Journal of
Cybersecurity and Privacy

Article

Generic Patterns for Intrusion Detection Systems in Service-Oriented Automotive and Medical Architectures

Andreas Puder [1,†], Marcel Rumez [2,†], Daniel Grimm [2,†] and Eric Sax [2,*]

1 Embedded Systems, Getinge AB, 76437 Rastatt, Germany
2 Institute for Information Processing Technologies (ITIV), Karlsruhe Institute of Technology (KIT), 76131 Karlsruhe, Germany
* Correspondence:eric.sax@kit.edu
† These authors contributed equally to this work.

Abstract: To implement new software functions and more flexible updates in the future as well as to provide cloud-based functionality, the service-oriented architecture (SOA) paradigm is increasingly being integrated into automotive electrical and electronic architectures (E/E architectures). In addition to the automotive industry, the medical industry is also researching SOA-based solutions to increase the interoperability of devices (vendor-independent). The resulting service-oriented communication is no longer fully specified during design time, which affects information security measures. In this paper, we compare different SOA protocols for the automotive and medical fields. Furthermore, we explain the underlying communication patterns and derive features for the development of an SOA-based Intrusion Detection System (IDS).

Keywords: information security; medical & automotive architectures; networks; service-oriented architecture (SOA); Intrusion Detection System (IDS)

check for updates

Citation: Puder, A.; Rumez, M.; Grimm, D.; Sax E. Generic Patterns for Intrusion Detection Systems in Service-Oriented Automotive and Medical Architectures. *J. Cybersecur. Priv.* **2022**, *2*, 731–749. https://doi.org/10.3390/jcp2030037

Academic Editor: Dimitris Gritzalis

Received: 31 July 2022
Accepted: 9 September 2022
Published: 14 September 2022

1. Introduction

The digitization and networking of technical devices with each other and with the Internet is constantly increasing, which is shown by, e.g., the number of Internet of Things (IoT) devices growing by 22% in 2022 [1]. IoT devices, which have multiple sensors, e.g., for light, voice, and temperature, make everyday life more comfortable. Beyond sensing the environment, Cyber-Physical Systems (CPSs) can influence their environment using actuators and software. Such connected embedded systems are becoming more and more established in various areas, for example, smart grids, industrial process control, medical devices, robots, or vehicles [2]. Through networking and increasing amounts of software, simple embedded systems that solve dedicated individual tasks are becoming intelligent and autonomous systems in many areas. Examples of this are automated driving, surgical robots, flexible production plants, or energy management systems. Even if the devices of different industries have different networking standards, protocols, and basic software (e.g., operating systems), there are some commonalities concerning the core challenges in development. Especially, the design of the CPS's software architecture is a challenge. A design paradigm that is therefore becoming increasingly popular is the service-oriented architecture (SOA) [3]. Each function is a self-contained service with clearly defined interfaces, and the implementation is considered a black box from the external perspective [4]. Larger functions are developed by combining ("orchestrating") several services, which increases the degree of reuse of the software.

A common factor of CPSs is the risk of cyber-attacks [5]. In addition to functional safety (protecting the environment from system failures), protecting systems from unauthorized access is a significant challenge. Especially in the case of medical devices, cyber-attacks can result in direct physical damage. As more and more medical devices are connected with each other and will become part of the Internet of Medical Things (IoMT) [6], the risk of unauthorized access increases. Healthcare data in particular are valuable, with cybercriminals stealing 707 million records worldwide in 2015 alone, and the healthcare sector accounted for 23% of attacks and nearly one-fifth of all data captured [7]. However, also in production plants or vehicles, human damage by vicious attackers should be taken into account. The best-known example in the vehicle sector is the so-called Jeep Hack [8], in which Miller and Valasek gained access to the vehicle controls miles away via a fake mobile network. In the case shown, 1.4 million vehicles could theoretically have been driven into the road ditch remotely.

The research for security measures for CPSs has gained interest in recent years. Especially measures that are well-known from traditional IT have been introduced to CPS research, e.g., firewalls, IDS, authentication, or encryption [5]. This is partially caused by CPS networks becoming more similar to IT networks. In the automotive [9] and medical sectors, for example, Ethernet technology was introduced in addition to the legacy bus systems, such as Controller Area Network (CAN).

Another typical approach of medical device manufacturers as well as the automotive industry was "security by obscurity", which means the usage of proprietary security measures and protocols. This has been shown to be not sufficient, as adversaries always will be able to break into such systems [10]. Especially for the automotive domain, reverse engineering to understand and break proprietary protocols and software is possible for each and every car, because cars and car parts are freely accessible on the market. An approach, that is viable for every system, is to limit the functionality that can be invoked via the network interface [11]. However, according to Teber et al. [12], especially future robotic surgical devices will have to compete on their supported connectivity interfaces for information exchange and integrate the information of other devices. This is also applicable for other CPSs.

In the early days of IT security, approaches such as *Demilitarized Zones* that segregate internal network hosts from the ones that need to communicate to the Internet via one or multiple firewalls were popular [13]. In this case, the network traffic from external hosts is untrusted, whereas internal traffic is trusted. Thus, IT security evolved toward a zero trust approach, where the entire traffic has to be authenticated [13]. Especially, IDS with an anomaly-based approach are viable, since they are able to detect previously unknown (*zero-day*) attacks [14].

IDS from IT, e.g., the well-known open-source solution *Snort*, lack a detailed analysis of CPS-specific protocols and applications. When it comes to analyzing the communication of an SOA, the capabilities of previous approaches are limited [15] (see also Section 2.4). Thus, our work focuses on the development of an IDS for the monitoring of CPSs that employ an SOA. To give the reader a more intuitive understanding of the generic approach outlined in the following, we apply our idea to two representative CPSs: an automotive SOA and a medical device SOA.

Problem: As service-oriented communication is no longer fully specified during design time and participants can enter and leave networks dynamically, it is increasingly difficult to determine rules to maintain the system security. Moreover, the flexibility that the paradigm is supposed to provide is accompanied by a loss of security, since the static communication relationships of traditional signal-based networks are absent.

Moreover, according to the review of Network Intrusion Detection System (NIDS) by Ahmad et al., research in CPSs is still in its early stages, although interest has increased massively in recent years [16]. Most of today's IDSs focus on traditional IT computer networks and therefore do not consider the dimension of the physical environment of networked CPSs.

In addition, despite the similar trend in several CPS domains to use SOA approaches, the underlying middlewares and protocol standards (see Section 2) are different and incompatible with each other. When using traditional IDSs, which often work directly on the protocol standard, they are strictly dependent on the underlying technology. Therefore, synergies between the different domains are hard to achieve.

Contribution: We propose a first step toward a generic IDS based on the communication standards Data Distribution Service (DDS), Scalable service-Oriented MiddlewarE over IP (SOME/IP) and Service-oriented Devices Connectivity (SDC) that can be adapted to other service-oriented middlewares such as Open Platform Communications Unified Architecture (OPCUA), which is widely used in automation engineering [17]. Furthermore, we chose a distributed approach using an NIDS that also collects data from Host Intrusion Detection System (HIDS) sensors of the different participants. Therefore, we identify different anomaly patterns which could occur in such a set of networked devices and apply them to a medical and an automotive use case. As shown in Section 2.4, current IDSs lack the necessary features for CPSs-specific protocols and applications. The capabilities of the already reported approaches are too limited to analyze the communication of an SOA for IDSs, especially for CPSs

Outline: First, we introduce the SOA paradigm in Section 2.1 and explain the background for service-oriented communication in the automotive as well as the medical field in Sections 2.2 and 2.3. We then examine the communication patterns common to the standards in question and present related work for both domains in Sections 2.5 and 2.6. Based on these patterns, we propose a generic SOA IDS in Section 3 on the basis of a medical and an automotive use case followed by a prototypical implementation in Section 4, and we summarize our results in Section 5.

2. Background and Related Work

2.1. Service-Oriented Architectures

In the SOA paradigm, distributed capabilities are organized by a unified concept called services. With the help of these, capabilities are made available and discoverable to each other. Therefore, service providers offer these services while service consumers are using them [18]. In the automotive setting, sensors, actuators, and computation entities provide and consume services. Computation can take place on an Electronic Control Units (ECUs) in the vehicle or on servers outside the car. The latter requires a reliable Wi-Fi or cellular network connection.

Since the entities in the domains considered here are software/hardware systems, SOA is implemented through communication protocols. Thus, the protocol uniformly regulates how data on offering, discovery and interaction are exchanged via a network connection. In both cases, networks are increasingly based on Ethernet topologies. However, the protocols used in the automotive field (see Section 2.3) and medical environment (see Section 2.2) differ.

In practice, most of the devices will provide as well as consume services, and therefore, a strict differentiation between both roles in a Operating Room (OR) setting and in the automotive domain will not be executed. Furthermore, both application domains of SOA have in common that the entities must meet special safety requirements, since malfunctions in the overarching system behavior could lead to life-threatening consequences.

2.2. Medical

In the medical field, the architecture comprises a large variety of self-contained devices that may be required in a surgical setting. In Figure 1, an example is depicted, which represents a use case for the OR. It is based on the concept of an Hybrid Operating Room (HOR) [19], which is already a state-of-the-art application based on proprietary protocols and signal-based communication. The aim of the HOR is to enable imaging techniques during a surgery in a single OR. Therefore, the angiography system, a C-shaped device for interoperative imaging with X-ray technology (Figure 1), requests information from the Operating Room Table (OR table) (e.g., joint positions) via an external interface to move around the patient without collision.

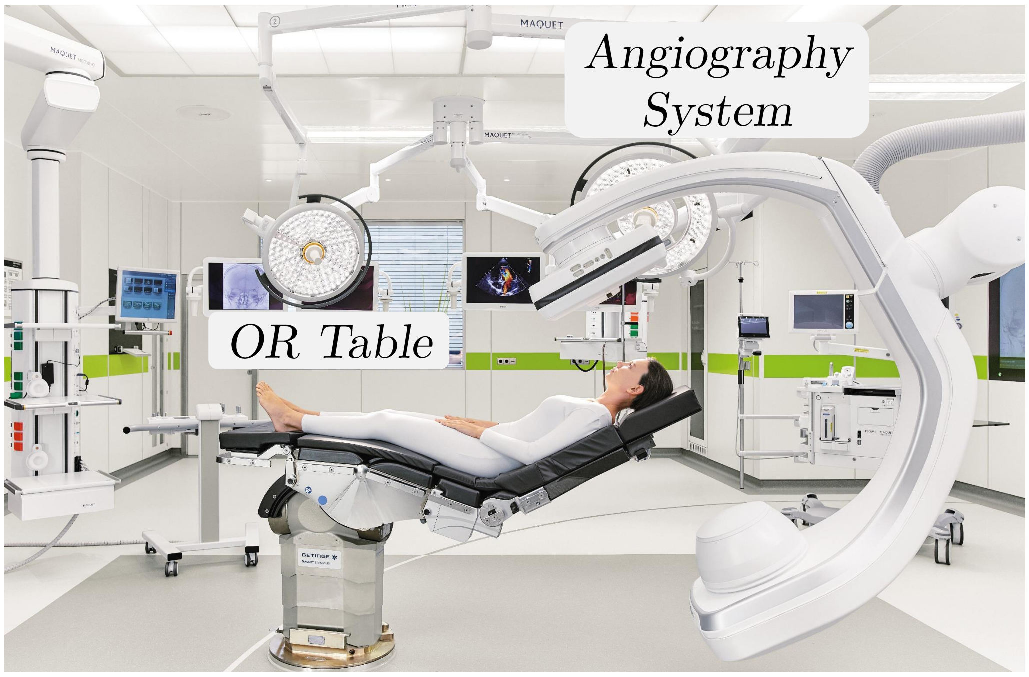

Figure 1. Image guided surgery in the HOR [20].

In [21], this scenario is also transferred to a service-oriented approach. The OR table can act as a service provider that can be moved in different directions, or an angiography system can provide a service that delivers X-ray images of the patient.

Figure 2 depicts the service-oriented communication of an angiography system with an OR table. Here, the angiography system uses the provided interface of the OR table to determine the current position of the OR table via the service *GetPosition* or to move the OR table into a desired position via the service *Move*. Thus, the angiography system acts as a service consumer, while the OR table represents the service provider.

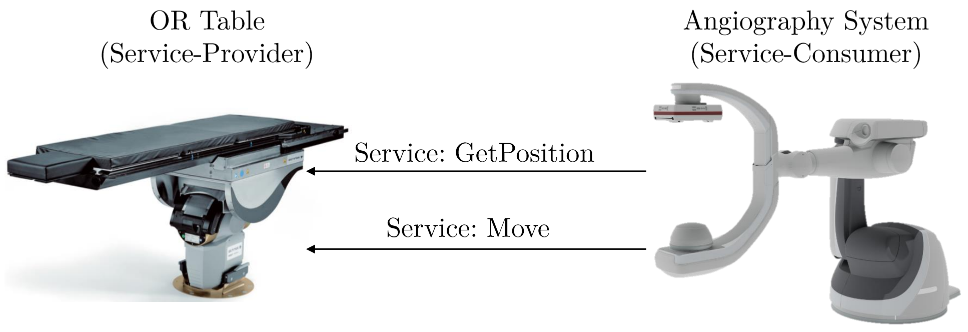

Figure 2. Example for service provider and consumer in the HOR (right picture inspired by [22], left picture from [23]).

There also have been several efforts by different medical device manufacturers to create a platform to support the interoperability of medical devices within the OR: Karl Storz (OR1™), Stryker (i-Suite™), Getinge (Tegris™), Olympus (EndoALPHA™), BrainLab (Brainsuite™) and Richard WOLF (core nova™). However, because these solutions are not based on open standards, the security measures and network protocols are proprietary. As stated in Section 1, *security by obscurity* does not provide any relevant security value. Therefore, manufacturers cannot be accused of not securing their systems sufficiently. Rather, the cooperation of multiple manufacturers is necessary to ensure the secure and safe communication of all medical devices in the OR.

Promising standards to enable interoperability for medical devices in the OR based on SOA have been started in the last two decades and are still ongoing:

Medical Device Plug and Play (MDPnP): The MDPnP program is developing integrated clinical systems and began as an offshoot of the *operating room of the future* at Massachusetts General Hospital. It was launched as an ORF-PnP initiative based on a symposium held back in May 2004 at CIMIT in Cambridge, MA [24]. With the MDPnP program, the ASTM F2761 Integrated Clinical Environment (ICE) standard [25] has been developed. Furthermore, with OpenICE, an open source reference implementation of this standard is available also relying on DDS (see Section 2.3) as middleware [26].

Service-oriented Device Connectivity (SDC): Since 2012, the German association OR-NET has been working on the ISO/IEEE 11073 family of communication standards to realize open integration in the OR of the future [27]. In contrast to other similar international aspirations, it already reached the level of a technical specification and standard [28]. The SDC standard is based on Devices Profile for Web Services (DPWS) and uses Hypertext Transfer Protocol (HTTP)/Hypertext Transfer Protocol Secure (HTTPS), Transmission Control Protocol (TCP) and User Datagram Protocol (UDP). As it has further extensions for the medical field, it is called a Medical DPWS (MDPWS), which is described in the IEEE 11073-20702 standard [29]. DPWS is based on the SOA concept [28] and provides three message exchange patterns for device-to-device communication defined in the Basic Integrated Clinical Environment Protocol Specification (BICEPS) [30]: publish–subscribe when a service provider pushes information to a service consumer (also for asynchronous communication), request/response when a service consumer pulls information (also for synchronous communication) from a service provider, and streaming when a service provider consecutively pushes information to 0 or more service consumers. In addition, it also realizes mechanisms for the dynamic exploration of services during runtime: the explicit discovery, where a service consumer actively searches for a specific service, and the implicit discovery, where a service provider publishes its services in the network. According to [28], the safety and security of SDC are not sufficiently examined.

Smart Cyber Operating Theater (SCOT®): The SCOT project is a Japanese initiative to introduce interoperability in the OR to connect medical devices with computing systems. Therefore, OPeLiNK® based on the industrial middleware Open Robot/Resource interface for the Network (ORiN) is used [31]. In contrast to SDC and MDPnP, there already are systems commercially available [32], but it concentrates more on the HOR, while the other standards include all medical devices in hospitals [28]. As SCOT does not rely on an SOA, it is only relevant in the context of this paper as integration within SDC as proposed by Berger et al. [33].

2.3. Automotive

Scalable Service-oriented Middleware over IP: SOME/IP is a service-oriented communication protocol standardized by the AUTomotive Open System ARchitecture (AUTOSAR) foundation [34]. SOME/IP defines three communication principles: events, fields, and methods. Thus, any service may offer several endpoints of these principles. In general, communication is initiated by either a *find* or an *offer* message of the SOME/IP Service Discovery (SOME/IP-SD) protocol to match the communication partners in the network. All patterns are in a 1-to-N fashion. Accordingly, each client/subscriber binds to a specific service/provider, which is selected for subscription if multiple service providers are available.

The publish/subscribe pattern is mainly implemented by *events*. Clients subscribe to required information from a matching provider by using the discovery mechanism of SOME/IP-SD. The subscriber gets notified when the information changes or cyclically. The Remote Procedure Call (RPC) pattern is implemented with *methods*. Depending on whether the request should return a response or not, SOME/IP distinguishes between request/response methods and fire-forget methods. In addition, SOME/IP specifies *fields* as a special case, which is a combination of events and methods. Fields include a get and set method but at the same time a notification when the value changes.

Data Distribution Service: The DDS standard [35] was developed by the Object Management Group (OMG) and is used in various application domains, e.g., automotive applications and healthcare [36]. At its core, DDS provides the stream pattern using its Data-Centric Publish-Subscribe (DCPS) mechanism. Unlike the SOME/IP approach, this is inherently not a pattern for searching and binding to a specific service instance but rather for searching and binding to a data stream. Data are transferred via topics with specific names and data types. Subscribers and publishers identify the topic they want to link to by its name and matching data type. A discovery mechanism for participants and available topics using four predefined topics (topic names: "DCPSParticipant," "DCPSSubscription", "DCPSPublication," and "DCPSTopic.") is part of DDS. OMG also specified a RPC standard [37] via DDS. The DDS-RPC standard provides a request–reply pattern to the network participants, on top of the substantial parts of DDS. For every request–reply connection, two topics are used: a request topic and a reply topic. A special feature of DDS is that it ensures flexible and comprehensive Quality of Service (QoS) settings. In addition, virtual subnetworks can be set up using different *Domain* identifiers.

2.4. Overview of SOA Standards

Based on the communication patterns of DDS, SOME/IP and SDC, the commonalities can be identified, although the protocols realize the patterns differently. These commonalities are summarized in Table 1 for patterns for the usage of services. Furthermore, the presented communication patterns represent a selection which is supported by several SOA standards. Additionally, each standard provides patterns for the binding of services such as find, offer, subscribe, stop offer or stop subscribe, as these are the basis for an SOA standard.

Table 1. Overview of SOA operation communication patterns.

Pattern	SDC	SOME/IP	DDS
publish/subscribe (1-N)	publish/subscribe (e.g., Description Event Service, State Event Service)	event, field	/
request/response (1-1)	request/response (e.g., Get/Set Service), Context Service	request/response, fire/forget	2x Topic with DDS-RPC
stream (N-M)	streaming (e.g., Waveform Service)	/	Topic

In our comparison, we differentiate between the *publish/subscribe*, *request/response* and the *stream* pattern. In particular, the term publish/subscribe is used in different standards with different meanings. We use it to refer to communication in a 1-N channel where a particular service provider is selected by any number of clients. There can also be multiple providers, but only one server can be selected by each client at any given time. In most cases, this pattern is used to implement event-based communication, in which a notification is sent in response to status changes. The request/response pattern, on the other hand, is typically used for a method call on the server. This also includes get and set functions that retrieve or change parameters from servers. The response can also be empty or non-existent (as in the case of SOME/IP fire/forget methods). Stream services are used to consecutively push information to 0 or more service consumers provided by 1 to N publishers (see SDC Section 2.2).

Since in an SOA, theoretically, any participant can consume and also provide a service based on the communication patterns presented, securing these mechanisms is central to the security of a SOA. By analyzing abnormal behavior in the relations of the services in a network depending on their communication patterns, an intrusion by an attacker could be the source of this behavior.

2.5. Security for SOAs in Medical Field

Papaioannou et al. [6] present a survey on security threats for the IoMT. Based on a categorization of existing and potential cybersecurity threats, the authors also provide a categorization of security countermeasures. While this study focuses on medical devices, they point out that IoMT is based on, and therefore strongly related to, general IoT technology. The proposed system architecture of IoMT-enabled healthcare systems uses a user's terminal device such as a smartphone as a gateway for receiving and forwarding patient data over the Internet to a cloud service. Nevertheless, this approach does not directly include medical devices for hospitals such as OR table or cardiopulmonary machines and is thus focused on wearable or implantable medical devices.

Javdani et al. [38] motivate a service-oriented approach by focusing on security for IoMT and provide a review of studies in this field. They emphasize that in scientific research, SOA is considered as one of the best solutions for IoT and thus consequently for IoMT. Furthermore, the authors conclude that in almost all surveyed articles, the need for more research on security for SOA in this field is stressed and that an adequate solution has not been found yet.

Leucker et al. [39] present a concept based on formal interface descriptions to improve the security of medical device interconnection in an SOA. Therefore, they propose to automatically generate monitors from described interfaces that check the correct usage of an interface at runtime. An example for this is the monitoring of allowed value ranges.

Arney et al. [40] focus on challenges in securing medical device networks based on the ICE standard, which is an SOA-based middleware (see Section 2.2). Furthermore, they present a comprehensive attack and adversary model for networked medical devices and discuss the associated challenges for security measures.

Although SOA opens up manufacturer-independent interoperability, there is a lack of concepts for service-oriented communication in the medical field to secure an entire network of medical devices as in the OR. To our awareness, current solutions consider either only the interface of a device or the communication between different devices directly. A more holistic architectural approach is a gap we aim to fill here.

2.6. Security for SOAs in Automotive Domain

Today, SOAs already reached the maturity to be used in critical embedded applications [41,42]. However, within the research sector, the topic of security for automotive SOAs is relatively new. To the best of our knowledge, only a few publications address these kinds of issues. In the following, we give a brief overview of existing ones.

In 2017, a comprehensive security evaluation based on the SOME/IP protocol was conducted by Kreissl [43]. The work starts with a Threat Analysis and Risk Assessment (TARA) to uncover possible attacks within an automotive on-board communication based on the SOME/IP protocol. As a result, 18 threats have been identified. The resulting risk analysis according to HEAling Vulnerabilities to Enhance Software, Security, and Safety (HEAVENS) [44] classifies 11 threats as high and three as critical. Kreissl justifies the overall problem for the potential attack ways with the lack of security features in the SOME/IP protocol. Therefore, the author defines various use cases as well as associated security properties and discusses suitable security mechanisms.

Gehrmann et al. [45] have started to discuss challenges and opportunities regarding an IDS for SOME/IP communication in vehicles. One challenge is the changed network behavior compared to established communication protocols such as CAN. The provider and consumer can decide autonomously to transmit any messages on the network. Therefore, the network traffic is more sporadically than periodic, and it is difficult to specify a normal behavior or detect abnormal network flows. Another challenge for an IDS is the property of reconfigurability, in case one network node is replaced by another due to a failure. As a result, a new network participant appears with new address information as well as changed network flows. The authors also mention the typical resource constraints on automotive gateways and thus limited possibilities for response in case of detected anomalies in real time. However, there are also features of SOME/IP which represent unique characteristics (e.g., maximum registered SOME/IP providers or established connections on a specific service) compared to other automotive protocols.

Moreover, the researchers propose in their work the first approach for an SOME/IP IDS by using these characteristics based on a predefined security policy (see Figure 3). The policy includes for example permitted services, which a specific provider can offer. Furthermore, the system is also able to support Intrusion Prevention System (IPS) functionalities (e.g., creating logs, alert messages for backends, termination of service calls). In their final evaluation, a lightweight implementation based on one security policy is shown. On the test platform used (Intel Core i7, Linux Ubuntu), verifying the policy causes an overhead of 6 μs.

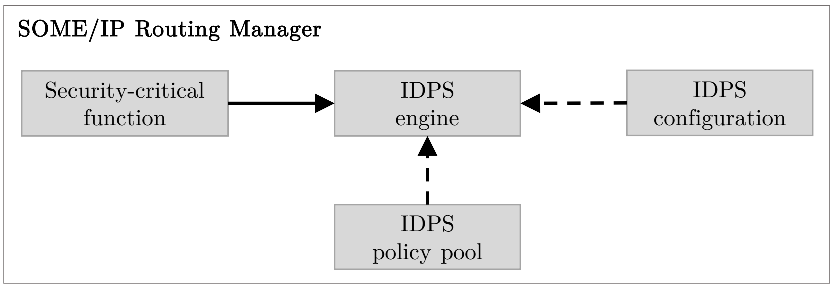

Figure 3. Intrusion Detection Prevention System (IDPS) Architecture proposed by Gehrmann et al. [45].

Iorio et al. [46] present a framework for securing the SOME/IP protocol, since the standard specification does not provide any security features. The framework defines three different security levels, which can be applied for services instances depending on their requirements regarding criticality. The first level (*nosec*) includes no specific security features but provides full compatibility with legacy services. The second level (*authentication*) comprises data authentication and integrity, which is verified by the middleware to guarantee message processing only from authorized sources. As a third level, (*confidentiality* includes all security objectives of the second level as well as the capability to encrypt payload data. To specify the permitted traffic, the framework uses a traffic matrix including high-level policies with entries such as application, service, role, or minimum security level. The subsequent evaluation includes different analyses (e.g, network throughput, timing, and computation usage). One main result is that the authentication-level causes a 15–25% overhead. The use of encryption (third security-level) further increases the computation time.

In our own work [15], we give a comprehensive overview about automotive SOAs. In detail, we describe and compare existing protocols and supported security features for on-board vehicle communication based on a hybrid (signal-based and service-oriented [47]) E/E architecture. In addition, we address specific security countermeasures such as Identity and Access Management (IAM), firewalls as well as IDS and discuss implications due to the upcoming SOA paradigm. Our previous work focused on describing the security challenges posed by the introduction of SOA but not on concrete technical solutions. In doing so, we also analyzed that new approaches are required for the implementation of network-based IDS (e.g., flow-based features, situation awareness) and, in particular, a generic approach based on the SOA-inherent communication properties is desirable. In comparison, in the present work, we start at this point with a first proposal.

2.7. Security for SOAs in Other Industries

In addition to the automotive domain and the medical field, there are other domains in the CPS area addressing SOAs and security.

El Mrabet et al. [48] present a cyber-security survey in the field of smart grids and discuss existing challenges. The authors classify 15 cyberattack techniques (e.g., traffic analysis, Denial of Service (DoS) or privacy violation) based on an attacking cycle typically used by malicious attackers (reconnaissance, scanning, exploitation, maintaining access). Furthermore, they explain three security categories (network security, cryptography for data security and device security) to map 25 known attacks to possible countermeasures. In addition, they discuss different challenges of heterogeneous systems (e.g., smart grids) due to various network protocols or limited hardware resources. The devices are mainly trimmed for low-power consumption and therefore not able to support computing-intensive security mechanisms. Another challenge is the distributed environment with different applications, domains (e.g., generation, transmission, customer, service provider) and security requirements.

Pliatsios et al. [49] present a comprehensive survey on Supervisory Control and Data Acquisition (SCADA) systems with a focus on security. First, they give an overview about established SCADA communication protocols, which is followed by a list of reported incidents. In the years 2000–2018, 15 attacks of high impact were reported. A well-known example is the *Stuxnet* worm in 2010, which targeted the sabotage of a uranium enrichment facility. Moreover, the authors discuss fundamental lacks of security (e.g., missing authentication or integrity) in SCADA protocols and provide a summary of eight published IDS approaches. Five of them are based on the traffic classification detection technique, which processes network flows. Two of eight approaches are based on a system variable inspection, which determines correlations of physical sensor values in different system states or conducts a critical state analysis. One approach uses a multilayer security framework consisting of IDS by using physical and cyber features and firewalls with predefined whitelists for filtering network packets. Moreover, the survey includes further overviews about specific security solutions for protocols such as Distributed Network Protocol 3 (DNP3), PROcess FIeld NET (PROFINET), EtherCAT and IEC 61850. As a conclusion, the authors mention open research problems and challenges such as the introduction of Software-defined Networking (SDN), which provides more networks with flexibility; however, adapted security mechanism are needed.

Baker et al. [50] mention a related challenge through the deployment of SOA-based SCADA. The corresponding services are connected by Ethernet and TCP/Internet Protocol (IP) protocols to ensure a compatibility to existing services on the web. As a result, the risk for potential cyberattacks increases due to these well-known protocols. Furthermore, the distributed architecture design of such systems across local system boundaries leads to changed security concepts compared to field networks.

2.8. Intrusion Detection Systems in Other Industries

Najjar and Azgomi [51] present a distributed IDS approach. In general, centralized IDSs can be deployed in host systems and operate locally, which are then referred to as HIDS, or there are network-based IDS, referred to as NIDS, which monitor network traffic with multiple agents on the network. The authors use a mixture of both types and also use information collected by IDS agents deployed on the server of a web service. They also apply anomaly detection mechanisms that monitor the normal behavior of a system as well as misuse detection mechanisms based on the signature of known attacks.

Carta et al. use a local feature engineering strategy [52] to create new features based on the original ones, which are computed locally in each network. These are then combined with discrete feature values to improve event characterization.

A two-stage deep learning structure for anomaly detection in NIDSs is proposed by Kao et al. [53]. As semi-supervised learning with Denoising Auto-Encoders (DAE) has good accuracy but insufficient precision, they combine it with a Gate Recurrent Unit (GRU). As a result, the precision could be improved by 0.83% compared to using a DAE alone, and also an accuracy of 90.21% has been achieved.

Jiang et al. [54] adopted a semi-supervised learning approach to improve the precision, recall, and F_1-score with their Tri-CAD anomaly detection framework. It applies to univariate time series and uses various techniques such as wavelet transform and deep learning autoencoders.

3. Application of Generic Patterns for an SOA IDS

Unlike firewalls, for example, IDSs are reactive measures in that they detect potential attacks only when they occur, which can be completed by analyzing traffic on networks with NIDSs, for instance [15]. IDSs are usually surveillance systems that detect suspicious behavior and issue alerts in case of detection. Furthermore, the detection scheme of an IDS can be divided into signature-based and anomaly-based intrusion detection [53]. According to Khalid et al., machine learning-based IDSs can help detect abnormal behavior in communication patterns [55].

Based on the patterns presented in Section 2.4, we extract features for our generic IDS approach and derive a categorization for anomaly patterns. For the following use cases, we provide approaches to detect these anomaly patterns. We assume a distributed IDS approach similar to [51]. For some applications, the amount of data generated to transmit to the central NIDS may be too large. Thus, an anomaly-based selection of transmitted data by the sensors could be applied similar to approaches presented by Hofmockel in [56].

3.1. Use Cases

3.1.1. Medical Use Case

Figure 4 shows a use case for interoperability in the OR where multiple medical devices form a network. Sample services that represent the patterns defined in Section 2.4 are also included.

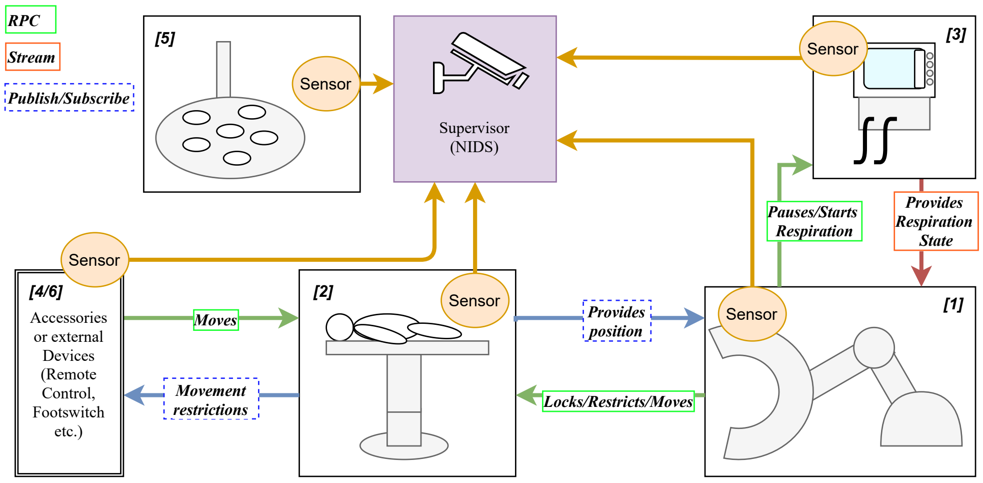

Figure 4. SDC network of different medical devices in an OR supervised by a distributed IDS.

1. **Angiography System:** The angiography system is able to move or restrict/lock the movements of an OR table by RPC to avoid collisions or to prohibit movements during an imaging process.
2. **OR Table:** The OR table is the central component in the OR on which the patient lies. It provides its position and possibly restricted movement ranges to other participants via publish/subscribe services.
3. **Ventilator:** The ventilator ventilates the patient and can be paused and restarted via an RPC to improve the imaging process by suppressing breathing movements (based on interoperability example from [57]). The information about ongoing respiration (paused or not) is streamed as a respiration state.
4. **Remote Control/Footswitch:** A connected remote control or footswitch which is either connected with the OR table via SDC or directly via a proprietary protocol. Both are capable of moving different joints via an RPC of the OR table.
5. **OR Light with Integrated Camera:** The integrated camera can be used to detect a table movement or to determine the position of the OR table (lift, longitudinal shift, etc.). Thus, the state of the OR table can be determined externally to provide more context to the IDS.

6. **External Accessory/Device:** Since OR tables have a variety of different accessories, it is conceivable in a future scenario that a device might be plugged into an OR table and use it as a gateway to the SDC-Network.

In our approach, each medical device has its own IDS sensor which is capable of determining the current state of the hosting device and detecting certain events. This additional information is then communicated to a central supervisor NIDS, which is able to provide a context of the entire OR. By using sensors in hosts, it is possible to also consider the payload of the services and the device context.

3.1.2. Automotive Use Case

Transferred to the automotive domain, related use cases can be described based on E/E architectures of vehicles. These architectures also include various actuators, sensors and associated ECUs to control vehicle dynamics as well as provide comfort functions for passengers. In comparison to OR tables, the vehicle operates within a more rapidly changing environment with partially activated automated driving functions. Different dependencies regarding communication patterns exist between the control functions involved in specific vehicle states. For example, such a state can represent whether an automated driving function such as Adaptive Cruise Control (ACC) is currently active. This results in specific communication patterns of involved functional components.

Figure 5 shows a use case for an automotive in-vehicle network, which is also observed by a distributed IDS, with different simplified service-oriented components. The foot pedals are used for accelerating or braking the car via the engine and the wheels. In addition, the distance needed by the ACC algorithm is determined by a camera system tracking the cars in front. Furthermore, the current velocity also needed as input for the ACC is provided by the wheels component.

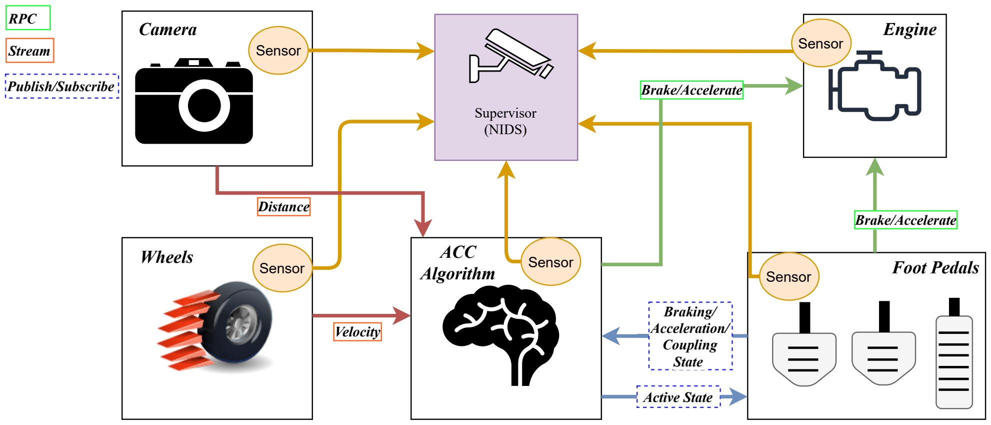

Figure 5. Simplified automotive network for different components realizing an ACC functionality supervised by a distributed IDS.

3.1.3. Differences in Use Cases

Although the architecture of the devices and components, as well as the proposed distributed IDS, is similar in both use cases, there are some differences. The in-vehicle network is rather static once the vehicle is manufactured, while in an OR environment, the devices can vary between different surgeries and even during a single surgery, e.g., when adding a high-frequency surgical device or a mobile X-ray machine [28]. Furthermore, the device composition of individual ORs also varies, and thus, the IDS must be individually configured. In contrast, the configuration effort for vehicles is lower because, apart from the different configurations, it can be applied to the entire product line. This advantage is canceled out when taking external vehicle communication into account, which is even more volatile than in the OR. In particular, the environment for the presented use cases also leads to different requirements: While it is difficult to gain physical access to a medical device in the OR due to access restrictions in hospitals, it is easier to gain physical access to the vehicle, e.g., by accessing the communication buses or manipulating sensors in public places.

3.2. Requirements for an SOA IDS Based on Generic Patterns

By analyzing the use cases and the literature mentioned above, the following are the requirements that should be considered for a generic IDS. It can be determined that the domain context is required, and thus, the service payload plays an important role, as do the device states. This domain-specific analysis can be based on a technology-independent and thus more abstract layer. Furthermore, as the literature shows, technology independence is provided using the same communication patterns.

R1 The system must use protocol independent SOA features.
R2 The system could use information contained in the specification (specification-based IDS).
R3 The system must be able to detect suspicious communication patterns dependent on defined states of the devices.
R4 The system could use a fused context information based on the states of the individual participants.
R5 The system could use the state of a single participant to detect suspicious service invocations based on the current device state.
R6 The system should be able to detect attacks by inspecting the payload.
R7 The system should detect suspicious communication patterns on the middleware-layers.
R8 The system should collect and classify possible features for the detection such as usage behavior or sensor data (current/voltage, movement information, patient/ user information).
R9 The system must be able to detect unknown attacks.

3.3. Detection of Anomaly Patterns

In the following, we describe several considerations to assist in the design of an SOA IDS. Since anomaly-based IDSs are already one of the most efficient solutions for network intrusion detection [52], we focus on anomaly detection in our generic IDS approach.

Transforming SOA communication into a generic feature set: One challenge in developing a generic approach for SOA IDSs is that the protocols used for implementing the SOA differ [R1]. Often, the *IDS approach is fitted to the protocol*. However, we propose to *fit the protocol to the IDS approach* and transform all network traffic of the different SOA implementation technologies into a generic feature set. This approach is similar to [58]. However, for SOA communication, we aim to differentiate between publish/subscribe, request/response, and stream communication, whereas [58] differentiates between push, request/response and write or call traffic.

In our approach, each packet shall be transformed in a generic *action* with an *ID*, *type* and metadata (*time*, *source* and *destination*).

Identification of Actions: To be able to identify an action of a service and its client or server uniquely (*ID*), a feature for abstraction has to be created from the data available by the different protocols [R1, R7]. For SOME/IP, this could be a combination of the Message-ID and the Request-ID (see [34]), for example.

Monitor Stationary States: In case all participants have started and all services are set up, a stationary state for the context is reached. Therefore, some values of states or payloads should remain in a specifiable range [R2, R3, R6].

Monitor Number of Active Connections: While for RPC, the number of connections is defined during the development phase, for stream and publish/subscribe service types, this is not applicable. However, an upper and lower threshold for an interval of possible active connections can be defined. If the number of connections during runtime is outside of such an interval, it represents a deviation from normal behavior [R2].

Service ID-Assignment Check: At the host level, there are exclusive features that can only be observed with a host-based IDS. As an example, each application is assigned to a specific service ID, which is used by specified request/response IDs. These assignments can be observed by middleware to verify if other combinations exist during runtime [R1, R2, R7].

Payload Plausibility: In contrast to the Service ID-Assignment Check, interception based on the service ID and requested parameter (API) is challenging because it requires decoding and plausibility checking of the contained data. Focusing on the payload of the services, these can be analyzed for anomalies in time series, for example, as presented by Weber et al. [59] [R6, R8]. Assuming that future architectures will increasingly consider the principle of zero trust, payload plausibility checks can only be performed at the host level, since connections are end-to-end encrypted.

Effect Chains for RPC: The invocation of RPCs may result in an effect chain because the orchestration of services generally is based upon other services. Thus, a specific service will result in other services being executed and thus can be trained, e.g., according to [60] [R9].

Another approach is also to take the timing of the specific services invoked within an effect chain into account. The order of occurrence for RPCs is predictable by the specification within an automotive or medical SOA network, as these are usually fixed for main-services/functions. For each function (e.g., ACC Section 3.1.2), dependencies to required services are defined. The corresponding network behavior (occurrence of the service IDs) can be plotted on a timeline (s. Figure 6). The black bars represent a benign behavior in terms of timing and order of service IDs in an SOA network. In contrast, the gray bars encompass a deviant behavior triggered by an attacker. In the case that an attacker interferes the communication behavior by compromising an SOA node, a modified communication pattern occurs [R9]. For example, a modified function can lead to a different calculation time, which results in a changed response time.

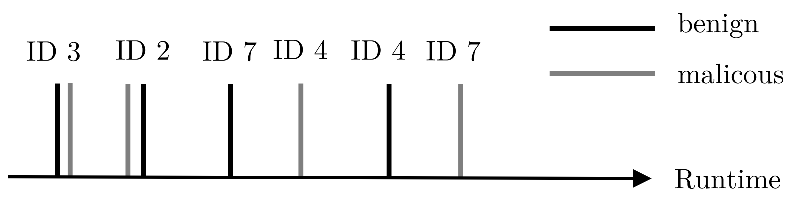

Figure 6. Exemplary timeline for the occurrence of Service IDs within an SOA network.

Publish/Subscribe Causality: For publish/subscribe patterns, it is determinable which change within the payload of a publish/subscribe service results in a change of the payload of another service [R5] or vice versa. This causal dependency can be determined with a machine learning algorithm which is trained with correlating data [R8, R9].

Stream Causality: For streams, expected behavior is harder to determine. In this case, it is more easy to check the causality according to an exclusion procedure [R5]. In case a stream-service A is missing, then a stream-service B might also be missing because it is based upon the stream-service A.

Context Plausibility: The state of different devices together with the payload of different services can be fused to create a context of a device ensemble (e.g., the OR). Thus, it is possible to make a plausibility check if a device state is possible within a given system context that is similar, as proposed by Grimm et al. [61]. In case the states and payloads are not plausible within a given overall context, an anomaly is detected [R3, R4, R8].

3.4. Detectable Anomalies in Use Cases

3.4.1. Detectable Anomalies in Medical Use Case

An effect chain of different RPCs can be built upon the angiography system, pausing the ventilator when taking an image and previously locking the OR table and afterwards restarting the ventilator. Furthermore, the timing is also interesting, as an imaging procedure and thus the pausing of the ventilator should approximately be identically in each iteration. When the angiography system pauses the respiration, the NIDS can check for the restarting of the respiration in a plausible time span. Furthermore, it could check that the HIDS sensor of the angiography system reports an imaging process during the respiration pause. The publish/subscribe causality check can be applied to the movement of the OR table together with the angiography system. During an imaging procedure, the movements of both systems must match in a certain workspace around the patient. In another scenario, the movements outside the restricted area of the OR table are detected by the supervisor NIDS.

Some systems create virtual pivot points for synchronized movements [62]. If both systems can determine this pivot point, it can be checked if both points are approximately identical during an imaging procedure. This can be monitored with an according to context plausibility check. Furthermore, the NIDS can check that the OR table is moving despite the lock having been set by the HIDS sensor of the OR table or the angiography system. In addition, the OR-Light camera can check in an advanced scenario that the OR table is moving. Therefore, the IDS sensors of the individual systems can be used to supervise other systems in the OR network. Another example is that an angiography system is in a state representing imaging in progress, while the payload data of a GetPosition service of an OR table indicates that a patient is unreachable for imaging.

In the OR context, it is hard to tell how many subscribers or publishers of services are present because each OR has an individual setup. Nevertheless, there is a chance to determine the according to amount for example at the start of the surgery. Another possibility is to determine static features during the commissioning process in which new medical devices are introduced into the OR [27]. These approaches are very static and might lack in practicability. Another approach is to couple the subscribers to the current devices with the so-called *location context* of SDC because they need to be especially reliable. Otherwise, a device from a different OR is able to control a device within another OR. In this way, a stationary state can be monitored.

3.4.2. Detectable Anomalies in Automotive Use Case

In reference to the use case provided in Figure 5, a context plausibility check can be done with neither the ACC algorithm nor the pedal control system of the car calling an RPC to accelerate, but the velocity is rising, provided it can be detected to drive downwards. The payload plausibility can be checked for the measured velocity, which should not contain implausible signal trajectories such as sudden jumps or direction changes. An example for publish/subscribe causality can be found in the ACC algorithm, which signals a short pause in its activation state when the pedals are used to brake or accelerate. It is also possible to check whether the speed measured at the wheels remains unchanged when braking via the foot pedals.

For SOME/IP in particular, at the design time, all servers and services as well as subscriptions are known, which results in a static quantity and can be used to monitor it as a stationary state. Only in the case of a software update does the number change. During runtime, each client is aware of all servers and services offered. Furthermore, any sequence consisting of a request/response is stateful. This means that the client ID of the request has to be included in the corresponding response message.

4. Prototypical Implementation for Medical Use Case

As a representative prototypical implementation, the medical device example has been partially implemented in ROS 2 [63], which uses DDS as middleware, in combination with Gazebo [64] for the use case depicted in Figure 4. A remote control, an angiography system, and an OR table are currently implemented each as ROS 2 nodes. The Supervisor-NIDS also is implemented in ROS2 and listens to the provided services and sensors in the network (Figure 7). Furthermore, the OR table and angiography system are connected to Gazebo models so that the positions of the joints can be simulated (Figure 8).

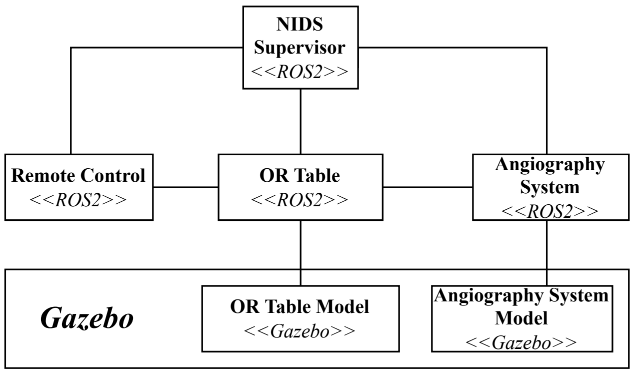
NIDS Supervisor <<ROS2>>
Remote Control <<ROS2>>
OR Table <<ROS2>>
Angiography System <<ROS2>>
Gazebo
OR Table Model <<Gazebo>>
Angiography System Model <<Gazebo>>

Figure 7. OR network implemented in ROS2.

Payload Plausibility Check for OR Table Joint Positions
As an example of payload plausibility, the *Provides Position* service for the OR table is implemented using the ROS2 sensor message *JointState* published by the Gazebo model. The NIDS supervisor also subscribes to this service of the OR table and generates a warning as soon as the position of a joint is not in a certain position. Each device in the OR relying on this service is capable of checking the payload plausibility this way.

Publish/Subscribe Causality for OR table Movements
To detect the implausible movement behavior of an OR table, the NIDS supervisor listens to the *Moves* service provided by the OR table. If the provided positions indicate a moving joint and there has been no movement request by the remote control for a specific moving joint, an anomaly is detected. The velocity is determined by deriving the provided position according to time. The comparison of the time stamp between the last movement call and a joint movement that has taken place provides information about whether the movement could have been triggered by the remote control.

Context Plausibility of a Locked OR table
A call to the motion service should not occur while the OR table is in a *locked* state, which prohibits movement of any kind. Therefore, this is taken into account by the NIDS supervisor, which listens to a sensor of the OR table that provides the current lock state as soon as it detects a call to the motion service. The sensor is implemented with a ROS2 topic that has the states true (locked) and false (unlocked). As soon as the position of the OR table changes while it is locked, a warning is generated.

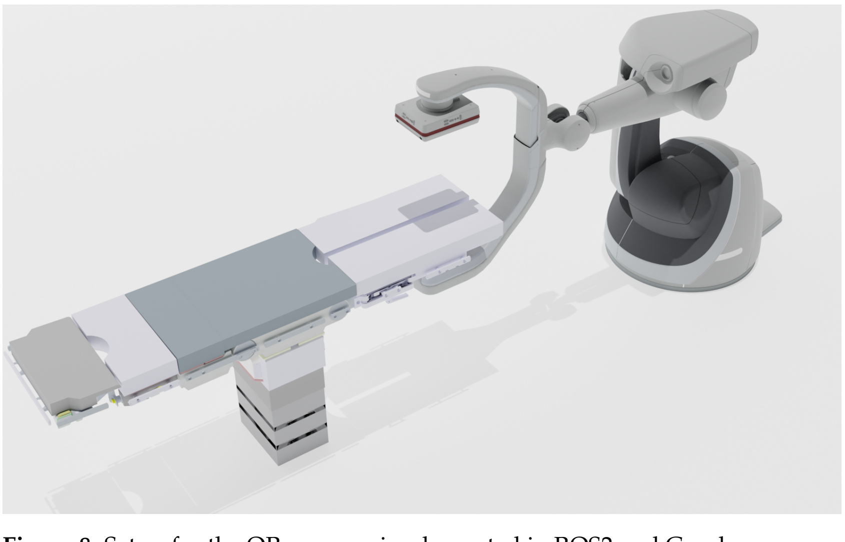

Figure 8. Setup for the OR use case implemented in ROS2 and Gazebo.

5. Conclusions and Future Work

The presented medical approaches currently are based on SDC networks in the OR. Nevertheless, today, there only exist demonstrators [65] for SDC connected devices. Although we focused on SOA features, there are some measures which are not directly restricted to SOA networks, e.g., the context plausibility check. Therefore, also, current OR integration software could apply these measures as they integrate different devices with a central integration platform. Thus, it is also possible to create a context of the whole OR and analyze the different device states as well as the payload.

In vehicles, the SOA paradigm is currently being introduced, making SOA IDS functionality increasingly relevant. The paper includes initial contributions toward an IDS based on SOME/IP and DDS protocol patterns. Compared to signal-based vehicle communication, IDS features for SOAs can no longer be derived exclusively based on the statically defined communication specification. Furthermore, an analysis of normal behavior during the runtime should also be performed to extract specific communication patterns. Context information should also be taken into account, for example, whether a specific function (e.g., ACC) is active, since the context has an impact on the corresponding communication behavior.

We do not focus on improving algorithms for IDSs such as the related works mentioned earlier but motivate a technology-independent and cross-domain architecture for IDSs. Moreover, our approach focuses more on the relationships between different services through their pattern type and overall context. The prototype presented in Section 4 also shows that security for the HOR can already be improved with the measures demonstrated.

Based on our presented assumptions, a prototypical implementation in Section 4 and proof of concept needs to be extended. One approach is to implement the presented OR use case within a simulated network based on available SDC libraries, such as OpenSDC [66]. In addition, more devices and examples need to be implemented for the remaining patterns.

As we discussed only the detection of intrusions, it needs to be evaluated which measures should be applied. Being very conservative and accepting false positives to detect all true positives leads to a reduced availability of systems and might be even worse than the actual attack. In addition, too many false alarms can lead to alarm fatigue, causing important alarms to be missed, as Sendelbach et al. [67] estimate that 72 to 99% are already false alarms. Thus, false alarms have to be considered, e.g., by the work of Kao et al. [53] to be reduced in real applications.

Transferred to the automotive domain, a prototypical implementation should also be carried out here to examine the patterns more closely. An important object of future investigation is the extraction of IDS features based on the normal SOA network behavior by using different techniques (e.g., Artificial Intelligence (AI)-based). One possible way to achieve the complete coverage of an entire system might be the use of a digital twin [68] of the devices including their SOA communication behavior, which is becoming increasingly important in the automotive domain.

Author Contributions: Conceptualization, A.P., M.R. and D.G.; software, A.P.; writing—original draft preparation, A.P., M.R. and D.G.; writing—review and editing, A.P., M.R., D.G. and E.S.; Supervision, E.S. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by KIT-Publication Fund of the Karlsruhe Institute of Technology grant number 2447.

Acknowledgments: We acknowledge support by the KIT-Publication Fund of the Karlsruhe Institute of Technology.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Hasan, M. State of IoT 2022: Number of Connected IoT Devices Growing 18% to 14.4 Billion Globally. 2022. Available online: https://iot-analytics.com/number-connected-iot-devices/ (accessed on 23 May 2022).
2. Khaitan, S.K.; McCalley, J.D. Design Techniques and Applications of Cyberphysical Systems: A Survey. *IEEE Syst. J.* **2015**, *9*, 350–365. [CrossRef]
3. Vetter, A.; Obergfell, P.; Guissouma, H.; Grimm, D.; Rumez, M.; Sax, E. Development Processes in Automotive Service-oriented Architectures. In Proceedings of the 2020 9th Mediterranean Conference on Embedded Computing (MECO), Budva, Montenegro, 8–11 June 2020; pp. 1–7. [CrossRef]
4. *ISO/IEC 18384-1:2016*; Information technology—Reference Architecture for Service Oriented Architecture (SOA RA)—Part 1: Terminology and concepts for SOA. ISO: Geneva, Switzerland, 2016.
5. Humayed, A.; Lin, J.; Li, F.; Luo, B. Cyber-Physical Systems Security—A Survey. *IEEE Internet Things J.* **2017**, *4*, 1802–1831. [CrossRef]
6. Papaioannou, M.; Karageorgou, M.; Mantas, G.; Sucasas, V.; Essop, I.; Rodriguez, J.; Lymberopoulos, D. A Survey on Security Threats and Countermeasures in Internet of Medical Things (IoMT). *Trans. Emerg. Telecommun. Technol.* **2020**, *33*, e4049. [CrossRef]
7. Wallenfels, M. Medizintechnik rüstet sich für den Kampf gegen Cyberkriminelle. *Gynäkol. Geburtshilfe* **2017**, *22*, 44. [CrossRef]
8. Miller, C.; Vasalek, C. Remote Exploitation of an Unaltered Passenger Vehicle. 2015. Available online: https://ioactive.com/remote-exploitation-of-an-unaltered-passenger-vehicle/ (accessed on 23 May 2022).
9. Matheus, K. One-Pair Ethernet in the Automotive Industry, 22 September 2020. Available online: https://singlepairethernet.com/wp-content/uploads/2021/07/SPE-in-Automotive_BMW_K.-Matthews.pdf13.pdf (accessed on 12 July 2022).
10. Ferguson, N.; Schneier, B.; Kohno, T. *Cryptography Engineering: Design Principles and Practical Applications/Niels Ferguson, Bruce Schneier, Tadayoshi Kohno*; Wiley: Indianapolis, IN, USA, 2010.
11. Lee, I.; Sokolsky, O.; Chen, S.; Hatcliff, J.; Jee, E.; Kim, B.; King, A.; Mullen-Fortino, M.; Park, S.; Roederer, A.; et al. Challenges and Research Directions in Medical Cyber—Physical Systems. *Proc. IEEE* **2012**, *100*, 75–90. [CrossRef]
12. Teber, D.; Engels, C.; Maier-Hein, L.; Ayala, L.; Onogur, S.; Seitel, A.; März, K. Wie weit ist Chirugie 4.0? *Der Urologe. Ausg. A* **2020**, *59*, 1035–1043. [CrossRef] [PubMed]
13. Harris, S.; Maymi, F. *CISSP All-In-One Exam Guide*, 7th ed.; McGraw-Hill Education: New York, NY, USA, 2016.
14. Hadžiosmanović, D.; Simionato, L.; Bolzoni, D.; Zambon, E.; Etalle, S. N-Gram against the Machine: On the Feasibility of the N-Gram Network Analysis for Binary Protocols. In *Research in Attacks, Intrusions, and Defenses*; Lecture Notes in Computer Science; Balzarotti, D.; Stolfo, S.J.; Cova, M., Eds.; Springer:: Berlin/Heidelberg, Germany, 2012; Volume 7462, pp. 354–373. [CrossRef]
15. Rumez, M.; Grimm, D.; Kriesten, R.; Sax, E. An Overview of Automotive Service-Oriented Architectures and Implications for Security Countermeasures. *IEEE Access* **2020**, *8*, 221852–221870. [CrossRef]
16. Ahmad, Z.; Shahid Khan, A.; Wai Shiang, C.; Abdullah, J.; Ahmad, F. Network intrusion detection system: A systematic study of machine learning and deep learning approaches. *Trans. Emerg. Telecommun. Technol.* **2021**, *32*, e4150. [CrossRef]
17. Langmann, R. *Vernetzte Systeme für die Automatisierung 4.0: Bussysteme— Industrial Ethernet—Mobile Kommunikation—Cyber-Physical Systems*; Hanser: New York, NY, USA, 2021.
18. OASIS. Reference Model for Service Oriented Architecture v1.0. 2006. Available online: http://docs.oasis-open.org/soa-rm/v1.0/soa-rm.html (accessed on 27 January 2022).
19. Nollert, G.; Hartkens, T.; Figel, A.; Bulitta, C.; Altenbeck, F.; Gerhar, V. The Hybrid Operating Room. In *Special Topics in Cardiac Surgery*; Narin, C., Ed.; InTech: Vienna, Austria, 2012. [CrossRef]
20. Getinge. Hybrid OR: Engineering Success. Pioneering Healthcare. 2022. Available online: https://www.getinge.com/int/products-and-solutions/operating-room/hybrid-and-imaging-or/siemens/ (accessed on 12 September 2022).
21. Puder, A.; Henle, J.; Rumez, M.; Vetter, A. A Mixed E/E-Architecture for Interconnected Operating Tables Inspired by the Automotive Industry. In Proceedings of the International Symposium on Medical Robotics, Atlanta, GA, USA, 13–15 April 2022.
22. Getinge. Hybrid OR: Engineering Success. Pioneering Healthcare. 2021. Available online: https://www2.getinge.com/de/loesungen/operationssaal/hybrid-op/siemens/ (accessed on 12 September 2022).
23. Getinge. Maquet Magnus Operating Table System. 2021. Available online: https://www.getinge.com/dam/hospital/documents/english/magnus_care_supporter_brochure-en-non_us.pdf (accessed on 12 September 2022).
24. Goldman, J.M.; Schrenker, R.A.; Jackson, J.L.; Whitehead, S.F. Plug-and-Play in the OperatingRoom of the Future. *Biomed. Instrum. Technol.* **2005**, *39*, 194–199.
25. ASTM. Medical Devices and Medical Systems: Essential Safety Requirements for Equipment Comprising the Patient-Centric Integrated Clinical Environment (ICE)—Part 1: General Requirements and Conceptual Model. 2013. Available online: http://www.astm.org/Standards/F2761.htm (accessed on 10 August 2022).
26. Arney, D.; Plourde, J.; Goldman, J.M. OpenICE medical device interoperability platform overview and requirement analysis. *Biomed. Technik. Biomed. Eng.* **2018**, *63*, 39–47. [CrossRef]
27. Pfeiffer, J.H.; Dingler, M.E.; Dietz, C.; Lueth, T.C. Requirements and architecture design for open real-time communication in the operating room. In Proceedings of the 2015 IEEE International Conference on Robotics and Biomimetics (ROBIO), Zhuhai, China, 6–9 December 2015; IEEE: 2015; pp. 458–463. [CrossRef]
28. Kasparick, M. Zuverlässige und herstellerübergreifende Medizingeräteinteroperabilität & Beiträge zur IEEE 11073 SDC-Normenfamilie. Ph.D. Thesis, Universität Rostock, Rostock, Germany, 2020. [CrossRef]

29. *11073-20702-2018*; Health informatics-Point-of Care Medical Device Communication: Part 20702: Medical Devices Communication Profile for Web Services. IEEE Standards Association: Piscataway, NJ, USA, 2018.
30. *P11073-10418/D15*; Health Informatics—Personal Health Device Communication: Part 10207: Domain Information and Service Model for Service-Oriented Point-Ofcare Medical Device Communication. ISO/IEEE: Piscataway, NJ, USA, 2019.
31. Okamoto, J.; Masamune, K.; Iseki, H.; Muragaki, Y. Development concepts of a Smart Cyber Operating Theater (SCOT) using ORiN technology. *Biomed. Technik. Biomed. Eng.* **2018**, *63*, 31–37. [CrossRef]
32. Sun, X.; Okamoto, J.; Masamune, K.; Muragaki, Y. Robotic Technology in Operating Rooms: A Review. *Curr. Robot. Rep.* **2021**, *2*, 333–341. [CrossRef] [PubMed]
33. Berger, J.; Rockstroh, M.; Schreiber, E.; Yoshida, Y.; Okamoto, J.; Masamune, K.; Muragaki, Y.; Neumuth, T. GATOR: Connecting integrated operating room solutions based on the IEEE 11073 SDC and ORiN standards. *Int. J. Comput. Assist. Radiol. Surg.* **2019**, *14*, 2233–2243. [CrossRef] [PubMed]
34. AUTOSAR Foundation. *SOME/IP Protocol Specification: Release 1.1.0*; Document ID 696; AUTOSAR Foundation: Munich, Germany, 2017.
35. Object Management Group. Data Distribution Service (DDS): Version 1.4. 2015. Available online: http://www.omg.org/spec/DDS/1.4 (accessed on 20 August 2022).
36. RTI Healthcare & Medical Connectivity and Autonomous System Software. 18 February 2022. Available online: https://www.rti.com/industries/healthcare (accessed on 19 February 2022).
37. Object Management Group. RPC Over DDS (DDS-RPC): Version 1.0. 2017. Available online: http://www.omg.org/spec/DDS-RPC/1.0 (accessed on 20 August 2022).
38. Javdani, H.; Kashanian, H. Internet of things in medical applications with a service-oriented and security approach: A survey. *Health Technol.* **2018**, *8*, 39–50. [CrossRef]
39. Leucker, M.; Schmitz, M. Secured SOA for the Safe Interconnection of Medical Devices (Position Paper). In Proceedings of the Gemeinsamer Tagungsband der Workshops der Tagung Software Engineering 2015, Dresden, Germany, 17–18 March 2015; Volume CEUR-WS 1337.
40. Arney, D.; Venkatasubramanian, K.K.; Sokolsky, O.; Lee, I. Biomedical devices and systems security. In Proceedings of the 2011 Annual International Conference of the IEEE Engineering in Medicine and Biology Society, Boston, MA, USA, 30 August–3 September 2011; Volume 2011, pp. 2376–2379. [CrossRef]
41. Staschulat, J.; Lütkebohle, I.; Lange, R. The rclc Executor: Domain-specific deterministic scheduling mechanisms for ROS applications on microcontrollers: Work-in-progress. In Proceedings of the 2020 International Conference on Embedded Software (EMSOFT), Shanghai, China, 20–25 September 2020; pp. 18–19. [CrossRef]
42. Kampmann, A.; Wüstenberg, A.; Alrifaee, B.; Kowalewski, S. A Portable Implementation of the Real-Time Publish-Subscribe Protocol for Microcontrollers in Distributed Robotic Applications. In Proceedings of the 2019 IEEE Intelligent Transportation Systems Conference (ITSC), Auckland, New Zealand, 27–30 October 2019; pp. 443–448. [CrossRef]
43. Kreissl, J. Absicherung der SOME/IP Kommunikation bei Adaptive AUTOSAR. Master's Thesis, Universität Stuttgart, Stuttgart, Germany, 2017.
44. Weschke, J.; Hesslund, F. Testing and Evaluation to Improve Data Security of Automotive Embedded Systems. Master's Thesis, Chalmers University of Technology, Gothenburg, Sweden, 2011.
45. Gehrmann, T.; Duplys, P. Intrusion Detection for SOME/IP: Challenges and Opportunities. In Proceedings of the 2020 23rd Euromicro Conference on Digital System Design (DSD), Kranj, Slovenia, 26–28 August 2020; pp. 583–587. [CrossRef]
46. Iorio, M.; Reineri, M.; Risso, F.; Sisto, R.; Valenza, F. Securing SOME/IP for In-Vehicle Service Protection. *IEEE Trans. Veh. Technol.* **2020**, *69*, 13450–13466. [CrossRef]
47. Schindewolf, M.; Stoll, H.; Guissouma, H.; Puder, A.; Sax, E.; Vetter, A.; Rumez, M.; Henle, J. A Comparison of Architecture Paradigms for Dynamic Reconfigurable Automotive Networks. In Proceedings of the 2022 International Conference on Connected Vehicle and Expo (ICCVE), Lakeland, FL, USA, 7–9 March 2022; pp. 1–7. [CrossRef]
48. El Mrabet, Z.; Kaabouch, N.; El Ghazi, H.; El Ghazi, H. Cyber-security in smart grid: Survey and challenges. *Comput. Electr. Eng.* **2018**, *67*, 469–482. [CrossRef]
49. Pliatsios, D.; Sarigiannidis, P.; Lagkas, T.; Sarigiannidis, A.G. A Survey on SCADA Systems: Secure Protocols, Incidents, Threats and Tactics. *IEEE Commun. Surv. Tutor.* **2020**, *22*, 1942–1976. [CrossRef]
50. Baker, T.; Mackay, M.; Shaheed, A.; Aldawsari, B. Security-Oriented Cloud Platform for SOA-Based SCADA. In Proceedings of the 2015 15th IEEE/ACM International Symposium 2015, Shenzhen, China, 4–7 May 2015; pp. 961–970. [CrossRef]
51. Najjar, M.S.; Abdollahi Azgomi, M. A distributed multi-approach intrusion detection system for web services. In Proceedings of the the 3rd International Conference on Security of Information and Networks, Russian Federation, 7–11 September 2010; Makarevich, O., Ed.; ACM: New York, NY, USA, 2010; p. 238. [CrossRef]
52. Carta, S.; Podda, A.S.; Recupero, D.R.; Saia, R. A Local Feature Engineering Strategy to Improve Network Anomaly Detection. *Future Internet* **2020**, *12*, 177. [CrossRef]
53. Kao, M.T.; Sung, D.Y.; Kao, S.J.; Chang, F.M. A Novel Two-Stage Deep Learning Structure for Network Flow Anomaly Detection. *Electronics* **2022**, *11*, 1531. [CrossRef]
54. Jiang, J.R.; Kao, J.B.; Li, Y.L. Semi-Supervised Time Series Anomaly Detection Based on Statistics and Deep Learning. *Appl. Sci.* **2021**, *11*, 6698. [CrossRef]

55. Khalid, F.; Rehman, S.; Shafique, M. Overview of Security for Smart Cyber-Physical Systems. In *Security of Cyber-Physical Systems*; Karimipour, H., Srikantha, P., Farag, H., Wei-Kocsis, J., Eds.; Springer International Publishing: Cham, Switzerland, 2020; pp. 5–24. [CrossRef]
56. Hofmockel, J. *Anomalieerkennung in Kommunikationsdaten zur Datenselektion im Fahrzeug*; Karlsruher Institut für Technologie: Karlsruhe, Germany, 2019. [CrossRef]
57. Arney, D.; Goldman, J.M.; Whitehead, S.; Lee, I. Synchronizing an X-Ray and Anesthesia Machine Ventilator: A Medical Device Interoperability Case Study. In *Proceedings of the International Conference on Biomedical Electronics and Devices*; SciTePress—Science and and Technology Publications: Porto, Portugal, 2009; pp. 52–60. [CrossRef]
58. Wolsing, K.; Wagner, E.; Henze, M. Facilitating Protocol-Independent Industrial Intrusion Detection Systems. In Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security, Virtual, 9–13 November 2020; Association for Computing Machinery: New York, NY, USA, 2020; pp. 2105–2107. [CrossRef]
59. Weber, M.; Pistorius, F.; Sax, E.; Maas, J.; Zimmer, B. A Hybrid Anomaly Detection System for Electronic Control Units Featuring Replicator Neural Networks. In *Advances in Information and Communication Networks*; Advances in Intelligent Systems and Computing; Arai, K., Kapoor, S., Bhatia, R., Eds.; Springer International Publishing: Cham, Switzerland, 2019; Volume 887, pp. 43–62. [CrossRef]
60. Rumez, M.; Lin, J.; Fuchß, T.; Kriesten, R.; Sax, E. Anomaly Detection for Automotive Diagnostic Applications Based on N-Grams. In Proceedings of the 2020 IEEE 44th Annual Computers, Software, and Applications Conference (COMPSAC), Madrid, Spain, 13–17 July 2020; pp. 1423–1429.
61. Grimm, D.; Sax, E. Context-aware vehicle and fleet security combining a Knowledge Graph and an object-oriented model. In Proceedings of the 2022 International Conference on Connected Vehicle and Expo (ICCVE), Lakeland, FL, USA, 7–9 March 2022; pp. 1–8. [CrossRef]
62. Hillrom Holding. Robotic Operating Rooms. 28 January 2022. Available online: https://www.hillrom.com/en/surgical-strategic-alliances/robotic-operating-rooms/ (accessed on 6 February 2022).
63. Maruyama, Y.; Kato, S.; Azumi, T. Exploring the performance of ROS2. In Proceedings of the 13th International Conference on Embedded Software, Chengdu, China, 13–14 August 2016; ACM: New York, NY, USA, 2016; pp. 1–10. [CrossRef]
64. Koenig, N.; Howard, A. Design and use paradigms for gazebo, an open-source multi-robot simulator. In Proceedings of the 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (IEEE Cat. No.04CH37566), Sendai, Japan, 28 September–2 October 2004; pp. 2149–2154. [CrossRef]
65. Kasparick, M.; Schmitz, M.; Andersen, B.; Rockstroh, M.; Franke, S.; Schlichting, S.; Golatowski, F.; Timmermann, D. OR.NET: A service-oriented architecture for safe and dynamic medical device interoperability. *Biomed. Technik. Biomed. Eng.* **2018**, *63*, 11–30. [CrossRef] [PubMed]
66. Schlichting, S.; Gregorczyk, D.; Andersen, B. OpenSDC Library. 21 February 2022. Available online: https://sourceforge.net/projects/opensdc/ (accessed on 7 March 2022).
67. Sendelbach, S.; Funk, M. Alarm Fatigue. *AACN Adv. Crit. Care* **2013**, *24*, 378–386. [CrossRef] [PubMed]
68. Holmes, D.; Papathanasaki, M.; Maglaras, L.; Ferrag, M.A.; Nepal, S.; Janicke, H. Digital Twins and Cyber Security—Solution or challenge? In Proceedings of the 2021 6th South-East Europe Design Automation, Computer Engineering, Computer Networks and Social Media Conference (SEEDA-CECNSM), Preveza, Greece, 24–26 September 2021; pp. 1–8.