SOAR4IoT: Securing IoT Assets with Digital Twins

Philip Empl*
philip.empl@ur.de
University of Regensburg
Germany

Daniel Schlette
daniel.schlette@ur.de
University of Regensburg
Germany

Daniel Zupfer
daniel.zupfer@ur.de
University of Regensburg
Germany

Günther Pernul
guenther.pernul@ur.de
University of Regensburg
Germany

ABSTRACT

As more and more security tools provide organizations with cybersecurity capabilities, security analysts are overwhelmed by security events. Resolving these events is challenging due to extensive manual processes, limited financial resources, and human errors. Security Orchestration, Automation, and Response (SOAR) is an established approach to manage security tools and assets. However, SOAR platforms typically integrate traditional IT systems only. Additional considerations are required to deal with the Internet of Things (IoT), its multiple devices and complex networks. Therefore, we adapt SOAR to IoT. We first aggregate existing research and information on SOAR and SOAR platforms. We envision the `SOAR4IoT` framework, making IoT assets manageable for SOAR via middleware. We implement a prototypical digital twin-based SOAR application integrating IoT assets and security tools to validate our framework. The experimental setup includes two playbooks coping with Mirai and Sybil attacks. Results show feasibility as our SOAR application enables securing IoT assets with digital twins.

CCS CONCEPTS

• **Security and privacy** → *Network security*; *Systems security*; *Security services*; • **Computer systems organization**; • **Information systems**;

KEYWORDS

Internet of Things, Security Orchestration, Incident Response, SOAR, Digital Twin

ACM Reference Format:
Philip Empl, Daniel Schlette, Daniel Zupfer, and Günther Pernul. 2022. SOAR4IoT: Securing IoT Assets with Digital Twins. In *The 17th International Conference on Availability, Reliability and Security (ARES 2022), August 23–26, 2022, Vienna, Austria.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3538969.3538975

1 INTRODUCTION

Attackers and defenders shape cybersecurity. Sophisticated attacks on networked information systems are countered by defenders' use of tools for security monitoring and operations. However, there is an ongoing challenge for security analysts. While more and more security tools are being used, analysts can face up to 11,000 security alerts per day (including false positives) [11]. Therefore, organizations use Security Orchestration, Automation and Response (SOAR) platforms promising tool integration, automation, and streamlined workflows for rapid incident response [19, 25].

SOAR platforms are based on security events. Security events concern traditional IT resources but also the Internet of Things (IoT). The new IoT frontier (e.g., smart factories or automated home systems) with its multitude of heterogeneous devices contributes to the ongoing datafication but currently neglects cybersecurity. Inadequate or missing security measures caused by a "set-it-and-forget-it manner" [20] are illustrative for the insecurity of IoT assets. Attackers notice these IoT security issues, as Kaspersky reports 1.5 billion attacks against their IoT honeypots in the first half of 2021 [30]. Eventually, networked IoT devices exposing default username/password authentication will become part of botnets. Estimates see the approximate time to compromise an IoT device at just five minutes [20]. Thus, it is necessary to extend security operations to IoT assets for which digital twins provide promising features [9]. Digital twins are used for security to simulate IoT attacks [8] and can assist incident response [7, 10].

Whether IoT-specific or not, security analysts cannot process security events manually. SOAR platforms greatly help analysts perform investigations and initiate adequate incident response actions. Analysts can reduce time and resources spent on low-priority events and manual actions using automated playbooks. Thus, SOAR documents a shift towards more effective security operations within organizations. As SOAR attracts attention in research and provides the dynamics to abstract complex environments, we investigate its potential for the IoT. Consequently, we ask *"how to use Security Orchestration, Automation and Response for the Internet of Things?"* We expect the general applicability of SOAR for IoT as it is a flexible construct. Still, it is crucial to showcase adaptation rigorously.

In this paper, we aim to answer the following questions: (1) What defines SOAR? (2) How to secure the IoT? (3) How to implement SOAR for IoT with digital twins? These questions lead to our main contributions:

- We enlighten SOAR core activities and platform features by analyzing the few academic works and current SOAR platforms.
- We envision our `SOAR4IoT` framework built on IoT attacks and mitigation strategies. Our framework encompasses IoT assets, middleware, SOAR platform, and security tools.
- We provide a SOAR4IoT implementation leveraging digital twins. The experimental setup documents the straightforward, ground-up implementation of a SOAR platform, including Eclipse Ditto-based digital twins, which researchers and practitioners can easily adapt and extend.

*Corresponding author

ARES 2022, August 23–26, 2022, Vienna, Austria

ACM ISBN 978-1-4503-9670-7/22/08.
https://doi.org/10.1145/3538969.3538975

- We explore two security issues of IoT assets. We address IoT security operations by designing and implementing two generic playbooks for orchestration and automated response to the Mirai botnet and the Sybil attack.

The paper is organized as follows. Section 2 outlines IoT, digital twins for cybersecurity, SOAR foundations and describes related work. Section 3 elaborates the framework defining the characteristics of SOAR, discussing the objectives of secure IoT assets, and describing technologies abstracting the IoT. Then, formal requirements lead to the overall SOAR4IoT framework. We validate our framework in Section 4 through the implementation of a digital twin-based SOAR platform integrating two use cases. We conclude our paper in Section 5.

2 BACKGROUND AND RELATED WORK

This section elaborates the background on IoT (Section 2.1), digital twins for cybersecurity (Section 2.2) and SOAR (Section 2.3), concluding with related work (Section 2.4).

2.1 Internet of Things

The IoT is characterized by identifiable networking objects (sensors or actuators) advertising their services to assemble semantic-rich applications [1]. Beyond scrutinizing particular devices, the IoT involves communication, applications, and processes. Heterogeneous devices and machines of widely ranging specifications and data operate seamlessly and collaboratively to assist business processes. The heterogeneity of IoT devices and networks is mainly caused by various manufacturers and (communication) protocols. As a result, there are plenty of cybersecurity issues demanding 1) automated security operations (detection and mitigation) and 2) orchestration of security functions for the IoT [17]. When it comes to integrating IoT assets, middleware is reliable, and a common choice [27, 32]. Organizations can choose between different types of middleware according to technology preferences and use cases (see Figure 1).

2.2 Digital Twins for Cybersecurity

In general, digital twins can be conceived as middleware. At its core, the digital twin links a virtual representation to a physical asset aiming to mirror the asset along its life cycle with semantic technologies [3]. The digital twin synchronizes system states using bidirectional communication with its physical counterpart. Implementing digital twins is a challenging task. Digital twins (e.g., Eclipse Ditto or Azure Digital Twins) can be used standalone or connected to IoT platforms (e.g., Eclipse Kapua or Azure IoT Hub).

From a security perspective, digital twins concern three primary security-operation modes: replication, simulation, and historical data analytics [8]. *Historical data analytics* deals with the documented behavior of IoT assets in the past and draws conclusions for the future. *Simulations* build on user-specific parameters and model the semantics of the real world. Last, the *replication* integrates real-world data to semantically model and operate a digital twin identical to its real-world counterpart. These operation modes assist security operations. For instance, behavior-based modeling supports more efficient intrusion detection, and the virtual representation of the digital twin is suitable for security training [9]. Moreover, replication-based digital twins indicate security orchestration and incident response features.

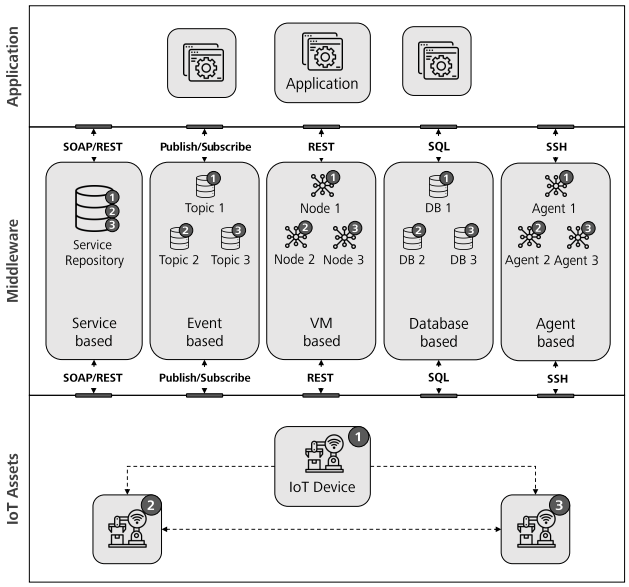


Figure 1: IoT architecture and middleware types

2.3 Security Orchestration, Automation and Response (SOAR)

Platforms promising *Security Orchestration, Automation and Response (SOAR)* capabilities for organizations are the latest solutions proposed by cybersecurity vendors [19]. Like other solutions before, the underlying concept has not received much research attention while products are being pushed to market. SOAR is not a standalone concept but part of continuous development. From related concepts like log management to Security Information and Event Management (SIEM), Cyber Threat Intelligence (CTI), and security orchestration, it can be observed that succeeding concepts build on previous ones. Examining SOAR, it becomes evident that platforms, system architectures, and data are crucial to understanding and implementing the concept.

In the organizational context, SOAR and corresponding platforms are associated with the Security Operations Center (SOC) or Computer Security Incident Response Team (CSIRT) [31]. Intuitively, SOAR aims to assist activities within the three domains of 1) security orchestration, 2) automation, and 3) incident response.

For *security orchestration*, SOAR subsumes the functionality of SIEM and integrates multiple devices, systems, and security tools [13]. Additionally, integration and unification aspects of SOAR relate to threat intelligence as relevant information about threats, attacks, and vulnerabilities is aggregated from internal and external sources. For *automation*, SOAR relies on events and defined courses of action to enable rapid security operations. Thus, automation bridges the gap between security orchestration and incident response. For *incident response*, containment, eradication, and recovery activities demand to derive and perform appropriate measures.

Therefore, SOAR includes the instrumentalization of endpoints and security tools to execute commands.

Related to SOAR is the standardization and representation of incident response [28]. While current systems are often based on ticketing systems for security incidents, incident response playbooks are central. In essence, incident response playbooks define how to conduct a specified defensive procedure. Towards standardization, the incident response community initiated the development of dedicated data formats. These formats specify structural elements and required meta-data for incident response use cases. For instance, the two formats *Open Command and Control (OpenC2)* [23] and *Collaborative Automated Course of Action Operations (CACAO) for Cyber Security* [24] document different focal areas such as executable commands and procedural workflows, respectively.

2.4 Related Work

IoT devices and networks are susceptible to cyberattacks. Providing security measures for IoT is a practical problem and has attracted researchers' attention. As outdated firmware enables attacks on IoT devices, the literature emphasizes security orchestration by using a firmware update scenario (e.g., [2]). RFC 9019 describes updating IoT firmware in detail [18] while others use distributed ledger technologies [5]. As a consequence, we consider IoT firmware updates to validate our work. From a network perspective, the European Telecommunication Standard Institute proposes central security orchestration based on automated configurations and deployments [15]. We build on existing research and unify security orchestration activities. We include network and device layers within a single SOAR framework.

Digital twins for incident response is a trending research topic. Digital twins assist analysts in SOC [8] and are proposed for response measures [12]. Especially for operational systems, digital twins should implement cybersecurity services (e.g., access control, intrusion detection, or incident response) [7]. In a recent publication, Eckhart and Ekelhart [10] emphasize digital twins of real-world IoT systems as a new method for incident response. Existing literature only conceptualizes digital twin-based incident response. We are taking research further and implement digital twins for incident response.

Scoping the topic of SOAR, we identified additional related work. Most notably, Islam et al. [13] provide a survey on security orchestration. In a follow-up work on SOAR architecture, the authors propose the layered integration of security tools and map tools to response activities [14]. For CTI sources in SOAR, security enumerations have been discussed in the context of the IoT [29]. We go beyond security tools and include application aspects and IoT assets in our approach.

Further, SOAR has been examined in the context of incident response. Complementary to incident response formats, Schlette et al. [28] outline the vast SOAR product landscape. As SOAR platforms assist organizations' incident response, research addressed the appropriate selection [22] and quantitative evaluation of features [21]. SOAR platforms evolve and existing works provide a snapshot. Based on these works, we aggregate common features of SOAR platforms and settle on agreed-upon characteristics.

Table 1: SOAR requirements

	Requirement	Description	IoT
Core activities	Security Orchestration	Integration of IT assets, security tools, and threat intelligence	*
	Automation	Use of technologies and logic to perform security operations	✓
	Incident Response	Investigation, mitigation, and remediation of incidents	*
Platform features	User Interface	Dashboard or console for human interaction	✓
	Playbooks	Workflows, courses of action, or scripts	✓
	Ticketing System	Case management for security incidents	✓
	User Management	Access control and communication	✓

✓ is applicable * requires modification

3 SOAR4IOT FRAMEWORK

To apply SOAR to IoT, we first identify general SOAR requirements (Section 3.1). Examining attacks on the IoT, we then derive IoT incident response objectives (Section 3.2). These objectives guide us towards required IoT security orchestration (Section 3.3). Based on our formal model (Section 3.4), we conceptualize a `SOAR4IoT` framework (Section 3.5) that integrates IoT systems using digital twins.

3.1 SOAR Requirements

SOAR requirements describe essential characteristics for the implementation of SOAR. Ultimately, SOAR requirements can assist the development of a SOAR platform, the evaluation of existing ones, or the adaptation to IoT devices and networks. In the following, we aggregate SOAR requirements from existing literature and validate the findings by examining current SOAR platforms. Table 1 describes core activities and platform features.

Core activities (i.e., security orchestration, automation, and incident response) constitute one group of requirements. They represent platform capabilities. For IoT, security orchestration demands modification as heterogeneous, dispersed devices form dynamic networks. Task automation remains largely unaffected, is conducted at SOAR platform level, and applies to IoT. Incident response measures directly involve IoT assets and thus demand modification.

Platform features constitute the second group of SOAR requirements. They represent technical aspects of a SOAR platform. Typically, a SOAR platform provides a user interface such as a dashboard or a console to assist orchestration and response activities [14]. More precisely, the user interface allows querying data and triggering courses of action. Playbooks are another dedicated SOAR

platform feature [21]. Playbooks represent workflows including actuators, actions, and artifacts to support automation and incident response. For instance, a remediation playbook can be designed and configured to make an orchestrated device (i.e., actuator) install (i.e., action) a new firmware version (i.e., artifact). Linked to security incidents or threat intelligence, (semi-)automation is possible. A ticketing system is a SOAR platform feature that helps to keep track of security incidents [13]. Tickets and case management also support prioritization and relate to security events. At last, SOAR platforms enable collaboration and include user management [22]. The platform-centric features above apply to SOAR for IoT.

Aside from literature and their analysis, we also analyzed a selected few SOAR platforms (Cortex XSOAR, D3 XGEN SOAR, Siemplify, Splunk SOAR, Tines). In addition, the latest Gartner market report [19] reveals some information on SOAR requirements. Our observations of SOAR platform characteristics include:

- Ready-to-use connectors, adapters, or similar interfaces
- No-code or low-code approach for playbooks
- SIEM functions included or integrated
- Ticketing system included or integrated

Most notably, SOAR platforms acknowledge the multitude of other security tools and provide necessary technical integrations. Playbook editors emphasize visualization and drag-and-drop functionality but also allow to generate scripts. Concerning SIEM functions, we consider log collection, detection, correlation, and alerts to be SIEM characteristics. However, some SOAR platforms directly include these functions. Moreover, there is only an arbitrary boundary between some SIEM and SOAR tools (e.g., Wazuh). Ticketing systems build an underlying foundation for SOAR platforms and are closely related to correlation and prioritization. Nevertheless, organizations can also integrate existing security ticketing systems.

As a result, core activities and platform features apply to current SOAR platforms. In the context of IoT and our framework, SOAR requirements are applicable but also demand adaptation.

3.2 IoT Incident Response Objectives

We discuss possible attacks and vulnerabilities of IoT systems to identify relevant assets that necessitate SOAR. The IoT provides a favored attack surface to different threat actors pooling their resources. As the number of IoT market participants grows, time-to-market is shortening, standards are lacking, and security is affected. Consequently, inadequate security of IoT assets is a call to incident response (e.g., update procedures or configuration). Research identifies several perspectives on IoT attacks and vulnerabilities, such as encryption attacks [16], attacks mapped to the ISO/OSI stack [4], or the most common vulnerabilities listed by OWASP IoT Top 10[1]. We distinguish IoT attacks on a higher level. Thereby we differentiate between attacks on device-level and network-level. We exclude attacks concerning other layers than the physical or network layer (e.g., attacks in cloud environments) because these attacks are not unique to the IoT. In summary, IoT attacks target:

- Device-level – hardware-based attacks, software-based attacks, and sensor data-based attacks
- Network-level – network-based attacks

[1]https://owasp.org

Table 2: IoT attacks and mitigations

Type	Attack	Mitigation
Hardware-based	Node tampering	Perimeter security
Software-based	Mirai malware	Firmware update
Data-based	False injection	Authentication
Network-based	Sybil attack	Offboarding

Hardware-based attacks target the physical layer to damage IoT devices systematically. These physical layer attacks include node tampering, node jamming, or other physical damage. Software-based attacks on IoT devices usually involve firmware vulnerabilities or the (embedded) operating system. These vulnerabilities are exploited by well-known malware, such as Mirai botnet, Industroyer, or Reaper. Attacks also target data, especially sensor data. Injecting false data, eavesdropping and task inference are data-based attacks and conclude the device-level attacks. The network-level scopes all attacks based on the ISO/OSI stack layers, e.g., Sybil attack, denial of service, or wormhole attack.

In order to mitigate and prevent these vulnerabilities and attacks, security measures concerning IoT are discussed [4]. These security measures constitute IoT incident response objectives. More generally, there are proactive and reactive security measures. For instance, over-the-air (OTA) firmware updates and strengthening of password security are proactive security measures and the on- and offboarding of IoT devices count to reactive security measures. SOAR platforms can orchestrate proactive and reactive security measures. We do not consider security-by-design decisions (e.g., encryption mechanisms).

The orchestration of IoT devices and networks is a prerequisite to incident response. Playbooks are a crucial platform feature of SOAR to enable automation. Referring back to SOAR requirements, the deployment of the other two core capabilities, namely orchestration and response, in the IoT is challenging. While the orchestration of security tools is similar to traditional SOAR and requires no further considerations, the orchestration of IoT devices and networks requires more attention. Table 2 summarizes attacks on IoT assets and example mitigations. Moreover, different means of IoT security orchestration exist, which we identify in the next section.

3.3 IoT Security Orchestration

IoT security orchestration is directed at *IoT devices* (device-level) and *IoT networks* (network-level). Security measures for hardware-based attacks are enabled by manual tasks only. Proactively locking IoT devices away is an illustrative physical security measure and not part of SOAR.

In general, middleware is used to abstract IoT devices and their functionalities. However, middleware can also serve security orchestration purposes. Commercial solutions address IoT devices with two common middleware concepts: Digital twins and IoT platforms. Our work takes on a broad perspective but emphasizes the digital twin concept for representing IoT assets.

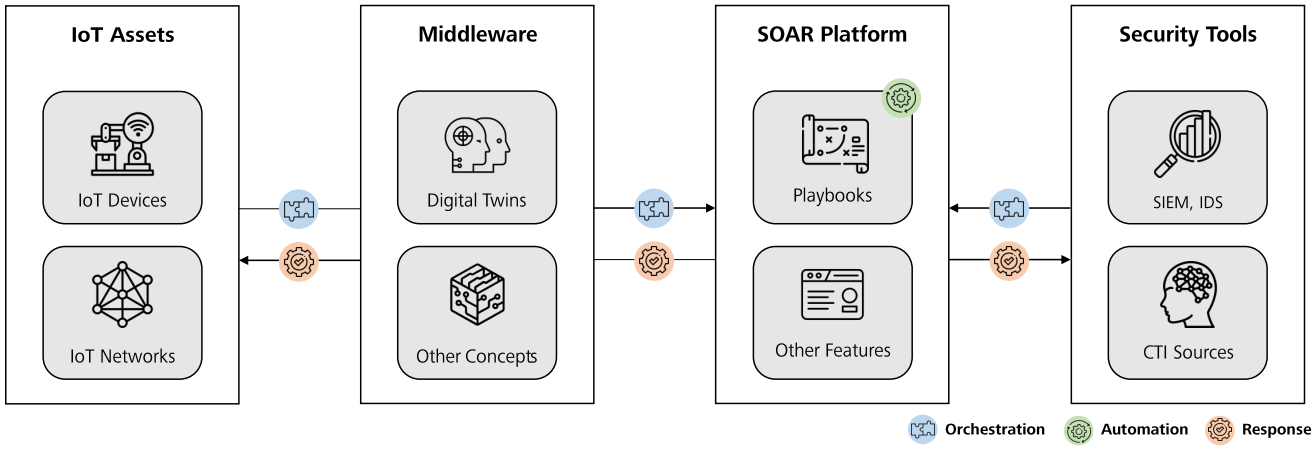


Figure 2: SOAR4IoT framework

Digital twins provide many features that enable security orchestration for IoT devices. They go beyond IoT platforms that are centered on common management tasks (e.g., onboard, monitor, and offboard devices). In particular, digital twins in replication mode provide IoT device modeling and security features. The bidirectional communication between the digital twin and IoT asset is beneficial as synchronizing sensor data and receiving commands can fulfill security orchestration. For instance, digital twins can store threat information acquired from third-party apps and synchronize information about vulnerabilities with their physical counterparts.

Besides IoT devices, digital twins and IoT platforms also extend to IoT networks. In this regard, digital twins allow the representation of dedicated edge nodes. Edge nodes are used in IoT networks as they control device communication. Using digital twins of edge nodes is thus a node-centric approach to communication-related security orchestration.

To sum things up, IoT device orchestration is enabled by digital twins. Further, IoT network orchestration requires the integration of edge nodes. Edge nodes are crucial as they control sub-networks containing several IoT devices. Therefore, we also include some node-centric aspects of IoT networks in our framework. We consider edge nodes represented by digital twins.

3.4 Formal Model

Concerning the security objectives of IoT, we define requirements targeting the three core capabilities of SOAR. These requirements are essential for the implementation of SOAR platforms and the definition of playbooks. The formal model includes:

Requirement 1 (Orchestration of IoT assets). We denote IoT assets as $A = \{a_1, a_2, ..., a_n\}$, whereby an asset is either a device, network or security tool. These assets are integrated into *SOAR*:

$$a \mapsto SOAR$$

Requirement 2 (Automation of security measures). Automation depends on security measures strategically executed for a specific event $E = \{e_1, e_2, ..., e_o\}$ mapping an asset to a playbook $P = \{p_1, p_2, ..., p_m\}$. Thereby, a playbook is generic and could be linked to one or more assets, located inside a SOAR platform. An asset does not necessarily require a playbook:

$$\exists e \in E: \; e \mapsto SOAR(p \circ a) \; \wedge \; SOAR(p) \mapsto a$$

$$\exists a \in A: \; \neg SOAR(p \circ a)$$

Requirement 3 (Deployment of responses to IoT assets). Response of the SOAR platform depends on whether at least one playbook fulfills or characterizes appropriate security measures for an event. Otherwise, no response is automatically deployed:

$$respond(e) = \begin{cases} SOAR(p) \mapsto a & if \; \exists p \in P: \; SOAR(p \circ e) \\ notify(e) & otherwise. \end{cases}$$

In the next step, we outline our framework, its components and middleware integration.

3.5 Framework Overview

Middleware integration complements our SOAR4IoT framework. We emphasize using digital twin middleware to extend existing SOAR platforms based on the previously established SOAR requirements and IoT security objectives. Figure 2 depicts the SOAR4IoT framework and the middleware integration.

IoT assets. The SOAR4IoT framework is based on IoT assets. IoT assets are classified as IoT devices (i.e., sensors or actuators) or IoT networks (i.e., edge nodes and communication). Intertwined, IoT devices and networks form complex IoT systems accessible through applications. IoT security orchestration implies that IoT assets are known to the SOAR platform. Consequently, there is an information flow from IoT assets to the SOAR platform. In the opposite direction, incident response measures target IoT assets.

Middleware. The SOAR4IoT framework integrates middleware. Besides digital twins, other middleware concepts (e.g., IoT platforms) exist. The middleware is located between IoT assets and the SOAR platform. We argue that middleware is beneficial for abstracting IoT assets. Also, IoT asset data is aggregated. Digital twins, in particular, provide semantic features (e.g., modeling components), a dedicated interface, and different perspectives (e.g., data views)

for orchestration and response. In our case, digital twins offer a comprehensive summary of the asset's (security) state and enable the validation of security measures.

SOAR platform. The SOAR4IoT framework contains a SOAR platform at its core. Most importantly, the SOAR platform emphasizes playbooks and their automation but includes other typical features such as ticketing, user interface, and user management. Data flows from the middleware and connected security tools to the SOAR platform for security orchestration. Then, appropriate incident response measures are disseminated.

Security tools. The SOAR4IoT framework includes security tools. Security tools (e.g., SIEM – Security Information and Event Management systems or IDS – Intrusion Detection Systems) are queried or actively provide security-relevant information. Various Cyber Threat Intelligence sources (e.g., CTI feeds) can also provide input to the SOAR platform and serve as a trigger to response actions. However, incident response actions also address security tools (e.g., updating SIEM rules or disseminating CTI).

4 PROOF OF CONCEPT

We implement the SOAR4IoT framework to validate its feasibility. Defining two use cases, we represent security measures in two playbooks (Section 4.1 and 4.2). More specifically, our experimental setup includes the SOAR platform, replication-based digital twin middleware, and IoT assets (Section 4.3). Further, we demonstrate security orchestration, automation, and incident response and show experimental results (Section 4.4). At last, we conclude our proof of concept by discussing the impact and limitations (Section 4.5).

4.1 Mirai Botnet – Use Case 1

The Mirai malware is scanning IoT devices for vulnerabilities. The attacker's goal is to use the IoT devices for malicious purposes. Consequently, IoT assets need to be secured at the device level. This scenario represents our first use case. The following SOAR playbook describes courses of action to address Mirai-like situations that require firmware updates.

Playbook 1 Mirai Botnet (proactive)

```
1: procedure MIRAI
2:     a ← IoT devices
3:     for all d ∈ a do
4:         e ← CTI for d
5:         if isVulnerable(e, d) and d.checkFirmware() then
6:             d.updateFirmware()
7:     if checkAuthentication(e, a) then
8:         changeAuthentication(a)
9:         a.permitJoin(true, 30s)
```

Organizational security operations to cope with Mirai or similar malware include threat intelligence. Organizations monitor their IoT devices and pay attention to vulnerabilities. Either manually or automated, organizations analyze CTI reports. CTI describes severe vulnerabilities and triggers security operations. Such security operations include checking affected IoT device status and whether a new firmware update is available. This procedure is necessary to keep IoT devices secure and ensure continuous operation. Otherwise, IoT devices can easily contribute to malicious activities, such as distributed denial of service (DDoS) attacks.

4.2 Sybil Attack – Use Case 2

A Sybil attack in IoT describes the fake creation of identities (i.e., IoT assets) in IoT networks [26]. Thereby, attackers attempt to forward data selectively, drop data packets or manipulate data. Consequently, IoT assets need to be secured at the network level. This scenario represents our second use case. The following SOAR playbook describes courses of action to address Sybil attack situations that require device removal.

Playbook 2 Sybil Attack (reactive)

```
1: procedure SYBIL
2:     e ← SIEM event
3:     a ← IoT network
4:     for all n ∈ a do
5:         if isSybilNode(e, n) then
6:             a.removeDevice(n)
7:         a.permitJoin(false)
```

Organizational security operations to cope with a Sybil attack center on adequate monitoring of additional edge nodes or other IoT network components. Digital twins include detailed information about trusted IoT assets. Thus, they can be leveraged once a trigger (e.g., a SIEM event containing the loss of several data packets) from a security tool is received. Assessing the IoT network components, organizations can identify additional fake nodes or even missing ones and start response measures. This procedure is crucial to avoid malfunctioning IoT applications.

Multiple attacks on IoT assets demand SOAR capabilities. We opted for the two exemplary use cases based on the Mirai botnet and Sybil attack to document our SOAR4IoT framework implementation. Next, we describe our technological setup, including hardware and software.

4.3 Experimental Setup

Our experimental setup implements the SOAR4IoT framework. The source code is available in Gitlab[2]. Figure 3 describes our prototypical implementation and documents technology and data flows. This overview is further specified by categorizing and listing the underlying hardware (see Table 3).

IoT assets. We deploy two Xiaomi Aqara temperature sensors and two IKEA Tradfri LED bulb actuators in our lab environment. The sensors measure temperature and humidity. The actuators control brightness, color temperature, and state of connected LEDs. For communication purposes, sensors and actuators use the Zigbee protocol. Additionally, we deploy a Raspberry Pi 3B+ edge node. Zigbee communication between IoT assets and the edge node is controlled with a CC2531 Zigbee USB-Stick. This Zigbee controller is physically plugged into the edge node, but communication is

[2] https://git.ur.de/soar4iot

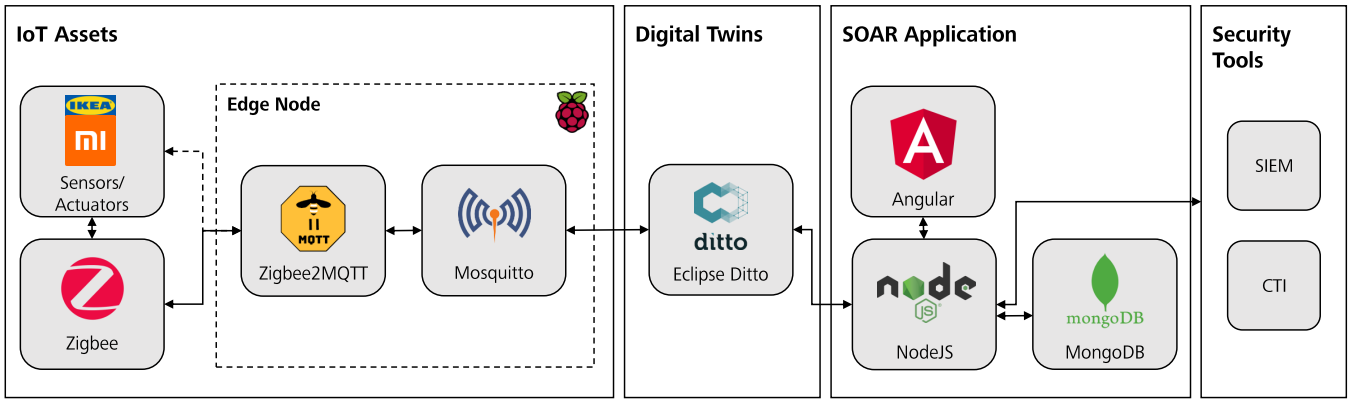


Figure 3: Experimental setting

Table 3: Hardware list

Device	Category	Characteristics
Xiaomi Aqara Temperature	Sensor	WSDCGQ01LM, Zigbee protocol
IKEA Tradfri LED Bulb E14	Actuator	LED1733G7, Zigbee protocol
CC2531 Zigbee USB flash drive	Controller	USB interface, Zigbee protocol
Raspberry Pi 3B+	Edge Node	Raspbian GNU/Linux 11, 1GB RAM, RJ-45 Ethernet
Virtual Machine	Server	Ubuntu 20.04.3 LTS, 16GB RAM, 8 cores, 80GB disc

wireless. At the edge node, the Zigbee data is transformed into MQTT data using the Zigbee2MQTT[3] bridge. Zigbee2MQTT acts as a client sending data from sensors and actuators to the MQTT broker. In our setup, the open-source MQTT broker Mosquitto[4] is installed on the edge node. As MQTT data is structured in topics, Zigbee2MQTT publishes/subscribes to an IoT asset-specific topic (e.g., SOAR4IoT/Lab_Actuator_Bulb1). In the same way, Mosquitto uses MQTT topics for upstream data. Similar IoT assets and edge nodes to our experimental setup might be used as part of an industrial oven or assembly line.

Digital twins. We implement digital twins representing the middleware of our SOAR4IoT framework. For each IoT asset there is one digital twin. Using the open-source digital twin software Eclipse Ditto[5] allows us to integrate and replicate heterogeneous IoT assets. Eclipse Ditto enables message-oriented communication with IoT assets through their digital twin. Besides, it supports the definition of policies (i.e., access control) and the integration of specific brokers for several IoT protocols (e.g., MQTT, AMQP, or CoAP). In our experimental setup, Eclipse Ditto runs on a virtual machine (Ubuntu, 16GB RAM, 8 kernels, and 80GB storage) and connects to Mosquitto.

We design and configure our Eclipse Ditto-based digital twins (see Figure 4). First, we define the primary policy. This policy grants an admin user read and write access to the digital twins and restricts a demo user to read access only. We then create five IoT assets, including the edge node. Each IoT asset is structured using JSON data serialization that defines a primary data schema for its digital twin. We further define messages in Eclipse Ditto. These messages allow users to interact directly with the digital twin of an IoT asset. Digital twins process all messages received from users separately and behave according to the message-defined function. However, not all messages are equally feasible for all IoT assets. While sensors and actuators implement firmware and state/effect messages, the edge node (network administrator) can remove or permit devices to join the network. For instance, if a new IoT asset is invited to onboard the network, the edge node temporarily allows new devices to join for 20 seconds by messaging *permitJoin(true,20)*. Last, we connect Eclipse Ditto to the Mosquitto MQTT broker to establish bidirectional communication between the digital twins and the IoT assets. On the one side, data received from the MQTT broker fills the pre-defined data schemata of the digital twins, and on the other side, digital twins can send commands to the IoT assets.

We opted for Eclipse Ditto because event-based middleware is most qualified for real-time data processing [6] and SOAR use cases. Eclipse Ditto implements the publish/subscribe approach with topics and events (see Figure 1). Nevertheless, there are several ways of implementing digital twins (e.g., physics-based modeling vs. data-driven techniques). Eclipse Ditto uses a data-driven technique with messages to represent IoT asset functions. This type of middleware fits SOAR best, as SOAR does not require simulation capabilities and other aspects of physics-based digital twins. Additionally, Eclipse Ditto is established and used by industrial companies (e.g., Bosch or Aloxy).

SOAR application. The SOAR platform application is deployed on the same virtual machine that runs Eclipse Ditto. We implemented the frontend of the SOAR platform using the Angular[6]

[3]https://www.zigbee2mqtt.io
[4]https://mosquitto.org
[5]https://www.eclipse.org/ditto
[6]https://angular.io

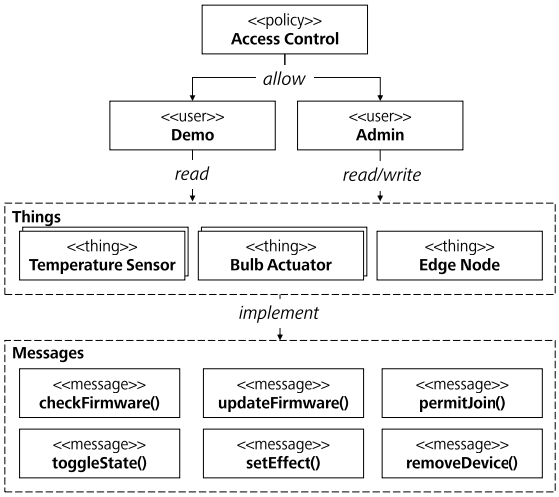


Figure 4: Digital twin setting in Eclipse Ditto

web application framework and Typescript[7]. The backend of our SOAR application is based on NodeJS[8] storing data in a MongoDB[9] database. Developing the SOAR application, we find microservice architecture to fit the purposes of SOAR best. Our SOAR4IoT implementation integrates four main microservices: core app (central microservice), Eclipse Ditto app, CTI app, and a SIEM app. The SIEM app generates pseudo-events used to trigger the execution of playbooks. The CTI app queries vulnerabilities, and the Eclipse Ditto app integrates IoT assets. For ease of deployment, we use Docker Compose and Docker Images. A detailed description of the SOAR platform features is described in Section 4.4.

Security tools. At last, our experimental setup includes the use of security tools. We pursue a twofold approach. First, we integrate existing CTI sources for security-relevant information. Thus, information about vulnerabilities in applications, hardware, or operating systems can be queried from the US National Vulnerability Database (NVD) and is structured by its Common Vulnerabilities and Exposures (CVE) enumeration. CVE descriptions are particularly relevant as attackers widely use available exploits for known vulnerabilities. Also, firmware update information can be queried for our actuators. Second, we directly include a security event feature. This feature is based on predefined security events representing SIEM alarms or incident notifications. Contrary to our experimental setting, organizations will integrate their existing SIEM systems or ticketing systems instead.

4.4 Results

Our research yields results concerning the demonstration of two IoT security use cases. Implementing our digital twins and IoT-centric SOAR application enables security workflows based on user interface (UI) and playbook execution.

We created three playbooks, of which two are addressing the Sybil attack and one the Mirai botnet use case. Therefore, our UI[10] includes an intuitive playbook editor for configuration. In general, the UI of our SOAR application follows a minimalistic approach and provides a single point of contact. Figure 5 documents three main views: (a) security event list, (b) IoT assets (digital twins), and (c) playbooks. Our digital twin and security-focused UI goes beyond the generic Mosquitto UI and the Ansible Semaphore UI[11]. We reason that designing and implementing a customized SOAR application along SOAR requirements is feasible with open-source technologies.

We define a generic `SOAR4IoT` workflow to showcase playbook execution. The workflow involves IoT assets (digital twins), apps, actions, playbooks, and events. Apps (i.e., individual microservices) implement specific actions (e.g., API calls) relevant for security operations. These actions are then structured and instantiated within playbooks. At last, given a specific security event (received by app or created via the UI), playbook execution is triggered. Playbook execution is dependent on event parameters and matching logic. As events are linked to IoT assets, matched playbooks must refer to the same IoT assets. During playbook execution the SOAR core service checks an app's availability, documents action status and starts subsequent actions. The playbook status indicates success, timeout or failure.

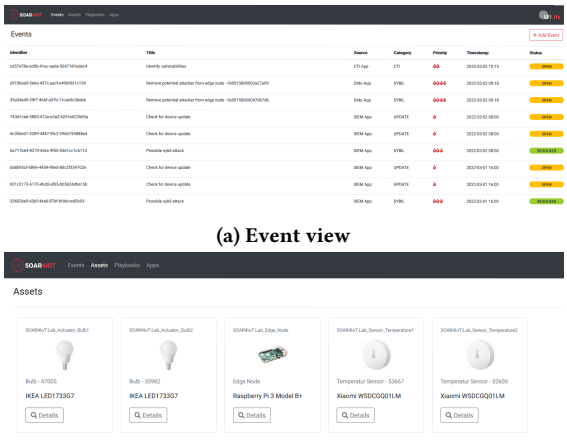
The *Mirai playbook* is used for vulnerable IoT assets (e.g., missing updates or default passwords). Its actions include fetching CTI data, updating IoT assets OTA, and requesting analysts to check the IoT assets' authentication manually. Our experimental setup includes no vulnerable IoT assets, so we define a repetitive update event. This event triggers playbook execution regularly. We successfully achieved firmware updates for the IKEA Tradfri LED bulb using digital twin messaging functions and Zigbee2MQTT. Changing authentication and validating playbook execution (e.g., comparing firmware versions) are subsequent manual tasks.

The *Sybil playbooks* address rogue devices. The actions include identifying and removing Sybil nodes from the network. This is followed by preventing new devices to join the network. Leveraging our SIEM app, we create events indicating a possible Sybil attack. In `SOAR4IoT`, the security analyst can then execute a playbook to analyze IoT assets not represented by a digital twin. If so, new removal events are created and listed with the asset's network address (see Figure 5a). A security analyst can also check manually if the network address is linked to a known IoT asset (see Figure 5b). The analyst is assisted in resolving the removal event by executing the corresponding playbook (see Figure 5c). Observing the status of playbook execution, the Sybil node is successfully removed. Validation might include comparing connected IoT assets at the edge node before and after playbook execution. In general, playbook selection depends on analyst's assessment of whether a playbook's actions meet the desired objective.

Lessons Learned. We learned that a logical separation of security orchestration and incident response using microservices benefits the SOAR application. We consider security orchestration a data collection task (e.g., querying device status or available CTI) and

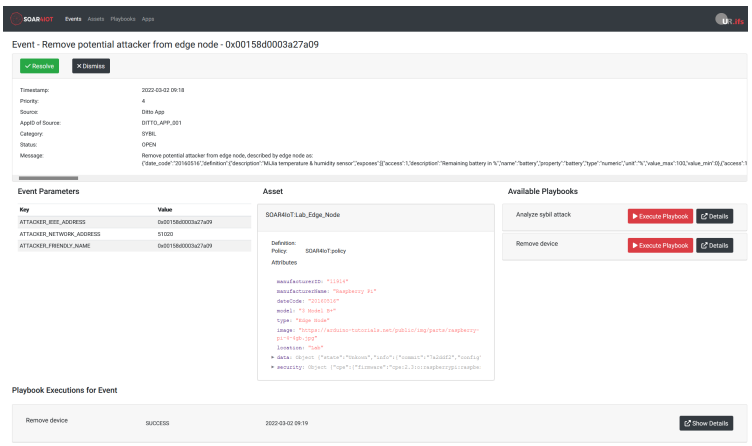
[7]https://www.typescriptlang.org
[8]https://nodejs.org
[9]https://www.mongodb.com

[10]https://soar4iot.ur.de
[11]https://github.com/ansible-semaphore/semaphore



(a) Event view

(b) Asset view



(c) Playbook execution view

Figure 5: SOAR4IoT UI

incident response a modification task. Digital twins prove relevant as they provide an additional layer with unified access and control to make the IoT manageable for security purposes. We experienced excellent feedback from the Eclipse Ditto community during development. It leads us to conclude that, in practice, digital twins go beyond the functional scope of IoT platforms, and digital twin research is relatively narrow. Overall, SOAR application development is a challenging task, but complexity can be reduced (e.g., via microservices, virtualization, and deployment tools).

4.5 Discussion

We discuss both the scientific and practical impact of our SOAR4IoT framework before mentioning limitations.

Scientific impact. Only a few academic works have addressed security orchestration and SOAR platforms. Our work is an attempt towards leveling the playing field with the large number of commercial SOAR platforms. This attempt includes a list of SOAR platform features. Eventually, documented by our SOAR application, open-source technologies can be used. We contrast user reluctance with the potential use cases for security and open-source frameworks like Eclipse Ditto digital twins. We direct attention to digital twins for security operations beyond current simulations.

Practical impact. To cope with the current IoT trend, organizations must manage IoT assets and extend existing SOAR platforms. Our work can be seen as an innovative approach using open-source technologies. Pointing at the benefits of small-scale, customized SOAR platforms, we contrast commercial SOAR platforms. Our publicly available source code can serve for future extensions.

Limitations. There are several aspects that our work does not address. We attempted to select appropriate technologies and justify our decisions, but there are no best practices for digital twins in cybersecurity. CPS Twinning[12] is an alternative digital twin framework worth further investigation. Additionally, we excluded security for IoT cloud applications (e.g., predictive maintenance) typically used with IoT assets. Our SOAR application does not consider communication features (e.g., messaging or task delegation) found in commercial SOAR platforms. Access control, available for digital twins, is missing at SOAR application level but is required in production environments. Due to the small quantity of IoT assets, we can not assess the scalability of our SOAR application. Since many IoT devices will never experience updates, organizations should pay attention when buying them. Also, we did not exploit the full range of possibilities as our SOAR application integrates only a few security tools.

[12]https://github.com/sbaresearch/cps-twinning

5 CONCLUSION AND FUTURE WORK

The question *"How to use Security Orchestration, Automation and Response for the Internet of Things?"* was the starting point of our work. While investigating the SOAR concept and SOAR platforms, we derived a detailed understanding of SOAR and its requirements. Defined by its orchestration, automation, and incident response capabilities, SOAR is mainly centered on playbooks and security tool integration for security operations. Extending the security operations to the IoT is a necessary step, as IoT attacks and IoT objectives show. Among different options to secure the IoT, digital twins provide a feasible, lightweight solution abstracting heterogeneous assets. Thus, our `SOAR4IoT` framework integrates a digital twin-based middleware. More precisely, we establish a prototypical implementation using Eclipse Ditto and a microservice SOAR application. Implications of our conceptual design and SOAR4IoT implementation include the following:

- Digital twins provide abstraction and a unified interface for the plethora of IoT assets. The security community should further compare different digital twin frameworks' abilities (e.g., advanced behavior or process modeling). To the best of our knowledge, our Eclipse Ditto implementation is the first, with security use cases built on top. It can serve as a

stepping stone for sophisticated intrusion detection, threat notifications, and life cycle analyses.
- SOAR is about playbooks. Thus, research should focus on the great potential of playbooks. We expect benefits of identifying additional use cases (e.g., execution of playbooks against a group of IoT assets) and formalizing playbook logic. Future work should assist security analysts from initial (automated) playbook creation based on manufacturers' course of action recommendations to playbook × event matching and (prioritized) execution. Therefore, playbooks must consider organizational incident response processes and their underlying principles.

From a security management perspective, SOAR4IoT has two great strengths. First, it is crucial to see the full picture and properly manage organizational assets. And second, security management must plan security operations strategically to maintain the security posture. Digital twins and SOAR playbooks foster both aspects. However, this requires initial resources to implement the SOAR4IoT framework and strategic decisions whether to use playbooks to their full extent. We believe it is worth the effort due to new avenues and security possibilities.

ACKNOWLEDGMENTS

This research was supported by the German Federal Ministry of Education and Research (BMBF) as part of the DEVISE project. This research was supported by the German Federal Ministry for Economic Affairs and Climate Action (BMWK) as part of the Secure Industrial Semantic Sensor Cloud (SISSeC) project.

REFERENCES

[1] Ala I. Al-Fuqaha, Mohsen Guizani, Mehdi Mohammadi, Mohammed Aledhari, and Moussa Ayyash. 2015. Internet of Things: A Survey on Enabling Technologies, Protocols, and Applications. *IEEE Communications Surveys & Tutorials* 17, 4 (2015), 2347–2376. https://doi.org/10.1109/COMST.2015.2444095

[2] Jan Bauwens, Peter Ruckebusch, Spilios Giannoulis, Ingrid Moerman, and Eli De Poorter. 2020. Over-the-Air Software Updates in the Internet of Things: An Overview of Key Principles. *IEEE Communications Magazine* 58, 2 (2020), 35–41. https://doi.org/10.1109/MCOM.001.1900125

[3] Stefan Boschert, Christoph Heinrich, and R. Rosen. 2018. Next Generation Digital Twin. In *Proceedings of the 12th. International Symposium on Tools and Methods of Competitive Engineering (TMCE'18)* (Las Palmas de Gran Canaria, Spain), I. Horvath, J.P. Suarez Riviero, and P.M. Hernandez Castellano (Eds.). 209–218.

[4] Ismail Butun, Patrik Österberg, and Houbing Song. 2020. Security of the Internet of Things: Vulnerabilities, Attacks, and Countermeasures. *IEEE Communications Surveys & Tutorials* 22, 1 (2020), 616–644. https://doi.org/10.1109/COMST.2019.2953364

[5] Seoyun Choi and Jong-Hyouk Lee. 2020. Blockchain-based distributed firmware update architecture for IoT devices. *IEEE Access* 8 (2020), 37518–37525. https://doi.org/10.1109/ACCESS.2020.2975920

[6] Mauro A. A. da Cruz, Joel José Puga Coelho Rodrigues, Jalal Al-Muhtadi, Valery Korotaev, and Victor Hugo C. de Albuquerque. 2018. A Reference Model for Internet of Things Middleware. *IEEE Internet of Things Journal* 5, 2 (2018), 871–883. https://doi.org/10.1109/JIOT.2018.2796561

[7] Violeta Damjanovic-Behrendt. 2018. A digital twin architecture for security, privacy and safety. *ERCIM News* 115 Special Issue "Digital Twins (2018).

[8] Marietheres Dietz, Manfred Vielberth, and Günther Pernul. 2020. Integrating digital twin security simulations in the security operations center. In *Proceedings of the 15th International Conference on Availability, Reliability and Security (ARES'20)* (Virtual Event), Melanie Volkamer and Christian Wressnegger (Eds.). 18:1–18:9. https://doi.org/10.1145/3407023.3407039

[9] Matthias Eckhart and Andreas Ekelhart. 2019. Digital twins for cyber-physical systems security: State of the art and outlook. *Security and quality in cyber-physical systems engineering* (2019), 383–412.

[10] Matthias Eckhart, Andreas Ekelhart, and Roland Eisl. 2021. Digital Twins for Cyber-Physical Threat Detection and Response. *ERCIM News* 127 (2021).

[11] Forrester Consulting. 2020. *The 2020 State Of Security Operations.* Technical Report E-46260. Forrester Research (commissioned by Palo Alto Networks), Cambridge, England.

[12] Janis Grabis, Janis Stirna, and Jelena Zdravkovic. 2021. A Capability Based Method for Development of Resilient Digital Services. In *Enterprise Information Systems*, Joaquim Filipe, Michał Śmiałek, Alexander Brodsky, and Slimane Hammoudi (Eds.). Vol. 417. 498–516. https://doi.org/10.1007/978-3-030-75418-1_23

[13] Chadni Islam, Muhammad Ali Babar, and Surya Nepal. 2019. A Multi-Vocal Review of Security Orchestration. *Comput. Surveys* 52, 2, Article 37 (2019), 45 pages. https://doi.org/10.1145/3305268

[14] Chadni Islam, Muhammad Ali Babar, and Surya Nepal. 2020. Architecture-Centric Support for Integrating Security Tools in a Security Orchestration Platform. In *Proceedings of the 14th. European Conference on Software Architecture (ECSA'20)* (L'Aquila, Italy), A. Jansen, I. Malavolta, H. Muccini, I. Ozkaya, and O. Zimmermann (Eds.). Springer, Cham, Germany, 165–181. https://doi.org/10.1007/978-3-030-58923-3_11

[15] Bernd Jäger. 2015. Security Orchestrator: Introducing a Security Orchestrator in the Context of the ETSI NFV Reference Architecture. In *Proceedings of the 14th. IEEE International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom'15)* (Helsinki, Finland). IEEE, New York, NY, USA, 1255–1260. https://doi.org/10.1109/Trustcom.2015.514

[16] Xingwei Liang and Yoohwan Kim. 2021. A Survey on Security Attacks and Solutions in the IoT Network. In *Proceedings of the 11th. IEEE Annual Computing and Communication Workshop and Conference (CCWC'21)* (Virtual Event). IEEE, New York, NY, USA, 853–859. https://doi.org/10.1109/CCWC51732.2021.9376174

[17] Parushi Malhotra, Yashwant Singh, Pooja Anand, Deep Kumar Bangotra, Pradeep Kumar Singh, and Wei-Chiang Hong. 2021. Internet of Things: Evolution, Concerns and Security Challenges. *Sensors* 21, 5 (2021), 1809. https://doi.org/10.3390/s21051809

[18] Brendan Moran, Hannes Tschofenig, David Brown, and Milosch Meriac. 2021. *A Firmware Update Architecture for Internet of Things.* Technical Report. RFC 9019. Internet Engineering Task Force (IETF).

[19] Claudio Neiva, Craig Lawson, Toby Bussa, and Gorka Sadowski. 2020. *2020 Market Guide for Security Orchestration, Automation and Response Solutions.* Technical Report. Gartner.

[20] Netscout. 2020. *Netscout Threat Intelligence Report (Issue 6).* Technical Report. Netscout.

[21] Savannah Norem, Ashley E Rice, Samantha Erwin, Robert A Bridges, Sean Oesch, and Brian Weber. 2021. A Mathematical Framework for Evaluation of SOAR Tools with Limited Survey Data. https://doi.org/10.48550/arXiv.2112.00100

[22] Megan Nyre-Yu. 2021. Identifying Expertise Gaps in Cyber Incident Response: Cyber Defender Needs vs. Technological Development. In *Proceedings of the 54th. Hawaii International Conference on System Sciences (HICCS'21)* (Wailea, Hawaii). 1978–1987.

[23] OASIS. 2020. *Open Command and Control (OpenC2) Language Specification Version 1.0 - Committee Specification 02.* OASIS. https://docs.oasis-open.org/openc2/oc2ls/v1.0/cs02/oc2ls-v1.0-cs02.html Last accessed 2021-11-20.

[24] OASIS. 2021. *CACAO Security Playbooks Version 1.0 - Committee Specification 01.* OASIS. https://docs.oasis-open.org/cacao/security-playbooks/v1.0/security-playbooks-v1.0.html Last accessed 2021-11-20.

[25] Palo Alto Networks. 2020. *Measuring the ROI of an Incident Response Platform.* Technical Report UC-031220. Palo Alto Networks, Santa Clara, CA, USA.

[26] Anjana Rajan, J. Jithish, and Sriram Sankaran. 2017. Sybil attack in IOT: Modelling and defenses. In *Proceedings of the 6th. International Conference on Advances in Computing, Communications and Informatics, ICACCI'17* (Manipal, India). IEEE, New York, NY, USA, 2323–2327. https://doi.org/10.1109/ICACCI.2017.8126193

[27] Mohammad Abdur Razzaque, Marija Milojevic-Jevric, Andrei Palade, and Siobhán Clarke. 2016. Middleware for Internet of Things: A Survey. *IEEE Internet of Things Journal* 3, 1 (2016), 70–95. https://doi.org/10.1109/JIOT.2015.2498900

[28] Daniel Schlette, Marco Caselli, and Günther Pernul. 2021. A Comparative Study on Cyber Threat Intelligence: The Security Incident Response Perspective. *IEEE Communications Surveys & Tutorials* 23, 4 (2021), 2525–2556. https://doi.org/10.1109/COMST.2021.3117338

[29] Daniel Schlette, Florian Menges, Thomas Baumer, and Günther Pernul. 2020. Security enumerations for cyber-physical systems. In *IFIP Annual Conference on Data and Applications Security and Privacy (DBSec'20)* (Virtual Event). Springer, Cham, Germany, 64–76.

[30] Tara Seils. 2021. IoT Attacks Skyrocket, Doubling in 6 Months. https://threatpost.com/iot-attacks-doubling/169224/. Last accessed 2021-02-21.

[31] Manfred Vielberth, Fabian Bohm, Ines Fichtinger, and Günther Pernul. 2020. Security Operations Center: A Systematic Study and Open Challenges. *IEEE Access* 8 (2020), 227756–227779. https://doi.org/10.1109/ACCESS.2020.3045514

[32] Jingbin Zhang, Meng Ma, Ping Wang, and Xiao-dong Sun. 2021. Middleware for the Internet of Things: A survey on requirements, enabling technologies, and solutions. *Journal of Systems Architecture* 117 (2021), 102098. https://doi.org/10.1016/j.sysarc.2021.102098