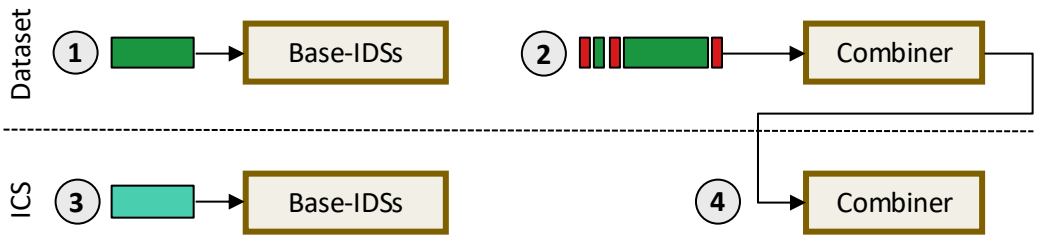


Figure 6.7 Our methodology to transfer an ensemble learned in the lab or one dataset to another ICS. Thereby, we avoid recording training data of attack scenarios in the final use case as it might not be available or difficult to obtain.

future time-aware ensembles considering inter-IIDS alert dependencies may even exceed our initial results, but likely require sophisticated methods of finding (or training) an adequate model.

6.6 A Chance for Learning-based Ensembles

Until now, we evaluated the potential of IIDS ensembles for each dataset in isolation. But one advantage of IIDSs is that they can operate in different environments, i.e., transfer to new industrial domains after another training phase (cf. Chapter 4). For example, MinMax, Gradient, Steadytime, Histogram, and GeCo have been designed, trained on, and evaluated for four datasets originating from different domains (cf. Section 5.1 and Section 5.2). Given that the set of base-IIDSs remains fixed, training a learning-based ensemble on one dataset (or in the lab) under attack conditions and then transferring the *ensemble's model* to a different deployment/dataset without known attacks might be feasible (cf. Figure 6.7). While it remains necessary to re-train the base-IIDSs on the new scenario, it hopefully suffices to keep the ensemble's model, i.e., the way in which the detection results are aggregated. This transfer-learning [TS10] is still in line with R1 as only the IIDSs would require retraining on benign-only data, while the pre-trained ensemble model is simply reused.

Transfer-learning promises a new level of flexibility for ensembles and may circumvent the issue of finding appropriate weights identified in Section 6.4.2. Also, it enables leveraging methods from related work such as ensemble stacking (cf. Section 6.3.2). To examine the feasibility of transfer-learning, we explain our new methodology in Section 6.6.1 and subsequently present the results in Section 6.6.2.

6.6.1 A New Methodology for Learning-based Ensembles

Learning-based ensembles require training on a dataset of exemplary attacks and expected labels of the outcome. Therefore, the methodology leveraged across related work usually bases on a *single* dataset artificially split into training and evaluation parts. Since we target an anomaly detection scenario, we cannot assume to observe the ensemble's IIDSs under attack conditions in the target ICS (R1).

	Transfer-learning	eTaP	eTaR	eTaF1	Scen.	FPAs
WADI → SWaT	Opt. Weights (Table 6.3)	85.4	71.4	77.8	32	2
	Opt. Weights	76.4	66.7	71.2	31	6
	SVM	62.2	62.5	62.3	32	18
	MLP	70.5	65.2	67.7	32	12
	LR	78.6	58.6	67.1	31	4
	Heuristic	70.4	50.5	58.8	30	3
SWaT → WADI	Opt. Weights (Table 6.3)	92.3	77.7	84.4	12	1
	Opt. Weights	86.3	69.2	76.8	12	1
	SVM	92.2	70.1	79.7	12	1
	MLP	86.8	64.6	74.1	12	1
	LR	91.3	60.0	72.5	12	0
	Heuristic	84.1	65.0	73.3	12	3

Detected Scenarios (Scen.) counts the number of absolute detected attacks.

Table 6.4 Transfer-learning proves helpful in training ensembles. For example, if trained on the dataset SWaT and applied to WADI, all ensembles outperform the manual strategies (cf. Majority) in eTaF1 and come close to the Opt. Weights.

However, assuming that the same set of base-IIDSs behaves similarly in a different ICS, operators may have a strong interest in reusing a well-performing, pre-trained ensemble model (cf. Figure 6.7). On the one hand, this approach enables assessing how well weights optimized for one scenario transfer to another ICS and, thus, helps to find these weights. On the other hand, misuse-based learning-based ensembles, which we neglected so far, may transfer too, such that approaches from related work can be leveraged even in the context of anomaly detection.

6.6.2 Putting Transfer-learning to the Test

To test the idea of transfer-learning, we use one dataset exclusively for training and evaluate the obtained ensemble model on another dataset. Yet, we only consider the transferability from SWaT to WADI and vice versa to limit the amount of possible combinations. Note that we use the "normalized" alerts according to Section 6.5 in this experiment to provide cleaner input data. As the first approach, we consider transferring the parameters of our weight-based ensembles. Furthermore, we analyze four learning-based variants from previous work. First, as leveraged in two publications [GGB+21, NLC22], an MLP resembles a neural network classifier. Further classifiers include SVM [KWP+22b] and LR [BSAM21, KWP+22b]. Besides these, Kus et al. [KWP+22b] proposed a heuristic that maps each possible alert vector to its most frequent output (benign or malicious), thereby optimizing the number of correct classifications.

Under the right conditions, transfer-learning can be an alternative to manual strategies (cf. Table 6.4). Yet, starting with ensembles trained on WADI and applied to SWaT (WADI → SWaT), only the Opt. Weights ensemble achieves a better eTaF1 score than the standalone GeCo IIDS but still with six FPA compared to GeCo's four FPA. The other ensembles perform even worse except for LR which also detects

the same amount of scenarios as GeCo with also four FPA. Still, no model learned on WADI and transferred to SWaT gets close to the optimum achievable with a weight-based ensemble (cf. Opt. Weights). Nonetheless, all methods outperform every manual strategies in eTaF1 (cf. Section 6.5.3).

The reverse direction (SWaT → WADI) is more promising. The weight-based ensemble detects twelve scenarios with one FPAs, which is equivalent to the previous Opt. Weights. Also, learning-based ensembles perform better if trained on SWaT. LR features zero FPA and twelve detected scenarios, an improvement over all ensemble methods before. SVM optimizes the eTaF1 score among all ensembles and exceeds the best base-IIDS GeCo by 10.0 % points, with only one FPAs.

Takeaway. Given suitable training data, transfer-learning can be successful. Yet, there is still room for improvement. Most importantly, transfer-learning can outperform simple strategies (SWaT → WADI), thus providing an alternative to manually finding performant ensembles for actual ICS deployments with nearly no FPAs.

6.7 Conclusion

While a research community continuously inventing intrusion detection algorithms should be a blessing for the cybersecurity of ICSs, capitalizing on the benefits of these efforts by bringing the achieved results into practice is another story (I5). To minimize the options ICS operators have to select an IIDS from, we analyzed misuse-based IIDSs, finding that they are rather unsuited in this use case (cf. Section 6.1). This observation is backed by related work with similar experiments [KWP+22a]. In contrast, anomaly detection promises to detect even zero-day attacks for which no training data exists.

Despite ruling out one type of intrusion detection, the remaining pool of IIDSs is still large. On our way to determine the best IIDS for a deployment, we have outlined that relying on a single detector is insufficient given their individual weaknesses (cf. Section 6.2). Thus, we propose ensemble learning, as, e.g., used in computer vision, to combine a set of IIDSs and their strengths into one detector.

We identify unused potential to improve not only the *combined* detection performance beyond what individual approaches can achieve but also the value of the alerts for ICS operators (cf. Section 6.4). Incorporating temporal correlations into the ensemble reduces the number of false positives further (cf. Section 6.5). Lastly, to ease the process of finding training data for ensembles, we consider *transfer-learning*, i.e., training ensembles on attacks from a different ICS (cf. Section 6.6).

In conclusion, ensemble learning poses an exciting solution to strengthen ICSs' cybersecurity by tightly integrating existing and upcoming methods for intrusion detection into an effective solution. Thereby, the protection of an ICS is improved because multiple IIDSs monitor the ICS simultaneously. Moreover, attackers have to defeat not only a single system, reducing the chances of successful attacks. While we have only scratched the surface of ensemble learning and demonstrated its potential, this research branch promises to be an opportunity for future work.

7

Conclusion

Industrial Control Systems (ICSs) greatly benefit from the advancements achieved through rapid digitization, which enable the automation of repetitive tasks for many industrial domains. Since cybersecurity has long been treated with lower priority, this neglect has led to an emerging growth in harmful cyberattacks. Additionally, due to legacy reasons, retrofitting cybersecurity measures to existing ICS facilities can be challenging or even impossible if the leveraged industrial hardware does not provide the necessary functionality, e.g., up-to-date cryptographic operations. As one remedy, the research community considers the concept of intrusion detection as a viable security solution for ICSs to identify and alert malicious activities to ICS operators before attacks can cause harm.

In that regard, ICS's largely repetitive and predictable communication patterns or physical processes offer great opportunities for realizing specialized Industrial Intrusion Detection Systems (IIDSs). The potential for IIDSs is also reflected in an active research community, inventing and evaluating complex detection methodologies for all types of ICS domains and protocols. However, despite their success, research struggles to work on intrusion detection collectively. The research landscape is split into smaller and separated fields, tackling only single industrial applications rather than providing generic solutions for the entire ICS domain. Thus, IIDS research still faces several hurdles, decelerating its efficiency and complicating the deployment of IIDS solutions to secure actual ICSs effectively.

This dissertation has surveyed the current state of IIDS research employing a Systematic Mapping Study (SMS) and worked out five issues in the IIDS research landscape. Summarizing our contributions and how we made progress on these issues, Section 7.1 presents the results obtained through our research activities. Following up, Section 7.2 lays out which research activities this dissertation has addressed and then points to interesting topics to be addressed in the future. Lastly, Section 7.3 discusses limitations of this dissertation and Section 7.4 wraps up the insights gained regarding generalizable and transferable intrusion detection for ICSs.

7.1 Addressing the Research Issues I1–I5

From the results of our SMS, we have distilled five research issues (I1–I5) in Section 3.6.1 that guided our research activities from Chapter 4 to Chapter 6. Our goal was to make IIDSs more accessible to ICSs. The following paragraphs summarize how this dissertation has contributed to the issues.

IIDS research takes place in isolated silos due to its inherent protocol- and domain-dependence, which slows down the collective achievements (I1).

One primary issue of IIDS research is its inherent focus on niece parts of the ICS landscape. To break up this pattern, we motivated and realized the concept of protocol- and domain-independent IIDS research by means of our Industrial Protocol Abstraction Layer (IPAL) in Chapter 4. As we observed the repetitive use of similar feature sets for intrusion detection regardless of the underlying industrial communication protocol or industrial domain, we could aggregate the required information into a novel and standardized data format. Thereby, IIDS detection methodologies based on the IPAL format can technically operate regardless of the actual protocols leveraged in a specific ICS domain. Consequently, IIDSs do not have to be reinvented for each ICS domain separately and can instead be transferred easily. The concept of IPAL proved successful during our studies, enabling us to test existing and novel IIDSs in a wide variety of industrial scenarios. Thereby, the idea of IPAL can contribute to a better consolidation of IIDS research over time.

Evaluations could be improved since datasets are leveraged scarcely and prominent metrics risk being inexpressive, leading to a low level of comparability (I2).

Our tooling and research artifacts around IPAL provide a holistic platform to conduct diverse and in-depth evaluations, which we have demonstrated throughout the dissertation. First, due to IPAL's standardized data format, existing datasets can be effortlessly transcribed into IPAL, giving researchers access to many datasets simultaneously. For example, all our evaluations of SIMPLE, GeCo, and IIDS ensembles were performed with four diverging and prominent ICS datasets. Next, we consistently evaluated IIDSs visually and with a diverse set of point-based and time-aware metrics. Lastly, by publishing IPAL and the (re-)implementations of existing and new IIDSs as an open-source project for the research community [FC24], we boost the reproducibility and comparability of the research itself. To this end, we compared the detection performance of our new IIDSs (SIMPLE and GeCo) to at least four publications from related work. Overall, IPAL can be a great tool for conducting high-quality evaluations and IIDS research, as demonstrated multiple times throughout this dissertation.

Existing approaches seem overly complex without proper justification (I3).

During our SMS, we observed an unjustified focus of research on IIDSs that leverage inherently complex detection methodologies. Questioning the proportionality of complexity to tackle the attacks contained in high-quality datasets, we wanted to determine whether intrusion detection must be so complicated. First, we show that even minimalistic approaches (e.g., MinMax or Gradient) provide a great detection performance that is on par with complex related work. Set out to improve their

detection performance, we invented GeCo. Our new IIDS provides a compromise between high detection capability, transferability, usability, and comprehensibility for ICS operators with little compromises to our S.I.M.P.L.E. philosophy (cf. Section 5.2.5.2). Overall, we proved that intrusion detection does not have to be inherently complex as in related work to unveil the attacks that research evaluates within their current datasets.

Transferability of IIDSs does not take place on a large scale (I4).

Despite the great opportunities in a diverse ICS environment with diverging domains, we have criticized that IIDSs are seldom transferred to use-cases outside the ones they were initially designed or evaluated for. Since IPAL provides the foundation for researching the generalizability and transferability of IIDSs to various industrial applications regardless of protocols or data structures, we explored these possibilities on multiple dimensions. As the first step, we repeatedly accessed the feasibility of IIDSs from related work (cf. Section 4.5) and our IIDSs (cf. Section 5.1 and Section 5.2) on datasets from different industrial backgrounds. In general, the underlying detection methodologies were indeed applicable outside of the domains for which they were initially evaluated. Especially the IIDSs, which were invented by us and intended to be transferable by design, outperform related work across all datasets on average (cf. Section 5.2). As another dimension, Section 5.3 studied whether we can transfer existing concepts of intrusion detection to a Marine Radar System (MRS). While existing IIDSs could detect many attacks in such a novel setting, some scenario-specific gaps in their detection capabilities remained. These remaining gaps were addressed with novel approaches. Thus, ICS operators can make fast progress on protecting their facilities with existing IIDSs before resorting to the elaborate process of designing entirely new IIDSs.

Research does not consider which IIDSs should be deployed (I5).

The last research activity of this dissertation in Chapter 6 concerned the deployment of IIDSs. In the first part, we questioned the general practicality of misuse-based IIDSs because of their extensive requirement for recording training samples of attacks. Moreover, these methodologies have a high risk of overfitting, potentially diminishing detection performance in real deployments (cf. Section 6.1). As the second option, anomaly detection is not affected by the previous issues, yet most IIDSs face the risk of false positives. To limit these risks without compromising detection performance, we studied the concept of ensemble learning for anomaly detection with great success (cf. Section 6.3). Thereby, ICS operators do not have to rely on a single detection approach and can instead flexibly pool the advantages of several IIDSs to their best. As a consequence, ICSs should be, at best, protected through different detection methodologies in practice.

In summary, we showed that addressing and tackling all of these issues is possible in theory. Therefore, this task is now up to the discipline of the research community in the future whether these findings and methodologies will be adopted or whether research will continue like before, as assessed initially in our SMS.

7.2 Guide and Inspiration for Future Research Topics

In the introduction, we enumerated nine dimensions in which IIDS research takes place. Based on the taxonomy from Figure 1.2, we now state for which dimension this dissertation has made progress and where further research is still required.

ICS Scenario. This dissertation augmented the current set of ICS scenarios with the MRS scenario discussed in Section 5.3. From this assessment, we can derive a resource-effective methodology for future work to tackle new ICS scenarios. By precisely analyzing and defining the attacker model, one can first transfer existing IIDSs and assess their detection capabilities in a novel scenario. This allows for identifying critical gaps in detection capabilities that can then be addressed with scenario-specific solutions. This process promises to avoid work being done twice and enhance the connectivity to the established research branches.

Attacker Model. Section 5.3 derived a taxonomy for attacks against MRS. But, besides attacks against an ICS itself that IIDSs aim to detect, another category of attacks has not been given attention in this dissertation. Related work has proven the feasibility of new threats for IIDSs, namely attacks targeting the detection methodologies [CGR13], and adversarial machine learning [ETG+20]. For their mitigation, one idea for future work would be to examine whether our ensemble learning concepts can help in that regard since attackers have to defeat multiple IIDSs simultaneously.

Data Type. Section 4.1.3 analyzed which data types IIDSs leverage for their detection mechanisms, finding that physical process data and network traffic were dominantly used. Our contribution, IPAL, is a new data format capable of aggregating existing features into a unified abstract format. IPAL enables protocol- and domain-independence from the ICS scenario. Another factor relevant for practical deployments is the amount of data points. While current datasets contain up to 127 process values (WADI), IIDSs likely have to scale to even higher numbers in practice [WBWS25], which could become another challenge.

Benchmarking Environments. Our SMS revealed that IIDS research has access to plenty of datasets for established domains. Still, for the MRS domain, we developed a new simulation environment (RadarSec-Lab), which we used to record the RadarPWN dataset [WSB+22]. Additionally, IPAL enhances the accessibility of existing datasets and has proved helpful in publishing novel datasets [ZHW+22, WBWS25]. Still, further meta-analyses on the quality of datasets can help provide additional guidance for high-qualitative evaluations [TETC20, LSAA23, SK25].

Evaluation Metrics. In this dissertation, only minor progress was achieved concerning evaluation metrics. While the IPAL IDS framework makes many metrics, especially newer time-aware ones, accessible, we did not invent any new metrics. Still, those metrics available now can already provide a holistic view of an IIDS's detection performance if leveraged appropriately and assessed in combination.

Detection Techniques. Across this dissertation, great efforts were invested in surveying detection methodologies (cf. Section 3.5.1), re-implementing existing approaches (cf. Section 4.4), inventing new IIDSs (cf. Chapter 5), and comparing the different

directions (cf. Section 6.1). Overall, the research landscape is well positioned regarding detection methodologies, and future work has to precisely elaborate on the research gap and why new methods are needed.

Alerts. Related work often resorts to solving the comprehensibility of IIDS decisions in retrospect, e.g., through explainable machine-learning [LSAA23], rather than incorporating comprehensibility during the IIDS design. In contrast, we put a special focus on the comprehensibility with GeCo, which we even validated in a user study in Section 5.2.4.2. Yet, how alerts are measured in research may not be optimal for deployments. For example, current metrics favor alerts with optimal attack coverage. But, in practice, operators have to investigate even short alerts. Making progress on these different conceptions of alerts in research and in practice might be necessary to ease the transition of IIDSs into deployments.

Deployment. While assessing related work in Section 3.5.2, we observed a surprising complexity in research. Consequently, we took special care while designing our IIDSs to tackle these shortcomings (cf. also Alerts). Discussing which IIDS to deploy, we propose leveraging ensemble learning to utilize the advantages of IIDSs in combination and effectively reduce false positives. While we demonstrated the feasibility of ensemble learning, future work can investigate more elaborated ensemble methods.

Reaction. Lastly, this dissertation has made no substantial progress on how to react to alerts. Alert management is crucial for intrusion detection, but also on vessels [SH24], to contain cyberattacks are exciting in ICS to not perform more harm than the attacks intend to do.

Overall, this dissertation has contributed to most dimensions concerning IIDS research. While some areas, e.g., detection methodologies, receive vast attention, other areas, e.g., reaction, that are especially relevant for deployments receive less attention during the design or evaluation of IIDSs as of now. Those are exciting areas for future work to narrow the gap between research and production systems.

7.3 Limitations

This section discusses limitations of this dissertation concerning the use of dataset, scalability issues, ICS process complexity, and vantage points.

To evaluate the IIDSs, we have leveraged multiple datasets (cf. Section 2.3.1) or invented the RadarSec-Lab simulation environment (cf. Section 5.3.3). While datasets enable the rapid development of new detection methodologies, they lag behind in realism. At least, most leveraged datasets [MTT15, GAJM16, APM17, SLYK20] were recorded on physical testbeds providing more realism than purely simulated environments. Still, it is known that deploying IIDSs in real-world environments comprises further challenges affecting detection performance or the effectiveness of an IIDS [SHK+24]. Thus, deploying IPAL or the other IIDSs in real use-cases is an interesting opportunity to gain insight in challenges that do not show in the lab.

Many of the problems in this dissertation were motivated and explained along the simplified water tank model from Section 2.1.1. Yet, in practice, more complex ICSs

are realistic regarding the size of the ICS and the complexity of the physical process. We discuss both aspects in the following.

WADI, the largest dataset concerning process values, contains 127 sensors and actuators. But realistic ICSs can encompass orders of magnitudes more data points. This can lead to scalability issues for the detection models during training and detection (cf. also Section 5.2.5.1). Possible solutions could be an optimization of the detection algorithms or splitting the data into multiple detection models, as already done by TABOR [LAVM18] and the Seq2SeqNN IIDS [KYK20]. Yet smaller models may also affect the overall detection performance.

While we considered datasets from diverse industrial domains [SLYK20] or modeling up to six process stages [GAJM16], ICS can become magnitudes more complex in reality [AQP+22]. Not only regarding the physical process but also due to human interactions, faults, or modifications to the process over time. This raises the question whether (our) IIDSs are capable of modeling such sophisticated processes as being already criticized by other researchers [Eta17, AMM20].

Another important aspect for detecting cyberattacks is the vantage point inside the ICS, i.e., where to deploy IPAL. While process anomalies can be visible up to the Supervisory Control and Data Acquisition (SCADA) system (cf. Section 2.1.3), network attacks, such as Denial of Service (DoS), occur at the network level. Therefore, especially for large ICS, e.g., power grids [WBWS25], the vantage point can become crucial, likely even requiring multiple installations.

7.4 Generalizable and Transferable Intrusion Detection

The overarching problem statement from Section 1.4 presumed that ICSs' diversity and research's complexity prevent IIDSs from being easily accessible for ICSs, which we confirmed and quantified in the literature. Following up, this dissertation presented three solutions to improve this situation. First, IPAL establishes a technical foundation for protocol- and domain-independent intrusion detection, paving the way for IIDSs' generalizability to arbitrary industrial protocols. Furthermore, we showed that the underlying detection methodologies are transferable to various industrial processes. Second, we showed that S.I.M.P.L.E. detection methodologies increase accessibility through increased comprehensibility while retaining high intrusion detection capabilities. Third, if those transferred and simplified IIDSs do not detect all attack vectors, ensemble learning can efficiently fuse them with domain-specific IIDSs to achieve flexible and holistic protection. Altogether, high-qualitative, generalizable, transferable, and comprehensible IIDSs become accessible to all kinds of ICS domains. While the technical foundations and detection methods are, therefore, sufficiently mature, it is now up to the ICS facilities to take the next step and apply these results in practice for better cybersecurity in ICSs.

Bibliography

[AA20] Wissam Aoudi and Magnus Almgren. A scalable specification-agnostic multi-sensor anomaly detection system for IIoT environments. *International Journal of Critical Infrastructure Protection*, 30:100377, 2020.

[AAG19] Malik Shahzad Kaleem Awan and Mohammed A. Al Ghamdi. Understanding the Vulnerabilities in Digital Components of An Integrated Bridge System (IBS). *Journal of Marine Science and Engineering*, 7(10), 2019.

[AAM22] Bushra A. Alahmadi, Louise Axon, and Ivan Martinovic. 99% False Positives: A Qualitative Study of SOC Analysts' Perspectives on Security Alarms. In *Proceedings of the 31st USENIX Security Symposium (USENIX Sec)*, pages 2783–2800, 2022.

[ABS20] Otis Alexander, Misha Belisle, and Jacob Steele. MITRE ATT&CK® for Industrial Control Systems: Design and Philosophy. Technical Report MP01055863, The MITRE Corporation, 2020.

[ACZ20] Tejasvi Alladi, Vinay Chamola, and Sherali Zeadally. Industrial Control Systems: Cyberattack trends and countermeasures. *Computer Communications*, 155:1–8, 2020.

[ADS20] Maged Abdelaty, Roberto Doriguzzi-Corin, and Domenico Siracusa. AADS: A Noise-Robust Anomaly Detection Framework for Industrial Control Systems. In *Proceedings of the 21th International Conference on Information and Communications Security (ICICS)*, 2020.

[ADS21] Maged Fathy Abdelaty, Roberto Doriguzzi Corin, and Domenico Siracusa. DAICS: A Deep Learning Solution for Anomaly Detection in Industrial Control Systems. *IEEE Transactions on Emerging Topics in Computing*, 10(2):1117–1129, 2021.

[AG22] Ahmed Amro and Vasileios Gkioulos. From Click to Sink: Utilizing AIS for Command and Control in Maritime Cyber Attacks. In *Proceedings of the 27th European Symposium on Research in Computer Security (ESORICS)*, pages 535–553, 2022.

[AIA18] Wissam Aoudi, Mikel Iturbe, and Magnus Almgren. Truth Will Out: Departure-Based Process-Level Detection of Stealthy Attacks on Control Systems. In *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security (CCS)*, page 817–831, 2018.

[AJK+21] Abdulrahman Al-abassi, Amir Namavar Jahromi, Hadis Karimipour, Ali Dehghantanha, Pierluigi Siano, and Henry Leung. A Self-Tuning Cyber-Attacks' Location Identification Approach for Critical Infrastructures. *IEEE Transactions on Industrial Informatics*, 18(7):5018–5027, 2022.

[AKDP20] Abdulrahman Al-Abassi, Hadis Karimipour, Ali Dehghantanha, and Reza M Parizi. An Ensemble Deep Learning-Based Cyber-Attack Detection in Industrial Control System. *IEEE Access*, 8:83965–83973, 2020.

[AKPI18] Ekta Aggarwal, Mehdi Karimibiuki, Karthik Pattabiraman, and André Ivanov. CORGIDS: A Correlation-Based Generic Intrusion Detection System. In *Proceedings of the Workshop on Cyber-Physical Systems Security and PrivaCy (CPS-SPC)*, page 24–35, 2018.

[ALGS19] Simon D. Duque Anton, Anna Pia Lohfink, Christoph Garth, and Hans Dieter Schotten. Security in Process: Detecting Attacks in Industrial Process Data. In *Proceedings of the Third Central European Cybersecurity Conference (CECC)*, pages 1–6, 2019.

[ALS23] Giovanni Apruzzese, Pavel Laskov, and Johannes Schneider. SoK: Pragmatic Assessment of Machine Learning for Network Intrusion Detection. In *Proceedings of the 2023 IEEE 8th European Symposium on Security and Privacy (EuroS&P)*, pages 592–614, 2023.

[Alv08] Victor M. Alvarez. YARA - The pattern matching swiss knife for malware researchers. `https://web.archive.org/web/20250126130055/https://virustotal.github.io/yara/`, 2008. [Online, accessed 2025-01-26].

[AM16a] Sridhar Adepu and Aditya Mathur. Distributed Detection of Single-Stage Multipoint Cyber Attacks in a Water Treatment Plant. In *Proceedings of the 11th ACM on Asia Conference on Computer and Communications Security (Asia CCS)*, page 449–460, 2016.

[AM16b] Sridhar Adepu and Aditya Mathur. Using Process Invariants to Detect Cyber Attacks on a Water Treatment System. In *Proceedings of the 31st Conference on ICT Systems Security and Privacy Protection*, pages 91–104, 2016.

[AMM20] Chuadhry Mujeeb Ahmed, Gauthama Raman MR, and Aditya P Mathur. Challenges in machine learning based approaches for real-time anomaly detection in industrial control systems. In *Proceedings of the 6th ACM on Cyber-Physical System Security Workshop (CPSS)*, pages 23–29, 2020.

[AMO20] Chuadhry Mujeeb Ahmed, Aditya P. Mathur, and Martín Ochoa. NoiSense Print: Detecting Data Integrity Attacks on Sensor Measurements Using Hardware-Based Fingerprints. *ACM Transactions on Privacy and Security*, 24(1):1–35, 2020.

[AMR17] Chuadhry Mujeeb Ahmed, Carlos Murguia, and Justin Ruths. Model-Based Attack Detection Scheme for Smart Water Distribution Networks. In *Proceedings of the ACM on Asia Conference on Computer and Communications Security (ASIA CCS)*, page 101–113, 2017.

[ANR17] Cesare Alippi, Stavros Ntalampiras, and Manuel Roveri. Model-Free Fault Detection and Isolation in Large-Scale Cyber-Physical Systems. *IEEE Transactions on Emerging Topics in Computational Intelligence*, 1(1):61–71, 2017.

[AOGK22] Ahmed Amro, Aybars Oruc, Vasileios Gkioulos, and Sokratis Katsikas. Navigation Data Anomaly Analysis and Detection. *Information*, 13(3), 2022.

[AOZ+18] Chuadhry Mujeeb Ahmed, Martin Ochoa, Jianying Zhou, Aditya P. Mathur, Rizwan Qadeer, Carlos Murguia, and Justin Ruths. NoisePrint: Attack Detection Using Sensor and Process Noise Fingerprint in Cyber Physical Systems. In *Proceedings of the 2018 on Asia Conference on Computer and Communications Security (ASIA CCS)*, page 483–497, 2018.

[Apa20] The Apache Software Foundation. PLC4X. https://plc4x.apache.org/, 2020. [Online, accessed 2020-08-14].

[APM13] Uttam Adhikari, Shengyi Pan, and Tommy Morris. Industrial Control System (ICS) Cyber Attack Datasets. `https://web.archive.org/web/20241130133235/https://sites.google.com/a/uah.edu/tommy-morris-uah/ics-data-sets`, 2013. [Online, accessed 2024-11-27].

[APM17] Chuadhry Mujeeb Ahmed, Venkata Reddy Palleti, and Aditya P. Mathur. WADI: A Water Distribution Testbed for Research in the Design of Secure Cyber Physical Systems. In *Proceedings of the 3rd International Workshop on Cyber-Physical Systems for Smart Water Networks (CySWATER)*, page 25–28, 2017.

[APQ+20] Chuadhry Mujeeb Ahmed, Jay Prakash, Rizwan Qadeer, Anand Agrawal, and Jianying Zhou. Process Skew: Fingerprinting the Process for Anomaly Detection in Industrial Control Systems. In *Proceedings of the 13th ACM Conference on Security and Privacy in Wireless and Mobile Networks (WiSec)*, page 219–230, 2020.

[AQP+22] Daniel Arp, Erwin Quiring, Feargus Pendlebury, Alexander Warnecke, Fabio Pierazzi, Christian Wressnegger, Lorenzo Cavallaro, and Konrad Rieck. Dos and Don'ts of Machine Learning in Computer Security. In *Proceedings of the 31st USENIX Security Symposium (USENIX Sec)*, pages 3971–3988, 2022.

[AS20] Simon D. Duque Anton and Hans Dieter Schotten. Intrusion Detection in Binary Process Data: Introducing the Hamming-distance to Matrix Profiles. In *Proceedings of the IEEE 21st International Symposium on "A World of Wireless, Mobile and Multimedia Networks" (WoWMoM)*, pages 347–353, 2020.

[AS21] Sergei K. Alabugin and Alexander N. Sokolov. Applying of Recurrent Neural Networks for Industrial Processes Anomaly Detection. In *Proceedings of the Ural Symposium on Biomedical Engineering, Radioelectronics and Information Technology (USBEREIT)*, pages 0467–0470, 2021.

[ASS19] Simon D. Anton, Sapna Sinha, and Hans Dieter Schotten. Anomaly-based Intrusion Detection in Industrial Data with SVM and Random Forests. In *International Conference on Software, Telecommunications and Computer Networks (SoftCOM)*, pages 1–6, 2019.

[AWBJ21] Eirini Anthi, Lowri Williams, Pete Burnap, and Kevin Jones. A three-tiered intrusion detection system for industrial control systems. *Journal of Cybersecurity*, 7(1):tyab006, 2021.

[AWT+20] Frederik Armknecht, Paul Walther, Gene Tsudik, Martin Beck, and Thorsten Strufe. ProMACs: progressive and resynchronizing MACs for continuous efficient authentication of message streams. In *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security (CCS)*, pages 211–223, 2020.

[AYT+14] Abdulmohsen Almalawi, Xinghuo Yu, Zahir Tari, Adil Fahad, and Ibrahim Khalil. An unsupervised anomaly-based detection approach for integrity attacks on SCADA systems. *Computers & Security*, 46:94 – 110, 2014.

[AZ15] Cristina Alcaraz and Sherali Zeadally. Critical infrastructure protection: Requirements and challenges for the 21st century. *International Journal of Critical Infrastructure Protection*, 8:53–66, 2015.

[AZM18] Chuadhry Mujeeb Ahmed, Jianying Zhou, and Aditya P. Mathur. Noise Matters: Using Sensor and Process Noise Fingerprint to Detect Stealthy Cyber Attacks and Authenticate Sensors in CPS. In *Proceedings of the 34th Annual Computer Security Applications Conference (ACSAC)*, page 566–581, 2018.

[Bak16] Monya Baker. 1,500 scientists lift the lid on reproducibility. *Nature*, 533(7604):452–454, 2016.

[BBF+19] Vaibhav Bajpai, Anna Brunstrom, Anja Feldmann, Wolfgang Kellerer, Aiko Pras, Henning Schulzrinne, Georgios Smaragdakis, Matthias Wählisch, and Klaus Wehrle. The Dagstuhl Beginners Guide to Reproducibility for Experimental Networking Research. *SIGCOMM Computer Communication Review*, 49(1):24–30, 2019.

[BDW09] Alan Bole, Bill Dineley, and Alan Wall. *Radar and ARPA Manual – Radar and Target Tracking for Professional Mariners, Yachtsmen and Users of Marine Radar*. Butterworth-Heinemann Ltd, 2nd edition, 2009.

[Ben96] S. Bennett. A brief history of automatic control. *IEEE Control Systems Magazine*, 16(3):17–25, 1996.

[BH17] Jahshan Bhatti and Todd E. Humphreys. Hostile Control of Ships Via False GPS Signals: Demonstration and Detection. *Journal of the Institute of Navigation*, 64(1):51–66, 2017.

[BHIM22] Asaad Balla, Mohamed Hadi Habaebi, MD Rafiqul Islam, and Sinil Mubarak. Applications of deep learning algorithms for Supervisory Control and Data Acquisition intrusion detection system. *Cleaner Engineering and Technology*, 9:100532, 2022.

[BJF21] Millot Baptiste, Francq Julien, and Sicard Franck. Systematic and Efficient Anomaly Detection Framework using Machine Learning on Public ICS Datasets. In *Proceedings of the IEEE International Conference on Cyber Security and Resilience (CSR)*, pages 292–297, 2021.

[BJL+21] Clet Boudehenn, Olivier Jacq, Maxence Lannuzel, Jean-Christophe Cexus, and Abdel Boudraa. Navigation anomaly detection: An added value for Maritime Cyber Situational Awareness. In *Proceedings of the International Conference on Cyber Situational Awareness, Data Analytics And Assessment (Cyber SA)*, pages 1–4, 2021.

[BKC+14] Thomas Bangemann, Stamatis Karnouskos, Roberto Camp, Oscar Carlsson, Matthias Riedl, Stuart McLeod, Robert Harrison, Armando W. Colombo, and Petr Stluka. *State of the Art in Industrial Automation*. Springer, 2014.

[BKS+23] Jan Bauer, Joris Kutzner, Philipp Sedlmeier, Anisa Rizvanolli, and Elmar Padilla. *Phish & Ships* and Other Delicacies from the Cuisine of Maritime Cyber Attacks. In *Proceedings of the European Workshop on Maritime Systems Resilience and Security (MARESEC)*, 2023.

[BPW14] Marco Balduzzi, Alessandro Pasta, and Kyle Wilhoit. A Security Evaluation of AIS Automated Identification System. In *Proceedings of the 30th Annual Computer Security Applications Conference (ACSAC)*, page 436–445, 2014.

[Bri22] bridgecommand. bc. `https://web.archive.org/web/20240620064720/https://github.com/bridgecommand/bc`, 2022. [Online, accessed 2024-06-20].

[BRJ15] Andreas Bathelt, N. Lawrence Ricker, and Mohieddine Jelali. Revision of the Tennessee Eastman Process Model. *IFAC-PapersOnLine*, 48(8):309–314, 2015.

[BSAM21] Madhumitha Balaji, Siddhant Shrivastava, Sridhar Adepu, and Aditya Mathur. Super Detector: An Ensemble Approach for Anomaly Detection in Industrial Control Systems. In *Proceedings of the 16th International Conference on Critical Information Infrastructures Security (CRITIS)*, pages 24–43, 2021.

[BSH25] BSH. Our Ships. `https://web.archive.org/web/20250108193637/https://www.bsh.de/EN/The_BSH/Our_ships/Our_ships_node.html`, 2025. [Online, accessed 2025-01-08].

[BSL+23] Lennart Bader, Martin Serror, Olav Lamberts, Ömer Sen, Dennis van der Velde, Immanuel Hacker, Julian Filter, Elmar Padilla, and Martin Henze. Comprehensively Analyzing the Impact of Cyberattacks on Power Grids. In *Proceedings of the 2023 IEEE 8th European Symposium on Security and Privacy (EuroS&P)*, pages 1065–1081, 2023.

[BSP12] Rafael Ramos Regis Barbosa, Ramin Sadre, and Aiko Pras. Towards Periodicity Based Anomaly Detection in SCADA Networks. In *Proceedings of IEEE 17th International Conference on Emerging Technologies Factory Automation (ETFA)*, pages 1–4, 2012.

[BTZ+19] Partha P Biswas, Heng Chuan Tan, Qingbo Zhu, Yuan Li, Daisuke Mashima, and Binbin Chen. A synthesized dataset for cybersecurity study of IEC 61850 based substation. In *Proceedings of the IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids (SmartGridComm)*, pages 1–7, 2019.

[BUH+22] Mohamed Amine Ben Farah, Elochukwu Ukwandu, Hanan Hindy, David Brosset, Miroslav Bures, Ivan Andonovic, and Xavier Bellekens. Cyber Security in the Maritime Industry: A Systematic Survey of Recent Advances and Future Trends. *Information*, 13(1), 2022.

[CAL+11] Alvaro A. Cárdenas, Saurabh Amin, Zong-Syun Lin, Yu-Lun Huang, Chi-Yen Huang, and Shankar Sastry. Attacks Against Process Control Systems: Risk Assessment, Detection, and Response. In *Proceedings of the 6th ACM Symposium on Information, Computer and Communications Security (ASIACCS)*, 2011.

[CBB15] Eduardo Esteban Casanovas, Tomas Exequiel Buchaillot, and Facundo Baigorria. Vulnerability of Radar Protocol and Proposed Mitigation. In *Proceedings of the 2015 ITU Kaleidoscope: Trust in the Information Society (K-2015)*, pages 1–6, 2015.

[CCG+11] Andrea Carcano, Alessio Coletta, Michele Guglielmi, Marcelo Masera, Igor Nai Fovino, and Alberto Trombetta. A Multidimensional Critical State Analysis for Detecting Intrusions in SCADA Systems. *IEEE Transactions on Industrial Informatics*, 7(2):179–186, 2011.

[CDF+07] Steven Cheung, Bruno Dutertre, Martin Fong, Ulf Lindqvist, Keith Skinner, and Alfonso Valdes. Using model-based intrusion detection for SCADA networks. In *Proceedings of the SCADA security scientific symposium*, pages 1–12, 2007.

[CDT21] Mauro Conti, Denis Donadel, and Federico Turrin. A Survey on Industrial Control System Testbeds and Datasets for Security Research. *IEEE Communications Surveys & Tutorials*, 23(4):2248–2294, 2021.

[CDV13] Manuel Cheminod, Luca Durante, and Adriano Valenzano. Review of Security Issues in Industrial Networks. *IEEE Transactions on Industrial Informatics*, 9(1):277–293, 2013.

[CFMT10] Andrea Carcano, Igor Nai Fovino, Marcelo Masera, and Alberto Trombetta. State-Based Network Intrusion Detection Systems for SCADA Protocols: A Proof of Concept. In *Critical Information Infrastructures Security*, pages 138–150, 2010.

[CGES19] Shai Cohen, Tomer Gluck, Yuval Elovici, and Asaf Shabtai. Security Analysis of Radar Systems. In *Proceedings of the Workshop on Cyber-Physical Systems Security & Privacy (CPS-SPC)*, page 3–14, 2019.

[CGR13] Igino Corona, Giorgio Giacinto, and Fabio Roli. Adversarial attacks against intrusion detection systems: Taxonomy, solutions and open issues. *Information Sciences*, 239:201–225, 2013.

[Che84] Chi-Tsong Chen. *Linear system theory and design*. Saunders college publishing, 1984.

[CHK24] Peter Christen, David J. Hand, and Nishadi Kirielle. A review of the F-measure: Its History, Properties, Criticism, and Alternatives. *ACM Computing Surveys*, 56(3), 2024.

[Chr17] Justyna Chromik. manipulatePcaps. `https://web.archive.org/web/20221208134552/http://github.com/jjchromik/manipulateTraces/`, 2017. [Online, accessed 2022-12-08].

[Chr19] Justyna Chromik. InTraVis. `https://web.archive.org/web/20221207221252/https://github.com/jjchromik/intravis`, 2019. [Online, accessed 2022-12-07].

[Cla04] B. Claise. Cisco Systems NetFlow Services Export Version 9. RFC 3954, RFC Editor, 2004.

[CLA+18] Hongjun Choi, Wen-Chuan Lee, Yousra Aafer, Fan Fei, Zhan Tu, Xiangyu Zhang, Dongyan Xu, and Xinyan Deng. Detecting Attacks Against Robotic Vehicles: A Control Invariant Approach. In *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security (CCS)*, page 801–816, 2018.

[CLL19] Ankang Chu, Yingxu Lai, and Jing Liu. Industrial Control Intrusion Detection Approach Based on Multiclassification GoogLeNet-LSTM Model. *Security and Communication Networks*, 2019, 2019.

[CLRM21] Danyang Cao, Di Liu, Xu Ren, and Nan Ma. Self-Adaption AAE-GAN for Aluminum Electrolytic Cell Anomaly Detection. *IEEE Access*, 9:100991–101002, 2021.

[CLS+22] Shai Cohen, Efrat Levy, Avi Shaked, Tair Cohen, Yuval Elovici, and Asaf Shabtai. RadArnomaly: Protecting Radar Systems from Data Manipulation Attacks. *Sensors*, 22(11), 2022.

[CPR+20] Maurantonio Caprolu, Roberto Di Pietro, Simone Raponi, Savio Sciancalepore, and Pietro Tedeschi. Vessels Cybersecurity: Issues, Challenges, and the Road Ahead. *IEEE Communications Magazine*, 58(6):90–96, 2020.

[CPS18] Yuqi Chen, Christopher M. Poskitt, and Jun Sun. Learning from Mutants: Using Code Mutation to Learn and Monitor Invariants of a Cyber-Physical System. In *Proceedings of the IEEE Symposium on Security and Privacy (SP)*, pages 648–660, 2018.

[CZ19] John Henry Castellanos and Jianying Zhou. A Modular Hybrid Learning Approach for Black-Box Security Testing of CPS. In *Proceedings of the Applied Cryptography and Network Security (ACNS)*, pages 196–216, 2019.

[CZK15] Marco Caselli, Emmanuele Zambon, and Frank Kargl. Sequence-aware Intrusion Detection in Industrial Control Systems. In *Proceedings of the 1st ACM Workshop on Cyber-Physical System Security (CPSS)*, page 13–24, 2015.

[CZLT18] Xiayang Chen, Lei Zhang, Yi Liu, and Chaojing Tang. Ensemble Learning Methods for Power System Cyber-Attack Detection. In *Proceedings of the 3rd International Conference on Cloud Computing and Big Data Analysis (ICCCBDA)*, pages 613–616, 2018.

[CZPK15] Marco Caselli, Emmanuele Zambon, Jonathan Petit, and Frank Kargl. Modeling Message Sequences for Intrusion Detection in Industrial Control Systems. In *Critical Infrastructure Protection IX*, pages 49–71, 2015.

[DAZ20] Tanmoy Kanti Das, Sridhar Adepu, and Jianying Zhou. Anomaly detection in Industrial Control Systems using Logical Analysis of Data. *Computers & Security*, 96:101935, 2020.

[DBAG22] Dimitrios Dagdilelis, Mogens Blanke, Rasmus H Andersen, and Roberto Galeazzi. Cyber-resilience for marine navigation by information fusion and change detection. *Ocean Engineering*, 266:112605, 2022.

[DBS11] Adrian Dabrowski, Sebastian Busch, and Roland Stelzer. A Digital Interface for Imagery and Control of a Navico/Lowrance Broadband Radar. In *Proceedings of Robotic Sailing*, pages 169–181, 2011.

[DH21] Ailin Deng and Bryan Hooi. Graph Neural Network-Based Anomaly Detection in Multivariate Time Series. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, pages 4027–4035, 2021.

[DHX+18] Derui Ding, Qing-Long Han, Yang Xiang, Xiaohua Ge, and Xian-Ming Zhang. A survey on security control and attack detection for industrial cyber-physical systems. *Neurocomputing*, 275:1674–1683, 2018.

[DLP+22] Markus Dahlmanns, Johannes Lohmöller, Jan Pennekamp, Jörn Bodenhausen, Klaus Wehrle, and Martin Henze. Missed Opportunities: Measuring the Untapped TLS Support in the Industrial Internet of Things. In *Proceedings of the 17th ACM ASIA Conference on Computer and Communications Security (ASIA CCS)*, pages 252–266, 2022.

[DS15] L. Dhanabal and S. P. Shantharajah. A Study on NSL-KDD Dataset for Intrusion Detection System Based on Classification Algorithms. *International Journal of Advanced Research in Computer and Communication Engineering*, 4(6):446–452, 2015.

[DSW76] Joseph J DiStefano, Allen R Stubberud, and Ivan J Williams. *Schaum's outline of theory and problems of feedback and control systems*. McGraw-Hill, 1976.

[DV93] James J Downs and Ernest F Vogel. A plant-wide industrial process control problem. *Computers & chemical engineering*, 17(3):245–255, 1993.

[dZS+22] Théobald de Riberolles, Yunkai Zou, Guthemberg Silvestre, Emmanuel Lochin, and Jiefu Song. Anomaly Detection for ICS Based on Deep Learning: A Use Case for Aeronautical Radar Data. *Annals of Telecommunications*, 77:749–761, 2022.

[Els12] Marc Elsberg. *Blackout: Tomorrow will be too late*. Black Swan, 2012.

[EMK20] Mariam Elnour, Nader Meskin, and Khlaed M. Khan. Hybrid Attack Detection Framework for Industrial Control Systems using 1D-Convolutional Neural Network and Isolation Forest. In *Proceedings of the IEEE Conference on Control Technology and Applications (CCTA)*, pages 877–884, 2020.

[EMKJ20] Mariam Elnour, Nader Meskin, Khaled Khan, and Raj Jain. A Dual-Isolation-Forests-Based Attack Detection Framework for Industrial Control Systems. *IEEE Access*, 8:36639–36651, 2020.

[ET22] Alessandro Erba and Nils Ole Tippenhauer. Assessing Model-free Anomaly Detection in Industrial Control Systems Against Generic Concealment Attacks. In *Proceedings of the 38th Annual Computer Security Applications Conference (ACSAC)*, page 412–426, 2022.

[Eta17] Sandro Etalle. From Intrusion Detection to Software Design. In *Proceedings of the 22nd European Symposium on Research in Computer Security (ESORICS)*, pages 1–10, 2017.

[ETG+20] Alessandro Erba, Riccardo Taormina, Stefano Galelli, Marcello Pogliani, Michele Carminati, Stefano Zanero, and Nils Ole Tippenhauer. Constrained Concealment Attacks against Reconstruction-based Anomaly Detectors in Industrial Control Systems. In *Proceedings of the 36th Annual Computer Security Applications Conference (ACSAC)*, page 480–495, 2020.

[FB20] David Formby and Raheem Beyah. Temporal Execution Behavior for Host Anomaly Detection in Programmable Logic Controllers. *IEEE Transactions on Information Forensics and Security*, 15:1455–1469, 2020.

[FC24] Fraunhofer FKIE-CAD. IPAL (Industrial Protocol Abstraction Layer). `https://web.archive.org/web/20241121123205/https://github.com/fkie-cad/ipal`, 2024. [Online, accessed 2024-11-12].

[FCCR18] Benedikt Ferling, Justyna Chromik, Marco Caselli, and Anne Remke. Intrusion Detection for sequence-based attacks with reduced traffic models. In *Measurement, Modelling and Evaluation of Computing Systems (MMB)*, pages 53–67, 2018.

[FCD+10] Igor Nai Fovino, Andrea Carcano, Thibault De Lacheze Murel, Alberto Trombetta, and Marcelo Masera. Modbus/DNP3 State-based Intrusion Detection System. In *Proceedings of the 24th IEEE International Conference on Advanced Information Networking and Applications (AINA)*, pages 729–736, 2010.

[Fen19] Cheng Feng. NDSS19_InvariantRuleAD. `https://web.archive.org/web/20240721231710/https://github.com/cfeng783/NDSS19_InvariantRuleAD`, 2019. [Online, accessed 2024-07-21].

[FFGS21] Luca Faramondi, Francesco Flammini, Simone Guarino, and Roberto Setola. A Hardware-in-the-Loop Water Distribution Testbed Dataset for Cyber-Physical Security Testing. *IEEE Access*, 9:122385–122396, 2021.

[FLC17] Cheng Feng, Tingting Li, and Deeph Chana. Multi-level Anomaly Detection in Industrial Control Systems via Package Signatures and LSTM Networks. In *Proceedings of the 47th International Conference on Dependable Systems and Networks (DSN)*, pages 261–272, 2017.

[FMC11] Nicolas Falliere, Liam O Murchu, and Eric Chien. W32.Stuxnet Dossier. `https://web.archive.org/web/20240318110846/https://pax0r.com/hh/stuxnet/Symantec-Stuxnet-Update-Feb-2011.pdf`, 2011. [Online, accessed 2024-05-14].

[FPMC19] Cheng Feng, Venkata Reddy Palleti, Aditya Mathur, and Deeph Chana. A Systematic Framework to Generate Invariants for Anomaly Detection in Industrial Control Systems. In *Network and Distributed System Security (NDSS)*, 2019.

[Fre21] Vitor Freitas. Parsifal. `https://web.archive.org/web/20241222201552/https://parsif.al/`, 2021. [Online, accessed 2024-12-22].

[FSL+22] Clement Fung, Shreya Srinarasi, Keane Lucas, Hay Bryan Phee, and Lujo Bauer. Perspectives from a Comprehensive Evaluation of Reconstruction-based Anomaly Detection in Industrial Control Systems. In *Proceedings of the 27th European Symposium on Research in Computer Security (ESORICS)*, pages 493–513, 2022.

[FZB24] Clement Fung, Eric Zeng, and Lujo Bauer. Attributions for ML-based ICS Anomaly Detection: From Theory to Practice. In *Proceedings of the Network and Distributed System Security (NDSS)*, pages 1–19, 2024.

[GAB+18] Hamid Reza Ghaeini, Daniele Antonioli, Ferdinand Brasser, Ahmad-Reza Sadeghi, and Nils Ole Tippenhauer. State-Aware Anomaly Detection for Industrial Control Systems. In *Proceedings of the 33rd Annual ACM Symposium on Applied Computing (SAC)*, pages 1620–1628, 2018.

[GAJM16] Jonathan Goh, Sridhar Adepu, Khurum Nazir Junejo, and Aditya Mathur. A Dataset to Support Research in the Design of Secure Water Treatment Systems. In *Proceedings of the International Conference on Critical Information Infrastructures Security (CRITIS)*, pages 88–99, 2016.

[GBP14] Annarita Giani, Russell Bent, and Feng Pan. Phasor measurement unit selection for unobservable electric power data integrity attack detection. *International Journal of Critical Infrastructure Protection*, 7(3):155–164, 2014.

[GCP21] Isaías González, Antonio José Calderón, and José María Portalo. Innovative multilayered architecture for heterogeneous automation and monitoring systems: Application case of a photovoltaic smart microgrid. *Sustainability*, 13(4):2234, 2021.

[GGB+21] Jun Gao, Luyun Gan, Fabiola Buschendorf, Liao Zhang, Hua Liu, Peixue Li, Xiaodai Dong, and Tao Lu. Omni SCADA Intrusion Detection Using Deep Learning Algorithms. *IEEE Internet of Things Journal*, 8(2):951–961, 2021.

[GH13] Brendan Galloway and Gerhard P. Hancke. Introduction to Industrial Control Networks. *IEEE Communications Surveys & Tutorials*, 15(2), 2013.

[GK20] Athanasios Goudosis and Sokratis Katsikas. Secure AIS with Identity-Based Authentication and Encryption. *TransNav*, 14(2), 2020.

[GM14] Wei Gao and Thomas H Morris. On cyber attacks and signature based intrusion detection for modbus based industrial control systems. *Journal of Digital Forensics, Security and Law*, 9(1), 2014.

[Gre21] Dor Green. pyshark. `https://web.archive.org/web/20241113211037/https://github.com/KimiNewt/pyshark`, 2021. [Online, accessed 2024-11-13].

[Gri19] Francesco Grillo. *Hero of Alexandria's Automata: a critical edition and translation, including a commentary on Book One*. PhD thesis, University of Glasgow, 2019.

[GS12] Béla Genge and Christos Siaterlis. An Experimental Study on the Impact of Network Segmentation to the Resilience of Physical Processes. In *Proceedings of the 11th International Networking Conference (NETWORKING)*, pages 121–134, 2012.

[GS14] André Gensler and Bernhard Sick. Novel Criteria to Measure Performance of Time Series Segmentation Techniques. In *Proceedings of the 2014 Workshop on Knowledge Discovery, Data Mining and Machine Learning (KDML)*, pages 193–204, 2014.

[GT16] Hamid Reza Ghaeini and Nils Ole Tippenhauer. HAMIDS: Hierarchical Monitoring Intrusion Detection System for Industrial Control Systems. In *Proceedings of the 2nd ACM Workshop on Cyber-Physical Systems Security and Privacy (CPS-SPC)*, page 103–111, 2016.

[GUC+18] Jairo Giraldo, David Urbina, Alvaro Cardenas, Junia Valente, Mustafa Faisal, Justin Ruths, Nils Ole Tippenhauer, Henrik Sandberg, and Richard Candell. A Survey of Physics-Based Attack Detection in Cyber-Physical Systems. *ACM Computing Surveys*, 51(4), 2018.

[GW13] Niv Goldenberg and Avishai Wool. Accurate Modeling of Modbus/TCP for Intrusion Detection in SCADA Systems. *International Journal of Critical Infrastructure Protection (IJCIP)*, 6(2):63–75, 2013.

[GZS+22] Astha Garg, Wenyu Zhang, Jules Samaran, Ramasamy Savitha, and Chuan-Sheng Foo. An Evaluation of Anomaly Detection and Diagnosis in Multivariate Time Series. *IEEE Transactions on Neural Networks and Learning Systems*, 33(6):2508–2517, 2022.

[HBP21] Christian Hemminghaus, Jan Bauer, and Elmar Padilla. BRAT: A BRidge Attack Tool for Cyber Security Assessments of Maritime Systems. *TransNav*, 15(1), 2021.

[HBW21] Christian Hemminghaus, Jan Bauer, and Konrad Wolsing. SIGMAR: Ensuring Integrity and Authenticity of Maritime Systems using Digital Signatures. In *Proceedings of the International Symposium on Networks, Computers and Communications (ISNCC)*, pages 1–6, 2021.

[HEF18] Kevin E. Hemsley and Dr. Ronald E. Fisher. History of Industrial Control System Cyber Incidents. Technical report, Idaho National Lab.(INL), Idaho Falls, ID (United States), 2018.

[HHS+18] Jens Hiller, Martin Henze, Martin Serror, Eric Wagner, Jan Niklas Richter, and Klaus Wehrle. Secure Low Latency Communication for Constrained Industrial IoT Scenarios. In *Proceedings of the IEEE 43rd Conference on Local Computer Networks (LCN)*, pages 614–622, 2018.

[HKKS10] Inseok Hwang, Sungwan Kim, Youdan Kim, and Chze Eng Seah. A Survey of Fault Detection, Isolation, and Reconfiguration Methods. *IEEE Transactions on Control Systems Technology*, 18(3):636–653, 2010.

[HL19] Zhongyuan Hau and Emil C. Lupu. Exploiting Correlations to Detect False Data Injections in Low-Density Wireless Sensor Networks. In *Proceedings of the 5th on Cyber-Physical System Security Workshop (CPSS)*, page 1–12, 2019.

[HL21] Chanwoong Hwang and Taejin Lee. E-SFD: Explainable Sensor Fault Detection in the ICS Anomaly Detection System. *IEEE Access*, 9:140470–140486, 2021.

[HLLL17] Abdulmalik Humayed, Jingqiang Lin, Fengjun Li, and Bo Luo. Cyber-Physical Systems Security—A Survey. *IEEE Internet of Things Journal*, 4(6):1802–1831, 2017.

[HMS21] HMS Networks. Continued growth for industrial networks despite pandemic. `https://web.archive.org/web/20220729044036/https://www.hms-networks.com/news-and-insights/news-from-hms/2021/03/31/continued-growth-for-industrial-networks-despite-pandemic`, 2021. [Online, accessed 2024-07-29].

[HNR22] Alexis Huet, Jose Manuel Navarro, and Dario Rossi. Local Evaluation of Time Series Anomaly Detection Algorithms. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*, pages 635–645, 2022.

[HSZH14] Dina Hadžiosmanović, Robin Sommer, Emmanuele Zambon, and Pieter H. Hartel. Through the Eye of the PLC: Semantic Security Monitoring for Industrial Processes. In *Proceedings of the 30th Annual Computer Security Applications Conference (ACSAC)*, page 126–135, 2014.

[HYKK19] Won-Seok Hwang, Jeong-Han Yun, Jonguk Kim, and Hyoung Chun Kim. Time-Series Aware Precision and Recall for Anomaly Detection: Considering Variety of Detection Result and Addressing Ambiguous Labeling. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management (CIKM)*, pages 2241–2244, 2019.

[HYKM22] Won-Seok Hwang, Jeong-Han Yun, Jonguk Kim, and Byung Gil Min. "Do You Know Existing Accuracy Metrics Overrate Time-Series Anomaly Detections?". In *Proceedings of the 37th ACM/SIGAPP Symposium on Applied Computing (SAC)*, pages 403–412, 2022.

[HYL+18] Yan Hu, An Yang, Hong Li, Yuyan Sun, and Limin Sun. A survey of intrusion detection on industrial control systems. *International Journal of Distributed Sensor Networks*, 14(8), 2018.

[IBM24] IBM. Cost of a Data Breach Report 2024. `https://web.archive.org/web/20241216184314/https://table.media/wp-content/uploads/2024/07/30132828/Cost-of-a-Data-Breach-Report-2024.pdf`, 2024. [Online, accessed 2024-12-16].

[Ics12] USDHS ICS-CERT. Gas Pipeline Cyber Intrusion Campaign. `https://web.archive.org/web/20250205154100/https://www.cisa.gov/sites/default/files/Monitors/ICS-CERT_Monitor_Apr2012.pdf`, 2012. [Online, accessed 2025-02-05].

[IEC13] IEC. IEC 62443-3-3 Industrial communication networks – Network and system security – Part 3-3: System security requirements and security levels, Edition 1.0. Technical report, International Electrotechnical Commission (IEC), 2013.

[IEC16] IEC. Enterprise-control system integration - Part 3: Activity models of manufacturing operations management. Standard, International Electrotechnical Commission (IEC), 2016.

[IEC18a] IEC. Maritime navigation and radiocommunication equipment and systems – Digital interfaces – Part 450: Multiple talkers and multiple listeners – Ethernet interconnection. Standard, International Electrotechnical Commission (IEC), 2018.

[IEC18b] IEC. Telecontrol equipment and systems - Part 5-104: Transmission protocols - Network access for IEC 60870-5-101 using standard transport profiles. Standard, International Electrotechnical Commission (IEC), 2018.

[IEC24] IEC. IEC 62381 Automation systems in the process industry - Factory acceptance test (FAT), site acceptance test (SAT), and site integration test (SIT). Standard, International Electrotechnical Commission (IEC), 2024.

[IM11] Rolf Isermann and Marco Münchhof. *Identification of dynamic systems: an introduction with applications*, volume 85. Springer, 2011.

[IMO82] IMO Resolution A.477(12). Performance Standards for Radar Equipment. Resolution, International Maritime Organization, 1982.

[iTr] iTrust. Invariants using Design-centri approach. `https://web.archive.org/web/20240622112839/https://itrust.sutd.edu.sg/wp-content/uploads/2017/08/comparison.pdf`. [Online, accessed 2024-07-22].

[Itu18] Mikel Iturbe. PASAD. `https://web.archive.org/web/20231128212048/https://github.com/mikeliturbe/pasad`, 2018. [Online, accessed 2023-11-28].

[IYC+17] Jun Inoue, Yoriyuki Yamagata, Yuqi Chen, Christopher M. Poskitt, and Jun Sun. Anomaly Detection for a Water Treatment System Using Unsupervised Machine Learning. In *Proceedings of the IEEE International Conference on Data Mining Workshops (ICDMW)*, pages 1058–1065, 2017.

[JG16] Khurum Nazir Junejo and Jonathan Goh. Behaviour-Based Attack Detection and Classification in Cyber Physical Systems Using Machine Learning. In *Proceedings of the 2nd ACM International Workshop on Cyber-Physical System Security (CPSS)*, page 34–43, 2016.

[JKB14] Austin Jones, Zhaodan Kong, and Calin Belta. Anomaly detection in cyber-physical systems: A formal methods approach. In *Proceedings of the 53rd IEEE Conference on Decision and Control (CDC)*, pages 848–853, 2014.

[JMAC20] Nick James, Max Menzies, Lamiae Azizi, and Jennifer Chan. Novel semi-metrics for multivariate change point analysis and anomaly detection. *Physica D: Nonlinear Phenomena*, 412, 2020.

[JTP16] Kai Jansen, Nils Ole Tippenhauer, and Christina Pöpper. Multi-Receiver GPS Spoofing Detection: Error Models and Realization. In *Proceedings of the 32nd Annual Conference on Computer Security Applications (ACSAC)*, page 237–250, 2016.

[KBC13] Fotios Katsilieris, Paolo Braca, and Stefano Coraluppi. Detection of malicious AIS position spoofing by exploiting radar information. In *Proceedings of the 16th International Conference on Information Fusion*, pages 1196–1203, 2013.

[KCC+22] Siwon Kim, Kukjin Choi, Hyun-Soo Choi, Byunghan Lee, and Sungroh Yoon. Towards a Rigorous Evaluation of Time-Series Anomaly Detection. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36(7):7194–7201, 2022.

[KCH18] Gary C. Kessler, Philip Craiger, and Jon C. Haass. A Taxonomy Framework for Maritime Cybersecurity: A Demonstration Using the Automatic Identification System. *International Journal on Marine Navigation and Safety of Sea Transportation (TransNav)*, 12(3):429–437, 2018.

[KFPG19] Mohamad Kaouk, Jean-Marie Flaus, Marie-Laure Potet, and Roland Groz. A Review of Intrusion Detection Systems for Industrial Control Systems. In *Proceedings of the 6th International Conference on Control, Decision and Information Technologies (CoDIT)*, pages 1699–1704, 2019.

[KGH15] István Kiss, Béla Genge, and Piroska Haller. A clustering-based approach to detect cyber attacks in process control systems. In *Proceedings of the IEEE 13th International Conference on Industrial Informatics (INDIN)*, pages 142–148, 2015.

[Kim19] Jonguk Kim. swat-seq2seq. `https://web.archive.org/web/20231203103052/https://github.com/jukworks/swat-seq2seq`, 2019. [Online, accessed 2023-12-03].

[Kit07] Barbara Ann Kitchenham. Guidelines for performing Systematic Literature Reviews in Software Engineering. Technical report, Keele University and University of Durham, 2007.

[KJB17] Zhaodan Kong, Austin Jones, and Calin Belta. Temporal Logics for Learning and Detection of Anomalous Behavior. *IEEE Transactions on Automatic Control*, 62(3):1210–1222, 2017.

[KJL24] Seok-Jin Kim, Tae-Youl Jeon, and Young-Chan Lee. Impact of Ship Noise on Seafarers' Sleep Disturbances and Daily Activities: An Analysis of Fatigue Increase and Maritime Accident Risk through a Survey. *Applied Sciences*, 14(9), 2024.

[KJM+14] Zhaodan Kong, Austin Jones, Ana Medina Ayala, Ebru Aydin Gol, and Calin Belta. Temporal Logic Inference for Classification and Prediction from Data. In *Proceedings of the 17th International Conference on Hybrid Systems: Computation and Control (HSCC)*, page 273–282, 2014.

[KK89] Tomihisa Kamada and Satoru Kawai. An algorithm for drawing general undirected graphs. *Information Processing Letters*, 31(1):7–15, 1989.

[KK18] Viacheslav Kulik and Ruslan Kirichek. The Heterogeneous Gateways in the Industrial Internet of Things. In *Proceedings of the 10th International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT)*, pages 1–5, 2018.

[KK21] Sokratis Katsikas and Georgios Kavallieratos. *Chapter: Cybersecurity of the Unmanned Ship.* Cybersecurity Issues in Emerging Technologies. Taylor & Francis Group, 1st edition, 2021.

[KKG19] Georgios Kavallieratos, Sokratis K. Katsikas, and Vasileios Gkioulos. Towards a Cyber-Physical Range. In *Proceedings of the 5th on Cyber-Physical System Security Workshop (CPSS)*, pages 25–34. ACM, 2019.

[KKG20] Georgios Kavallieratos, Sokratis Katsikas, and Vasileios Gkioulos. Modelling Shipping 4.0: A Reference Architecture for the Cyber-Enabled Ship. In *Intelligent Information and Database Systems*, pages 202–217, 2020.

[KKL22] Hee-Yong Kwon, Taesic Kim, and Mun-Kyu Lee. Advanced Intrusion Detection Combining Signature-Based and Behavior-Based Detection Methods. *Electronics*, 11(6), 2022.

[KL14] Eric D Knapp and Joel Thomas Langill. *Industrial Network Security: Securing critical infrastructure networks for smart grid, SCADA, and other Industrial Control Systems.* Syngress, 2014.

[KLG15] Marina Krotofil, Jason Larsen, and Dieter Gollmann. The Process Matters: Ensuring Data Veracity in Cyber-Physical Systems. In *Proceedings of the 10th ACM Symposium on Information, Computer and Communications Security (ASIA CCS)*, page 133–144, 2015.

[KPH+15] William Knowles, Daniel Prince, David Hutchison, Jules Ferdinand Pagna Disso, and Kevin Jones. A survey of cyber security management in industrial control systems. *International Journal of Critical Infrastructure Protection*, 9, 2015.

[KS15] Abdullah Khalili and Ashkan Sami. SysDetect: A systematic approach to critical state determination for Industrial Intrusion Detection Systems using Apriori algorithm. *Journal of Process Control*, 32:154 – 160, 2015.

[KS18] Moshe Kravchik and Asaf Shabtai. Detecting Cyber Attacks in Industrial Control Systems Using Convolutional Neural Networks. In *Proceedings of the Workshop on Cyber-Physical Systems Security and PrivaCy (CPS-SPC)*, page 72–83, 2018.

[KS21] Moshe Kravchik and Asaf Shabtai. Efficient Cyber Attack Detection in Industrial Control Systems Using Lightweight Neural Networks and PCA. *IEEE Transactions on Dependable and Secure Computing*, 19(4):2179–2197, 2021.

[KSC21] Ajit Kumar, Neetesh Saxena, and Bong Jun Choi. Machine Learning Algorithm for Detection of False Data Injection Attack in Power System. In *Proceedings of the International Conference on Information Networking (ICOIN)*, pages 385–390, 2021.

[KSK+19] Jun-Sung Kim, Seong Min So, Joong-Tae Kim, Jung-Won Cho, Hee-Jeong Park, Fauzan Hanif Jufri, and Jaesung Jung. Microgrids platform: A design and implementation of common platform for seamless microgrids operation. *Electric Power Systems Research*, 167:21–38, 2019.

[KWP+22a] Dominik Kus, Eric Wagner, Jan Pennekamp, Konrad Wolsing, Ina Berenice Fink, Markus Dahlmanns, Klaus Wehrle, and Martin Henze. A False Sense of Security? Revisiting the State of Machine Learning-Based Industrial Intrusion Detection. In *Proceedings of the 8th ACM Cyber-Physical System Security Workshop (CPSS)*, page 73–84, 2022.

[KWP+22b] Dominik Kus, Konrad Wolsing, Jan Pennekamp, Eric Wagner, Martin Henze, and Klaus Wehrle. Poster: Ensemble Learning for Industrial Intrusion Detection. Technical Report RWTH-2022-10809, RWTH Aachen University, 2022.

[KYK20] Jonguk Kim, Jeong-Han Yun, and Hyoung Chun" Kim. Anomaly Detection for Industrial Control Systems Using Sequence-to-Sequence Neural Networks. In *Proceedings of the ESORICS 2019 International Workshops, CyberICPS, SECPRE, SPOSE, and ADIoT*, pages 3–18, 2020.

[LA15] Alexander Lavin and Subutai Ahmad. Evaluating Real-Time Anomaly Detection Algorithms – The Numenta Anomaly Benchmark. In *Proceedings of the IEEE 14th International Conference on Machine Learning and Applications (ICMLA)*, pages 38–44, 2015.

[LAC16] Robert M. Lee, Michael J. Assante, and Tim Conway. Analysis of the Cyber Attack on the Ukrainian Power Grid. *Electricity Information Sharing and Analysis Center (E-ISAC)*, 2016.

[LAVM18] Qin Lin, Sridha Adepu, Sicco Verwer, and Aditya Mathur. TABOR: A Graphical Model-based Approach for Anomaly Detection in Industrial Control Systems. In *Proceedings of the 2018 on Asia Conference on Computer and Communications Security (AsiaCCS)*, page 525–536, 2018.

[LCGN18] Dan Li, Dacheng Chen, Jonathan Goh, and See-kiong Ng. Anomaly Detection with Generative Adversarial Networks for Multivariate Time Series. In *Proceedigns of the 7th International Workshop on Big Data, Streams and Heterogeneous Source Mining: Algorithms, Systems, Programming Models and Applications (KDD BigMine)*, 2018.

[LDSR05] Pavel Laskov, Patrick Düssel, Christin Schäfer, and Konrad Rieck. Learning Intrusion Detection: Supervised or Unsupervised? In *Proceedings of the 13th International Conference on Image Analysis and Processing (ICIAP)*, pages 50–57, 2005.

[LF16] Antoine Lemay and José M. Fernandez. Providing SCADA Network Data Sets for Intrusion Detection Research. In *Proceedings of the 9th USENIX Workshop on Cyber Security Experimentation and Test (CSET)*, 2016.

[LFK+14] Heiner Lasi, Peter Fettke, Hans-Georg Kemper, Thomas Feld, and Michael Hoffmann. Industry 4.0. *Business & Information Systems Engineering*, 6:239–242, 2014.

[LGH+18] Mass Soldal Lund, Jørgen Emil Gulland, Odd Sveinung Hareide, Øyvind Jøsok, and Karl Olav Carlsson Weum. Integrity of Integrated Navigation Systems. In *Proceedings of the Conference on Communications and Network Security (CNS)*, pages 1–5, 2018.

[LGH+21] Tim Lackorzynski, Gregor Garten, Jan Sönke Huster, Stefan Köpsell, and Hermann Härtig. Enabling and optimizing macsec for industrial environments. *IEEE Transactions on Industrial Informatics*, 17(11):7599–7606, 2021.

[LGH+23] Maxime Lanvin, Pierre-François Gimenez, Yufei Han, Frédéric Majorczyk, Ludovic Mé, and Eric Totel. Towards Understanding Alerts raised by Unsupervised Network Intrusion Detection Systems. In *Proceedings of the 26th International Symposium on Research in Attacks, Intrusions and Defenses (RAID)*, page 135–150, 2023.

[LKP+19] George Loukas, Eirini Karapistoli, Emmanouil Panaousis, Panagiotis Sarigiannidis, Anatolij Bezemskij, and Tuan Vuong. A taxonomy and survey of cyber-physical intrusion detection approaches for vehicles. *Ad Hoc Networks*, 84:124–147, 2019.

[LKS19] Tim Lackorzynski, Stefan Köpsell, and Thorsten Strufe. A comparative study on virtual private networks for future industrial communication systems. In *2019 15th IEEE International Workshop on Factory Communication Systems (WFCS)*, pages 1–8, 2019.

[LLL+11] Jung-Jin Lee, Pyoung-Hean Lee, Seong-Whan Lee, Alan Yuille, and Christof Koch. AdaBoost for Text Detection in Natural Scene. In *Proceedings of the International Conference on Document Analysis and Recognition (ICDAR)*, pages 429–434, 2011.

[LLN+18] Richard Liaw, Eric Liang, Robert Nishihara, Philipp Moritz, Joseph E. Gonzalez, and Ion Stoica. Tune: A Research Platform for Distributed Model Selection and Training, 2018.

[LMAR23] Giacomo Longo, Alessio Merlo, Alessandro Armando, and Enrico Russo. Electronic Attacks as a Cyber False Flag against Maritime Radars Systems. In *Proceedings of the IEEE 48rd Conference on Local Computer Networks (LCN)*, pages 1–6, 2023.

[LNT18] Chih-Yuan Lin and Simin Nadjm-Tehrani. Understanding IEC-60870-5-104 Traffic Patterns in SCADA Networks. In *Proceedings of the 4th ACM Workshop on Cyber-Physical System Security (CPSS)*, page 51–60, 2018.

[LNT19] Chih-Yuan Lin and Simin Nadjm-Tehrani. Timing Patterns and Correlations in Spontaneous SCADA Traffic for Anomaly Detection. In *Proceedings of the 22nd International Symposium on Research in Attacks, Intrusions and Defenses (RAID)*, pages 73–88, 2019.

[LNTA17] Chih-Yuan Lin, Simin Nadjm-Tehrani, and Mikael Asplund. Timing-based Anomaly Detection in SCADA Networks. In *Critical Information Infrastructures Security (CRITIS)*, pages 48–59, 2017.

[LRAM23] Giacomo Longo, Enrico Russo, Alessandro Armando, and Alessio Merlo. Attacking (and Defending) the Maritime Radar System. *IEEE Transactions on Information Forensics and Security*, pages 3575–3589, 2023.

[LSAA23] Eric Lanfer, Sophia Sylvester, Nils Aschenbruck, and Martin Atzmueller. Leveraging Explainable AI Methods Towards Identifying Classification Issues on IDS Datasets. In *Proceedings of the 2023 IEEE 48th Conference on Local Computer Networks (LCN)*, pages 1–4, 2023.

[LSW24] Max Landauer, Florian Skopik, and Markus Wurzenberger. A Critical Review of Common Log Data Sets Used for Evaluation of Sequence-Based Anomaly Detection Techniques. *Proceedings of the ACM on Software Engineering*, 1(FSE), 2024.

[LT21] Maya Hilda Lestari Louk and Bayu Adhi Tama. Exploring Ensemble-Based Class Imbalance Learners for Intrusion Detection in Industrial Control Networks. *Big Data And Cognitive Computing*, 5(4), 2021.

[LWW+23] Olav Lamberts, Konrad Wolsing, Eric Wagner, Jan Pennekamp, Jan Bauer, Klaus Wehrle, and Martin Henze. SoK: Evaluations in Industrial Intrusion Detection Research. *Journal of Systems Research (JSys)*, 3(1), 2023.

[LXC+21] Yuan Luo, Ya Xiao, Long Cheng, Guojun Peng, and Danfeng Yao. Deep Learning-Based Anomaly Detection in Cyber-Physical Systems: Progress and Opportunities. *ACM Computing Surveys*, 54(5), 2021.

[LXW+21] Yunfeng Li, Wenli Xue, Ting Wu, Huaizhi Wang, Bin Zhou, Saddam Aziz, and Yang He. Intrusion detection of cyber physical energy system based on multivariate ensemble classification. *Energy*, 218, 2021.

[Lyn24] Sean Lyngaas. Russia-linked hacking group suspected of carrying out cyberattack on Texas water facility, cybersecurity firm says. `https://web.archive.org/web/20240424195516/https://amp.cnn.com/cnn/2024/04/17/politics/russia-hacking-group-suspected-texas-water-cyberattack`, 2024. [Online, accessed 2024-04-24].

[MAM21] Gauthama Raman M. R., Chuadhry Mujeeb Ahmed, and Aditya Mathur. Machine learning for intrusion detection in industrial control systems: challenges and lessons from experimental evaluation. *Cybersecurity*, 4, 2021.

[MBF19] Mohamad Houssein Monzer, Kamal Beydoun, and Jean-Marie FLAUS. Model based rules generation for Intrusion Detection System for industrial systems. In *Proceedings of the International Conference on Control, Automation and Diagnosis (ICCAD)*, pages 1–6, 2019.

[MBJ+25] Jose Carlos Mondragon, Paula Branco, Guy-Vincent Jourdan, Andres Eduardo Gutierrez-Rodriguez, and Rajesh Roshan Biswal. Advanced ids: a comparative study of datasets and machine learning algorithms for network flow-based intrusion detection systems. *Applied Intelligence*, 55(7):608, 2025.

[MC99] David E Mann and Steven M Christey. Towards a common enumeration of vulnerabilities. In *Proceedings of the 2nd Workshop on Research with Security Vulnerability Databases*, page 9, 1999.

[MC14] Robert Mitchell and Ing-Ray Chen. A Survey of Intrusion Detection Techniques for Cyber-Physical Systems. *ACM Computing Surveys*, 46(4), 2014.

[MDES18] Yisroel Mirsky, Tomer Doitshman, Yuval Elovici, and Asaf Shabtai. Kitsune: an ensemble of autoencoders for online network intrusion detection. *Network and Distributed System Security (NDSS)*, 2018.

[MIT24] MITRE. MITRE ATT&CK Matrix for ICS. `https://web.archive.org/web/20241122235627/https://attack.mitre.org/versions/v16/matrices/ics/`, 2024. [Online, accessed 2024-11-22].

[MJ14] Leandros A. Maglaras and Jianmin Jiang. Intrusion detection in SCADA systems using machine learning techniques. In *Science and Information Conference (SAI)*, pages 626–631, 2014.

[MJC16] Leandros A. Maglaras, Jianmin Jiang, and Tiago J. Cruz. Combining ensemble methods and social network metrics for improving accuracy of OCSVM on intrusion detection in SCADA systems. *Journal of Information Security and Applications*, 30:15–26, 2016.

[MMA+16] Ariana Mirian, Zane Ma, David Adrian, Matthew Tischer, Thasphon Chuenchujit, Tim Yardley, Robin Berthier, Joshua Mason, Zakir Durumeric, J. Alex Halderman, and Michael Bailey. An Internet-wide view of ICS devices. In *Proceedings of the 14th Annual Conference on Privacy, Security and Trust (PST)*, pages 96–103, 2016.

[MMSJS12] Joao Mendes-Moreira, Carlos Soares, Alípio Mário Jorge, and Jorge Freire De Sousa. Ensemble approaches for regression: A survey. *ACM Computing Surveys*, 45(1), 2012.

[Mod18] Modbus. Modbus Application Protocol Specification V1.1b. Standard, Modbus Organization, 2018.

[Mor19] Steve Morgan. 2019 Official Annual Cybercrime Report. `https://web.archive.org/web/20240514132252/https://www.empresas.hsbc.com.mx/-/media/media/mexico/pdfs/cv-hg-2019-official-annual-cybercrime-report.pdf`, 2019. [Online, accessed 2024-05-14].

[MPA17] Houssem Mansouri, Al-Sakib Khan Pathan, and Makhlouf Aliouat. A snapshot security protocol for radar network protection. In *Proceedings of the Seminar on Detection Systems Architectures and Technologies (DAT)*, pages 1–6, 2017.

[MSM+21] Thomas Miller, Alexander Staves, Sam Maesschalck, Miriam Sturdee, and Benjamin Green. Looking back to look forward: Lessons learnt from cyber-attacks on Industrial Control Systems. *International Journal of Critical Infrastructure Protection*, 35:100464, 2021.

[MSRF18] David Myers, Suriadi Suriadi, Kenneth Radke, and Ernest Foo. Anomaly detection for industrial control systems using process mining. *Computers & Security*, 78:103–125, 2018.

[MTT15] Thomas H Morris, Zach Thornton, and Ian Turnipseed. Industrial control system simulation and data logging for intrusion detection system research. *Proceedings of the 7th annual southeastern cyber security summit (SCSS)*, pages 3–4, 2015.

[MW19] Mayra Macas and Chunming Wu. An Unsupervised Framework for Anomaly Detection in a Water Treatment System. In *Proceedings of the 18th International Conference On Machine Learning And Applications (ICMLA)*, pages 1298–1305, 2019.

[MYZ+23] Jie Meng, Zeyu Yang, Zhenyong Zhang, Yangyang Geng, Ruilong Deng, Peng Cheng, Jiming Chen, and Jianying Zhou. SePanner: Analyzing Semantics of Controller Variables in Industrial Control Systems based on Network Traffic. In *Proceedings of the 39th Annual Computer Security Applications Conference (ACSAC)*, page 310–323, 2023.

[NBHF21] Nataliia Neshenko, Elias Bou-Harb, and Borko Furht. A Behavioral-Based Forensic Investigation Approach for Analyzing Attacks on Water Plants Using GANs. *Forensic Science International: Digital Investigation*, 37:301198, 2021.

[NHB14] Patric Nader, Paul Honeine, and Pierre Beauseroy. l_p-norms in One-Class Classification for Intrusion Detection in SCADA Systems. *IEEE Transactions on Industrial Informatics*, 10(4):2308–2317, 2014.

[NIS] Directive (EU) 2022/2555 of the european parliament and of the council of 14 december 2022 on measures for a high common level of cybersecurity across the union, amending regulation (EU) no 910/2014 and directive (EU) 2018/1972, and repealing directive (EU) 2016/1148 (nis 2 directive). `https://web.archive.org/web/20240820192449/https://eur-lex.europa.eu/legal-content/EN/TXT/PDF/?uri=CELEX:32022L2555`. [Online, accessed 2024-08-29].

[NJMP20] Dusan M. Nedeljkovic, Zivana B. Jakovljevic, Zoran Dj Miljkovic, and Miroslav Pajic. Detection of Cyber-Attacks in Systems With Distributed Control Based on Support Vector Regression. *TELFOR Journal*, 12(2):104–109, 2020.

[NLC22] Duc-Duong Nguyen, Minh-Thuy Le, and Thanh-Long Cung. Improving intrusion detection in SCADA systems using stacking ensemble of tree-based models. *Bulletin of Electrical Engineering and Informatics*, 11(1):119–127, 2022.

[NTNK21] Susumu Naito, Yasunori Taguchi, Kouta Nakata, and Yuichi Kato. Anomaly Detection for Multivariate Time Series on Large-Scale Fluid Handling Plant Using Two-Stage Autoencoder. In *Proceedings of the International Conference on Data Mining Workshops (ICDMW)*, pages 542–551, 2021.

[Ope22] opencpn-radar pi. radar_pi. `https://web.archive.org/web/20240405021852/https://github.com/opencpn-radar-pi/radar_pi`, 2022. [Online, accessed 2024-04-05].

[Ope23] OpenCPN. OpenCPN Chart Plotter. `https://web.archive.org/web/20241217000214/https://github.com/OpenCPN/OpenCPN`, 2023. [Online, accessed 2024-12-17].

[ORL20] Felix O. Olowononi, Danda B. Rawat, and Chunmei Liu. Resilient Machine Learning for Networked Cyber Physical Systems: A Survey for Machine Learning Security to Securing Machine Learning for CPS. *IEEE Communications Surveys & Tutorials*, 23(1):524–552, 2020.

[ORZ+24] Neil Ortiz, Martin Rosso, Emmanuele Zambon, Jerry den Hartog, and Alvaro A. Cardenas. From Power to Water: Dissecting SCADA Networks Across Different Critical Infrastructures. In *Proceedings of 25th International Conference on Passive and Active Measurement (PAM)*, pages 3–31, 2024.

[PA16] Stanislav Ponomarev and Travis Atkison. Industrial Control System Network Intrusion Detection by Telemetry Analysis. *IEEE Transactions on Dependable and Secure Computing*, 13(2):252–260, 2016.

[PAG17] Koyena Pal, Sridhar Adepu, and Jonathan Goh. Effectiveness of Association Rules Mining for Invariants Generation in Cyber-Physical Systems. In *Proceedings of the 18th International Symposium on High Assurance Systems Engineering (HASE)*, pages 124–127, 2017.

[PASE18] Rocio Lopez Perez, Florian Adamsky, Ridha Soua, and Thomas Engel. Machine Learning for Reliable Network Attack Detection in SCADA Systems. In *Proceedings of the 17th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/ 12th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE)*, pages 633–638, 2018.

[PB07] Sean Peisert and Matt Bishop. How to Design Computer Security Experiments. In *Proceedings of the 5th World Conference on Information Security Education (WISE)*, pages 141–148, 2007.

[PB18] Ranjit Panigrahi and Samarjeet Borah. A detailed analysis of CICIDS2017 dataset for designing Intrusion Detection Systems. *International Journal of Engineering & Technology*, 7(3.24):479–482, 2018.

[PBB19] Philipp Probst, Anne-Laure Boulesteix, and Bernd Bischl. Tunability: Importance of Hyperparameters of Machine Learning Algorithms. *Journal of Machine Learning Research*, 20(53):1–32, 2019.

[PBD+21] Jan Pennekamp, Erik Buchholz, Markus Dahlmanns, Ike Kunze, Stefan Braun, Eric Wagner, Matthias Brockmann, Klaus Wehrle, and Martin Henze. Collaboration is not Evil: A Systematic Look at Security Research for Industrial Use. In *Proceedings of the Workshop on Learning from Authoritative Security Experiment Results (LASER)*, pages 1–16, 2021.

[PFH+20] Angel Luis Perales Gomez, Lorenzo Fernandez Maimo, Alberto Huertas Celdran, and Felix J. Garcia Clemente. MADICS: A Methodology for Anomaly Detection in Industrial Control Systems. *Symmetry*, 12(10), 2020.

[Pit23a] Martin Pitt. Taking a Look Back at Control: Part 1. `http://web.archive.org/web/20240620230451/https://www.thechemicalengineer.com/features/taking-a-look-back-at-control-part-1`, 2023. [Online, accessed 2024-06-20].

[Pit23b] Martin Pitt. Taking a Look Back at Control: Part 2. `http://web.archive.org/web/20231207105641/https://www.thechemicalengineer.com/features/taking-a-look-back-at-control-part-2`, 2023. [Online, accessed 2023-12-07].

[PMA15] Shengyi Pan, Thomas Morris, and Uttam Adhikari. Developing a Hybrid Intrusion Detection System Using Data Mining for Power Systems. *IEEE Transactions on Smart Grid*, 6(6):3104–3113, 2015.

[PMS+20] James Pavur, Daniel Moser, Martin Strohmeier, Vincent Lenders, and Ivan Martinovic. A Tale of Sea and Sky On the Security of Maritime VSAT Communications. In *IEEE Symposium on Security and Privacy (SP)*, pages 1384–1400, 2020.

[Poc93] Manel Poch. Water Treatment Plant. `https://web.archive.org/web/20240724173336/https://archive.ics.uci.edu/dataset/106/water+treatment+plant`, 1993. [Online, accessed 2024-07-24].

[Pow11] David M. W. Powers. Evaluation: From Precision, Recall and F-Measure to ROC, Informedness, Markedness & Correlation. *Journal of Machine Learning Technologies*, 2(1):37–63, 2011.

[PTS18] Venkata Reddy Palleti, Yu Chong Tan, and Lakshminarayanan Samavedham. A mechanistic fault detection and isolation approach using Kalman filter to improve the security of cyber physical systems. *Journal of Process Control*, 68:160–170, 2018.

[QGS+20] Raul Quinonez, Jairo Giraldo, Luis Salazar, Erick Bauman, Alvaro Cardenas, and Zhiqiang Lin. SAVIOR: Securing autonomous vehicles with robust physical invariants. In *Proceedings of the 29th USENIX Security Symposium (USENIX Security)*, pages 895–912, 2020.

[Raj20] R. Raj. PyPASAD. `https://web.archive.org/web/20240617083150/https://github.com/rahulrajpl/PyPASAD`, 2020. [Online, accessed 2024-06-17].

[RAMH18] Daniel Ramotsoela, Adnan Abu-Mahfouz, and Gerhard Hancke. A Survey of Anomaly Detection in Industrial Wireless Sensor Networks with Critical Water System Infrastructure as a Case Study. *Sensors*, 18(8), 2018.

[RCL11] Ø. J. Rødseth, M. J. Christensen, and K. Lee. Design challenges and decisions for a new ship data network. In *Proceedings of the International Symposium on Information on Ships (ISIS)*, pages 1–21, 2011.

[RM21] Ondřej Ryšavý and Petr Matoušek. A Network Traffic Processing Library for ICS Anomaly Detection. In *7th Conference on the Engineering of Computer Based Systems (ECBS)*, pages 1–7, 2021.

[Roc21] Rocio. ML-NIDS-for-SCADA. `https://web.archive.org/web/20221207155200/https://github.com/Rocionightwater/ML-NIDS-for-SCADA`, 2021. [Online, accessed 2022-12-07].

[Roe99] Martin Roesch. Snort: Lightweight intrusion detection for networks. In *Proceedings of LISA '99: 13th Systems Administration Conference*, pages 229–238, 1999.

[Rok10] Lior Rokach. Ensemble-based Classifiers. *Artificial Intelligence Review*, 33(1–2):1–39, 2010.

[RSE+20] Panagiotis Radoglou-Grammatikis, Panagiotis Sarigiannidis, George Efstathopoulos, Paris-Alexandros Karypidis, and Antonios Sarigiannidis. DIDEROT: An Intrusion Detection and Prevention System for DNP3-Based SCADA Systems. In *Proceedings of the 15th International Conference on Availability, Reliability and Security (ARES)*, pages 1–8, 2020.

[RSMP20] Slavica V. Boštjančič Rakas, Mirjana D. Stojanović, and Jasna D. Marković-Petrović. A Review of Research Work on Network-Based SCADA Intrusion Detection Systems. *IEEE Access*, 8:93083–93108, 2020.

[RT14] Ørnulf Jan Rødseth and Åsmund Tjora. A system architecture for an unmanned ship. In *Proceedings of the 13th International Conference on Computer and IT Applications in the Maritime Industries (COMPIT)*, 2014.

[SAH16] Robin Sommer, Johanna Amann, and Seth Hall. Spicy: A unified deep packet inspection framework dissecting all your data. In *Proceedings of the 32nd Annual Conference on Computer Security Applications (ACSAC)*, page 558–569, 2016.

[SAP24] Guilherme Saraiva, Filipe Apolinário, and Miguel L. Pardal. IM-DISCO: Invariant Mining for Detecting IntrusionS in Critical Operations. In *Proceedings of the ESORICS 2023 International Workshops: CPS4CIP, ADIoT, SecAssure, WASP, TAURIN, PriST-AI, and SECAI*, pages 42–58, 2024.

[SB21] William Stallings and Lawrie Brown. *Computer Security: Principles and Practice*. Pearson, fourth edition edition, 2021.

[SBB19] Tim Claudius Stratmann, Dierk Brauer, and Susanne Boll. Supporting the Perception of Spatially Distributed Information on Ship Bridges. In *Proceedings of Mensch Und Computer (MuC)*, page 475–479, 2019.

[SBJ+08] Frank Steinicke, Gerd Bruder, Jason Jerald, Harald Frenz, and Markus Lappe. Analyses of Human Sensitivity to Redirected Walking. In *Proceedings of the Symposium on Virtual Reality Software and Technology (VRST)*, page 149–156, 2008.

[SCPA21] Bruno Sousa, Tiago Cruz, Vasco Pereira, and Miguel Arieiro. Denial of Service and Man in The Middle attacks in Programmable Logic Controllers, 2021.

[SCW+16] Wenli Shang, Junrong Cui, Ming Wan, Panfeng An, and Peng Zeng. Modbus Communication Behavior Modeling and SVM Intrusion Detection Method. In *Proceedings of the 6th International Conference on Communication and Network Security (ICCNS)*, page 80–85, 2016.

[SFL18] Dmitry Shalyga, Pavel Filonov, and Andrey Lavrentyev. Anomaly Detection for Water Treatment System Based on Neural Network With Automatic Architecture Optimization. In *ICML Workshop for Deep Learning for Safety-Critical in Engineering Systems*, 2018.

[SGAL22] So Seng, Joaquin Garcia-Alfaro, and Youssef Laarouchi. Why Anomaly-Based Intrusion Detection Systems Have Not Yet Conquered the Industrial Market? In *Proceedings of the 14th International Symposium on Foundations and Practice of Security (FPS)*, volume 13291, pages 341–354, 2022.

[SGLG17] Iman Sharafaldin, Amirhossein Gharib, Arash Habibi Lashkari, and Ali A Ghorbani. Towards a reliable intrusion detection benchmark dataset. *Software Networking*, 2017(1):177–200, 2017.

[SH24] Asger C. Schliemann-Haug. Alarm Management in the Maritime Industry. `https://web.archive.org/web/20241120003106/https://safety4sea.com/wp-content/uploads/2024/09/LR-Alarm-Management-Report-Spreads-Vol.1-2024_09.pdf`, 2024. [Online, accessed 2024-11-20].

[SHH+21] Martin Serror, Sacha Hack, Martin Henze, Marko Schuba, and Klaus Wehrle. Challenges and Opportunities in Securing the Industrial Internet of Things. *IEEE Transactions on Industrial Informatics*, 17(5):2985–2996, 2021.

[SHK+24] Jon-Martin Storm, Siv Hilde Houmb, Pallavi Kaliyar, Laszlo Erdodi, and Janne Merete Hagen. Testing commercial intrusion detection systems for industrial control systems in a substation hardware in the loop testlab. *Electronics*, 13(1), 2024.

[Shu11] Martyn Shuttleworth. Heron's Inventions. `https://web.archive.org/web/20241123143708/https://explorable.com/heron-inventions`, 2011. [Online, accessed 2024-11-23].

[SK25] Franka Schuster and Hartmut König. Questioning the Myth: Investigating ICS Traffic Homogeneity from an Anomaly Detection Perspective. In *Proceedings of the 19th International Conference on Critical Information Infrastructures Security (CRITIS)*, pages 21–42, 2025.

[SLHG19] Iman Sharafaldin, Arash Habibi Lashkari, Saqib Hakak, and Ali A. Ghorbani. Developing realistic distributed denial of service (ddos) attack dataset and taxonomy. In *Proceedings of the 2019 International Carnahan Conference on Security Technology (ICCST)*, pages 1–8, 2019.

[SLP+13] Keith Stouffer, Suzanne Lightman, Victoria Pillitteri, Marshall Abrams, and Adam Hahn. Guide to Industrial Control Systems (ICS) Security. *NIST special publication*, 800(82r3), 2013.

[SLYK20] Hyeok-Ki Shin, Woomyo Lee, Jeong-Han Yun, and HyoungChun Kim. HAI 1.0: HIL-based Augmented ICS Security Dataset. In *Proceedings of the 13th USENIX Workshop on Cyber Security Experimentation and Test (CSET)*, 2020.

[Smi16] J. Smith. ICS-pcap. `https://web.archive.org/web/20221207213607/https://github.com/automayt/ICS-pcap/blob/master/Additional%20Captures/4SICS-GeekLounge-151022/4SICS-GeekLounge-151022.pcap`, 2016. [Online, accessed 2022-12-07].

[SMLC21] Ruimin Sun, Alejandro Mera, Long Lu, and David Choffnes. SoK: Attacks on Industrial Control Logic and Formal Verification-Based Defenses. In *Proceedings of the 2021 IEEE European Symposium on Security and Privacy (EuroS&P)*, pages 385–402, 2021.

[SMRM20] Subin Sapkota, A K M Nuhil Mehdy, Stephen Reese, and Hoda Mehrpouyan. FALCON: Framework for Anomaly Detection in Industrial Control Systems. *Electronics*, 9(8), 2020.

[SOL02] SOLAS Chapter V – 1/7/02. Safety of Navigation, 2002. IMO.

[Som03] Robin Sommer. Bro: An open source network intrusion detection system. In *Security, E-learning, E-Services, 17. DFN-Arbeitstagung über Kommunikationsnetze*, pages 273–288, 2003.

[SP10] Robin Sommer and Vern Paxson. Outside the closed world: On using machine learning for network intrusion detection. In *Proceedings of the IEEE symposium on security and privacy (SP)*, pages 305–316, 2010.

[SR18] Omer Sagi and Lior Rokach. Ensemble learning: A survey. *WIREs Data Mining and Knowledge Discovery*, 8(4), 2018.

[SRFM20] Boris Sviličić, Igor Rudan, Vlado Frančić, and Djani Mohović. Towards a Cyber Secure Shipboard Radar. *Journal of Navigation*, 73(3), 2020.

[SSA+22] Rahul Anand Sharma, Ishan Sabane, Maria Apostolaki, Anthony Rowe, and Vyas Sekar. Lumen: a framework for developing and evaluating ML-based IoT network anomaly detection. In *Proceedings of the 18th International Conference on Emerging Networking EXperiments and Technologies (CoNEXT)*, page 59–71, 2022.

[SSR19] Maneet Singh, Richa Singh, and Arun Ross. A comprehensive overview of biometric fusion. *Information Fusion*, 52:187–205, 2019.

[Sur21] Open Information Security Foundation (OISF). Suricata. `https://web.archive.org/web/20241202100515/https://suricata.io/`, 2021. [Online, accessed 2024-12-02].

[SWL+21] Yongtao Shui, Yu Wang, Yu Li, Yongzhi Shan, Naigang Cui, and Baojun Pang. Consensus-Based Distributed Target Tracking with False Data Injection Attacks over Radar Network. *Applied Sciences*, 11(10), 2021.

[SWWB24] Antoine Saillard, Konrad Wolsing, Klaus Wehrle, and Jan Bauer. Exploring Anomaly Detection for Marine Radar Systems. In *Proceedings of the 10th Workshop on the Security of Industrial Control Systems & of Cyber-Physical Systems (CyberICPS), co-located with the 29th European Symposium on Research in Computer Security (ESORICS)*, pages 361–381, 2024.

[TETC20] Federico Turrin, Alessandro Erba, Nils Ole Tippenhauer, and Mauro Conti. A Statistical Analysis Framework for ICS Process Datasets. In *Proceedings of the 2020 Joint Workshop on CPS&IoT Security and Privacy (CPSIOTSEC)*, page 25–30, 2020.

[TGT+18] Riccardo Taormina, Stefano Galelli, Nils Ole Tippenhauer, Elad Salomons, Avi Ostfeld, Demetrios G. Eliades, Mohsen Aghashahi, Raanju Sundararajan, Mohsen Pourahmadi, M. Katherine Banks, B. M. Brentan, Enrique Campbell, G. Lima, D. Manzi, D. Ayala-Cabrera, M. Herrera, I. Montalvo, J. Izquierdo, E. Luvizotto, Sarin E. Chandy, Amin Rasekh, Zachary A. Barker, Bruce Campbell, M. Ehsan Shafiee, Marcio Giacomoni, Nikolaos Gatsis, Ahmad Taha, Ahmed A. Abokifa, Kelsey Haddad, Cynthia S. Lo, Pratim Biswas, M. Fayzul Pasha, Bijay Kc, Saravanakumar Lakshmanan Somasundaram, Mashor Housh, and Ziv Ohar. Battle of the Attack Detection Algorithms: Disclosing Cyber Attacks on Water Distribution Networks. *Journal of Water Resources Planning and Management*, 144(8), 2018.

[THM+22] Kimberly Tam, Rory Hopcraft, Kemedi Moara-Nkwe, Juan Palbar Misas, Wesley Andrews, Avanthika Vineetha Harish, Pablo Giménez, Tom Crichton, and Kevin Jones. Case Study of a Cyber-Physical Attack Affecting Port and Ship Operational Safety. *Journal of Transportation Technologies*, 12(1), 2022.

[TJ19a] Kimberly Tam and Kevin Jones. Factors Affecting Cyber Risk in Maritime. In *Proceedings of the International Conference on Cyber Situational Awareness, Data Analytics And Assessment (Cyber SA)*, pages 1–8, 2019.

[TJ19b] Kimberly Tam and Kevin Jones. MaCRA: a model-based framework for maritime cyber-risk assessment. *WMU Journal of Maritime Affairs*, 18, 2019.

[TLZ+18] Nesime Tatbul, Tae Jun Lee, Stan Zdonik, Mejbah Alam, and Justin Gottschlich. Precision and Recall for Time Series. In *Proceedings of the 32nd Conference on Neural Information Processing Systems (NeurIPS)*, 2018.

[TPSJ12] André Teixeira, Daniel Pérez, Henrik Sandberg, and Karl Henrik Johansson. Attack Models and Scenarios for Networked Control Systems. In *Proceedings of the 1st International Conference on High Confidence Networked Systems (HiCoNS)*, pages 55–64, 2012.

[TS08] Patrick P. Tsang and Sean W. Smith. YASIR: A low-latency, high-integrity security retrofit for legacy SCADA systems. In *Proceedings of the IFIP TC 11 23rd International Information Security Conference (SEC)*, pages 445–459, 2008.

[TS10] Lisa Torrey and Jude Shavlik. Transfer learning. In *Handbook of research on machine learning applications and trends: algorithms, methods, and techniques*, pages 242–264. IGI global, 2010.

[UGC+16] David I. Urbina, Jairo A Giraldo, Alvaro A. Cardenas, Nils Ole Tippenhauer, Junia Valente, Mustafa Faisal, Justin Ruths, Richard Candell, and Henrik Sandberg. Limiting the impact of stealthy attacks on industrial control systems. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security (CCS)*, pages 1092–1105, 2016.

[UHH+21] Rafael Uetz, Christian Hemminghaus, Louis Hackländer, Philipp Schlipper, and Martin Henze. Reproducible and Adaptable Log Data Generation for Sound Cybersecurity Experiments. In *Proceedings of the 37th Annual Computer Security Applications Conference (ACSAC)*, page 690–705, 2021.

[UHH+24] Rafael Uetz, Marco Herzog, Louis Hackländer, Simon Schwarz, and Martin Henze. You Cannot Escape Me: Detecting Evasions of SIEM Rules in Enterprise Networks. In *Proceedings of the 33rd USENIX Security Symposium (USENIX Sec)*, pages 5179–5196, 2024.

[UJJM22] Muhammad Azmi Umer, Khurum Nazir Junejo, Muhammad Taha Jilani, and Aditya P. Mathur. Machine learning for intrusion detection in industrial control systems: Applications, challenges, and recommendations. *International Journal of Critical Infrastructure Protection (IJCIP)*, 38, 2022.

[UMJA20] Muhammad Azmi Umer, Aditya Mathur, Khurum Nazir Junejo, and Sridhar Adepu. Generating Invariants Using Design and Data-Centric Approaches for Distributed Attack Detection. *International Journal of Critical Infrastructure Protection (IJCIP)*, 28:100341, 2020.

[UMZS21] Darshana Upadhyay, Jaume Manero, Marzia Zaman, and Srinivas Sampalli. Intrusion Detection in SCADA Based Power Grids: Recursive Feature Elimination Model With Majority Vote Ensemble Algorithm. *IEEE Transactions on Network Science and Engineering*, 8(3):2559–2574, 2021.

[Vas23] Christian Vasquez. Did someone really hack into the Oldsmar, Florida, water treatment plant? New details suggest maybe not. `https://web.archive.org/web/20241205235024/https://cyberscoop.com/water-oldsmar-incident-cyberattack/`, 2023. [Online, accessed 2024-12-05].

[VC09] Alfonso Valdes and Steven Cheung. Communication Pattern Anomaly Detection in Process Control Systems. In *IEEE Conference on Technologies for Homeland Security*, pages 22–29, 2009.

[vRS+22] Merlin von Rechenberg, Nina Rößler, Mari Schmidt, Konrad Wolsing, Florian Motz, Michael Bergmann, Elmar Padilla, and Jan Bauer. Guiding Ship Navigators through the Heavy Seas of Cyberattacks. In *Proceedings of the European Workshop on Maritime Systems Resilience and Security (MARESEC)*, 2022.

[VVKK04] Fredrik Valeur, Giovanni Vigna, Christopher Kruegel, and Richard A. Kemmerer. A Comprehensive Approach to Intrusion Detection Alert Correlation. *IEEE Transactions on Dependable and Secure Computing (TDSC)*, 1(3):146–169, 2004.

[vWW+25] Christian van Sloun, Vincent Woeste, Konrad Wolsing, Jan Pennekamp, and Klaus Wehrle. Detecting Ransomware Despite I/O Overhead: A Practical Multi-Staged Approach. In *Proceedings of the 22nd Annual Network and Distributed System Security Symposium (NDSS)*, 2025.

[WAM20] Herman Wijaya, Maurício Aniche, and Aditya Mathur. Domain-Based Fuzzing for Supervised Learning of Anomaly Detection in Cyber-Physical Systems. In *Proceedings of the 42nd International Conference on Software Engineering Workshops (ICSEW)*, pages 237–244, 2020.

[Wan09] Jie Wang. *The Art of Intrusion Detection*, pages 317–347. Springer, 2009.

[WBH22] Eric Wagner, Jan Bauer, and Martin Henze. Take a Bite of the Reality Sandwich: Revisiting the Security of Progressive Message Authentication Codes. In *Proceedings of the 15th ACM Conference on Security and Privacy in Wireless and Mobile Networks (WiSec)*, page 207–221, 2022.

[WBWS25] Eric Wagner, Lennart Bader, Konrad Wolsing, and Martin Serror. Sherlock: A Dataset for Process-aware Intrusion Detection Research on Power Grid Networks. In *Proceesings of the 15th ACM Conference on Data and Application Security and Privacy (CODASPY)*, pages 419–424, 2025.

[Wei16] Joe Weiss. *Aurora generator test.* CRC Press, 2016.

[WKW+24] Konrad Wolsing, Dominik Kus, Eric Wagner, Jan Pennekamp, Klaus Wehrle, and Martin Henze. One IDS Is Not Enough! Exploring Ensemble Learning for Industrial Intrusion Detection. In *Proceedings of the 28th European Symposium on Research in Computer Security (ESORICS)*, pages 102–122, 2024.

[WMV20] Hilde JP Weerts, Andreas C Mueller, and Joaquin Vanschoren. Importance of tuning hyperparameters of machine learning algorithms, 2020.

[WRBW22] Konrad Wolsing, Linus Roepert, Jan Bauer, and Klaus Wehrle. Anomaly Detection in Maritime AIS Tracks: A Review of Recent Approaches. *Journal of Marine Science and Engineering (JMSE)*, 10(1), 2022.

[WRW+23] Eric Wagner, Nils Rothaug, Konrad Wolsing, Lennart Bader, Martin Henze, and Klaus Wehrle. Retrofitting Integrity Protection into Unused Header Fields of Legacy Industrial Protocols. In *Proceedings of the 48th IEEE Conference on Local Computer Networks (LCN)*, pages 1–9, 2023.

[WSB+22] Konrad Wolsing, Antoine Saillard, Jan Bauer, Eric Wagner, Christian van Sloun, Ina Berenice Fink, Mari Schmidt, Klaus Wehrle, and Martin Henze. Network Attacks Against Marine Radar Systems: A Taxonomy, Simulation Environment, and Dataset. In *Proceedings of the 47th Conference on Local Computer Networks (LCN)*, pages 114–122, 2022.

[WSJ+18] Jingyi Wang, Jun Sun, Yifan Jia, Shengchao Qin, and Zhiwu Xu. Towards 'verifying' a water treatment system. In *Formal Methods*, pages 73–92, 2018.

[WTvS+22] Konrad Wolsing, Lea Thiemt, Christian van Sloun, Eric Wagner, Klaus Wehrle, and Martin Henze. Can Industrial Intrusion Detection Be SIMPLE? In *Proceedings of the 27th European Symposium on Research in Computer Security (ESORICS)*, pages 574–594, 2022.

[WWB+24] Konrad Wolsing, Eric Wagner, Frederik Basels, Patrick Wagner, and Klaus Wehrle. Deployment Challenges of Industrial Intrusion Detection Systems. In *Proceedings of the 10th Workshop on the Security of Industrial Control Systems & of Cyber-Physical Systems (CyberICPS), co-located with the 29th European Symposium on Research in Computer Security (ESORICS)*, pages 453–473, 2024.

[WWH20] Konrad Wolsing, Eric Wagner, and Martin Henze. Poster: Facilitating Protocol-independent Industrial Intrusion Detection Systems. In *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security (CCS)*, pages 2105–2107, 2020.

[WWL+25] Konrad Wolsing, Eric Wagner, Luisa Lux, Martin Henze, and Klaus Wehrle. GeCos Replacing Experts: Generalizable and Comprehensible Industrial Intrusion Detection. In *Proceedings of the 34rd USENIX Security Symposium (USENIX Security)*, 2025.

[WWLQ20] Chao Wang, Bailing Wang, Hongri Liu, and Haikuo Qu. Anomaly Detection for Industrial Control System Based on Autoencoder Neural Network. *Wireless Communications and Mobile Computing (WCMC)*, 2020.

[WWSH22] Konrad Wolsing, Eric Wagner, Antoine Saillard, and Martin Henze. IPAL: Breaking up Silos of Protocol-dependent and Domain-specific Industrial Intrusion Detection Systems. In *Proceedings of the 25th International Symposium on Research in Attacks, Intrusions and Defenses (RAID)*, pages 510–525, 2022.

[WZR20] Patrick Wiener, Philipp Zehnder, and Dominik Riemer. Managing geo-distributed stream processing pipelines for the IIoT with StreamPipes edge extensions. In *Proceedings of the 14th ACM International Conference on Distributed and Event-based Systems (DEBS)*, pages 165–176, 2020.

[XAY21] Qinghua Xu, Shaukat Ali, and Tao Yue. Digital Twin-Based Anomaly Detection in Cyber-Physical Systems. In *Proceedings of the 14th Conference on Software Testing, Verification and Validation (ICST)*, pages 205–216, 2021.

[XWWT20] Xin Xie, Bin Wang, Tiancheng Wan, and Wenliang Tang. Multivariate Abnormal Detection for Industrial Control Systems Using 1D CNN and GRU. *IEEE Access*, 8:88348–88359, 2020.

[YB09] Mo Yilin and Sinopoli Bruno. Secure Control Against Replay Attacks. In *Proceedings of the 47th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pages 911–918, 2009.

[YC14] Man-Ki Yoon and Gabriela F Ciocarlie. Communication pattern monitoring: Improving the utility of anomaly detection for industrial control systems. In *Proceedings of the NDSS workshop on Security of Emerging Network Technologies (SENT)*, pages 1–10, 2014.

[YFZ+18] Chaoqun Yang, Li Feng, Heng Zhang, Shibo He, and Zhiguo Shi. A Novel Data Fusion Algorithm to Combat False Data Injection Attacks in Networked Radar Systems. *IEEE Transactions on Signal and Information Processing over Networks (TSPIN)*, 4(1):125–136, 2018.

[YHC+20] Zeyu Yang, Liang He, Peng Cheng, Jiming Chen, David K.Y. Yau, and Linkang Du. PLC-Sleuth: Detecting and Localizing PLC Intrusions Using Control Invariants. In *International Symposium on Recent Advances in Intrusion Detection (RAID)*, pages 333–348, 2020.

[YHL+18] Jeong-Han Yun, Yoonho Hwang, Woomyo Lee, Hee-Kap Ahn, and Sin-Kyu Kim. Statistical Similarity of Critical Infrastructure Network Traffic Based on Nearest Neighbor Distances. In *Research in Attacks, Intrusions, and Defenses*, pages 577–599, 2018.

[YKP+23] Abbas Yazdinejad, Mostafa Kazemi, Reza M Parizi, Ali Dehghantanha, and Hadis Karimipour. An ensemble deep learning model for cyber threat hunting in industrial internet of things. *Digital Communications and Networks*, 9(1):101–110, 2023.

[YKSA19] Hyunguk Yoo, Sushma Kalle, Jared Smith, and Irfan Ahmed. Overshadow PLC to Detect Remote Control-Logic Injection Attacks. In *Proceedings of the 16th International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment (DIMVA)*, pages 109–132, 2019.

[YXG+17] Yi Yang, Hai-Qing Xu, Lei Gao, Yu-Bo Yuan, Kieran McLaughlin, and Sakir Sezer. Multidimensional Intrusion Detection System for IEC 61850-Based SCADA Networks. *IEEE Transactions on Power Delivery*, 32(2):1068–1078, 2017.

[Zeek] The Zeek Project. Zeek. `http://web.archive.org/web/20240108132036/https://zeek.org/`, 2024. [Online, accessed 2024-01-08].

[Zho09] Jianying Zhou. Top Cyber Security Conferences Ranking. `https://web.archive.org/web/20250126132325/http://jianying.space/conference-ranking.html`, 2009. [Online, accessed 2025-01-26].

[Zho21] Zhi-Hua Zhou. *Machine Learning*. Springer, 1st edition, 2021.

[ZHW+22] Sven Zemanek, Immanuel Hacker, Konrad Wolsing, Eric Wagner, Martin Henze, and Martin Serror. PowerDuck: A GOOSE Data Set of Cyberattacks in Substations. In *Proceedings of the 15th Workshop on Cyber Security Experimentation and Test (CSET)*, page 49–53, 2022.

[ZHX+15] Chunjie Zhou, Shuang Huang, Naixue Xiong, et al. Design and Analysis of Multimodel-Based Anomaly Intrusion Detection Systems in Industrial Process Automation. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 45(10):1345–1360, 2015.

[ZLZS16] Feng Zhang, Min Liu, Zhuo Zhou, and Weiming Shen. An IoT-Based Online Monitoring System for Continuous Steel Casting. *IEEE Internet of Things Journal*, 3(6):1355–1363, 2016.

[ZM12] Cha Zhang and Yunqian Ma. *Ensemble Machine Learning: Methods and Applications*. Springer, 1st edition, 2012.

[ZWF+21] Dan Zhang, Qing-Guo Wang, Gang Feng, Yang Shi, and Athanasios V. Vasilakos. A survey on attack detection, estimation and control of industrial cyber–physical systems. *ISA Transactions*, 116:1–16, 2021.

[ZWSR20] Philipp Zehnder, Patrick Wiener, Tim Straub, and Dominik Riemer. StreamPipes Connect: Semantics-Based Edge Adapters for the IIoT. In *The Semantic Web*, pages 665–680, 2020.

Abbreviations and Acronyms

PLC Programmable Logic Controller 3
PoC Proof-of-Concept 156
PPI Plot Position Indicator 148
radar radio detection and ranging 145
RAT Radar Attack Tool x
RF Random Forest 24
RMSE Root Mean Square Error 161
SCADA Supervisory Control and Data Acquisition 10
SIMPLE Sufficient, Independent, Meaningful, Portable, Local, and Efficient 104
SLR Systematic Literature Review 43
SMS Systematic Mapping Study iii
SOLAS International Convention for the Safety of Life at Sea 148
SVM Support Vector Machine 24
TABOR Timed Automata and Bayesian netwORk 93
TEP Tennessee Eastman Process 92
TPA True-Positive Alarm 36
TTD Time To Detection 160
VSAT Very-Small-Aperture Terminal 145