

RTS-ABAC: Real-Time Server-Aided Attribute-Based Authorization & Access Control for Substation Automation Systems

Moritz Gstür[✉], Gustav Keppler[✉], Mohammed Ramadan[✉], Ghada Elbez[✉], and Veit Hagemeyer[✉]

Karlsruhe Institute of Technology (KIT)

{moritz.gstuer,gustav.keppler,mohammed.ramadan,ghada.elbez,veit.hagemeyer}@kit.edu

Abstract. Critical energy infrastructures increasingly rely on information and communication technology for monitoring and control, which leads to new challenges with regard to cybersecurity. Recent advancements in this domain, including attribute-based access control (ABAC), have not been sufficiently addressed by established standards such as IEC 61850 and IEC 62351. To address this issue, we propose a novel real-time server-aided attribute-based authorization and access control for time-critical applications called RTS-ABAC. We tailor RTS-ABAC to the strict timing constraints inherent to the protocols employed in substation automation systems (SAS). We extend the concept of conventional ABAC by introducing real-time attributes and time-dependent policy evaluation and enforcement. To safeguard the authenticity, integrity, and non-repudiation of SAS communication and protect an SAS against domain-typical adversarial attacks, RTS-ABAC employs mandatory authentication, authorization, and access control for any type of SAS communication using a bump-in-the-wire (BITW) approach. To evaluate RTS-ABAC, we conduct a testbed-based performance analysis and a laboratory-based demonstration of applicability. We demonstrate the applicability using intelligent electronic devices, merging units, and I/O boxes communicating via the GOOSE and SV protocol. The results show that RTS-ABAC is able to secure low-latency communication between SAS devices, as up to 99.82 % of exchanged packets achieve a round-trip time below 6 ms. Moreover, the results of the evaluation indicate that RTS-ABAC is a viable solution to enhance the cybersecurity not only in a newly constructed SAS but also via retrofitting of existing substations.

Keywords: Attribute-Based Access Control (ABAC) · Smart Grid, Digital Substation · Substation Automation System · Cyber-Physical System · Low-Latency Communication · Bump-in-the-Wire · IEC 61850 · IEC 62351

1 Introduction

Modern operational technology (OT) increasingly relies on information and communication technology (ICT) for monitoring and control [29]. This development leads to new possibilities including the integration of distributed OT into supervisory control and data acquisition (SCADA) systems. Nevertheless, new challenges arise from the increased usage of ICT in OT systems. Although a variety of cybersecurity solutions exist for IT, migration of existing approaches to the OT domain may not be a viable solution due to the differing system characteristics, risks, and priorities. While the prevention of unauthorized access represents the core objective of IT security approaches, OT systems and especially OT-based critical infrastructure prioritize system availability and reliability.

In the energy-related sector, the infrastructure currently transforms from traditional top-down energy transmission and distribution systems to so-called smart grids with bidirectional data and energy flows [8]. The distribution of formerly centralized entities, such as power plants and control centers, necessitates not only changes in energy infrastructure but also leads to an increased reliance on communication solutions. Historical evidence indicates that economically or politically motivated adversaries pose a risk to OT systems, including energy-related systems [2]. The Communications Security Establishment Canada [2] published a list of 28 OT-related cybersecurity incidents between 2010 and 2020, including incidents in energy-related sectors. These incidents comprise 13 state-sponsored incidents, 13 cybercrime incidents, and two incidents perpetrated by thrill-seeking individuals. The state-sponsored incidents include the Stuxnet malware deployed in

Iranian nuclear power and enrichment facilities in 2010 [9], the Shamoon malware used against Saudi Aramco in 2012 [10], the Blackenergy malware used to attack Ukrainian power distribution systems in 2015 [4], the Industroyer/CrashOverride malware used to shut down remote terminal units of a Ukrainian power transmission facility in 2016 [31, 3], and the Triton/Trisis malware used to attack Triconex safety instrumented system controllers in 2017 [20].

In the following, we focus on a particular part of smart electricity grids called substation automation systems (SAS). An SAS represents the entirety of communication and control equipment of a substation [26], which is a facility of a high-voltage electricity grid connecting power transmission and distribution lines that use different voltage levels [25]. The tasks of an SAS are time-critical and have to be executed reliably, as the electricity sector and its substations are critical infrastructures. Although standards regarding the communication within an SAS are widely accepted and utilized, cybersecurity continues to confront unresolved challenges. The IEC 61850 series provides standards for the communication networks of digital energy systems, which ensure seamless communication and interoperability in smart grids [14]. However, cybersecurity is not an objective of these standards. To overcome this problem, the IEC 62351 standard series provides standardized security means for communication compliant to IEC 61850 and a role-based access control (RBAC) model for power systems [16, 18].

The present paper focuses on enhancing the cybersecurity of SAS communication without compromising its time criticality, as the strict time constraints of the low latency communication in an SAS are key factors for the cybersecurity [19, 5]. Thus, we propose a real-time server-aided attribute-based authorization and access control (RTS-ABAC) approach. The objective of RTS-ABAC is to provide secure protocols and algorithms for SAS communication, which satisfy the SAS security requirements, including integrity, authenticity, access control, and non-repudiation. Furthermore, RTS-ABAC takes the specific characteristics, risks, and priorities of OT, ICS, and SAS into account. To address the aforementioned objectives and considerations, the present paper comprises the following main contributions:

- C1** A novel function-oriented component-based SAS communication architecture, which integrates SAS components as well as components responsible for authentication, authorization, and access control.
- C2** Two novel strategies for the employment of ABAC in an SAS:
 - C2.1** A strategy for the representation, specification, and classification of ABAC policies, which introduces the notion of time-dependent attributes and policies.
 - C2.2** A strategy for the evaluation of time-dependent access control policies and the enforcement of the respective access control decisions in distributed real-time systems.
- C3** Two novel communication protocols specifying the message exchanges of RTS-ABAC:
 - C3.1** A delegated attribute-based authorization protocol responsible for the access control policy creation, management, storage, and distribution.
 - C3.2** A delegated decision enforcement protocol responsible for the exchange, verification, and enforcement of access decisions.
- C4** An experimental evaluation analyzing the performance and applicability of RTS-ABAC:
 - C4.1** A performance analysis of RTS-ABAC using different authentication algorithms in a testbed based on off-the-shelf hardware.
 - C4.2** A laboratory-based demonstration of applicability with industrial SAS devices communicating via the GOOSE and SV protocol.

The remainder of the paper is organized as follows. In section 2, we present the fundamentals and related literature of RTS-ABAC. In section 3, the architecture and communication protocols of RTS-ABAC are introduced and our novel access control policy specification, classification, and evaluation strategies are discussed. We present the evaluation of RTS-ABAC, which consists of a performance analysis and a laboratory-based demonstration of applicability, in section 4. In section 5, we present a summary and provide insights into prospective future research.

2 Preliminaries

2.1 Attribute-Based Access Control (ABAC)

ABAC is an access control model enabling access decisions based on attributes associated with subjects, objects, actions, and the environment of a system [30]. Within the context of ABAC,

an attribute is a characteristic containing information in the form of a name-value pair [13]. A policy represents a rule based on which an access decision is taken for specific attributes. As a consequence, a policy can be seen as a relationship between subject, object, environment, and operation attributes describing under which circumstances the access control mechanism grants or denies an access request.

As the eighth part of the IEC 62351 series provides a RBAC concept for power systems [18], RBAC represents the predominant access control model employed in the smart grid domain and related literature. Since a role can be seen as a subject attribute evaluated by the access control mechanism to take an access decision, RBAC represents a special and limited case of ABAC regarding the attributes used [13]. Accordingly, an advantage of ABAC is the higher flexibility regarding multifactor policy expression. Moreover, ABAC can take access control decisions based on ad-hoc knowledge.

2.2 Substation Automation Systems (SAS): Architecture & Communication Protocols

RTS-ABAC is tailored to the communication and control systems of digital substations in smart electricity grids. However, the main concepts may also be applied to other ICSs with similar requirements and constraints. The SAS architecture shown in Figure 1 is based on the IEC 61850 standards [14] and consists of three layers called station, bay, and process level. Each of the layers consists of different devices and provides different control and automation functions. Devices restricted to the transformation and provision of measurement and control values are referred to as merging units (MU). Intelligent electronic devices (IED) at bay level supervise the operation of lower-level devices. Human machine interfaces (HMI) as well as wide area network (WAN) gateways are required for on-site and remote monitoring and control of an SAS.

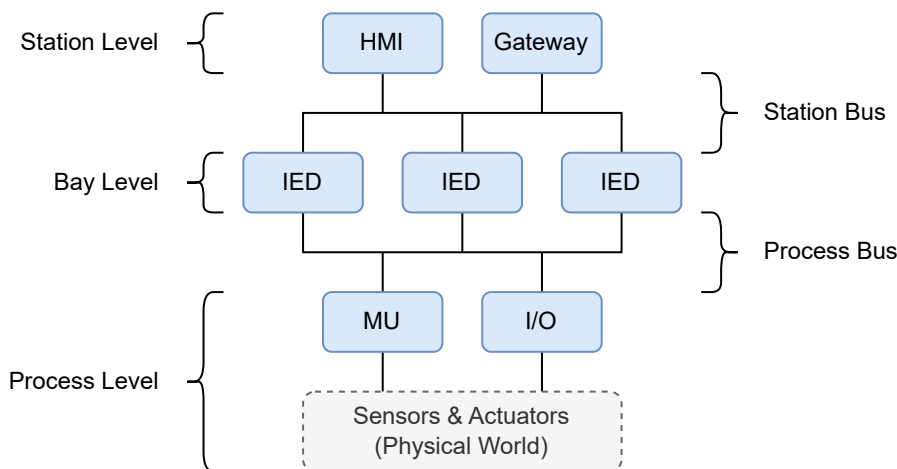


Fig. 1: Three-layered architecture of an SAS.

The types of messages exchanged between devices of an SAS are shown in Table 1. In addition to the IEC 61850-5 message types and performance classes [14], the table shows the associated SAS protocols of each message type [17]. Communication within a substation is segregated into two networks: the process bus and the station bus. The process bus operates using protocols based on Ethernet multicast or broadcast communication. The generic object oriented substation event (GOOSE) protocol is used to transmit commands and statuses like protection trips and interlocking signals. The sampled values (SV) protocol is used by MUs to transmit digitized measurements, such as current and voltage measurements, to IEDs at bay level. The real-time nature of these protocols necessitates time synchronization, provided by the (simple) network time protocol (NTP / SNTP) and the precision time protocol (PTP). The communication at the station bus is session-based unicast communication with less strict time requirements compared to the process bus. The

manufacturing message specification (MMS) protocol is used for station bus communication in an SAS. The configuration of data exchanges, for all buses and protocols, is formally defined by the system configuration description language (SCL), specified in IEC 61850-6 [15].

Table 1: Types of messages exchanged between SAS devices.

Message Name	Message Type	Performance Class	Transfer Time	SAS Protocol
Fast Messages	1A (Trip)	P1	$\leq 3 \text{ ms}$	GOOSE
		P2	$\leq 10 \text{ ms}$	
	1B (Others)	P3	$\leq 20 \text{ ms}$	
Medium Speed Messages	2	P4	$\leq 100 \text{ ms}$	MMS
Low Speed Messages	3	P5	$\leq 500 \text{ ms}$	MMS
		P6	$\leq 1000 \text{ ms}$	
Raw Data Messages	4	P7	$\leq 3 \text{ ms}$	SV
		P8	$\leq 10 \text{ ms}$	
File Transfer Functions	5	P9	$\leq 10000 \text{ ms}$	MMS
Command Messages & File Transfer with Access Control	6	P10	$\leq 500 \text{ ms}$	i.a. PTP & SNTP
		P11	$\leq 1000 \text{ ms}$	
		P12	$\leq 10000 \text{ ms}$	

2.3 Related Work

An authenticated communication approach for network packets between IEDs and MUs is presented by Ishchenko and Nuqui [19]. Based on the concept of retrofitting, a bump-in-the-wire (BITW) device called security filter is presented as an add-on device between IEDs and Ethernet-based communication busses using the GOOSE or SV protocol. The security filter appends MAC tags to outgoing messages of an IEDs and verifies incoming MAC tags. Consequently, the communication busses are secured against unauthenticated messages achieving the security goals integrity and authenticity. The authors show that the security filter is able to meet the IEC 61850 performance requirements of GOOSE and SV [14] using a hash message authentication code (HMAC) and galois message authentication code (GMAC) algorithm even on commodity off-the-shelf ARM hardware. The architecture and security procedures of RTS-ABAC are inspired by the security filter [19]. We use the concept of authenticated communication as a foundation for secure communication in substations. However, RTS-ABAC is as an authentication-dependent but scheme-agnostic approach that safeguards not only authentication-related but also authorization-related security requirements. Moreover, RTS-ABAC extends the employed access control from identity-based to attribute-based authorization to enable more complex access control policies.

An authentication and encryption approach for GOOSE and SV communication is presented by Rodriguez *et al.* [27]. The authors propose an FPGA-implementable hardware architecture for the encryption and authentication of GOOSE and SV packets at wire-speed conforming to IEC 62351:2020 [16]. The authors conduct an evaluation of the presented architecture using simulation-based and hardware-based timing results. As stated by the authors, the hardware implementation is able to process GOOSE and SV packets with a fixed latency in the order of microseconds. Besides securing the intra-substation communication based on the GOOSE and SV protocol, RTS-ABAC extends the idea of providing integrity, authenticity, and non-repudiation to any type of SAS communication. In contrast to the hardware architecture by Rodriguez *et al.* [27], RTS-ABAC is software-based. Our software-based solution achieves flexibility and interoperability with regard to different ICS environments including different protocols and algorithms used. Moreover, RTS-ABAC is cost-efficient and requires neither special equipment nor specialized human operators due to its deployment onto inexpensive off-the-shelf hardware.

An access control approach driven by ABAC policies for smart grid systems including substations is presented by Ruland and Sassmannshausen [28]. The presented access control approach is based on an access control firewall, which splits the station bus and connects the outer and inner station bus by processing access requests of connected devices. The access control firewall enforces access request decisions based on ABAC policies. The policies used in the presented approach are defined, communicated, and evaluated using the extensible access control markup language (XACML) [24]. Accordingly, the access request decisions are made by a policy decision point (PDP) that can either be part of the access control firewall or be implemented as an external server on the outer station bus. RTS-ABAC employs ABAC similarly to the access control approach presented by Ruland and Sassmannshausen [28]. In addition to the employment of ABAC to secure the communication between devices on the station bus, RTS-ABAC controls access requests to any device within the substation that requires access control. For this purpose, we use a distributed ABAC firewall instead of a single firewall. As a consequence, RTS-ABAC does not represent a communication bottleneck or single point of failure of an SAS.

A real-time capable ABAC approach is presented by Burmester, Magkos, and Chrissikopoulos [1]. According to the authors, employing ABAC in real-time availability scenarios can be challenging due to the dynamic and large event space determining the attribute values. In these real-time availability scenarios, events threatening the system state might not be addressed within strict time limits if attribute values are not available in time. For this purpose, the authors propose an extended ABAC model that is based on time-dependent attributes to support availability within the strict time constraints of cyber-physical systems. The authors demonstrate the real-time ABAC for IP multicast in Trusted Computing (TC) compliant networks. Moreover, as stated by the authors, the extended ABAC model is applicable to an SAS and medical cyber-physical systems. The access policy classification and evaluation approach proposed in our paper is inspired by the authors' concept of real-time attributes [1]. While relying on a similar concept of real-time attributes, RTS-ABAC differs in its handling of these attributes. In our approach, real-time attributes together with static attributes form dynamic and static ABAC policies, i.e., access control policies whose evaluation does or does not rely on time-variable subject, object, environment, or action attributes. Consequently, we extend the notion of real-time attributes to dynamic and static ABAC policies. Moreover, we introduce novel strategies to represent, specify, classify, evaluate, and enforce these novel policies.

A distributed RBAC (DRBAC) approach for subscription-based remote network services is presented by Ma and Woodhead [22, 23]. DRBAC is a distributed authentication and role-based authorization framework. The distributed authentication is realized by delegating the authentication of users to the corresponding subscribing institutions. The DRBAC framework extends conventional RBAC by adding the concept of distributed roles shared by the resource provider and resource subscribers. Thus, it enables access control policies associated with distributed roles rather than subject identities, which leads to an increase in scalability and manageability of access control. Moreover, the authors state that the DRBAC approach supports temporal, contextual, and cardinality constraints to enhance the semantic expressiveness of access control and enable the definition of higher-level organizational policies. Our authentication and authorization approach utilizes the concept of delegation presented by Ma and Woodhead [22, 23]. While the authors illustrate the concept of delegation within the context of subscription-based remote network service environments, RTS-ABAC employs delegation in substations. RTS-ABAC not only relies on the concept of delegation for authentication but also for authorization and access control via PDPs that make access control decisions for resource requests in place of SAS devices.

An ABAC approach for the data distribution service (DDS), a data-centric machine-to-machine publisher-subscriber standard, is presented by Kim, Kim, and Alaerjan [21]. Due to an increasing utilization of publisher-subscriber communication across different domains, including the Internet of things (IoT) as well as OT systems, DDS security is increasingly relevant. To tackle this problem, the authors present an approach enhancing the authorization of DDS by employing ABAC. The authors evaluate the approach by implementing it based on XACML and applying it to a patient monitoring system in the healthcare domain. The presented implementation is capable of handling 150 access requests simultaneously with an average communication time of 4.95 ms. According to the authors, the approach is applicable to the healthcare domain and satisfies the transfer time requirements of all message types of IEC 61850-5 [14] except for type 1A and type 4. While Kim, Kim, and Alaerjan demonstrate the applicability to medium and low speed IEC 61850-5 message

types [21], RTS-ABAC provides security to all SAS communication protocols, including the low-latency message types 1A and 4. Moreover, RTS-ABAC highlights the importance of not only authorization but also authentication in order to enhance the security of low-latency communication in an SAS in a timely and reliable manner.

3 RTS-ABAC Approach

In this section, we introduce our real-time server-aided attribute-based authorization and access control (RTS-ABAC) approach. RTS-ABAC represents a significant advancement over our initial server-aided attribute-based authorization and access control (SABAAC) approach for SAS [12]. RTS-ABAC safeguards the principle of least privilege and separation of duties. Furthermore, as all message exchanges are secured using cryptographic authentication mechanisms, RTS-ABAC safeguards the data frame payload integrity, data frame sender authenticity, and data frame authorship non-repudiation. However, the availability of secure cryptographic authentication mechanisms is assumed by RTS-ABAC, as it is beyond the scope of the present paper. To satisfy the strict time constraints of the SAS domain, the expressive and flexible yet computationally expensive ABAC policies are handled in a server-aided manner.

As the objective of an adversary is typically unachievable by a single operation but rather requires a sequence of attacks to be achieved, we model the objectives of RTS-ABAC as an attack tree, which is shown in Figure 2. The attacks depicted in orange are integrity-focused, i.e., aim to disturb the system’s integrity by transitioning the system into a state that is beneficial for an adversary. The attacks depicted in purple are authenticity-focused and enable an adversary to impersonate a legitimate subject of the system, thereby abusing the subject’s granted privileges. We identify two primary types of adversaries for RTS-ABAC. The first adversary has the objective of compromising the system by disrupting communication between two or more legitimate system subjects. In order to compromise the communication integrity, the adversary may either replay messages that have been captured on the network, or modify them. While replaying only requires the adversary to be able to eavesdrop on communication, message modification additionally requires the adversary to masquerade as a legitimate subject. This can be achieved by breaking authenticity protection mechanisms, such as digital signatures, or by colluding with other adversaries, such as infiltrated system devices. The second adversary has the objective of compromising the system by directly attacking the SAS devices. This adversary may seek to either disrupt the device’s availability or compromise its integrity. A device’s integrity can be compromised by modifying its state using accessible service interfaces. Accordingly, an adversary may either create a legitimate request, in case of an unsecured service, or forge a request.

3.1 Authorization & Access Control Architecture

RTS-ABAC is based on a function-oriented component-based architecture. An overview of the architecture, components, and protocols is shown in Figure 3. The component-based architecture consists of four functional units. The functional units are inspired by the access control mechanism functional points presented by Hu *et al.* [13]. Each functional unit is represented by a component that offers a set of services. The four components of the architecture are defined below:

Policy Administration & Storage Point (PASP) The PASP offers services for policy creation, storage, management, and distribution. Accordingly, it provides interfaces for policy management services to human operators. Moreover, the PASP acts as a repository for created policies and changes to policies.

Policy Decision Point (PDP) The PDP takes access control decisions by evaluating policies. The PDP takes an access decision on request of another device and provides the decision to the requestor. Thus, it is responsible for the policy and attribute evaluation workflow, including the retrieval of required attributes and speedup techniques such as access decision caching and policy evaluation precomputation. Moreover, the PDP incorporates the services provided by the context handler, which is introduced by Hu *et al.* [13].

Attribute Administration & Storage Point (AASP) The AASP offers services for attribute creation, management, and distribution. The AASP supports the authorization procedure by providing attributes to the PDPs. Accordingly, the AASP corresponds to the Policy Information Point (PIP) introduced by Hu *et al.* [13].

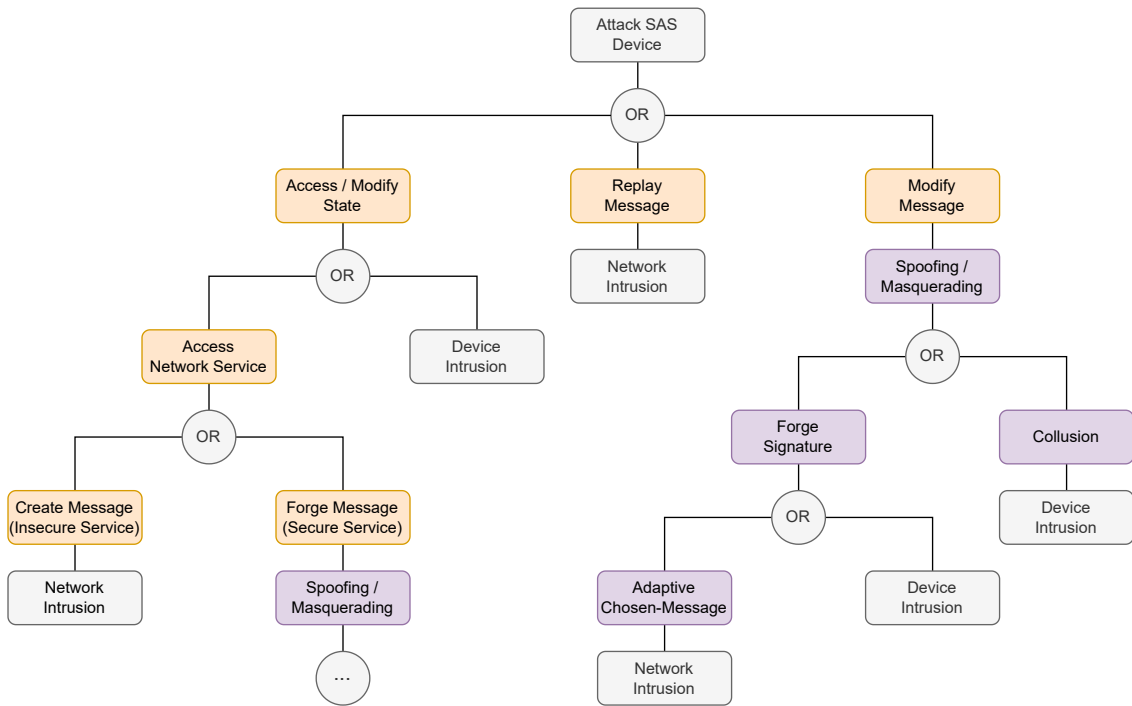


Fig. 2: Attack tree comprising cyberattacks that endanger SAS devices and message exchange protocols.

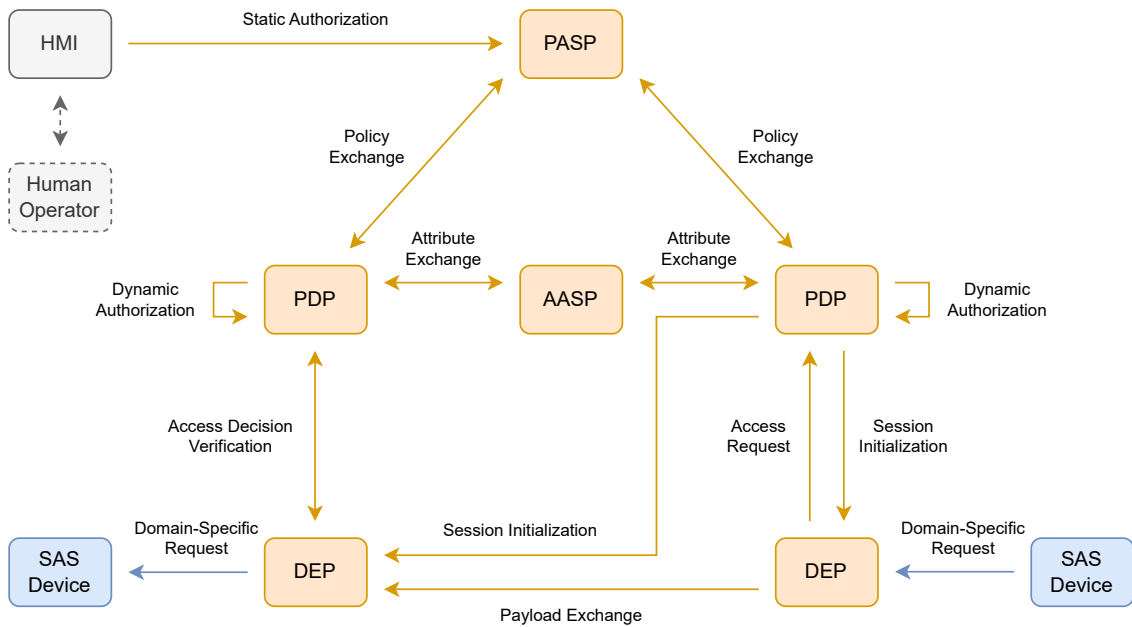


Fig. 3: Function-oriented component-based architecture of the RTS-ABAC approach.

Decision Enforcement Point (DEP) The DEP enforces access control decisions by controlling access to protected objects. In contrast to the PEP introduced by Hu *et al.* [13], the DEP neither has direct access to policies nor attributes but enforces policies indirectly via access decisions taken by a PDP. In addition to the tasks of conventional PEPs, our DEPs are responsible for providing authentication-related services to SAS devices. The services for SAS devices are provided in the form of a BITW solution, i.e., these services are provided to the corresponding service consumers automatically and invisibly.

3.2 Policy Specification

RTS-ABAC features a novel strategy for the representation and specification of access control policies. The policies are enforced for data frame flows, i.e., sets of data frames with common properties passing through the SAS network. Accordingly, the objective of an access control policy is twofold. On the one hand, a policy has to contain information that specifies if the policy is applicable to a certain data frame flow. On the other hand, a policy must specify which actions should be taken for a matching data frame flow.

Access Control Policy: $\rho = (action_\rho, flow_\rho, auxiliary_\rho)$.

A policy ρ is represented by a three-tuple, which contains the *action* to be taken, the *flow*-specifying pattern, and a set of attribute-based *auxiliary* predicates specifying additional non-*flow*-related system characteristics.

Action: $action \in \{GRANT, DENY\}$.

The action of a policy specifies whether a matching data frame flow should be granted or denied. A data frame from a granted flow may be delivered to an SAS device, whereas a data frame from a denied flow is dropped by a DEP. The default action is *DENY*, which leads to dropping of all data frames of non-explicitly granted flows.

Flow Pattern: $flow = \{p_1, \dots, p_n\}, p_i(f) \in \{TRUE, FALSE\}$.

The flow pattern of a policy specifies whether the policy is applicable to a specific data frame or not. It consists of a set of frame predicates p_1, \dots, p_n . A frame predicate p_i is a function that assigns a boolean value to an arbitrary data frame. A predicate p_i matches a frame f , if $p_i(f) = TRUE$. A policy is applicable to a specific data frame if all frame predicates match the frame. In other words, a policy $\rho = (action_\rho, flow_\rho, auxiliary_\rho)$ is applicable to a frame f if the following equation holds:

$$\forall p_i \in flow_\rho : p_i(f) = TRUE \iff flow_\rho(f) = p_1(f) \wedge \dots \wedge p_n(f) = TRUE$$

To increase the efficiency of the predicate evaluation and simplify the flow specification, each flow pattern is represented as a directed graph. By enclosing one pattern into another pattern, tree-like flow patterns can be constructed. A tree-like flow pattern of a transmission control protocol (TCP) segment is shown in Figure 4. Each node of the tree-like pattern represents one flow predicate. RTS-ABAC distinguishes between three types of flow predicates. The predicates depicted in blue are hierarchy predicates. Hierarchy predicates determine the structure of the tree, i.e., these predicates determine which other predicates can be added to the tree-like pattern. The predicates depicted in purple are so-called hierarchy-constrained predicates. Hierarchy-constrained predicates are non-parametric and are derived from the structure of the tree. The predicates depicted in green are parametric predicates. Parametric predicates can be configured to match the expected data frame flows within an SAS.

To check if a flow pattern matches a frame, the frame is transformed into its pattern representation. The pattern representing a frame is referred to as access request pattern. While the flow pattern consists of predicates, the access request pattern consists of constant values or facts. A flow pattern matches a frame's access request pattern, if all predicates of the flow pattern are true for their corresponding constant values of the access request pattern. To identify the pairs of predicates and facts that belong together, hierarchy predicates are used as anchor points of the patterns. Figure 5 shows a pattern matching process of a GOOSE flow pattern and two access request patterns representing Ethernet frames. The hierarchy predicates and facts used as anchor

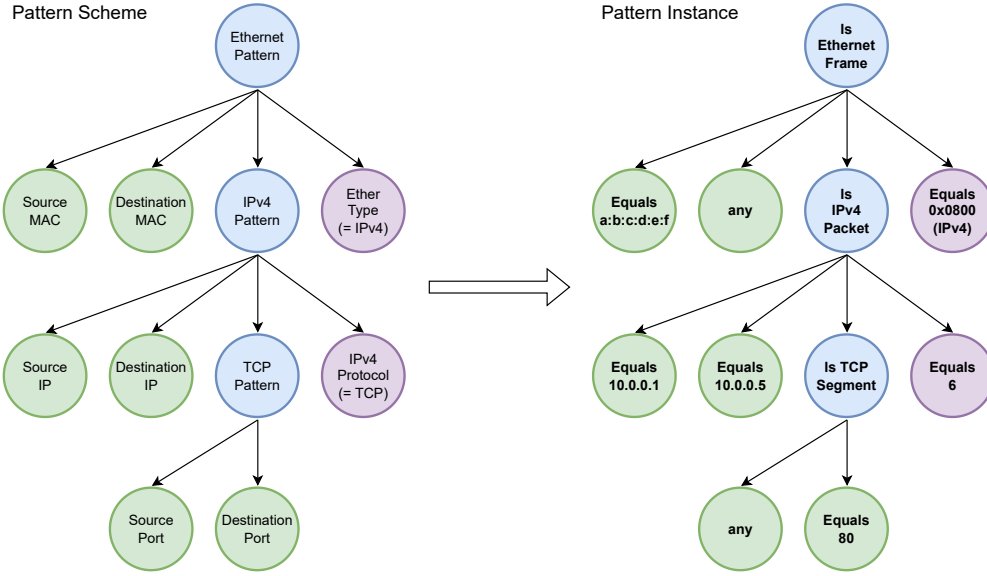


Fig. 4: Scheme and instance of a tree-like flow pattern representing the flow predicates of a TCP segment.

points for the pattern matching are depicted in blue. The remaining facts of the access request patterns are depicted in gray. As modern network communication is based upon layering of protocols, RTS-ABAC supports matching of nested patterns. A pattern matching process of a GOOSE flow pattern and a UDP/IP-encapsulated GOOSE access request pattern is shown in Figure 6. To find the nested pattern match, RTS-ABAC performs pattern matching of the flow pattern at all anchor points of the access request pattern. The pattern matching ends if either a match has been found or the flow pattern does not match at any of the anchor points.

Auxiliary Precondition: $auxiliary = \{a_1, \dots, a_n\}$, $a_i(system) \in \{TRUE, FALSE\}$.

The auxiliary precondition or system precondition of a policy specifies constraints for non-flow-related system attributes that have to be taken into account in order to apply an access control policy. The auxiliary precondition consists of a set of auxiliary predicates a_1, \dots, a_n . An auxiliary predicate a_i is a function that assigns a boolean value to the system state at a given point in time. The system state is represented by the set of all non-flow-related system attributes $ATT_{system} \subseteq system$ available at a given time. Accordingly, an auxiliary predicate is a function

$$a_i(system) = a_i(k_1, \dots, k_l \in ATT_{system}) = b \in \{TRUE, FALSE\}$$

that maps an arbitrary number of system attributes k_1, \dots, k_l to a boolean value b . The set of system attributes $ATT_{a_i} := \{k_1, \dots, k_j\} \subseteq ATT_{auxiliary_\rho} \subseteq ATT_{system}$ denotes the attributes required for the evaluation of the system predicate a_i . The set $ATT_{auxiliary_\rho} := \bigcup_{i=1}^n ATT_{a_i}$ denotes all system attributes required for the evaluation of the system precondition $auxiliary_\rho = \{a_1, \dots, a_n\}$ of a policy ρ . The auxiliary precondition is said to match the system state at a given time if all auxiliary predicates evaluate to true. A policy $\rho = (action_\rho, flow_\rho, auxiliary_\rho)$ is applicable in a system if the auxiliary precondition matches the current system state, i.e., a policy

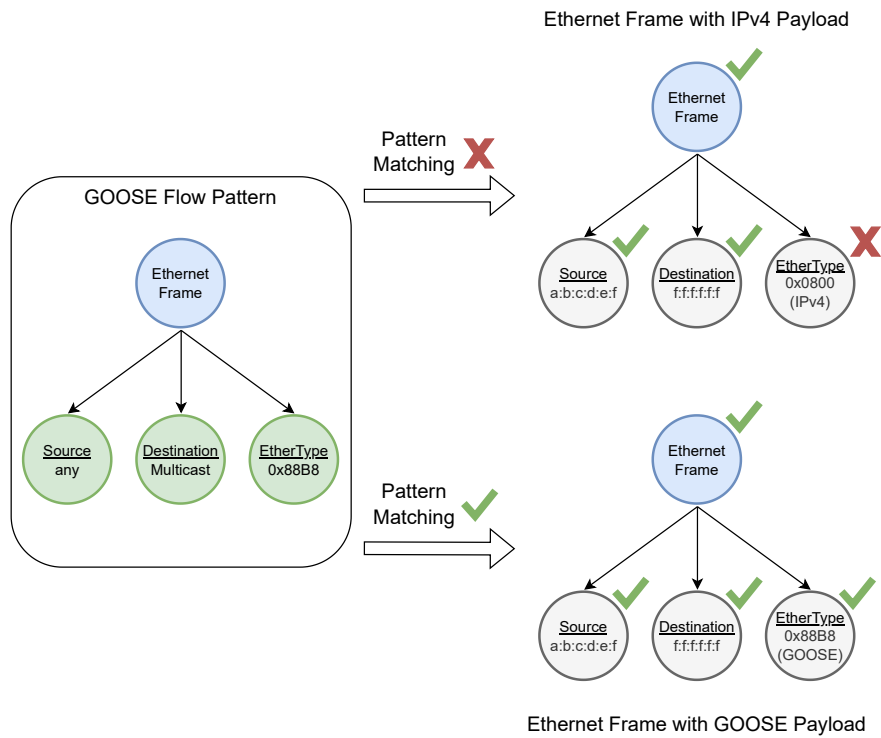


Fig. 5: Pattern matching of a GOOSE flow pattern and two Ethernet access request patterns.

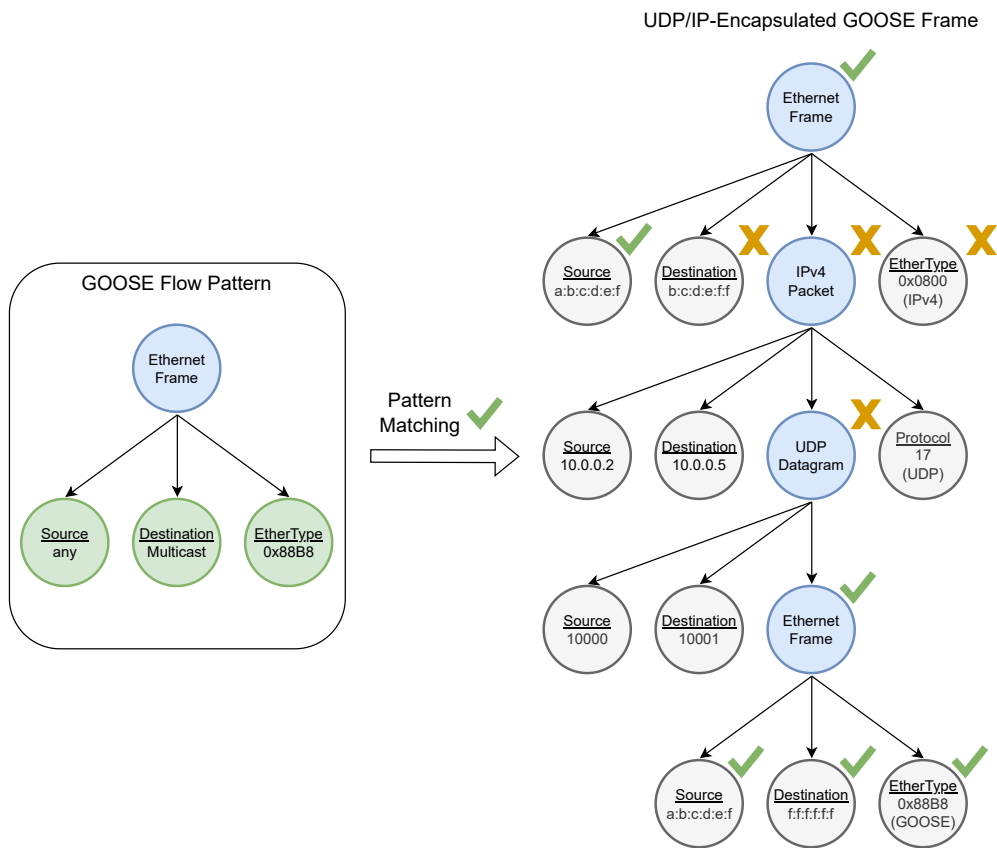


Fig. 6: Pattern matching of a GOOSE flow pattern and a UDP/IP-encapsulated GOOSE access request pattern.

is applicable if the following equation holds:

$$\begin{aligned}
\text{auxiliary}_\rho(\text{system}) &= \text{auxiliary}_\rho(ATT_{\text{system}}) \\
&= \text{auxiliary}_\rho(ATT_{\text{auxiliary}_\rho}) \\
&= \bigwedge_{a_i \in \text{auxiliary}_\rho} a_i(ATT_{\text{auxiliary}_\rho}) \\
&= \bigwedge_{a_i \in \text{auxiliary}_\rho} a_i(ATT_{a_i}) \\
&= a_1(ATT_{a_1}) \wedge \dots \wedge a_n(ATT_{a_n}) = \text{TRUE}
\end{aligned}$$

While the logical evaluation of numerous predicates without any hierarchy is an efficient task for modern computers, human operators might experience problems with this representation when specifying complex auxiliary preconditions. To overcome this issue, RTS-ABAC features an auxiliary precondition representation based on boolean binary expression trees. This representation is referred to as auxiliary predicate tree. Figure 7 shows a predicate tree of an auxiliary precondition consisting of five predicates. The inner nodes of a predicate tree are unary or binary logical connectives, including AND, OR, XOR, and NOT. The leaves of a predicate tree are auxiliary predicates. The predicate tree can be transformed into the conjunctive form by transforming the tree into a logical expression, calculating the conjunctive normal form (CNF) of this expression, and merging the disjunctions of predicates into new atomic predicates.

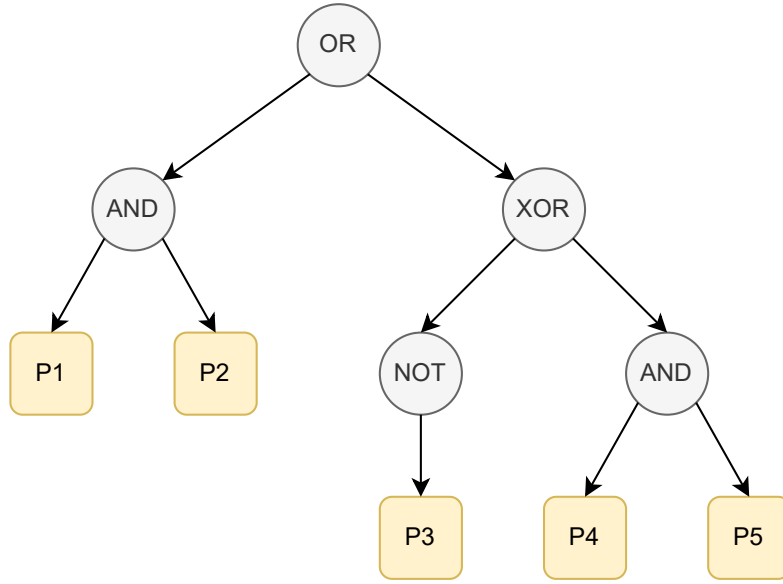


Fig. 7: Auxiliary precondition represented by a boolean binary expression tree.

3.3 Policy Classification

The dynamic or ad-hoc evaluation of ABAC policies allows the utilization of attributes from a time-variable environment. A real-time attribute represents an attribute whose value is time-dependent [1]. Given an attribute evaluation function E_{ATT} and a point of time t , the value λ_k of a real-time attribute k is defined by $\lambda_k := E_{ATT}(k, t)$.

Each policy ρ is related to a set of attributes $ATT_{\text{auxiliary}_\rho}$. The auxiliary attributes $ATT_{\text{auxiliary}_\rho}$ consist of all non-flow-related system attributes required for the evaluation of the auxiliary predicates a_1, \dots, a_n . To handle policies based on their degree of time-variability, we classify policies as follows:

Static Policy A static policy ρ is an ABAC policy whose evaluation does not rely on time-variable attributes. A policy ρ is static if $\forall k \in ATT_{auxiliary_\rho} : \forall t_i, t_j : E_{ATT}(k, t_i) = E_{ATT}(k, t_j)$. Consequently, access decisions can be cached. Due to the non-frequent attribute retrieval, non-frequent evaluation, and access decision caching, static policies are a viable solution for low latency message exchange. A static policy is also referred to as non-real-time policy.

Dynamic Policy A dynamic policy ρ is an ABAC policy whose evaluation relies on at least one time-variable attribute. A policy ρ is dynamic if $\exists k \in ATT_{auxiliary_\rho} : \exists t_i \neq t_j : E_{ATT}(k, t_i) \neq E_{ATT}(k, t_j)$. Due to the time-variable evaluation of dynamic policies, access decisions must have a limited time of validity that corresponds to the change rate of the underlying attribute values. As a result, caching of access decisions that are based on dynamic policies should be avoided. A dynamic policy is also referred to as real-time policy.

3.4 Policy Evaluation & Decision Enforcement

While the access control policy may depend upon the current system state, the corresponding access control decision is static during a specified period of validity. As a consequence, the process of handling access control policies can be divided into two related processes: The PDP-driven dynamic authorization derives an access control decision from an access control policy for the current system state. The DEP-driven decision enforcement utilizes the result of the dynamic authorization to enforce the policies in a timely manner.

Access Control Decision: $decision_\rho = \{flow_\rho, action_\rho, nexthop, validity\}$.

An access control decision consists of a *flow*-specifying pattern, an *action* to be taken, a set of *nexthop* DEPs, and a period of *validity*. Accordingly, an access decision is valid for a specific data frame *flow* in a specific system during a specified period of time. For each access policy p there exists exactly one access decision $decision_\rho$ at any point in time.

Dynamic Authorization: $E_{POL} : Policy \times System \rightarrow Decision$.

The policy evaluation of RTS-ABAC is server-aided, i.e., it is delegated to a PDP. To derive a decision from a policy at the PDP, the dynamic authorization function E_{POL} is used. The dynamic authorization function is defined as follows:

$$E_{POL}(\rho, s_t) = \begin{cases} (flow_\rho, action_\rho, nexthop, validity) & , \text{ if } \forall a_i \in auxiliary_\rho : a_i(s_t) = TRUE, \\ (flow_\rho, DENY, \emptyset, validity) & , \text{ otherwise.} \end{cases}$$

Thus, if the auxiliary predicates a_1, \dots, a_n of the policy ρ match the current system state s_t , the policy is applicable to matching data frame flows in the system. To evaluate the auxiliary predicates, the PDP fetches the auxiliary attributes $ATT_{auxiliary_\rho} \subseteq ATT_{s_t}$ prior to the evaluation of a_1, \dots, a_n . The validity of an access decision of a policy ρ at a specific time t equals the minimal validity of the attribute values in $\{\lambda_k = E_{ATT}(k, t) | k \in ATT_{auxiliary_\rho}\}$. Consequently, the validity of an access decision of a dynamic policy is determined by the attribute value that expires first. The validity of an access decision of a static policy may be limited by specifying a non-attribute-related maximum time of validity, to avoid the utilization of outdated access decisions.

Decision Enforcement: $E_{DEC} : Decision \times Frame \times Time \rightarrow (Action, Nexthop)$.

A DEP uses an access decision d_ρ to enforce a policy ρ in the SAS. For this purpose, each data frame f traversing a DEP is matched using the flow pattern $flow_\rho$ to identify the corresponding access decision. The process of enforcing a policy based on a priorly taken access decision is referred to as decision enforcement. At a DEP, the decision enforcement for a decision $d_\rho = \{flow_\rho, action_\rho, nexthop, validity\}$ and a frame f at a point in time t is performed using the decision enforcement function E_{DEC} :

$$E_{DEC}(d_\rho, f, t) = \begin{cases} (action_\rho, nexthop) & , \text{ if } \forall p_i \in flow_\rho : p_i(f) = TRUE \wedge d_\rho.valid(t), \\ (DENY, \emptyset) & , \text{ otherwise.} \end{cases}$$

Thus, if the flow predicates match a data frame, and the decision is still valid, the decision enforcement function returns the action to be taken and the DEPs the frame has to be forwarded to. However, two cases have to be distinguished, if $E_{DEC}(d_\rho, f, t) = (action_\rho, nexthop)$:

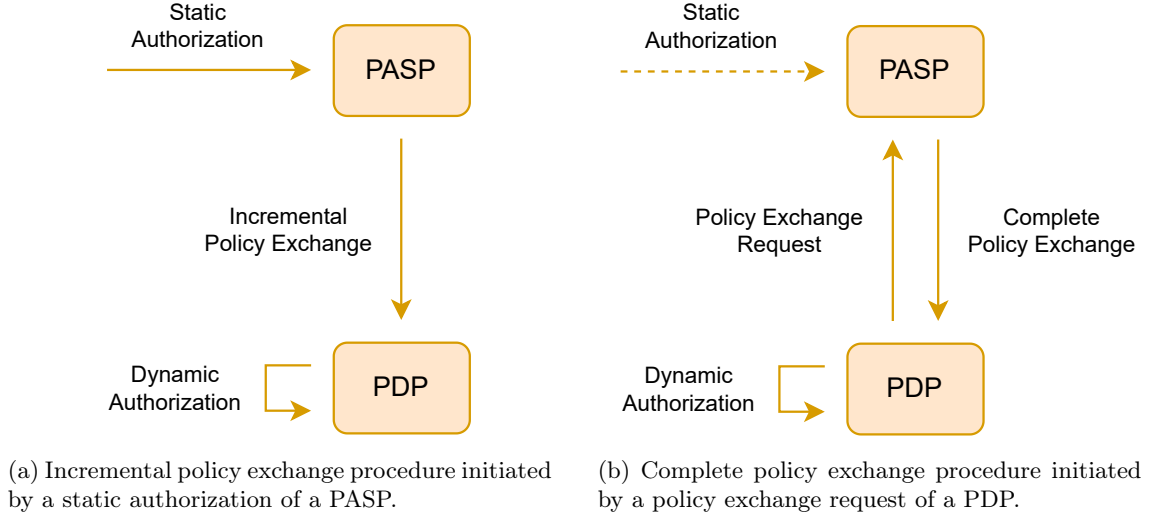


Fig. 9: Components and exchanged messages involved in the policy exchange process.

are visualized in Figure 10. Algorithm 1 shows the steps of the dynamic authorization algorithm. The dynamic authorization is based on the dynamic authorization function E_{POL} . Prior to its evaluation, the PDP fetches the required auxiliary attributes from one or multiple AASPs via attribute requests. An AASP responds to an attribute request with an attribute resolution message, which contains the values and periods of validity of the auxiliary attributes.

Algorithm 1 Dynamic authorization process of a PDP deriving an access control decision for each access control policy in a given system.

```

function DYNAMICAUTHORIZATION(policies, system)
  decisions  $\leftarrow$  {} // Initialize an empty set of access decisions
  ATTsystem  $\leftarrow$  {} // Initialize an empty system state

  for all  $\rho = (action_\rho, flow_\rho, auxiliary_\rho) \in policies$  do
    ATTauxiliary $\rho$   $\leftarrow$  {} // Initialize an empty set of auxiliary attributes
    for all  $a_i \in auxiliary_\rho$  do
      ATTai  $\leftarrow$  REQUESTATTRIBUTES( $a_i$ ) // Request the attributes from AASPs
      ATTauxiliary $\rho$   $\leftarrow$  ATTauxiliary $\rho$   $\cup$  ATTai
    end for

    // Add the auxiliary attributes of policy  $\rho$  to the current system state ATTsystem
    ATTsystem  $\leftarrow$  ATTsystem  $\cup$  ATTauxiliary $\rho$ 

    // Derive an access decision from policy  $\rho$  for the current system state ATTsystem
    decisions  $\leftarrow$  decisions  $\cup$   $E_{POL}(\rho, ATT_{system})$ 
  end for
  return decisions
end function

```

3.6 Delegated Decision Enforcement Protocol

The delegated decision enforcement protocol is responsible for the exchange, verification, and enforcement of access decisions. The workflow of the protocol is divided into three mandatory phases and an optional verification phase. The phases are defined as follows:

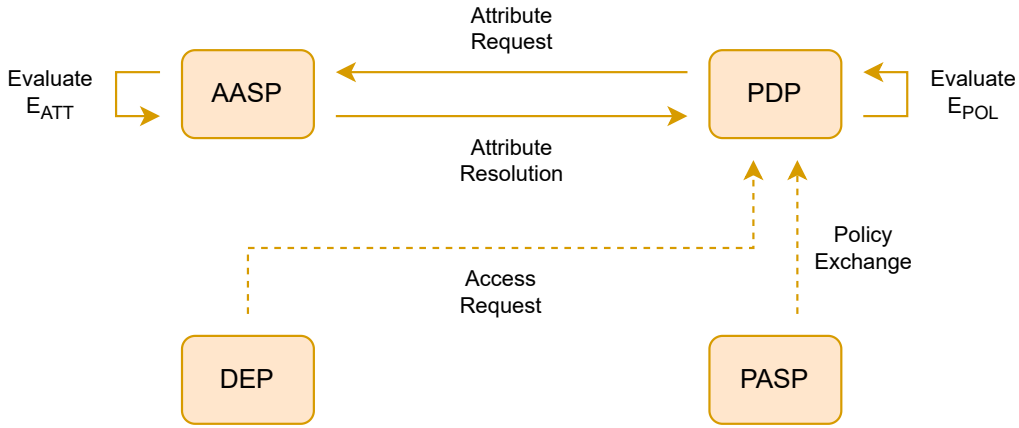


Fig. 10: Components and exchanged messages involved in the dynamic authorization process.

Access Request The access request phase is initiated by a DEP on behalf of an SAS device. To identify the corresponding access decision and access policy, the DEP appends the access request pattern of a domain-specific data frame to the access request.

Session Initialization The session initialization is performed by a PDP as a response to an access request. A session initialization between a requesting DEP and a single *nexthop* DEP is shown in Figure 11. The PDP identifies all applicable policies via flow pattern matching and fetches the corresponding access decisions from its cache or derives them via dynamic authorization. Then the PDP sends session initialization messages to the requesting DEP and the $DEP_i \in nexthop$. Each session initialization message contains one or multiple access decisions. If no access control policy is applicable, the PDP sends the requesting DEP a session initialization with a denying access decision $d_{Default}$ that matches the access request pattern from the access request exactly. A DEP initializes a local session state by adding the encapsulated access decisions to a local set of access decisions for incoming or outgoing data frames.

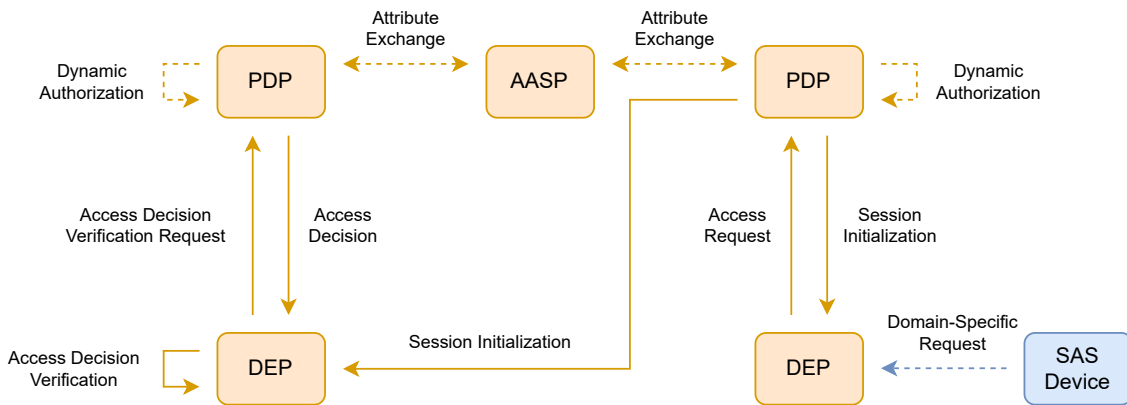


Fig. 11: Components and exchanged messages involved in an access request, session initialization, and access decision verification.

Access Decision Verification The optional server-aided access decision verification enables a DEP to verify a session initialization. This optional step is used to reduce the trust in a single PDP. The verification is shown in Figure 11. To initiate the verification, a DEP sends an access verification request to a PDP. The request contains the flow pattern of the access decision to be verified. The PDP sends the corresponding access decisions to the requesting DEP without initializing sessions at the *nexthop* DEPs. The DEP performs the verification by checking if any conflicts exist between the session initialization and received access decisions.

Payload Exchange The goal of the payload exchange is to securely exchange domain-specific data frames between SAS devices. It is initiated by an SAS device via delivery of a domain-specific data frame to its DEP. On receipt of a data frame f , the sender's DEP identifies all matching access control decisions d_1, \dots, d_n via flow pattern matching with the access request pattern of f . As discussed in subsection 3.4, a DEP uses the access decision with the most specific flow pattern match, or a composite decision in case of a conflict. If no matching access decision is available, a DEP initiates an access request procedure. As soon as the access decision is available, the DEP executes the decision enforcement function E_{DEC} . If the access decision is granting, the DEP encapsulates the domain-specific data frame in a payload exchange request, and forwards the request to each $DEP_i \in nexthop$.

On receipt of the payload exchange request, the receiving DEP fetches the most specific access decision and evaluates E_{DEC} . If the access decision is granting and the receiving $DEP \in nexthop$, the domain-specific data frame is extracted from the payload exchange request and forwarded to the receiving SAS device. Otherwise, the payload exchange request is discarded. A successful payload exchange between two SAS devices is shown in Figure 12.



Fig. 12: Components and exchanged messages involved in a payload exchange between two SAS devices.

4 Evaluation

The evaluation of RTS-ABAC is performed experimentally using industrial SAS devices and a testbed based on off-the-shelf hardware. The experiments utilize a software implementation of RTS-ABAC. The software is implemented component-wise using the object-oriented high-level programming languages Java and Kotlin. The software implementation of RTS-ABAC is published open source on GitHub [11] under the European Union Public Licence (EUPL) [7].

4.1 Performance Analysis

The objective of the performance analysis is to demonstrate that RTS-ABAC is a viable solution to secure time-critical message exchanges. For this purpose, we conduct an experimental estimation of message exchange latencies using a testbed based on off-the-shelf hardware.

Experimental Setup The testbed of the performance analysis consists of eight devices. The hardware devices and network topology of the testbed is shown in Figure 13. Two Raspberry Pi 5 mimic SAS devices, two Raspberry Pi 5 protect the SAS devices as BITW DEPs, and a fifth Raspberry Pi 5 provides the services of the PASP, AASP, and PDP. We interconnect the devices using Ethernet over twisted-pair. Since the Raspberry Pi 5 only possesses a single on-board RJ45 Ethernet connector, we additionally use a USB-A to RJ45 Ethernet adapter made by Bechtle for both DEPs.

Procedure & Results To demonstrate the viability of RTS-ABAC with regard to securing time critical message exchanges, we evaluate the extent to which RTS-ABAC is capable of handling different message types. For this purpose, we conduct an experiment to estimate the end-to-end communication latency of a message exchange between two domain entities. The sequence of events

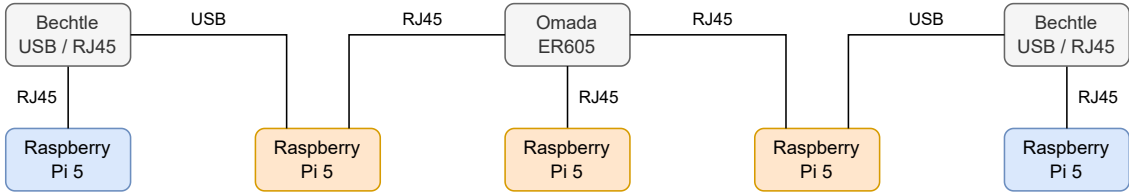


Fig. 13: Devices and network topology of the performance analysis testbed.

and the corresponding messages exchanged are shown in Figure 14. We realize the message exchange latency estimation by implementing a benchmark program and deploying it to the domain entities. The benchmark program is implemented in Python and is published open source on GitHub [11] alongside the implementation of RTS-ABAC. The program estimates the end-to-end latency between the two domain entities based on the round-trip time (RTT) of a bidirectional message exchange. We utilize the user datagram protocol (UDP) as a message exchange protocol for the experiment in order to avoid external latency influences, such as the flow control and congestion control of the transmission control protocol (TCP). To measure the RTT, the so-called active benchmark entity sends a timestamp to the passive benchmark entity. The passive entity replies to the message with the received timestamp. Thus, after receiving the response from the passive entity, the active entity is able to calculate the RTT by subtracting the received timestamp from the current timestamp. As a consequence, no time synchronization is required between the domain entities. Furthermore, under the assumption of symmetric transmission times, the accuracy of the RTT measurements only depends on the accuracy of the active entity’s system clock. To avoid an offset caused by the router’s buffering and forwarding strategy, messages are sent sequentially, i.e., the active entity waits for the arrival of a response before sending another timestamp message. Moreover, to compensate for fluctuations in the RTT measurements and to increase the confidence in the RTT estimation, the active entity repeats the RTT measurement procedure.

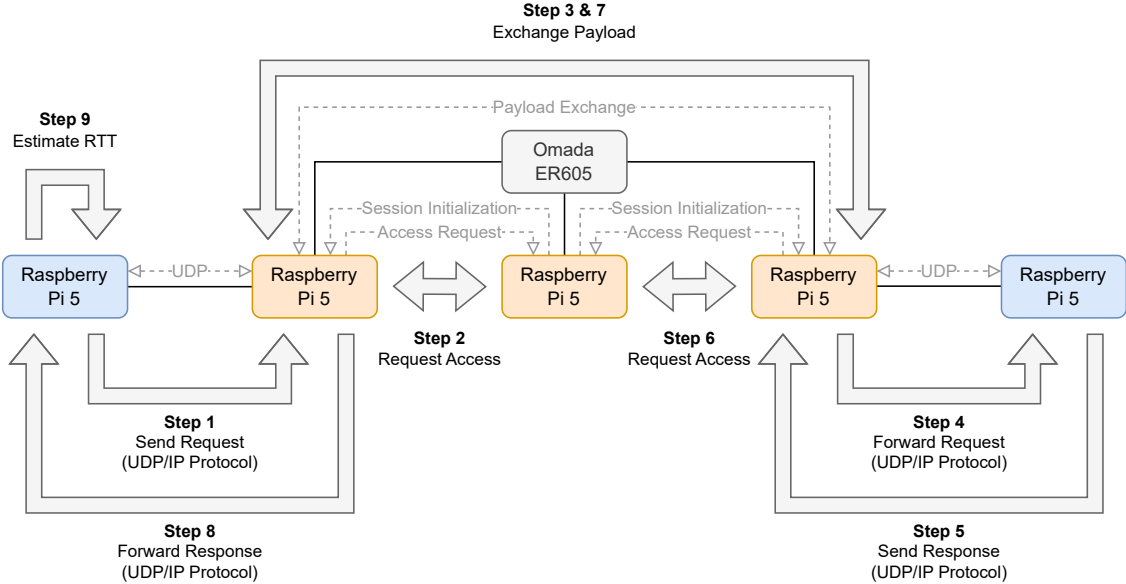


Fig. 14: Sequence of events of the experimental message exchange latency estimation.

The results of the latency estimation experiment are shown in Table 2 and Table 3. The raw data of the individual experiment runs are published open source on GitHub [11] alongside the implementation of RTS-ABAC. Since RTS-ABAC is cryptography-dependent but scheme-agnostic, we conduct the latency estimation procedure for different authentication algorithms. For each of the algorithms, the benchmark program sends 5000 sequential packets to estimate the RTT. Based

on the measurements, we calculate the arithmetic mean, median, standard deviation, and extrema-related values for each algorithm. Furthermore, we calculate the throughput of the DEPs in packets per second (PPS), and the cumulative share of the 5000 packets in the message types of IEC 61850 listed in Table 1.

To measure the latency offset caused by packet capturing and forwarding, we perform the initial latency estimation without authentication, authorization, or access control. The results indicate that the bidirectional message exchange between two domain entities requires 0.644 ms on average, which leads to a PEP throughput of 1487.2 sequential PPS. Without authentication, authorization, and access control in place, the measured RTTs are consistent, and each bidirectional message exchange is finished in less than 6 ms RTT.

To measure the influence of the authorization and access control, we perform a latency estimation with authorization and access control but without authentication. For this purpose, we implement a so-called no-operation (NoOp) authenticator, which processes the packets without signing or verifying them. The measurements using the NoOp authenticator show that the authorization and access control workflow leads to an RTT increase of 1.555 ms on average. Moreover, the increased range of minimum and maximum time indicates that authorization and access control lead to an increase in RTT fluctuation. Nevertheless, 99.9 % of the messages are exchanged in less than 6 ms RTT.

In order to evaluate the performance of RTS-ABAC in combination with symmetric cryptography, we conduct a latency estimation for HMAC authentication based on SHA-512. The results show that 99.82 % of the messages are exchanged in less than 6 ms RTT. The remaining nine packets or 0.18 % of the 5000 packets satisfy the 40 ms time constraint of the IEC 61850 message type 1B. By employing HMAC authentication, RTS-ABAC achieves a throughput of 435.2 PPS at the DEPs.

To evaluate the performance of RTS-ABAC in combination with public key cryptography (PKC), we conduct latency estimations for two PKC algorithms. For the latency estimations we use the Ed25519 algorithm and the RSA-2048 algorithm. Both PKC approaches do not satisfy the time constraints of the message types 1A and 4. However, Ed25519 and RSA-2048 satisfy the time constraint of the message type 1B. The data indicates that the RTTs of the PKC approaches are subject to fluctuations with higher magnitude compared to the fluctuations caused by symmetric cryptography. Furthermore, the throughput of the DEPs is reduced by more than 78 % compared to the HMAC authentication.

Table 2: Results of the RTT estimation based on 5000 measurements per algorithm.

Algorithm	Mean \bar{x}	Median \tilde{x}	Deviation σ	Extrema			
				Min	Max	Range	Mid-Range
None	0.644	0.612	0.122	0.560	4.810	4.250	2.685
RTS-ABAC + NoOp	2.199	2.026	0.430	1.676	9.807	8.131	5.741
RTS-ABAC + HMAC	2.276	2.160	0.491	1.738	9.350	7.612	5.544
RTS-ABAC + Ed25519	10.478	10.721	1.465	8.426	15.775	7.348	12.101
RTS-ABAC + RSA-2048	12.221	12.063	1.591	10.453	57.004	46.551	33.729

4.2 Demonstration of Applicability

We conduct a demonstration of applicability with industrial SAS devices in our smart grid cybersecurity laboratory [6]. The demonstration serves to complement the performance analysis by demonstrating that RTS-ABAC is a viable solution for enhancing the communication security in a newly constructed or retrofitted SAS. Accordingly, we demonstrate that the SAS behavior and functionality are not negatively influenced by RTS-ABAC, and that SAS devices protected by RTS-ABAC are capable of providing their services and exchanging information.

Experimental Setup The hardware devices used for the experiment are listed in Table 4. Nine of these devices are industrial SAS devices, which are organized in three so-called bays. Each bay

Table 3: Throughput and cumulative message type share of the analyzed algorithms.

Algorithm	Throughput	Cumulative Share in Message Types			
		1A / 4 $\leq 6 \text{ ms}$	1B $\leq 40 \text{ ms}$	2 $\leq 200 \text{ ms}$	3 / 5 / 6 $\leq 1000 \text{ ms}$
None	1487.2 PPS	5000 (100 %)	5000 (100 %)	5000 (100 %)	5000 (100 %)
RTS-ABAC + NoOp	448.8 PPS	4995 (99.9 %)	5000 (100 %)	5000 (100 %)	5000 (100 %)
RTS-ABAC + HMAC	435.2 PPS	4991 (99.82 %)	5000 (100 %)	5000 (100 %)	5000 (100 %)
RTS-ABAC + Ed25519	94.9 PPS	0 (0 %)	5000 (100 %)	5000 (100 %)	5000 (100 %)
RTS-ABAC + RSA-2048	81.4 PPS	0 (0 %)	4998 (99.96 %)	5000 (100 %)	5000 (100 %)

consists of an MU, a protection IED, and an I/O box. We conduct the experiment for each of the three bays individually. To simulate a power grid, we use a relay test device from Omicron to generate three-phase electric power, which is measured by the MUs. Each I/O box is connected to an Omicron process simulator, which mimics the behavior of a substation circuit breaker. Besides these SAS-related devices, Raspberry Pi 5 computers provide the services of the RTS-ABAC components.

Table 4: Hardware used for the laboratory-based experimental demonstration of applicability.

	Manufacturer	Device	Task
Bay I	General Electric	Reason MU320	Process Bus MU
	General Electric	Multilin F60	Protection IED
	Siemens	SIPROTEC 6MD84	Input/Output Box
Bay II	SEL	SEL-401	Process Bus MU
	Hitachi ABB	REL670	Protection IED
	Siemens	SIPROTEC 6MD84	Input/Output Box
Bay III	Siemens	SIPROTEC 6MU85	Process Bus MU
	Siemens	SIPROTEC 7SX85	Protection IED
	Siemens	SIPROTEC 6MD84	Input/Output Box
Grid Simulation	OMICRON	CMC 356	Universal Relay Test Set
	OMICRON	Process Simulator	Circuit Breaker
Networking	Weidmüller	IE-SW-SL28M-SV	Industrial Ethernet Switch
	FS.COM	UMC-1F1T	Ethernet Media Converter
	Bechtle	ARTICONA Adapter	USB-A to RJ45 Adapter
RTS-ABAC	Raspberry Pi Ltd	Raspberry Pi 5 8GB	DEP, PDP, PASP, & AASP

The network topology of the devices is shown in Figure 15. To improve the readability of the shown topology, the USB-A to RJ45 Ethernet adapters and Ethernet media converters are omitted from the figure. In accordance with the layered SAS architecture shown in Figure 1, we introduce a layering of devices for the setup of the laboratory experiment.

Procedure & Results At the initial state of the experiment, the voltages and currents of all three phases generated by the Omicron CMC 356 are within certain boundaries to be detected as a valid grid situation. Accordingly, the circuit breakers connected to the I/O boxes are closed and allow electric power to flow. The procedure of the experiment comprised four key events. The sequence of events and the corresponding messages exchanged between the devices are shown in Figure 16 and are discussed in the following:

Step 1 "Generate Overcurrent": To start the experiment, we adjust the generated three-phase electric power of the relay test device. We set the current to a level that represents an overcurrent situation in our laboratory grid. This situation is communicated to the MUs via a direct analog connection, i.e., the MUs measure the voltages and currents of the three phases.

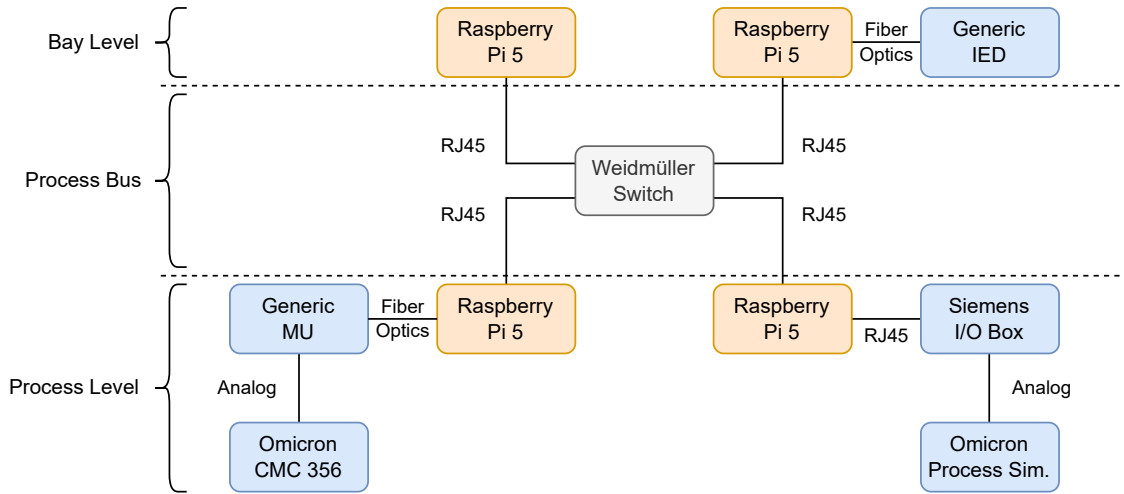


Fig. 15: Network topology of the laboratory-based demonstration of applicability.

Step 2 "Send Sampled Values": The MUs sample the voltage and current values provided by our relay test device. The MUs send the sampled values to the protection IEDs using the SV protocol. As the MUs are protected by a DEP, the outgoing SV frames are captured by the DEPs and processed as discussed in subsection 3.6. The authenticated and authorized payload exchange messages, which contain the SV frames, are then forwarded to the DEPs of the IEDs.

Step 3 "Send Trip Signal": The IEDs receive and process the SV frames, and detect the overcurrent situation. To resolve the overcurrent situation, the IEDs send GOOSE frames to their I/O boxes to open the corresponding circuit breakers. As the IEDs are protected by a DEP, the outgoing GOOSE frames are captured by the DEPs and processed as discussed in subsection 3.6. The authenticated and authorized payload exchange messages, which contain the GOOSE frames, are then forwarded to the DEPs of the I/O boxes.

Step 4 "Trigger Circuit Breaker": The I/O boxes receive the GOOSE frames and, as the GOOSE frames signal an overcurrent situation, the I/O boxes use an analog signal to open their circuit breakers. At the end of the experiment, we are able to verify electrically and visually that all circuit breakers are open.

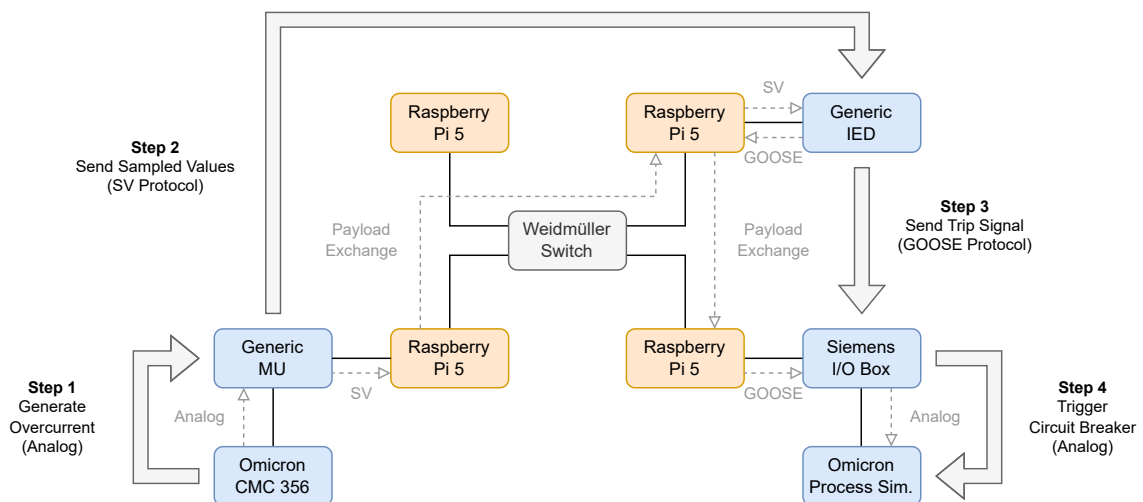


Fig. 16: Sequence of events of the laboratory-based demonstration of applicability.

4.3 Discussion & Limitations

As the chain of events successfully propagates through our laboratory SAS network, we demonstrate that RTS-ABAC does not disrupt the operation of the MUs, IEDs, and I/O boxes. Since the security-related components are deployed to inexpensive off-the-shelf hardware, we are able to demonstrate that RTS-ABAC is a feasible solution for SAS environments, not only security-wise and performance-wise, but also cost-wise. Due to its BITW concept, no adaptations have to be made to the SAS devices. Thus, the interoperability and interchangeability requirements of an IEC 61850 substation remain satisfied, while the communication security is enhanced. This indicates that RTS-ABAC is a viable solution for the retrofitting of existing substations.

The performance analysis reveals that the attribute-based authorization and access control workflow of RTS-ABAC has only a limited impact on the overall message exchange latency. Accordingly, the results emphasize the appropriateness of expressive and flexible yet computationally expensive access control approaches, such as ABAC, even in time and resource constrained environments. Nevertheless, the results of the performance analysis indicate that computational performance is the primary challenge for the deployment of cybersecurity approaches in an SAS. The findings show that lightweight yet inflexible symmetric cryptography seems to be the only viable solution to authenticate low latency message exchanges at the moment. We anticipate that the required computation time can be decreased, leading to an increase in message throughput, through the utilization of hardware accelerators for cryptographic algorithms. However, factors such as algorithm compatibility, costs per acceleration unit, and computation time consistency may result in a less beneficial influence on the system than currently expected.

The evaluation demonstrates that RTS-ABAC is capable of securing application protocols of an SAS, as well as multipurpose transport protocols. However, by employing RTS-ABAC in a substation the attack surface of the corresponding SAS might change. While IEDs, MUs, and other SAS devices benefit from being protected by using our BITW approach, the increased total number of devices in the SAS and their communication relations lead to an increased risk of denial-of-service (DoS) attacks. In particular, the dependence on certain locally or globally centralized components, such as PDP, PASP, and AASP, might introduce new attack vectors for SAS adversaries.

Furthermore, network time protocols and lower-layer network management protocols, such as the address resolution protocol (ARP), parallel redundancy protocol (PRP), and media redundancy protocol (MRP), are not secured but forwarded by the DEPs. The protocols are bypassed since they provide services not only to SAS devices but also to auxiliary intermediate devices, including network switches and routers, and as the operation of these protocols is susceptible to temporal inconsistencies resulting from authentication, authorization, and access control. Nevertheless, the usage of static bypass rules in the DEPs suggests that incompatible or legacy devices and protocols could continue their operation in retrofitted substations even if RTS-ABAC is deployed.

5 Conclusions & Outlook

To address the increasing relevance of cybersecurity for smart grid systems and to overcome the limitations of existing standards like IEC 61850 and IEC 62351, we present RTS-ABAC, a novel real-time server-aided attribute-based authorization and access control approach. RTS-ABAC enables the creation, management, storage, distribution, evaluation, and enforcement of ABAC policies for time-critical and time-variable distributed systems. We introduce the concept of time-dependency for ABAC attributes and policies. Furthermore, we discuss methods to manage, distribute, and enforce such expressive and flexible yet computationally expensive access control policies and access control decisions. We propose a delegated attribute-based authorization protocol, which is responsible for the policy creation, management, storage, distribution, and evaluation of RTS-ABAC policies. Finally, we additionally introduce a delegated decision enforcement protocol, which is responsible for the exchange, verification, and enforcement of access control decisions.

RTS-ABAC is implemented in software and is published open source on GitHub [11] under the European Union Public Licence (EURL) [7]. Based on the implementation we assess the applicability of RTS-ABAC for SAS communication protocols by conducting a performance analysis. The results of the performance analysis are published open source alongside the software implementation [11]. The performance analysis demonstrates the ability of RTS-ABAC to secure time-critical

message exchanges, as up to 99.82 % of authenticated, authorized, and access controlled network packets achieve an RTT below 6 ms. In particular, the performance analysis shows that the time required to perform cryptographic authentication procedures exceeds the time to evaluate and enforce RTS-ABAC policies. Furthermore, the performance analysis identifies the advantages and disadvantages of different authentication schemes with regard to satisfied time constraints. In accordance with the related literature, we identify the strict time constraints of low latency communication in an SAS as a key challenge for cybersecurity.

We demonstrate the applicability of RTS-ABAC for SAS devices in our smart grid cybersecurity laboratory [6]. The laboratory-based demonstration of applicability shows the ability of RTS-ABAC to secure the GOOSE and SV protocol between industrial-grade IEDs, MUs, and I/O boxes made by ABB, General Electric, Siemens, and SEL. Furthermore, we show that RTS-ABAC is a feasible solution for SAS environments, not only security-wise and performance-wise, but also cost-wise and due to its highly-compatible BITW concept, which allows retrofitting of existing systems. Accordingly, the results of the evaluation indicate that RTS-ABAC is a viable approach to enhance the communication security in a newly constructed or retrofitted substation.

While SAS-typical cyberattacks can be mitigated by employing RTS-ABAC, we also discuss the change of the attack surface, leading to an increased risk of DoS as a result of the additional components and protocols deployed in an SAS. To simplify the architectural complexity of RTS-ABAC, reduce its attack surface, and reduce the overall costs of deployment, we propose the integration of RTS-ABAC into network switches as an alternative realization. For this purpose, further research could investigate the potential benefits of realizing RTS-ABAC through the use of software-defined networking (SDN) solutions. SDN-based realization could aggregate the tasks of multiple DEPs by deploying a virtual DEP for each port of a network switch. Furthermore, distributed SDN controllers might provide the PASP, AASP, and PDP services. While the proposed PASP entities provide policy management services for human operators, future research could investigate how RTS-ABAC might benefit from the utilization of artificial intelligence (AI). The integration of AI-based intrusion detection could facilitate the creation and modification of security policies that are enforced within an SAS, thereby enabling the mitigation of a wider range of cyberattacks in a timelier manner. In addition to the deployment in an SAS, further research is required to evaluate the applicability of RTS-ABAC for other time-critical distributed systems. Therefore, we propose its evaluation in systems that have similar requirements as an SAS. Systems that potentially benefit from the enhanced communication security provided by RTS-ABAC include industry 4.0, robotics, avionics, and medical systems.

Acknowledgments

This work was supported by funding from the topic Engineering Secure Systems of the Helmholtz Association (HGF) and by KASTEL Security Research Labs (structure 46.23.02).

References

1. Burmester, M., Magkos, E., Chrissikopoulos, V.: T-ABAC: An attribute-based access control model for real-time availability in highly dynamic systems. In: 2013 IEEE Symposium on Computers and Communications (ISCC). IEEE (2013). <https://doi.org/10.1109/iscc.2013.6754936>
2. Communications Security Establishment Canada, Cyber threat bulletin: Cyber threat to operational technology, (2021). <https://open.canada.ca/data/dataset/98bad300-28f1-49b9-9b34-2d46de4c9a58>
3. Cybersecurity & Infrastructure Security Agency (CISA), CrashOverride Malware, (2021). <https://www.cisa.gov/news-events/alerts/2017/06/12/crashoverride-malware>
4. Cybersecurity & Infrastructure Security Agency (CISA), Cyber-Attack Against Ukrainian Critical Infrastructure, (2021). <https://www.cisa.gov/news-events/ics-alerts/ir-alert-h-16-056-01>
5. Elbez, G., Keller, H.B., Hagenmeyer, V.: Authentication of GOOSE Messages under Timing Constraints in IEC 61850 Substations. In: Electronic Workshops in Computing. BCS Learning & Development (2019). <https://doi.org/10.14236/ewic/icscsr19.17>
6. Elbez, G., Sánchez, G., Canbolat, S., Corallo, S., Fruböse, C., Lanzinger, F., Kellerer, N., Keppler, G., Neumeister, F., Beckert, B., Koziolok, A., Zitterbart, M., Hagenmeyer, V.: Insights and Lessons Learned from a Realistic Smart Grid Testbed for Cybersecurity Research. In: Proceedings of the 16th ACM International Conference on Future and Sustainable Energy Systems. E-Energy '25, pp. 805–812. ACM (2025). <https://doi.org/10.1145/3679240.3734649>

7. European Union, European Union Public Licence (EURL) Version 1.2, (2017). <https://joinup.ec.europa.eu/collection/eupl>
8. Fang, X., Misra, S., Xue, G., Yang, D.: Smart Grid — The New and Improved Power Grid: A Survey. *IEEE Communications Surveys & Tutorials* **14**(4), 944–980 (2012). <https://doi.org/10.1109/surv.2011.101911.00087>
9. Fildes, J.: Stuxnet worm targeted high-value Iranian assets, (2010). <https://www.bbc.com/news/technology-11388018>
10. Finkle, J.: Insiders suspected in Saudi cyber attack, (2012). <https://www.reuters.com/article/technology/exclusive-insiders-suspected-in-saudi-cyber-attack-idUSBRE8860CR>
11. Gstuer, M.: Real-Time Server-Aided Attribute-Based Authorization & Access Control for Substation Automation Systems, (2026). <https://github.com/gstuer/RTS-ABAC>
12. Gstür, M.: Certificateless Attribute-Based Server-Aided Cryptosystem for Substation Automation Systems (CASC-SAS). MA thesis, Karlsruher Institut für Technologie (KIT) (2025). <https://doi.org/10.5445/IR/1000182038>.
13. Hu, V.C., Ferraiolo, D., Kuhn, R., Schnitzer, A., Sandlin, K., Miller, R., Scarfone, K.: Guide to Attribute Based Access Control (ABAC) Definition and Considerations. Tech. rep. NIST Special Publication 800-162, National Institute of Standards and Technology (2014). <https://doi.org/10.6028/nist.sp.800-162>
14. International Electrotechnical Commission, Part 5: Communication requirements for functions and device models. Communication networks and systems for power utility automation (IEC 61850) (2014)
15. International Electrotechnical Commission, Part 6: Configuration description language for communication in power utility automation systems related to IEDs. Communication networks and systems for power utility automation (IEC 61850) (2020)
16. International Electrotechnical Commission, Part 6: Security for IEC 61850. Power systems management and associated information exchange - Data and communications security (IEC 62351) (2020)
17. International Electrotechnical Commission, Part 8-1: Specific communication service mapping (SCSM) - Mappings to MMS (ISO 9506-1 and ISO 9506-2) and to ISO/IEC 8802-3. Communication networks and systems for power utility automation (IEC 61850) (2022)
18. International Electrotechnical Commission, Part 8: Role-based access control for power system management. Power systems management and associated information exchange - Data and communications security (IEC 62351) (2020)
19. Ishchenko, D., Nuqui, R.: Secure Communication of Intelligent Electronic Devices in Digital Substations. In: 2018 IEEE/PES Transmission and Distribution Conference and Exposition (T&D). IEEE (2018). <https://doi.org/10.1109/tdc.2018.8440438>
20. Johnson, B., Caban, D., Krotofil, M., Scali, D., Brubaker, N., Glycer, C.: Attackers Deploy New ICS Attack Framework "TRITON" and Cause Operational Disruption to Critical Infrastructure, (2017). <https://cloud.google.com/blog/topics/threat-intelligence/attackers-deploy-new-ics-attack-framework-triton>
21. Kim, H., Kim, D.-K., Alaerjan, A.: ABAC-Based Security Model for DDS. *IEEE Transactions on Dependable and Secure Computing* **19**(5), 3113–3124 (2022). <https://doi.org/10.1109/tdsc.2021.3085475>
22. Ma, M., Woodhead, S.: Authentication delegation for subscription-based remote network services. *Computers & Security* **25**(5), 371–378 (2006). <https://doi.org/10.1016/j.cose.2006.03.006>
23. Ma, M., Woodhead, S.: Constraint-Enabled Distributed RBAC for Subscription-Based Remote Network Services. In: The Sixth IEEE International Conference on Computer and Information Technology (CIT'06). IEEE (2006). <https://doi.org/10.1109/cit.2006.63>
24. OASIS Open, eXtensible Access Control Markup Language (XACML) Version 3.0, (2013). <https://docs.oasis-open.org/xacml/3.0/xacml-3.0-core-spec-os-en.pdf>
25. Occupational Safety and Health Administration (OSHA), Illustrated Glossary - Substations, <https://www.osha.gov/etools/electric-power/illustrated-glossary/sub-station>
26. Padilla, E.: Substation Automation Systems: Design and Implementation. Wiley (2015)
27. Rodriguez, M., Lazaro, J., Bidarte, U., Jimenez, J., Astarloa, A.: A Fixed-Latency Architecture to Secure GOOSE and Sampled Value Messages in Substation Systems. *IEEE Access* **9**, 51646–51658 (2021). <https://doi.org/10.1109/access.2021.3069088>
28. Ruland, C., Sassmannshausen, J.: Firewall for Attribute-Based Access Control in Smart Grids. In: 2018 IEEE International Conference on Smart Energy Grid Engineering (SEGE). IEEE (2018). <https://doi.org/10.1109/sege.2018.8499306>
29. Stouffer, K., Pease, M., Tang, C., Zimmerman, T., Pillitteri, V., Lightman, S., Hahn, A., Saravia, S., Sherule, A., Thompson, M.: Guide to Operational Technology (OT) Security. Tech. rep. NIST Special Publication 800-82, Rev.3, National Institute of Standards and Technology (2023). <https://doi.org/10.6028/nist.sp.800-82r3>

30. Task Force Interagency Working Group, J.: Security and Privacy Controls for Information Systems and Organizations. Tech. rep. NIST Special Publication 800-53, Rev. 5, National Institute of Standards and Technology (2020). <https://doi.org/10.6028/nist.sp.800-53r5>
31. Zinets, N.: Ukraine hit by 6,500 hack attacks, sees Russian cyberwar, (2016). <https://www.reuters.com/article/us-ukraine-crisis-cyber-idUSKBN14I1QC>