The Need of Security Inside a Microservices Architecture in the Insurance Industry

Arne Koschel, Andreas Hausotter, Robin Buchta
Hochschule Hannover
University of Applied Sciences & Arts
Faculty IV, Department of Computer Science
Hannover, Germany
Email: {arne.koschel, andreas.hausotter, robin.buchta}
@hs-hannover.de

Pascal Niemann, Christin Schulze,
Christopher Rust, Alexander Grunewald
Hochschule Hannover
University of Applied Sciences & Arts
Faculty IV, Department of Computer Science
Hannover, Germany
Email: {pascal.niemann, christin.schulze, christopher.rust,
alexander.grunewald}@stud.hs-hannover.de

***Abstract*—Even for the more traditional insurance industry, the *Microservices Architecture (MSA)* style plays an increasingly important role in provisioning insurance services. However, insurance businesses must operate legacy applications, enterprise software, and service-based applications in parallel for a more extended transition period. The ultimate goal of our ongoing research is to design a microservice reference architecture in co-operation with our industry partners from the insurance domain that provides an approach for the integration of applications from different architecture paradigms. In Germany, individual insurance services are classified as part of the critical infrastructure. Therefore, German insurance companies must comply with the Federal Office for Information Security requirements, which the Federal Supervisory Authority enforces. Additionally, insurance companies must comply with relevant laws, regulations, and standards as part of the business's compliance requirements. Note: Since Germany is seen as relatively 'tough' with respect to privacy and security demands, fullfilling those demands might well be suitable (if not even 'over-achieving') for insurances in other countries as well. The question raises thus, of how insurance services can be secured in an application landscape shaped by the MSA style to comply with the architectural and security requirements depicted above. This article highlights the specific regulations, laws, and standards the insurance industry must comply with. We present initial architectural patterns to address authentication and authorization in an MSA tailored to the requirements of our insurance industry partners.**

***Keywords*—Security; Authorization; Authentication; Insurance Industry; Microservices Architecture.**

I. INTRODUCTION

Information Technology (IT)-Security is absolutely a 'must have' for insurance companies, especially for customer data, self-written and 3rd party applications, and their IT infrastructure in general. General regulations, such as the European *General Data Protective Regulation (GDPR)* [1], are applied to insurance as well as insurance specific laws and rules regarding security and other regulations (cf. [2] and [3]), for example, data protection and secured IT communication infrastructure. This article mainly focuses on securing insurance business applications (cf. [4]). Over time, several technologies from monolithic mainframe applications, functional decomposition-based software, traditional *Service-Oriented Architecture (SOA)*, and 3rd party enterprise software, such as SAP systems, were and are used together in insurance business applications.

Recently, the MSA style (cf. [5], [6]) and cloud computing joined the field. The ultimate goal of our currently ongoing research [7] is to develop a "*Microservice Reference Architecture for Insurance Companies (RaMicsV)*" jointly with partner companies from the insurance domain, which is taking all those typical cornerstones from (overtime grown) insurances into account. Placed within our work on RaMicsV is the question: "how to help secure (insurance) business applications using potentially several logical parts from RaMicsV, mainly including microservices combined with other typical insurance applications technologies"?

Only a few authors (see Section II) look at such technology combinations, and especially they do not take (German) insurance domain specifics into account. Thus, the present article constitutes an initial step in that direction.

In particular, we contribute here our ongoing work and intermediate results regarding:

- An introduction to IT-Security Regulations in Germany for insurance companies, including:
  - A brief explanation of when an institution is considered critical infrastructure and the resulting consequences.
  - Functions and regulations of the *Federal Office of Information Security (BSI)* and the *Federal Financial Supervisory (BaFin)* in this context.
- Evaluate existing patterns for achieving protection goals and weigh their pros and cons.
- To take a brief look at service- and edge-level authentication.
- To take a deeper look at service- and edge-level authorization.
- Consider the pattern concerning the requirements of the insurance industry with SOA and an *Enterprise Service Bus (ESB)*.

The remainder of this article is structured as follows: After discussing related work in Section II, we place our current work into our initial logical reference architecture from [7] in Section III. Next, Section IV looks at requirements for German insurance companies, and Section V examines known authorization and authentication patterns and their potential application within our work. Finally, Section VI summarizes the results, draws a conclusion, and looks at future work.

II. Related Work

Our research is based on literature of well-known authors in microservices, especially Chris Richardson (Microservices Pattern) [5]. His book describes fundamental statements for the advantages and disadvantages of the edge-level security pattern and the service-level security pattern.

We adopted our definition of components for authorization and authentication from the *National Institute of Standards and Technology (NIST)* [8] and the patterns described in Section V originate from [9].

Regarding legal regulations and specifications, we use, among others, *the Act on Federal Office for Information Security (BSIG)* [10]. Here the part for critical infrastructures and, correspondingly, *the Regulation for the Determination of Critical Infrastructures according to the BSI Act (BSI-KritisV)* [11] is used to reinforce the relevance of our reference architecture. In addition, this is supplemented with *the insurance regulatory requirements for IT (VAIT)* [2] from the BaFin, as this is the responsible authority of the insurance industry.

In our previous work [7], we presented the logical microservice reference architecture that we created in the German insurance domain with our partners by logical and technical details in the area of logging and monitoring components. So far, components in the area of security have not been considered within this reference architecture, which is now started in the present article.

Additionally, in [12], we dealt with the consistency of microservices, among other things. Here, compliance aspects were described, which arose during the service design using Domain Driven Design. The requirements specific to German insurance companies were briefly mentioned. Based on this, the legal constraints and controlling constitutions are described in more detail.

To the authors' knowledge, this is the first work to address the legal regulations for German insurance companies in the context of a reference architecture for microservices with a focus on patterns for security and, in particular, authentication and authorization. In addition, we address the requirement of this reference architecture for microservices to work together or side by side with an ESB (see III).

III. Reference Architecture for Insurance Companies

This Section will present our logical reference architecture for microservices in the insurance industry (RaMicsV).

RaMicsV defines the setting for the architecture and the design of a microservices-based application for our industry partners. The application's architecture is out of scope, as it heavily depends on the specific functional requirements.

When designing RaMicsV, a wide range of restrictions and requirements given by the insurance company's IT management have to be taken into account. Concerning this contribution, the most relevant are:

- ESB: The ESB as part of the SOA must not be questioned. It is part of a succesfully operated SOA landscape, which seems suitable for our industry partners for several years to come. Thus, from their perspective, the MSA style is only suitable as an additional enhancement and only a partial replacement of parts from their SOA or other self-developed applications.
- Coexistence: Legacy applications, SOA, and microservices-based applications will be operated in parallel for quite a extended transition period (several years to come). This means that RaMicsV has to provide approaches for integrating applications from different architectural paradigms – looking at it from a high-level perspective, allowing an 'MSA style best-of-breed' approach at the enterprise architectural level as well.

Figure 1 depicts the building blocks of RaMicsV, which comprises layers, components, interfaces, and communication relationships. Components of the reference architecture are colored yellow; those out of scope are greyed out.

A component may be assigned to one of the following *responsibility areas*:

- **Presentation** includes components for connecting clients and external applications such as SOA services.
- **Business Logic & Data** contains the set of microservices to provide the desired application-specific behavior.
- **Governance** consists of components that contribute to meeting the IT governance requirements of our industrial partners.
- **Integration** contains system components to integrate microservices-based applications into the industrial partner's application landscape.
- **Operations** consist of system components to realize unified monitoring and logging, which encloses all systems of the application landscape.
- **Security** consists of components to provide the goals of information security, i.e., confidentiality, integrity, availability, privacy, authenticity & trustworthiness, non-repudiation, accountability, and audibility.

Components communicate via HTTP—using a RESTful API, or message-based—using a *Message-Oriented Middleware (MOM)* or the ESB. The ESB is part of the *integration* responsibility area, which contains a message broker (see Figure 1).

In addition to data transformation and message routing and delivery, an ESB also implements security policies. For example, WS02 ESB supports *Web Services (WS)*-Security and WS-Policy specifications [13]. Beyond that, the WSO2 Identity Server can be used to generate an *OAuth Base Security Token* that microservices may employ to authenticate and authorize client applications and API clients. This corresponds to the edge- level authentication & authorization depicted in Section V.

In the next sections, we will look at the *security* responsibility area.

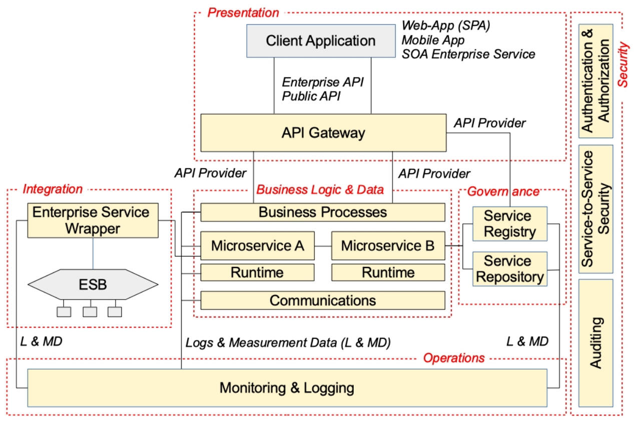

Fig. 1. Building Blocks of the Logical Reference Architecture RaMicsV

IV. Requirements for German Insurance Companies

Security is a fundamental aspect of any architecture and should never be neglected, mainly when there is a legislative framework where specific regulations exist. In Germany, insurance companies, which are regarded as critical infrastructure, are obligated to comply with the requirements of the BSIG, which the BaFin enforces. This consideration has been determined by the Federal Office for Information Security. Note: In our work we did not look at regulations and legal requirements in other countries, but, as stated above, German regulations are seen as 'somewhat tough' already.

A. Federal Office for Information Security and Critical Infrastructures

The BSI is a federal agency in Germanys responsible for security standards inside federal authoritie ando is a central reporting point for security incidents. Companies that are running critical infrastructures are obligated to report to the BSI. The Council of the European Union defined that a critical infrastructure "... is essential for the maintenance of vital societal functions, health, safety, security, economic or social well-being of people, and the disruption or destruction of which would have a significant impact in a Member State ..." [14]. Therefore, in Germany, an ordinance (BSI-KritisV [11]) from 2016 defines which infrastructures are critical. It could easily have dramatic consequences for the economy, state, and society if an infrastructure from one of the seven mentioned sectors (energy, water, food, information technology and telecommunications, health, finance and insurance, transport, and traffic) were attacked. Under Section 7 (1) no. 1 to 5, examples are given of critical financial and insurance services, which are of corresponding importance. Some examples mentioned are payment transactions or, among other things, insurance services and social security benefits. However, either a system or a part of it must be assigned to column B (System category) of Annex 6 Part 3 and, at the same time, exceed the corresponding threshold value in column D of the specific metric to be considered critical infrastructure. A general example would be a contract administration system in which the number of life insurance claims per year exceeds 500,000. Therefore, some systems from our partners are considered critical infrastructure and are liable to other requirements.

Because of the BSIG from 2009 [10], under section 8a "Security regarding the information technology of critical infrastructures," institutions with critical infrastructures are obligated to a security standard. They need to provide each two years evidence to the BSI that they took precautionary measures to achieve the protective goals of IT-Security. Specifically mentioned are **availability, integrity, authenticity, and confidentiality**. In addition, precautions are described here as reasonable if the effort required to secure the protection goals is in proportion to the consequences of the failure. Moreover, the BSI has published a document [15] that specifies the requirements imposed by Section 8a (1) BSIG.

Section 8a (2) of the BSIG states that it is possible to establish an industry-specific security standard that meets the requirements. The Federal Office of Civil Protection and Disaster Assistance and the corresponding regulatory authority will determine whether this standard is appropriate. Thus, there has to be a Federal Office that determines whether the company is complying with the requirements.

B. Federal Financial Supervisory Authority

The BaFin is responsible for the supervision of banks and financial and insurance providers. They published VAIT [2] in the year 2018. This publication gives the general conditions and specifications for IT risk and security management. There is a reference to the BSI-KritisV, and it has a entire section dedicated to critical infrastructures. All aspects, from detection over definition to implementation of security measurements, are essential. The goal is to secure the protective objectives of IT-Security, which are named in IV-A, and to minimize all risk factors inside the critical infrastructure. Therefore, German insurance companies must provide evidence through audits, certificates, or examinations every two years to fulfill their obligations. That is why every aspect of security needs to be addressed while or even better before implementing new systems.

C. Further Motivation for the Commitment to Confidentiality

There is a wide range of security aspects that need to be addressed. At this point, we would like to refer to a document published by the BSI entitled "Supervision of critical infrastructures in finance and insurance" [3]. This briefly discusses the legal requirements for critical infrastructures and the introduction of these requirements in 2019. It states that most of the deficiencies and shortcomings did not pose a direct threat to maintaining the operation of the infrastructures concerned. Nevertheless, according to ISO/IEC 27002, eight percent of the deficiencies were attributable to access control.

Additionally, in 2021 on the *Open Web Application Security Project (OWASP)* Top Ten 2021, first place is ”Broken Access Control,” and seventh place is ”Identification and Authentication Failures” [16]. Compared to 2017,” Broken access control“ came up from place 5 [17]. This shows that the importance of authorization and authentication continues to increase. As a result, it is increasingly important to find mechanisms that protect system boundaries with a low potential for error by business logic development teams.

Concerning Sections IV-A and IV-B, the four security properties that are explicitly named are listed below:

- **Confidentiality** includes read access by authorized subjects only.
- **Integrity** describes writing access by authorized subjects only.
- **Availability** implies access by authorized subjects at any time.
- **Authenticity** verifies the identity of the sender.

Through conversations with our partners, the focus of this paper will first be on different patterns of the service-level authorization aspect as part of the confidentiality and partly the integrity protection goal. Since authorization can be close to authentication in terms of implementation, it will also be included in the following section concerning the implementation location.

V. Authorization and Authentication Patterns

In distributed systems, authentication and authorization can be completed at different locations. While there is typically one place where authentication and authorization is performed in monolithic systems, there are various system locations where authentication and authorization might occur in distributed systems. This Section, thus, looks at well-known patterns for authentication and authorization for microservices.

Authentication and authorization have a crucial difference in the choice of location. Scalability is the critical factor in positioning authentication, as there is no business reason to prefer edge-level or service-level. Authentication needs a database to check credentials and calculate any security token; domain knowledge is not necessary [5]. In the case of authorization, on the other hand, it is not only scalability that is important but also how access is controlled. If *role-based access control (RBAC)* is the only requirement, decisions can be made without domain knowledge, e.g., by roles per URL path. In this case, edge-level authorization is usable. When a more explicit authorization is required, an *access control list (ACL)* is called. In this case, Domain information is needed, and service-level authorization is practical.

This Section does not discuss technical authentication and authorization solutions but highlights the authentication and authorization positioning and the resulting properties for the system’s performance and development. For both authentication and authorization, two fundamentally different approaches are possible. At the edge-level, the required components are frequently located in an API Gateway, whereas at the service-level, the components are located in each service. In the following Section, we first discuss edge-level authentication.

A. Edge-level Authentication

If there is an API Gateway, it may be used for authentication decisions. This is a quick-to-develop but hard-to-scale solution. Using an API Gateway has the following properties [5]:

- Domain logic development teams have very little involvement with authentication.
- API Gateway development teams have to deal with more complexity.
- Only one team is responsible for the authentication. This lowers the risk of security vulnerability.
- Faster development by lower complexity.
- Poor scalability due to a single point of control.
- Risk of too strong coupling of API Gateway and microservices, independent deployment is usually impossible.

B. Service-level Authentication

An alternative to the API Gateway implementation is the authentication at the service-level. This solution is slow and expensive to develop but scales well. The service-level authentication has the following properties [5]:

- Domain logic development teams have to deal with more complexity.
- Higher risk for security vulnerabilities due to multiple development teams.
- Slower development due to higher complexity in any microservice.
- Higher scalability, which stresses one of the most essential properties of an MSA.
- If there is only RBAC and a role, e.g., an admin has his microservice, the user database is small. Authorization errors have a more minor impact because a regular user can not log in.

The difference between authentication at the edge- and service-level should have become clearer now: Both approaches provide the authentication basis for the protection goals of confidentiality and integrity, which are described in Section IV.

In the next Section, edge-level and service-level authorization will be discussed.

C. Edge-level Authorization

With edge-level authorization, all the logic resides in the API Gateway. This brings the following characteristics:

- Easy implementation and maintenance.
- May create problems when scaling.
- Complex systems can be challenging to design.
- Back-end microservices must only be accessible via the API Gateway.

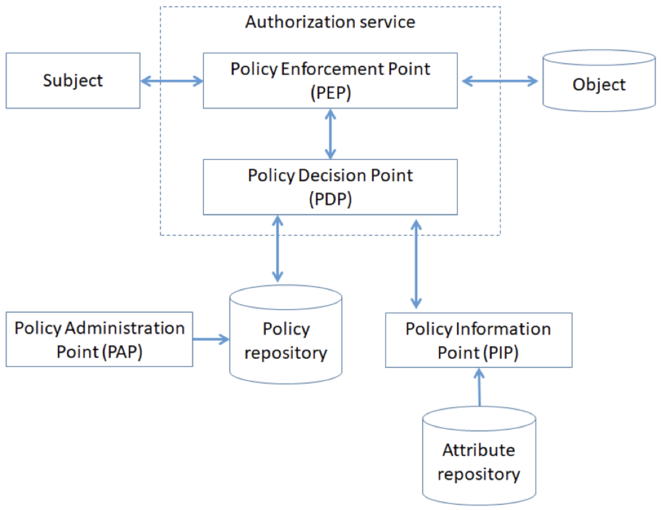
Authorization service
Subject
Policy Enforcement Point (PEP)
Object
Policy Decision Point (PDP)
Policy Administration Point (PAP)
Policy repository
Policy Information Point (PIP)
Attribute repository

Fig. 2. Fundamental points of ACM [8].

- Risk of too strong coupling of API Gateway and microservices—no independent deployment is possible.

This is a suitable solution for a lightweight MSA with few roles. Next, we will look at service-level authorization, which is increasingly attractive for more complex systems [9].

D. Service-Level Authorization

Like authentication, authorization can also be implemented at the service-level. An additional component is added to each microservice for authorization, authentication, or both. In this context, the following terms are important (Figure 2) [8]:

- ***Policy Enforcement Point (PEP)*** enforces the authorization decision.
- ***Policy Decision Point (PDP)*** computes the authorization decision.
- ***Policy Administration Point (PAP)*** comprises an interface to administrate the policies.
- ***Policy Information Point (PIP)*** provides additional information for the PDP to make authorization decisions [8].

As shown in Figure 2, the PEP and PDP together form the authorization.

The subsequent patterns are where PEP and PDP reside in the microservices environment. PAP and PIP are only mentioned for completeness. At first, we consider the general properties change compared to edge-level:

- Responsibility moves from the API development team to the microservices development team.
- Complex microservices environments are possible.
- Implementation and maintenance are more complex because changes affect each microservice.

1) Decentralized pattern: The decentralized pattern is the solution to create a microservice that is wholly controlled by the development team. All software and data components for making authorization decisions reside inside the microservice. This is optimal for scaling, but it requires a lot of effort to implement and maintain since any change in the authorization process requires changes in each microservice. Another challenge is propagating policy or attribute changes to all microservices. This is a complex pattern in the context of the required ESB (Section III). On the other hand, there are scenarios where this pattern may be suitable, e.g., if there is a microservice with a high number of requests [9].

2) Centralized pattern with single PDP: With the centralized single PDP pattern, the PEP is located within each microservice, and the PDP resides in a different central location. This implies that every request to the microservice will result in a network call to the PDP. Thus, if a very low response time is required, this is not a suitable solution. Also, if high scalability is needed, a single-point-of-decision is associated with limitations.

However, in the case of a central PDP, all microservices are independent of changes within the PDP. Moreover, thus approach could be faster to be implemented in cooperation with a required ESB (Section III), because then, the PEP resides in each microservice, and the PDP is provided by the ESB [9].

3) Centralized pattern with embedded PDP: In the centralized pattern with embedded PDP, the data and attributes are centralized, but the PDP is part of each microservice. Unlike the decentralized pattern (V-D1), the PDP is not part of the code but is embedded using a microservices library. So, the PDP is part of the microservice for quick decisions, but the development team doesn't have a lot of development work.

For interoperation with the required ESB (Section III), this pattern combines the advantages of a decentralized pattern and a quick implementation. The ESB could be used for data and attribute sharing. All other components could make fast decisions through the microservices [9]. Concerning the protection goals described in Section IV, the authorization enforces confidentiality and integrity.

E. Summary

Insurance companies are running large and complex systems with many different services and fine-grained access control. For this reason, edge-level authorization is suitable only in specific scenarios, for example, if RBAC can be used for a given microservice.

The application landscape of our partners of the insurance industry comprises an ESB as part in the reference architecture (Section III). Therefore, each pattern has its use case as we explained above. The decentralized pattern (V-D1) is recommended when performance is the most crucial requirement. The centralized pattern with a single PDP (V-D2) is suitable if performance is less critical and RBAC is needed. The centralized pattern with embedded PDP (V-D3) brings together the advantages of the previously mentioned patterns and is, therefore, from our point of view the most promising one.

VI. Conclusion and Future Work

The security aspect is indispensable in any realization or evolution of application architecture. Especially in Germany, insurance companies have to fulfill legal requirements according to the BSIG if general framework conditions are met and the resulting status of critical infrastructure is achieved. Every two years, proof must be provided to the BSI that the corresponding security standard is met. The BaFin is responsible for the regulation of this proof. Our partners from the insurance industry, thus, should still be compliant with those requirements if adding a critical (defined based on BSI-KritisV) system part based on RaMicsV.

For better guidance on authorization patterns from a confidentiality perspective, authentication has also been included, as the two security properties are usually close in terms of implementation. Relevant points regarding the implementation at the service-level and edge-level have been included. The paper's main focus was on the different patterns of service-level authorization, which were considered and evaluated in the context of our partners within the insurance industry.

Finally, the advantages and disadvantages of the individual patterns were weighed up. The pattern of choice, depends on the requirements for scalability and performance. In the context of (grown) insurance and microservices, implementation at the service-level seems the most appropriate. Furthermore, the centralized pattern with the single or the embedded policy decision point comes in closer selection due to the use of the required ESB within RaMicsV. Thus, an important part of the protection goal confidentiality was addressed. Still, it also took another step closer to answering the initially asked question: ”how to help secure (insurance) business applications using potentially several logical parts from RaMicsV, mainly including microservices combined with other typical insurance applications technologies”?

Within this publication, some guidelines for selecting patterns regarding authorization and authentication of critical infrastructure have been started and will be continued within our future work. In addition, our future work also deals with the approach of validity and consistency of embedded policies. To continue to remain oriented towards the protection goals, a prominent topic, service-to-service authentication, will be addressed in more detail in future work as well. Here, the available options for implementing authentication will be considered inside RaMicsV, and the respective advantages and disadvantages will be weighed against each other. Furthermore, relevant and current aspects of the broad subject's availability and integrity will then be evaluated one by one, to address later emerging security aspects of the MSA, such as deployment options and resulting security domains. The exact order is made in consultation with our partners from the insurance industry, depending on current topics or preferences.

Initial prototypes and proof of concepts have been developed and implemented for the reference architecture and were described in previous publications [12] and [7]. While similar work has not yet been done for the security domain from this publication, the effort required to implement parts or all of the reference architecture in a commercial system depends on the existing SOA, specific functional requirements, and the number of critical systems components to be implemented.

References

[1] The European Parliament and the Council of the European Union, “Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46/EC (General Data Protection Regulation - GDPR)),” [Online]. Available from: https://eur-lex.europa.eu/eli/reg/2016/679/oj/eng. [accessed: 2022-04-15].

[2] Bundesanstalt für Finanzdienstleistungsaufsicht (BaFin) - Federal Financial Supervisory (BaFin), “Versicherungsaufsichtliche Anforderungen an die IT (VAIT) (Insurance Supervisory Requirements for IT (VAIT)),” [Online]. Available from: https://www.bafin.de/SharedDocs/Downloads/EN/Rundschreiben/dl_rs_1810_vait_va_en.pdf?__blob=publicationFile&v=5. [accessed: 2022-04-15].

[3] Bundesamt für Sicherheit in der Informationstechnik (BSI) - Federal Office of Information Security (BSI), “Aufsicht über Kritische Infrastrukturen im Finanz- und Versicherungswesen (Supervision of Critical Infrastructures in the Finance and Insurance Industry),” [Online]. Available from: https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/KRITIS/Nachweispruefungen_im_Finanz-und_Versicherungswesen.pdf?__blob=publicationFile&v=3. [accessed: 2022-04-15].

[4] Gesamtverband der Deutschen Versicherungswirtschaft e.V. (General Association o.t. German Insurance Industry)), “VAA Final Edition. Das Fachliche Komponentenmodell (VAA Final Edition. The Functional Component Model),” 2001.

[5] C. Richardson, *Microservices Patterns: With examples in Java*. Shelter Island, New York: Manning Publications, 2018.

[6] S. Newman, *Building microservices: designing fine-grained systems*. Sebastopol, California: O'Reilly Media, Inc., 2015.

[7] A. Koschel, A. Hausotter, R. Buchta, A. Grunewald, M. Lange, and P. Niemann, “Towards a Microservice Reference Architecture for Insurance Companies,” in *SERVICE COMPUTATION 2021, 13th Intl. Conf. on Advanced Service Computing*, Online, 2021.

[8] V. C. Hu, D. Ferraiolo, R. Kuhn, A. R. Friedman, A. J. Lang, M. M. Cogdell, A. Schnitzer, K. Sandlin, R. Miller, and K. Scarfone, “Guide to attribute based access control (abac) definition and considerations (draft),” *NIST special publication*, vol. 800, no. 162, pp. 1–54, 2013.

[9] A. Barabanov and D. Makrushin, “Authentication and authorization in microservice-based systems: survey of architecture patterns,” *CoRR*, vol. abs/2009.02114, 2020, [Online]. Available from: https://arxiv.org/abs/2009.02114. [accessed: 2022-04-15].

[10] Bundesamt für Sicherheit in der Informationstechnik (BSI) - Federal Office of Information Security (BSI), “Act on the Federal Office for Information Security (BSI Act - BSIG) - courtesys translation -,” [Online]. Available from: https://www.bsi.bund.de/SharedDocs/Downloads/EN/BSI/BSI/BSI_Act_BSIG.pdf?__blob=publicationFile&v=4. [accessed: 2022-04-15].

[11] Bundesamt für Sicherheit in der Informationstechnik (BSI) - Federal Office of Information Security (BSI), “Verordnung zur Bestimmung Kritischer Infrastrukturen nach dem BSI-Gesetz (BSI-Kritisverordnung - BSI-KritisV) (Regulation for the Determination of Critical Infrastructures according to the BSI Act (BSI-Kritisverordnung - BSI-KritisV)),” [Online]. Available from: https://www.gesetze-im-internet.de/bsi-kritisv/BJNR095800016.html. [accessed: 2022-04-15].

[12] A. Koschel, A. Hausotter, M. Lange, and S. Gottwald, “Keep it in Sync! Consistency Approaches for Microservices - An Insurance Case Study,” in *SERVICE COMPUTATION 2020, The Twelfth International Conference on Advanced Service Computing*, Nice, France, 2020, [Online]. Available: http://www.thinkmind.org/index.php?view=article&articleid=service_computation_2020_1_20_10016. [accessed: 2022-04-15].

[13] WSO2, “WS02 Enterprise Service Bus Documentation: Securing APIs,” [Online]. Available: https://docs.wso2.com/display/ESB481/Securing+APIs. [accessed: 2022-04-15].

[14] The Council of the European Union, “COUNCIL DIRECTIVE 2008/114/EC,” [Online]. Available: https://eur-lex.europa.eu/legal-content/EN/TXT/HTML/?uri=CELEX:32008L0114. [accessed: 2022-04-15].

[15] Bundesamt für Sicherheit in der Informationstechnik (BSI) - Federal Office of Information Security (BSI), “Konkretisierung der Anforderungen an die gemäß § 8a Absatz 1 BSIG umzusetzenden Maßnahmen (Specification of the requirements for the measures to be implemented in accordance with Section 8a (1) BSIG),” [Online]. Available from: https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/KRITIS/Konkretisierung_Anforderungen_Massnahmen_KRITIS.pdf. [accessed: 2022-04-15].

[16] CWE Content Team, “Weaknesses in OWASP Top Ten (2021),” 2022, [Online]. Available: https://cwe.mitre.org/data/definitions/1344.html. [accessed: 2022-04-15].

[17] OWASP, “Welcome to the OWASP Top 10 - 2021,” 2022, [Online]. Available: https://owasp.org/Top10/. [accessed: 2022-04-15].