Amazon Web Services Cloud Compliance Automation with Open Policy Agent

Alen Paul
Department of Computer Science and Engineering
Amrita School of Computing, Chennai
Amrita Vishwa Vidyapeetham
ch.en.u4cys20002@ch.students.amrita.edu

Rishi Manoj
Department of Computer Science and Engineering
Amrita School of Computing, Chennai
Amrita Vishwa Vidyapeetham
ch.en.u4cys20065@ch.students.amrita.edu

Udhayakumar S
Department of Computer Science and Engineering
Amrita School of Computing, Chennai
Amrita Vishwa Vidyapeetham
s_udhayakumar@ch.amrita.edu

2024 International Conference on Expert Clouds and Applications (ICOECA) | 979-8-3503-8579-3/24/$31.00 ©2024 IEEE | DOI: 10.1109/ICOECA62351.2024.00063

***Abstract*— The security challenges posed by Infrastructure as code (IaC) are outgrowing established security procedures in an era of rapidly adopting cloud computing and DevOps methodologies. Using security principles to be integrated into the CI/CD pipeline for continuous validation and remediation, this research work proposes a DevSecOps approach to enable cloud environment IaC security. The proposed architecture evaluates CloudFormation Template file to the specified security policy by utilizing Open Policy Agent (OPA). Any violations are flagged by OPA, which may force the deployment process to stop. By incorporating security and compliance considerations into the development pipeline, this method guarantees that vulnerabilities are kept out of production systems. Organizations can improve the overall security posture of their cloud systems by proactively identifying and remediating security problems by implementing a DevSecOps approach. By ensuring that IaC deployments adhere to specified security policies based on the compliance requirements, OPA's continuous security validation reduces the possibility of misconfigurations and security flaws.**

Keywords*— *DevSecOps, Infrastructure as Code (IaC), Cloud Security, Continuous Integration/Continuous Deployment, Open Policy Agent, Security Policies, IAM Roles, CloudFormation Templates.

I. INTRODUCTION

Organizations frequently require assistance in keeping an eye on such a complicated and interconnected infrastructure as they grow and adjust to the cloud. Misconfigurations are frequent problems that put companies at risk and can result in costly, sometimes irreversible data breaches. Furthermore, making sure regulations are followed can be difficult. Under the shared responsibility model that governs their operations, cloud providers and their clients work together to ensure security. Organizations may create a strong security posture and guarantee ongoing protection of their cloud-based assets by utilizing cloud security products. In the dynamic world of cloud-native computing, maintaining, and enforcing policies across a wide range of apps and infrastructure components can be a difficult task. In response, Open Policy Agent (OPA) emerges as a robust and flexible open-source platform that helps companies create, manage, and apply rules consistently and successfully. The objective is to use Open Policy Agent (OPA) to automate compliance processes, hence improving the security of AWS cloud infrastructure. Using OPA, an open-source policy engine, we can create and implement fine-grained rules for AWS resources that guarantee compliance with best practices and industry standards.

Defining, upholding, and enforcing security and compliance rules in the cloud consistently may be accomplished with policy as code. Applying a set of guidelines or best practices to a company's cloud resources programmatically is known as "treating policy like code." This project intends to minimize human error, and maintain a safe AWS environment by automating compliance tests. OPA is an open-source policy engine that allows you to develop and validate policy as code. Declarative query languages like as Rego are used in OPA to create policy in the form of code validations. These validations evaluate data, such as infrastructure as a code, considering your company's security and compliance policies. For example, this suggests that you may create a Rego policy to determine whether resources would not comply with industry standards prior to development. As cloud-native technologies and microservices architectures are used, organizations have an inherent complexity in managing security, compliance, access control, and other requirements to contend with. These settings are dynamic, and a strong and adaptable policy management solution is required due to the sheer number of services and data.

A. Open Policy Agent

It is a general-purpose, open-source policy engine designed to give a standard foundation for using policy-as-code in any area. You may describe policy as code using OPA's high-level declarative language, Rego (pronounced "ray-go"), which was created specifically for policies. Policies may be defined, put into practice, and enforced via CI/CD pipelines, API gateways, microservices, Kubernetes, and more. OPA, to put it briefly, operates by severing the link between policy enforcement and decision-making.

B. Features of Open Policy Agent

- **Declarative Policy Language—Rego:** OPA utilizes a high-level declarative language called Rego. Rego may be used by stakeholders with and without technical expertise since it is meant to be expressive and readable by humans. Rego policies are logical guidelines that outline acceptable conduct and prohibited actions. This policy demonstrates the readability and simplicity of Rego by granting "admin" read access to "document" resources.

- **Dynamic Policy Evaluation:** In real-time, OPA may evaluate policies dynamically based on the processing of requests and data. Unlike static, pre-defined rules, OPA's dynamic assessment adapts to changing conditions and requirements, which makes it suitable for microservices and cloud-native applications.

979-8-3503-8579-3/24/$31.00 ©2024 IEEE
DOI 10.1109/ICOECA62351.2024.00063

- **“Policy as Code” Paradigm:** OPA advances the "Policy as Code" paradigm by considering policies as code objects that are versioned, tested, and maintained alongside infrastructure settings, application code, and other data sources. This method enhances the cooperation between the operators, developers, and security teams.

- **Cloud-Native Design:** OPA is consistent with cloud-native concepts and appropriate for orchestration and containerized systems like as Kubernetes. It integrates well with cloud-native environments' most popular tools and technologies.

Overall, cloud compliance automation can help organizations to achieve a few important benefits, including reduced risk, improved efficiency, increased accuracy, improved visibility, and improved scalability, Organizations may lower their risk of non-compliance, save time and money, and strengthen their overall security posture by automating compliance operations. In addition to the benefits mentioned above, cloud compliance automation can also help organization to:

- **Improve collaboration:** Automation may aid in the dismantling of organizational silos amongst various departments including compliance, security, and IT. This may facilitate better cooperation and communication about compliance-related concerns.

- **Reduce costs:** Because automation eliminates the need for manual labor, it can assist lower compliance costs. This can free up money for further IT projects.

- **Improve agility:** Automation can make it easier for businesses to react quickly to changes in the law. This is so that automation may be easily adjusted to meet the changing requirements.

Cloud compliance automation is an essential tool for any organization that operates in a regulated industry. By automating compliance processes, organizations can reduce their risk, improve efficiency, and gain a competitive advantage.

II. LITERATURE SURVEY

Modeling continuous security: A conceptual model for Automated DevSecOps using Open-source software (OSS) over Cloud (ADOC) provides a conceptual framework for automated DevSecOps utilizing open-source software over cloud. It was developed by Rinkaj Goyal and Rakesh Kumar. The paradigm is built around a conceptual framework for continuous security, which highlights four fundamental ideas for maintaining security throughout the software development lifecycle:

- **Security as a shared responsibility:** Everyone should be accountable for security, including operations teams and developers.
- **Automation:** To guarantee that security measures are implemented consistently and effectively, automation is necessary.
- **Continuous monitoring:** Continuous security monitoring is necessary to spot risks and take prompt action.
- **DevSecOps culture:** The whole organization, from the top down, should embrace a culture of security.

OSS tools are also used by the ADOC model to automate security control implementation, testing, and monitoring. This makes it easier to guarantee that security is implemented consistently and effectively across the DevOps process. The ADOC model was assessed in the report using a software development business case study. The case study's findings demonstrated how the ADOC model enhanced the software development process' security at the organization. One significant addition to the field of DevSecOps is the ADOC model. It offers a useful method for utilizing OSS tools to include security into the DevOps process. In addition, the model is adaptable and may be used in many corporate settings.

The goal of creating a safe and controlled cloud environment using OPA and AWS services is in line with the paper "Implementation of DevSecOps by Integrating Static and Dynamic Security Testing in CI/CD Pipelines," which by Agung Maulana Putra and Herman Kabetta offers a useful method for doing so. The authors stress the need of integrating security testing throughout the software development lifecycle and promote the use of DevSecOps approaches. This strategy fits with our project's objective of leveraging OPA to automate policy enforcement for CloudFormation Template files. The advantages of including both static and dynamic security testing methods in CI/CD pipelines are emphasized in the article. While Dynamic Application Security Testing (DAST) assesses apps while they are in use, Static Application Security Testing (SAST) examines source code to find vulnerabilities.

The authors propose a five-stage approach to implement DevSecOps:

1. **Continuous Development:** Incorporates security considerations into the development process from the outset.
2. **Continuous Testing:** Integrates automated security testing tool into the CI/CD pipeline to detect vulnerabilities early.
3. **Continuous Integration:** Seamlessly integrated security testing results into the overall CI/CD process.
4. **Continuous Deployment:** Deploys secure applications with minimal disruption.
5. **Continuous Monitoring:** Continuously monitors applications for vulnerabilities and potential threats.

The necessity of security automation is emphasized in the article, which aligns with our proposed use of OPA to automate policy enforcement. The entire security posture is improved and manual intervention is decreased by automated security procedures.

A thorough explanation of DevSecOps concepts and how they apply to infrastructure as code (IaC) can be found in the paper "DevSecOps: A Security Model for Infrastructure as Code

Over the Cloud," written by Amr Ibrahim et al. The authors support the inclusion of security practices in the IaC phase as well as the software development lifecycle. This is in line with the goal to use OPA to automate policy enforcement for CloudFormation template files.

The relevance of IaC security is emphasized in the paper, along with its role in defending cloud infrastructure from vulnerabilities and misconfigurations. This is consistent with the goal of our research, which is to secure cloud resources by comparing CloudFormation Template files to predetermined security guidelines. The writers go over several IaC security strategies, such as continuous monitoring, dynamic testing, and static code analysis.

.

The paper introduces several key concepts related to DevSecOps and IaC security:

- **Shifting security left:** Embedding security controls into the development and deployment process to prevent vulnerabilities from reaching production environments.
- **Continuous security validation:** Integrating automated security checks into the CI/CD pipeline to continuously assess the security posture of IaC templates.
- **Policy-driven security:** Utilizing policy engines like Open Policy Agent (OPA) to enforce security policies and flag violations.

The study concludes that DevSecOps is a crucial strategy for resolving the security issues raised by IaC and cloud adoption. Organizations may improve the overall security posture of their cloud systems by regularly validating and resolving security threats by including security practices into the DevOps lifecycle and utilizing tools such as OPA.

III. Proposed Methodology

Deployments of Infrastructure as Code (IaC) and the human verification of security guidelines and compliance requirements result in errors, delays, and inefficiencies. It is critically necessary to have an automated system that seamlessly integrates with the deployment process to support real-time assessment, security policy enforcement, and proactive detection of non-compliance issues. Deployments of Infrastructure as Code (IaC) and the human verification of security guidelines and compliance requirements result in errors, delays, and inefficiencies. It is critically necessary to have an automated system that seamlessly integrates with the deployment process to support real-time assessment, security policy enforcement, and proactive detection of non-compliance issues. Fig 1. shows the proposed system architecture.

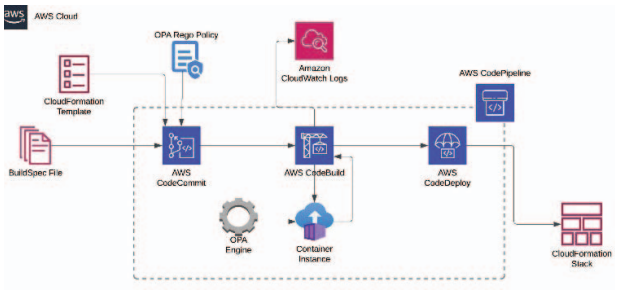

Fig 1. System Architecture

A. Research Motivation

The proposed model acts a proactive security measure that addresses cloud resource misconfigurations before they occur, The proposed model adds a security layer to the cloudformation deployments so that the IaC Templates adheres to a set of compliance requirements for the particular AWS resources it creates. Resource misconfigurations in the cloud is a security issue that needs to be addressed in almost every infrastructure that functions on the cloud. While cloud governance frameworks on AWS exist, current approaches often lack the flexibility and granularity needed to enforce complex security and compliance policies across diverse cloud resources This project proposes a solution that leverages Open Policy Agent (OPA) and its Rego policy language on AWS. OPA will enable the creation of rego policies can be written to define security and compliance requirements in a declarative and reusable manner. These policies can be easily updated to adapt to changing needs.

Continuous Integration / Continuous Deployment (CI/CD) pipelines can be coupled with OPA to automatically assess cloud resources in compliance with predefined policies. Any infractions found may trigger remedial warnings. The objective is to use AWS CodePipeline, AWS CodeCommit, and AWS CodeBuild to create a CI/CD pipeline in our AWS environment. CodeCommit shown in Fig. 2 contains the cloudformation template, the rego policy file and the buildspec file used by CodeBuild. The pipeline starts when a new code commit is made to the CodeCommit repository, and the buildspec file contains commands that makes use of the OPA engine and evaluates the cloudformation template against the set compliance policy written in rego.

The validation procedure will stop the pipeline if any policy is broken. The pipeline moves into the deployment stage only if the template is validated as “true” and it adheres to the set policy. There are specific IAM Roles designed for deployment and management to follow the least privilege security measure.

Fig 2. AWS CodeCommit Repository

IV. OPERATIONAL MECHANSIM

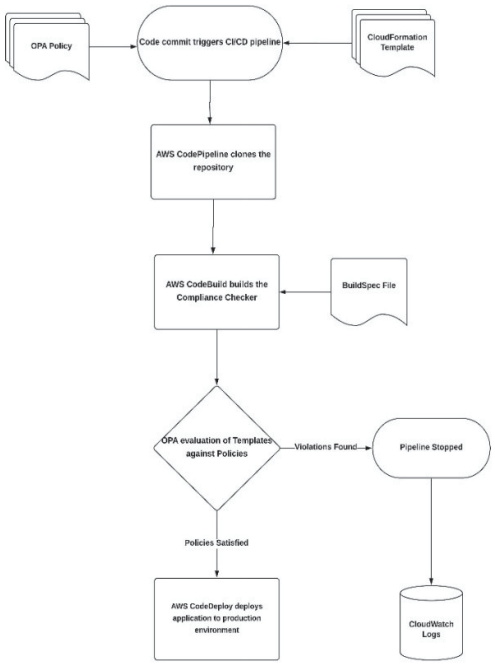

Fig 3. Flowchart of Operational Mechanism

Fig 3. depicts the flow of the operational mechanism.

Starting the CI/CD Pipeline: A code commit to the AWS CodeCommit repository starts the CI/CD pipeline which means whenever a new cloudformation template or a rego policy is added to the CodeCommit repository or even changes in the buildspec file are recorded by the pipeline and the pipeline is started.

Code Cloning and Building: To guarantee that the most recent code changes are available for the build process, AWS CodePipeline smoothly clones the repository holding the modified code. This is then used by AWS CodeBuild for further actions.

Policy Evaluation using Open-Source Policy Engine (OPA): The buildspec file shown in Fig. 4, which resides in the CodeCommit repository contains commands that would download the OPA binary into a container Instance which would be used by CodeBuild and also this file contains commands that performs the evaluation of the cloudformation template to a set policy in rego, The OPA engine is present in this container instance during evaluation.

OPA-Static-Prevention / buildspec.yaml Info

```
version: 0.2

phases:
  install:
    runtime-versions:
      python: 3.11
    commands:
      - echo Installing OPA...
      - curl -L -o opa https://openpolicyagent.org/downloads/latest/opa_linux_amd64
      - chmod 755 ./opa
  build:
    commands:
      - echo Running OPA...
      - |
        result=$(./opa eval --format=pretty --data policy.rego --input template.json "data.opa_policies.allow")
        echo $result
        if [ "$result" == "false" ]; then
          exit 1
        fi
```

Fig 4. Build-spec File

Enforcement of Policy Compliance: OPA prevents the deployment of non-compliant infrastructure by immediately stopping the pipeline at the Build phase if it finds any violations of the control policies. This rigorous enforcement process guarantees that code only moves through the pipeline in accordance with the organization's policies.

Deployment Upon Policy Satisfaction: The pipeline moves on to the deployment phase if the CloudFormation template files pass OPA's stringent inspection. The continuous deployment service offered by AWS CodeDeploy takes charge and carefully deploys the application to the production environment.

Monitoring and Logging: AWS CloudWatch Logs keeps track of, and archives log data produced by the different components involved in the CI/CD pipeline. This thorough logging system makes troubleshooting easier if there are any problems and offers insightful information about how the pipeline is being executed.

V. RESULTS AND DISCUSSION

The model has demonstrated the ability to deliver infrastructure changes automatically and consistently using our CI/CD pipeline, which is run by AWS CodePipeline, CodeCommit, and CodeBuild as shown in Fig. 5. Changes to the CodeCommit repository triggers the pipeline. This starts the process of creating, testing, and distributing CloudFormation Template files. If OPA discovers policy violations during the evaluation of CloudFormation Templates, the pipeline responds by terminating the process. Prompt alerts are generated to notify relevant parties so remediation can begin immediately. This proactive approach ensures that only compliant configurations advance to the deployment stage.

A Cloud Formation template was developed to create an EC2 instance in the ap-southeast-1a availability zone, with a t2.micro instance type. It uses an AMI and a key pair that you specify. It also creates a security group that allows SSH and HTTP access from any IP address.

Here, we have a set of specifications for the EC2 instances which should be created in our cloud environment and we also developed a rego policy file that ensures that only the cloudformation templates that adhere to these EC2 specifications gets deployed into the cloud environment through CloudFormation Stack. The pipeline was successful to the deployment stage because the cloudformation template was in compliance with the set policy.

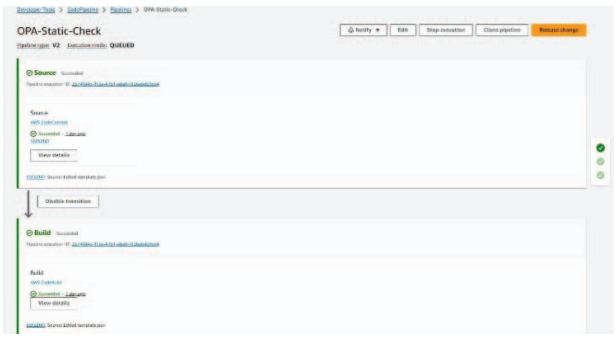

Fig 5. The CodePipeline Source and Build Phase

Fig 6. Evaluation of Policy with the Template File

The Build phase is only successful if the *opa eval* command returns *true.* This is evaluated on a container instance of our choice while setting up CodeBuild as illustrated in Fig. 6. The pipeline then moves onto to the Deployment phase shown in Fig. 7, where it deploys the template through cloudformation stack shown in Fig. 8 and 9.

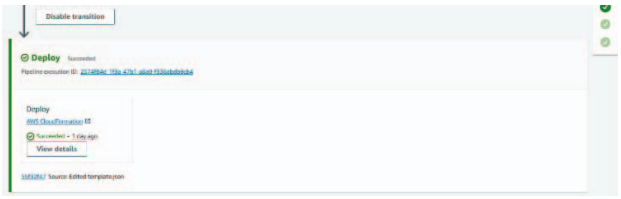

Fig 7. Deployment Phase of the Pipeline

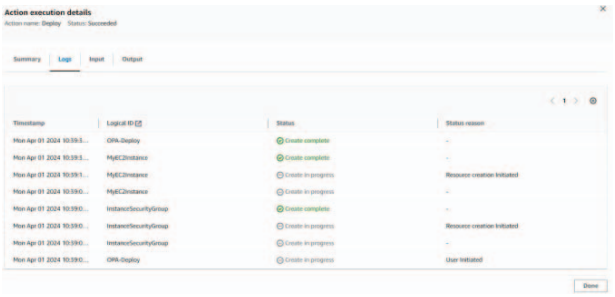

Fig 8. CloudFormation Stack Deployed by the pipeline

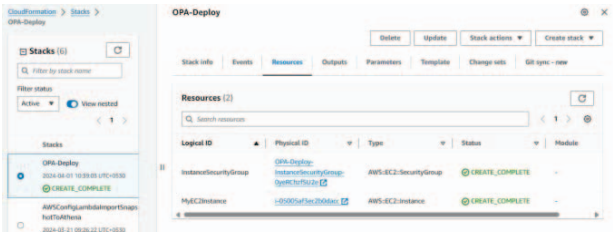

Fig 9. Resources Deployed by the CloudFormation Stack

VI. Conclusion and Future Work

The integration of Open Policy Agent (OPA) into a CI/CD pipeline has successfully enhanced the security, compliance, and overall quality of cloud infrastructure provisioning. By enforcing a set of predefined control policies, OPA effectively prevents the deployment of non-compliant infrastructure, ensuring that only secure and compliant configurations reach production environments.

The successful implementation of OPA in the CI/CD pipeline has demonstrated its effectiveness in addressing the challenges of cloud infrastructure provisioning. Its ability to enforce policies, detect violations early, and automate compliance checks has significantly improved the security, compliance, and quality of cloud deployments. The integration of OPA into CI/CD pipelines offers a promising future scope for enhancing application development and deployment processes.

References

[1] Manohar Marandi, A. Bertia, Salaja Silas, "Implementing and Automating Security Scanning to a DevSecOps CI/CD Pipeline", 2023 World Conference on Communication & Computing (WCONF), pp.1-6, 2023.

[2] Ruslan Shevchuk, Mikolaj Karpinski, Mykhailo Kasianchuk, Ihor Yakymenko, Andriy Melnyk, Roman Tykhyi, "Software for Improve the Security of Kubernetes-based CI/CD Pipeline", 2023 13th International Conference on Advanced Computer Information Technologies (ACIT), pp.420-425, 2023

[3] Federico Lombardi, Alberto Fanton, "From DevOps to DevSecOps is not enough. CyberDevOps: an extreme shifting-left architecture to bring cybersecurity within software security lifecycle pipeline", Software Quality Journal, 2023.

[4] Narasimha Rao Vajjhala, "An Exploratory Analysis of Cloud Security Models in Social Networks", 2023 3rd International Conference on Pervasive Computing and Social Networking (ICPCSN), pp.935-941, 2023.

[5] Vikas Agarwal, Chris Butler, Lou Degenaro, Arun Kumar, Anca Sailer, Gosia Steinder, "Compliance-as-Code for Cybersecurity Automation in Hybrid Cloud", 2022 IEEE 15th International Conference on Cloud Computing (CLOUD), pp.427-437, 2022.

[6] Rahul Mishra, Manish Gupta, Vikram Rajpoot, "Identifing the Future Security Issues Methods for Secure Data in the Cloud Computing", 2021 5th International Conference on Information Systems and Computer Networks (ISCON), pp.1-4, 2021.

[7] Christian Banse, Immanuel Kunz, Angelika Schneider, Konrad Weiss, "Cloud Property Graph: Connecting Cloud Security Assessments with Static Code Analysis", 2021 IEEE 14th International Conference on Cloud Computing (CLOUD), pp.13-19, 2021.

[8] Tarik Eltaeib, Nazrul Islam, "Taxonomy of Challenges in Cloud Security", 2021 8th IEEE International Conference on Cyber Security and Cloud Computing (CSCloud)/2021 7th IEEE International Conference on Edge Computing and Scalable Cloud (EdgeCom), pp.42-46, 2021.

[9] Constantin Adam, Muhammed Fatih Bulut, Milton Hernandez, Maja Vukovic, "Cognitive Compliance: Analyze, Monitor and Enforce Compliance in the Cloud", 2019 IEEE 12th International Conference on Cloud Computing (CLOUD), pp.234-242, 2019.

[10] Rahul Balabhadruni, Prajwal Kharvi and Monto Manu, "Implementation of DevSecOps using Open-Source tools", *International Journal of Advance Research Ideas and Innovations in Technology 5*, pp. 1050-1051, 2019.

[11] Z. Ahmed and S. C. Francis, "Integrating Security with DevSecOps: Techniques and Challenges", *2019 International Conference on Digitization (ICD)*, pp. 178182, 2019.

[12] Juncal Alonso, Radosław Piliszek, Matija Cankar, "Embracing IaC Through the DevSecOps Philosophy: Concepts, Challenges, and a Reference Framework", *IEEE Software*, vol.40, no.1, pp.56-62, 2023.

[13] Arunava Roy, Kanchan Patil, "Framework for Cloud Security Initiatives in Small and Medium-Sized Enterprises", *2023 International Conference on Advancement in Computation & Computer Technologies (InCACCT)*, pp.444-449, 2023.

[14] Rajesh Rompicharla, Bhaskar Reddy P. V, "Continuous Compliance model for Hybrid Multi-Cloud through Self-Service Orchestrator", *2020 International Conference on Smart Technologies in Computing, Electrical and Electronics (ICSTCEE)*, pp.589-593, 2020.

[15] Patrick Lukas Schubert, Bernd Wachter, "IT-Sicherheit und Compliance in heterogenen Cloud Umgebungen—Compliance-as-Code als Schlüssel zur Umsetzung regulatorischer Anforderungen", *HMD Praxis der Wirtschaftsinformatik*, 2023.