Security-as-Code Concept for Fulfilling ISO/IEC 27001:2022 Requirements

Oleksandr Vakhula[1], Yevhenii Kurii[1], Ivan Opirskyy[1], and Vitalii Susukailo[1]

[1] *Lviv Polytechnic National University, Information Security Department, Lviv, 79000, Ukraine*

Abstract

This article thoroughly examines and analyzes the latest iteration of the ISO/IEC 27001:2022 standard, outlining the most contemporary requirements for information security management systems. The primary focus of this study centers on the novel control A.8.9, which is dedicated to configuration management. The article delves into the necessity and significance of effectively implementing this control element to ensure a high level of infrastructure security and strategically reduce the risks of security breaches. Particular attention is given to the innovative security approach known as "Secure as Code." This methodology involves integrating security measures during the early stages of software development to effectively counter potential threats and ensure system resilience from the initial development phase to the operational stage. Serving as a comprehensive analysis and review of current trends in information security, the article not only provides readers with a broad perspective on the standard but also offers specific recommendations for the successful implementation of control A.8.9 of the ISO/IEC 27001:2022 standard. Additionally, practical tips are shared for seamlessly integrating the Secure as Code strategy into development practices. To support the research, an extensive review of literature and articles providing information on the implementation of ISO 27001 and the security-as-code approach were conducted.

Keywords

Information security, cybersecurity, ISO/IEC 27001:2022, information security framework, configuration management, security-as-code, infrastructure-as-code, DevSecOps, cloud environments, cloud service provider, software development cycle, cloud security threats, shift-left security approach.

1. Introduction

Virtually all modern business processes are defined and driven by information and data. In our digital economy, almost nothing functions without the exchange of information. Our core services rely on critical infrastructures [1, 2], and their functionality is highly dependent on the exchange of information and data. Information security is integral to the reality of our work and life. Therefore, enterprises of any size must protect daily information operations, critical data, and intellectual property from cyber threats [3].

In this age of industrialized cyberattacks, adapting to ever-changing information security risks requires a timely and flexible approach to building a resilient and secure enterprise. Taking into account the constant evolution of cyber threats and their rapid increase, the relevance of updating information security practices becomes obvious. The need to adapt to the latest challenges of the digital environment has become critical [4].

And that's where the new ISO/IEC 27001:2022 standard comes into play [5], focusing on a process-oriented approach to information security management. For more than two decades, the ISO 27001 standard has

CPITS-2024: Cybersecurity Providing in Information and Telecommunication Systems, February 28, 2024, Kyiv, Ukraine
EMAIL Oleksandr.p.vakhula@lpnu.ua (O. Vakhula); kuriy.yevgeniy@gmail.com (Y. Kurii); iopirsky@gmail.com (I. Opirskyy); vitalii.susukailo@gmail.com (V. Susukailo)
ORCID: 0009-0008-5367-3344 (O. Vakhula); 0000-0002-3423-5655 (Y. Kurii); 0000-0002-8461-8996 (I. Opirskyy); 0000-0003-4431-9964 (V. Susukailo)

remained the recognized framework for building information security management systems. Despite the potential obsolescence of the practices outlined in it, according to the latest report from the International Standardization Organization, in the year 2022, the number of standard setters increased by 21% compared to 2021 [6].

In response to global challenges, an updated and improved version of the ISO/IEC 27001 standard was released at the end of 2022. As the most prominent standard in the world of information security management, this standard plays a critical role in helping organizations protect their information assets. [7] Given the vital importance of this task in today's digital environment, the update of the standard reflects a continuous effort to improve and adapt to growing demands and threats. This is aimed at creating more reliable and effective information protection mechanisms and increasing trust in the digital space [8].

In this article, the authors investigate and analyze the notable changes in the latest iteration of the international standard ISO/IEC 27001:2022, as compared to the previous 2013 version. Specifically, the focus lies on conducting an in-depth examination of control A.8.9—Configuration Management.

The article aims to develop practical recommendations in alignment with the ISO/IEC 27001:2022 standard for implementing secure configurations of critical infrastructure assets through the application of the "Security as Code" approach.

2. Overview of the ISO/IEC 27001:2022 Standard

2.1. Description of the New Controls

Many changes in the new version of the standard are editorial, for example, changing "international standard" to "document" throughout and rearranging phrases to allow for better international translation [9].

However, the new security controls in Annex A demand the utmost attention. There are 11 of them in the updated version of the ISO/IEC 27001:2022 standard, namely:

1. Threat intelligence
2. Information security for the use of cloud services
3. ICT readiness for business continuity
4. Physical security monitoring
5. Configuration management
6. Information deletion
7. Data masking
8. Data leakage prevention
9. Monitoring activities
10. Web filtering
11. Secure coding.

A more detailed description of the new controls, as well as recommended activities for transitioning to the new version of the standard, are provided in Table 1.

Table 1

Description of new controls and recommended activities to transition according to ISO/IEC 27001:2022

Clause	Name of control	Control	Purpose	Action items
A.5.7	Threat intelligence	Information relating to information security threats shall be collected and analyzed to produce threat intelligence.	Understanding attackers and their methods in the context of your IT landscape.	• Update the Vulnerability Management Policy • Update the Contacts with special interest groups
A.5.23	Information security for use of cloud services	Processes for acquisition, use, management, and exit from cloud services shall be established by the organization's information security requirements.	The introduction through operation to exit strategy regarding cloud initiatives now needs to be considered comprehensively.	• Create the 3rd party evaluation register
A.5.30	ICT readiness for business continuity	ICT readiness shall be planned, implemented, maintained, and tested based on business continuity objectives and ICT continuity requirements.	The requirements for the IT landscape should be derived from the overall business processes and the ability to recover operational capabilities.	• Update the Business Continuity Management Policy/Plan
A.7.4	Physical security monitoring	Premises shall be continuously monitored for unauthorized physical access.	The use of alarm and monitoring systems to prevent unauthorized physical access has gained more emphasis.	• Update the Physical and Environmental Security Policy • Ensure using of the alarm and monitoring systems for

				protection against unauthorized physical access
A.8.9	Configuration management	Configurations, including security configurations, of hardware, software, services, and networks shall be established, documented, implemented, monitored, and reviewed.	Hardening and secure configuration of IT systems.	• Create the hardening baselines/configuration documentation for key systems and services (servers, network equipment, workstations) • Enforce the secure configuration/hardening of key information systems and company assets
A.8.10	Information deletion	Information stored in information systems, devices, or in any other storage media shall be deleted when no longer required.	Compliance with external requirements, such as data protection deletion concepts needs to be implemented.	• Update the Information Classification Policy • Create the Data Retention Policy • Ensure the timely deletion of obsolete information
A.8.11	Data masking	Data masking shall be used by the organization's topic-specific policy on access control and other related topic-specific policies, and business requirements, considering applicable legislation.	Use techniques that mask data, such as anonymization and pseudonymization, to bolster your data protection.	• Update the Information Classification Policy
A.8.12	Data leakage prevention	Data leakage prevention measures shall be applied to systems, networks, and any other devices that process, store, or transmit sensitive information.	Taking steps to help prevent sensitive data from being leaked.	• Consider using the DLP system • Otherwise, implement compensating controls or review and accept the risk
A.8.16	Monitoring activities	Networks, systems, and applications shall be monitored for anomalous behavior and appropriate actions taken to evaluate potential information security incidents.	Your organization should be monitoring network security and application behavior to detect any network anomalies.	• Update the Logging and Monitoring Policy • Install and configure the SIEM system
A.8.23	Web filtering	Access to external websites shall be managed to reduce exposure to malicious content.	Helps prevent users from viewing specific URLs containing malicious code.	• Update the Malware Protection Policy • Enforce the web content filtering function
A.8.28	Secure coding	Secure coding principles shall be applied to software development.	Using tools, commenting, tracking changes, and avoiding insecure programming methods are ways to ensure secure coding.	• Enforce the Secure SDLC for company products

As can be seen from the table, the new controls are, in fact, an addition and a certain extension of the existing domains of the previous version of the standard. Therefore, they can be relatively easily integrated into the existing processes of the organization, whose information security management system is built according to ISO/IEC 27001:2013.

2.2. Detailed Overview of the A.8.9—Configuration Management Control

Configurations—whether acting as a single configuration file, or a group of configurations linked together—are the underlying parameters that govern how hardware, software, and even entire networks are managed.

As an example, a firewall's configuration file will hold the baseline attributes that the device uses to manage traffic to and from an organization's network, including block lists, port forwarding, virtual LANs, and VPN information.

Configuration management is an integral part of an organization's broader asset management operation. Configurations are key in ensuring that an infrastructure not only operating as it should but also in securing devices against unauthorized changes or incorrect amendments on the part of maintenance staff and/or vendors [10–13].

Configuration management safeguards find diverse interpretations across various information security frameworks, ranging from SOC2 and HIPAA to PCI DSS and CIS Critical Security Controls [14].

Table 2 illustrates the mappings of the CIS Critical Security Controls (CIS Controls) and CIS Safeguards to ISO/IEC 27001:2022 concerning the management of configurations [15].

Table 2
Mapping of the CIS Controls to ISO/IEC 27001:2022 concerning the management of configurations

CIS Safeguard	Title	Description	Control #	Control Title
4,1	Establish and Maintain a Secure Configuration Process	Establish and maintain a secure configuration process for enterprise assets (end-user devices, including portable and mobile, non-computing/IoT devices, and servers) and software (operating systems and applications). Review and update documentation annually, or when significant enterprise changes occur that could impact this Safeguard.	A8.9	Configuration management
4,2	Establish and Maintain a Secure Configuration Process for Network Infrastructure	Establish and maintain a secure configuration process for network devices. Review and update documentation annually, or when significant enterprise changes occur that could impact this Safeguard.	A8.9	Configuration management
4,3	Configure Automatic Session Locking on Enterprise Assets	Configure automatic session locking on enterprise assets after a defined period of inactivity. For general-purpose operating systems, the period must not exceed 15 minutes. For mobile end-user devices, the period must not exceed 2 minutes.	A8.9	Configuration management
4,7	Manage Default Accounts on Enterprise Assets and Software	Manage default accounts on enterprise assets and software, such as root, administrator, and other pre-configured vendor accounts. Example implementations can include: disabling default accounts or making them unusable.	A8.9	Configuration management
4,8	Uninstall or Disable Unnecessary Services on Enterprise Assets and Software	Uninstall or disable unnecessary services on enterprise assets and software, such as an unused file-sharing service, web application module, or service function.	A8.9	Configuration management

Overall, organizations need to draft and implement configuration management policies for both new systems and hardware, and any that are already in use. Internal controls should include business critical elements such as security configurations, all hardware that holds a configuration file, and any relevant software applications or systems.

Control 8.9 asks organizations to consider all relevant roles and responsibilities when implementing a configuration policy, including the delegated ownership of configurations on a device-by-device, or application-by-application basis.

Where possible, organizations should use standardized templates to secure all hardware, software, and systems. Templates should:

- Attempt to utilize publicly available, vendor-specific, and/or open-source guidance on how best to configure hardware and software assets.
- Meet minimum security requirements for the device, application, or system that they apply to.
- Work in harmony with the organization's broader information security efforts, including all relevant ISO controls.
- Keep in mind the organization's unique business requirements—especially where security configurations are concerned—including how feasible it is to apply or manage a template at any given time.
- Be reviewed at appropriate intervals to cater for system and/or hardware updates or any prevailing security threats [15].

When addressing the crucial aspect of asset protection and security, a widely acknowledged approach to safeguard an organization's assets and adhere to industry best practices is the adoption of configuration standards provided by the Center for Internet Security (CIS), commonly referred to as CIS Benchmarks.

CIS Benchmarks is a set of globally recognized and consensus-driven best practices to help security practitioners implement and manage their cybersecurity defenses. Developed with a global community

of security experts, the guidelines help organizations proactively safeguard against emerging risks. Companies implement the CIS Benchmark guidelines to limit configuration-based security vulnerabilities in their digital assets.

Tools such as the CIS Benchmarks are important because they outline security best practices, developed by security professionals and subject matter experts, for deploying over 25 different vendor products. These best practices are a good starting point for creating a new product or service deployment plan or for verifying that existing deployments are secure [16].

3. Security as a Code—Overview and Advantages

We suggest considering an effective and reliable way to implement the above-described control using the security-as-code approach, based on the CIS Benchmark for cloud environments. Let's first understand the essence of this approach. Primarily, it's important to understand that this is a general name for an approach that can encompass the implementation of various types of security controls. Currently, we can distinguish the following:

- Policy or compliance as code.
- Security testing as code.
- Detection and response as code.

In our case, we will consider Policy or compliance as code, but we will refer to the approach generally as Security as Code.

The "Security as Code" approach in cloud environments means integrating security measures directly into the software development and deployment process. This approach allows for the automation of many aspects of security, making it more consistent and effective. It is particularly important in cloud environments, where it is necessary to respond quickly and flexibly to changes and new security challenges. "Security as Code" aids in the early detection of potential vulnerabilities and ensures adherence to regulatory and security standards.

"Security as Code" in cloud environments also involves integrating security policies and standards directly into Infrastructure as Code (IaC) templates, which are used for automating the deployment and management of cloud resources. This ensures that all deployed systems automatically comply with established security standards. Such an approach allows the use of code as a mechanism for ensuring continuous compliance with security requirements, reducing human error and potential configuration mistakes.

Using the "Security as Code" approach can significantly simplify configuration management processes, which are a new control envisaged by the ISO/IEC 27001:2022 standard, especially in cloud environments. This approach integrates security standards and policies directly into the code, automating the implementation and adherence to configuration requirements. Using the CIS Benchmark for cloud environments allows for the standardization and optimization of security settings, ensuring compliance with standards. This includes automatic checking and adjustment of configurations according to updated recommendations and best practices, enhancing efficiency and reducing risks in cloud environments [17].

In cloud environments, the use of the "security-as-code" approach is particularly important due to its high dynamics and flexibility. Cloud environments are often characterized by rapid changes, scaling, and distribution of resources, creating challenges for traditional configuration management. Let's take a closer look at these issues and their essence:

Rapid Scaling: Cloud environments often expand and contract, requiring flexible configuration management.

Diversity of Resources: Various types of resources may be used in the cloud, complicating the standardization of security settings.

Automation: Due to the large number of resources and services, effective configuration management requires a high level of automation. "Security as Code" allows for the automation and standardization of security processes, adapting to rapid changes in cloud environments, which helps ensure compliance with configuration management requirements in the ISO/IEC 27001:2022 standard.

4. Detailed Explanation of the Approach

IaC is an important and growing aspect of modern IT infrastructure management. It is increasingly used for automating the configuration, setup, and management of systems, especially in complex, large-scale environments. The growth of IaC meets the need for more efficient, scalable, and error-reducing approaches in IT infrastructure, indicating its growing importance in the industry. This trend signifies a significant and ongoing shift towards IaC practices in IT operations and development.

Most cloud service consumers agree that IaC allows for the automation of rapid service deployment in the cloud, eliminating any manual configuration and, consequently, errors. "Security as Code" further develops this approach by programmatically defining security policies, standards, and best practices to be used by default in configuration scripts already used for setting up cloud services and systems. IT departments can move from perpetually balancing business flexibility and security to realizing that these elements can be combined, ensuring an appropriate level of both without sacrificing either [18, 19].

Let's review the simplified scheme of implementation SaC (Fig. 2).

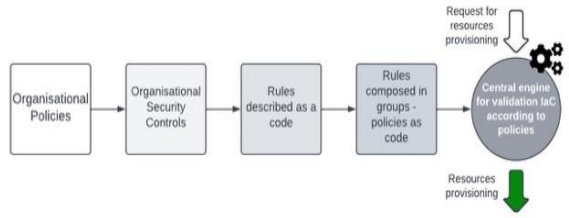

Figure 1: Simplified scheme of SaC concept

Organizational policies contain a list of required security controls, which are broken down into rules that are transformed into code understandable by the Centralized Policy Compliance Verification Service. These rules are later grouped into hierarchically organized policies with a structure of inheritance. This centralized service acts as a control barrier, checking the infrastructure code for compliance with established policies before deploying resources. For example, if an organization adopts a policy requiring the encryption of personal or financial data in databases, this requirement is integrated into the procedure as a rule that automatically activates when the DevSecOps team attempts to deploy code. If the code does not comply with the policy, it is rejected. Other policy examples may include requirements for using container or virtual machine images only from verified sources, the necessity of data backup, resource duplication in different availability zones, encryption of virtual machine disks, and requirements for resource labeling and nomenclature. Such policies can be sourced from standards, regulations, best practices, and recommendations from external organizations like the Cloud Security Alliance (CSA), Center for Internet Security (CIS), NIST, GDPR, HIPAA, PCI DSS, SOC2, as well as internal directives. Most of these requirements can be expressed in code, serving for prevention, detection, and response to violations [20].

IaC serves as the foundation for modules that perform static policy compliance analysis. IaC can be deployed using tools like CloudFormation for AWS, Deployment Manager for GCP, and Resource Manager for Azure, with Terraform or Pulumi being ideal for cross-platform applications. Static policy compliance checks should be integrated into the CI/CD process of infrastructure code, adhering to GitOps principles to prevent misconfigurations and correct discrepancies early in the development process [21].

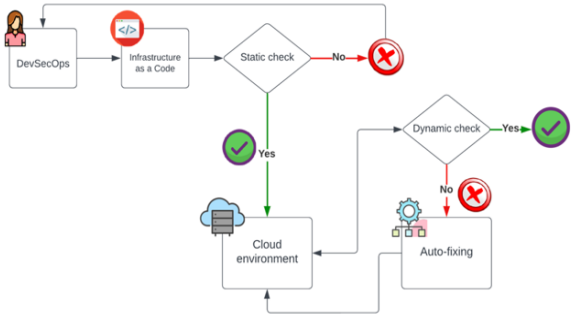

Figure 2: Process of static and dynamic validation according to policy

The component of the Centralized Policy Compliance Verification Service can be implemented using Open Policy Agent (OPA) or Regula, both open-source software. OPA, accepted into the Cloud Native Computing Foundation (CNCF) on March 29, 2018, and advancing to the "Graduated" maturity level by January 29, 2021, offers flexibility and numerous advantages. It operates independently of any

specific service or platform, allowing consistent policy application across environments and technologies. OPA provides fine-grained control over policy enforcement and integrates with major cloud providers and Kubernetes, making it a versatile choice for different infrastructures. With OPA, policies can be managed centrally and uniformly across platforms, reducing the risk of inconsistencies. As an open-source project, OPA benefits from a strong community and ecosystem, ensuring reliable support and continuous improvements.

These features make OPA a powerful tool for organizations seeking to implement a unified, flexible, and scalable Security as a code approach [22, 23].

5. Implementing CIS Benchmark Requirements Through Security-as-Code for Compliance with Control A.8.9 ISO 27001 (Configuration Management)

Policies will be written based on the CIS Amazon Web Services Foundations Benchmark v2.0.0. All CIS standards focus on technical configuration settings for maintaining or enhancing the security of the discussed technology, and they should be used in conjunction with other essential cyber hygiene tasks. The document, CIS Amazon Web Services Foundations Benchmark, is a set of security configuration best practices for Amazon Web Services (AWS). It typically covers areas like Identity and Access Management (IAM), logging and monitoring strategies, network resource configurations for information and resource protection, security measures for services like EC2, S3, RDS, etc., and general security and compliance guidelines.

This standard is intended to be the best practice for AWS customers to enhance the security of their AWS environments. It's also often used as a compliance and security standard for organizations using AWS for their infrastructure. We will take several requirements and describe them in the Rego language for further use in OPA [24].

Cloud misconfiguration involves various oversights and mistakes that can make cloud environments vulnerable to security risks. These vulnerabilities may lead to security incidents like breaches, unauthorized access by external hackers, and internal threats such as ransomware or malware. These issues exploit weaknesses in the cloud setup, allowing malicious entities to infiltrate the network [25].

Cloud misconfiguration is a widespread issue in the realm of cloud computing. Due to the complexity and rapidly evolving nature of cloud environments, organizations often find it challenging to maintain optimal security configurations. These misconfigurations can easily become prevalent, leading to significant vulnerabilities. As cloud adoption continues to grow, so does the risk of these misconfigurations, making it a prevalent concern for cybersecurity in modern IT environments. This underscores the importance of regular audits and reviews of cloud configurations to ensure security and compliance.

5.1. CIS Amazon Web Services Foundations Benchmark v2.0.0 - 06-28-2023 - 1.16. Ensure IAM Policies that Allow Full "*:*" Administrative Privileges are Not Attached

```
        statement.Action ==
    administrativePrivileges[_]
        statement.Effect == "Allow"
        policyName ==
    resource["values"]["name"]["new"]
    } }
     default allow = true
```

The policy imports the input file input.tfplan, representing the Terraform plan. It uses a deny rule to check each IAM policy resource in the Terraform plan. If a policy contains any statements allowing full administrative privileges (specified as "*:*"), it generates a denial message. The function hasFullAdminPrivileges checks if the IAM policy document contains any statements allowing "*:*" (full administrative privileges). The default statement allow = true at the end of the policy permits all other resources not matched by the deny rule.

5.2. CIS Amazon Web Services Foundations Benchmark v2.0.0 - 06-28-2023 -2.1.1. Ensure the S3 Bucket Policy is Set to Deny HTTP Requests

```
package
terraform.aws_s3_bucket_policy_validation
import input.tfplan
deny[msg] {
    resource = tfplan.resources[_]
    resource["type"] ==
"aws_s3_bucket_policy"
    not
hasDenyHttpStatement(resource["values"][
"policy"]["new"])
    msg = sprintf("S3 Bucket policy '%v'
does not deny HTTP requests and should
be denied.",
[resource["values"]["bucket"]])
}
hasDenyHttpStatement(policyDoc) {
    statements := policyDoc["Statement"]
    some i, statement := statements {
    statement.Effect == "Deny"
    statement.Action ==
"s3:GetObject"
containsHttpCondition(statement.Conditio
n)
} }
containsHttpCondition(condition) {
    keys := keys(condition)
    "IpAddress" in keys
    condition["IpAddress"] ==
{"aws:SourceIp": "HTTP request IP
address"}
}
default allow = true
  package
terraform.aws_iam_admin_policies
 import input.tfplan
 deny[msg] {
    resource = tfplan.resources[_]
    resource["type"] ==
"aws_iam_policy"  # Adjust the resource
type as per your Terraform
configuration.
 hasFullAdminPrivileges(resource["values
"]["name"]["new"])
    msg = sprintf("IAM policy '%v'
allows full administrative privileges
and should not be attached.",
[resource["values"]["name"]["new"]])
 }
 hasFullAdminPrivileges(policyName) {
    # Define a list of administrative
privileges you want to deny.
    administrativePrivileges := ["*:*"]
    resource_policy :=
data.aws_iam_policy_document[resource["
v
alues"]["policy"]["new"]]
    statements :=
resource_policy["Statement"]
    some i, statement := statements {62
```

It checks each S3 Bucket Policy resource in the Terraform plan. If the policy does not contain a Deny statement that denies HTTP requests, it generates a denial message.

The function hasDenyHttpStatement checks if the policy document contains a Deny statement that specifically denies HTTP requests for s3:GetObject actions.

The containsHttpCondition function checks if the Deny statement contains a condition that involves an HTTP request IP address.

The default allow = true statement at the end of the policy permits all other resources not matched by the deny rule.

5.3. CIS Amazon Web Services Foundations Benchmark v2.0.0 - 06-28-2023 - 2.2.1. Ensure EBS Volume Encryption is Enabled in all Regions)

```
package
terraform.aws_ebs_volume_encryption
import input.tfplan
deny[msg] {
    resource = tfplan.resources[_]
    resource["type"] == "aws_ebs_volume"
# Adjust the resource type as per your
Terraform configuration.
    not isEBSEncrypted(resource)
    msg = sprintf("EBS volume encryption
is not enabled in all regions in the
Terraform configuration.")
}
isEBSEncrypted(resource) {
    encryption_enabled :=
resource["values"]["encrypted"]["new"]
    encryption_enabled == true
}
default allow = true
```

The policy imports input data from 'input.tfplan', which represents the Terraform plan.

It uses a deny rule to check each AWS EBS volume resource in the Terraform plan. If the 'encrypted' attribute is not set to 'true' (meaning EBS volume encryption is not enabled), it generates a denial message.

The default 'allow = true' statement at the end of the policy permits all other resources that do not match the deny rule.

5.4. CIS Amazon Web Services Foundations Benchmark v2.0.0 - 06-28-2023 - 2.3.1. Ensure that Encryption-at-Rest is Enabled for RDS Instances

```
import input.tfplan
deny[msg] {
    resource = tfplan.resources[_]
    resource["type"] ==
"aws_db_instance"
    not isEncryptionEnabled(resource)
    msg = sprintf("RDS instance %s is
not configured with encryption at
rest.", [resource["name"]])
}
isEncryptionEnabled(resource) {
    # Modify this rule to match the
naming convention of your encryption
attribute.
    attribute_exists :=
resource["values"]["storage_encrypted"]
    attribute_value :=
resource["values"]["storage_encrypted"][
"new"]
    attribute_value == true
}
default allow = false
```

The policy imports input data from 'input.tfplan', which represents the Terraform plan.

It applies a denial rule to check each AWS RDS instance resource in the Terraform plan. If the 'storage_encrypted' attribute is not set to 'true' (meaning encryption at rest is not active), it generates a denial message.

5.5. CIS Amazon Web Services Foundations Benchmark v2.0.0 - 06-28-2023 - 3.1. Ensure CloudTrail is Enabled in all Regions

```
package terraform.aws_cloudtrail
import input.tfplan
deny[msg] {
    resource = tfplan.resources[_]
    resource["type"] == "aws_cloudtrail"
    not isCloudTrailEnabled(resource)
    msg = sprintf("AWS CloudTrail is not
enabled in all regions in the Terraform
configuration.")
}
isCloudTrailEnabled(resource) {
    # Modify this rule to match the
naming convention of your CloudTrail
attributes.
    attribute_exists :=
resource["values"]["is_multi_region_trai
l"]
    attribute_value :=
resource["values"]["is_multi_region_trai
l"]["new"]
    attribute_value == true
default allow = true
```

The policy imports the 'input.tfplan' data, representing the Terraform plan. It uses a deny rule for evaluating each AWS CloudTrail resource in the Terraform plan. If the 'is_multi_region_trail' attribute is not set to 'true' (indicating that CloudTrail is not configured to operate in all regions), a denial message is generated. The standard statement 'allow = true' at the end of the policy permits all other resources not covered by the deny rule.

5.6. CIS Amazon Web Services Foundations Benchmark v2.0.0 - 06-28-2023 - 5.2. Ensure no Security Groups Allow Ingress from 0.0.0.0/0 to Remote Server Administration Ports

```
package
terraform.aws_security_group_validation
import input.tfplan
deny[msg] {
    resource = tfplan.resources[_]
    resource["type"] ==
```

```
"aws_security_group_rule"
isRemoteAdminPort(resource["values"]["fr
om_port"])
isEverywhereAllowed(resource["values"]["
cidr_blocks"])
    msg = sprintf("Security group rule
allows ingress from 0.0.0.0/0 to remote
server administration ports: %v",
[resource["values"]["from_port"]])
}
isRemoteAdminPort(port) {
port == 22 // Add more remote server
administration ports as needed (e.g., 3389
for RDP)
}
isEverywhereAllowed(blocks) {
    "0.0.0.0/0" in blocks
}
default allow = true
```

This policy uses the 'input.tfplan' input, representing the Terraform plan. It checks each AWS Security Group Rule in the Terraform plan. If the rule allows ingress from 0.0.0.0/0 (anywhere) to remote server administration ports (e.g., SSH on port 22), it generates a denial message. The 'isRemoteAdminPort' function checks if the rule's 'from_port' matches a remote server administration port (like 22 for SSH). The 'isEverywhereAllowed' function verifies if 0.0.0.0/0 is present in the rule's 'cidr_blocks', indicating it allows ingress from anywhere. The default 'allow = true' statement at the end of the policy permits all other resources not covered by the deny rule.

5.7. CIS Amazon Web Services Foundations Benchmark v2.0.0 - 06-28-2023 - 4.9. Ensure AWS Config Configuration Changes are Monitored

```
package terraform.aws_config_monitoring
import input.tfplan
deny[msg] {
    resource = tfplan.resources[_]
resource["provider"] ==
"provider[\"aws\"]"
resource["type"] ==
"aws_config_configuration_recorder"
    not hasConfigMonitoring(resource)
msg = sprintf("AWS Config configuration
changes must be monitored.")
}
hasConfigMonitoring(recorder) {
recorder["values"]["recording_group"][0]
["all_supported"] == true
}
default allow = true
```

It checks each Configuration Recorder resource in the Terraform plan. If the recorder is not monitoring all supported resource types (all_supported set to true), it generates a denial message.

The function hasConfigMonitoring checks if the Configuration Recorder has all_supported set to true, indicating that it's monitoring all supported resource types.

The default allow = true statement at the end of the policy permits all other resources not matched by the deny rule.

However, Rego is a language that operates quite differently than most and can be quite unintuitive at first glance. It's more similar to SQL than to common imperative languages like Python, which means that the learning curve can be quite steep [26].

6. Practical Implementation Directly into the Development Cycle, Specifically into the Continuous Integration and Continuous Deployment (CI/CD) System

Policies are created, but the full potential of Terraform and OPA is realized when integrated into a CI/CD pipeline. In our example, we use GitHub Actions to trigger these checks on every pull request. GitHub Actions is a feature on the GitHub platform that allows automating software development workflows directly in a GitHub repository. This implementation ensures that any proposed infrastructure changes are checked against the policies we created earlier before they are merged and deployed in the production environment.

Below we provide an example of YAML code that will create a compliance check process, in our case, regarding the secure configuration of the AWS environment according to the CIS Benchmark.

Let's examine each step of the GitHub Actions workflow:

```
    Validate Terraform
    on:
```

```
pull_request:
  branches:
    - main
jobs:
  terraform:
    runs-on: ubuntu-latest
    steps:
      - name: Checkout repository
        uses: actions/checkout@v3
        with:
          fetch-depth: 0
      - name: Set up Terraform
        uses: hashicorp/setup-terraform@v2
      - name: Configure AWS Credentials
        uses: aws-actions/configure-aws-credentials@v2
        with:
          aws-access-key-id: ${{ secrets.AWS_ACCESS_KEY_ID }}
          aws-secret-access-key: ${{ secrets.AWS_SECRET_ACCESS_KEY }}
          aws-region: ${{ secrets.AWS_REGION }}
      - name: Terraform Init
        run: terraform init
      - name: Terraform Validate
        run: terraform validate
      - name: Generate Terraform Plan JSON Output
        run: |
          terraform plan -var="db_username=${{ secrets.DB_USERNAME }}" -var="db_password=${{ secrets.DB_PASSWORD }}" -out=tfplan.binary
          terraform show -json tfplan.binary | grep -v "::debug::" | tail -n +2 > plan.json
      - name: Install OPA
        run: |
          curl -L -o opa https://openpolicyagent.org/downloads/latest/opa_linux_amd64
          chmod 755 ./opa
          sudo mv opa /usr/local/bin/
      - name: Evaluate Rego Policies
        id: evaluate_policies
        run: |
          opa eval --data ./policies/aws_tags_policy.rego --input plan.json --format pretty "data.main.deny" > tags_policy_result.txt
          if [[ -s tags_policy_result.txt ]]; then
            exit 1
          fi
      - name: Add policy results to job summary
        if: failure()
        run: |
          if [[ -s tags_policy_result.txt ]]; then
            echo "Tags policy violations:" >> $GITHUB_STEP_SUMMARY
            cat tags_policy_result.txt >> $GITHUB_STEP_SUMMARY
          fi
```

[27]

1. Trigger Workflow on Pull Requests to 'main' Branch: The workflow initiates when a pull request is made to the 'main' branch.

2. Job Execution on Ubuntu Latest Runner: The job named 'terraform' runs on the latest Ubuntu virtual environment provided by GitHub Actions.

3. Checkout Repository: The repository's code is checked out for use in the workflow.

4. Set up Terraform: Installs and configures the specified version of Terraform.

5. Configure AWS Credentials: Sets up AWS credentials using secrets stored in the GitHub repository for secure access to AWS services.

6. Terraform Init: Initializes Terraform, setting up the working directory for Terraform operations.

7. Terraform Validate: Validates the Terraform files for syntax correctness.

8. Generate Terraform Plan: Creates a Terraform execution plan and outputs it in binary format, then converts it to JSON.

10. Evaluate Rego Policies: Evaluates custom policies written in Rego against the Terraform plan and outputs results to text files.

11. Add Policy Results to Job Summary: If there are policy violations, they are added to the job summary, which is visible in the GitHub Actions UI.

Each step ensures that code merged into the 'main' branch complies with defined policies and is safe to deploy (Fig.3).

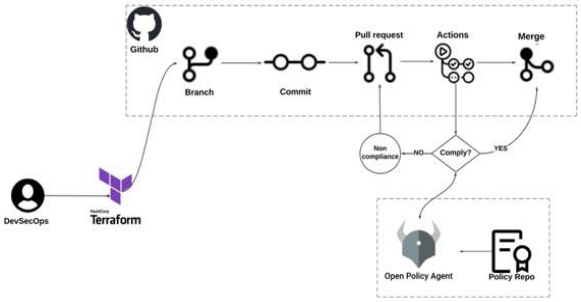

Figure 3: Scheme of implementation SaC with Open Policy Agent and GitHub Actions

The integration of Terraform and Open Policy Agent (OPA) into the Continuous Integration/Continuous Deployment (CI/CD) pipeline via GitHub Actions represents a significant advancement in IaC practices. This approach not only automates the process of infrastructure deployment but also enforces a high standard of compliance and security. By conducting rigorous policy evaluations on every pull request, the system ensures that only code that adheres to the established security and operational standards is merged into the 'main' branch. This methodology effectively minimizes human error and streamlines the deployment process, ensuring that all infrastructure changes are consistent, secure, and aligned with the Center for Internet Security (CIS) benchmarks for AWS environments. The detailed breakdown of each step in the GitHub Actions workflow highlights the thoroughness of the process, offering a clear and efficient pathway for maintaining the integrity of the infrastructure. This practice not only enhances security but also contributes to a more robust and reliable development cycle, ensuring that the infrastructure evolves in lockstep with the application it supports.

7. Summary

The integration of the Security as Code (SaC) concept into organizational practices is a pivotal move towards complying with the ISO/IEC 27001:2022 standards, with a special emphasis on Control A.8.9—Configuration Management. This article has delved into how SaC not only streamlines security processes but also directly supports the stringent requirements of ISO/IEC 27001:2022, particularly in the realm of managing and safeguarding information assets through effective configuration management. Utilizing SaC, especially in the context of Control A.8.9, automates and standardizes the configuration management processes. This harmonizes with ISO/IEC 27001:2022's mandate for maintaining an inventory of assets and ensuring the integrity of operational systems. Automated tools in SaC can track configurations and changes, ensuring that all assets are consistently configured in compliance with the defined security policies. SaC methodologies facilitate continuous monitoring of configuration states, ensuring that any deviations from the standard configurations are immediately identified and rectified. This continuous oversight is crucial for meeting ISO/IEC 27001:2022's requirements for Control A.8.9, which emphasizes the importance of maintaining secure configurations and promptly addressing any anomalies. By translating configuration policies into code, organizations can enforce consistent settings across their IT environment. This practice aligns with Control A.8.9's focus on ensuring the security of operational systems through proper configuration management, thereby enhancing the overall security posture as mandated by ISO/IEC 27001:2022. SaC offers comprehensive logging capabilities which are essential for documenting changes in system configurations. This level of documentation and traceability is not only a key aspect of Control A.8.9 but also a fundamental requirement of ISO/IEC 27001:2022, aiding in audits and compliance verification processes.

SaC methodologies enable organizations to assess risks associated with configuration changes proactively. This is in line with the objectives of Control A.8.9, which aims to manage risks stemming from changes in operational environments, ensuring that security measures are always up-to-date and effective.

In conclusion, the adoption of Security as Code is not just a compliance measure but a strategic approach that aligns closely with ISO/IEC 27001:2022, particularly in the context of Control A.8.9—Configuration Management. It provides a robust framework for managing configurations securely, automating compliance processes, and ensuring continuous monitoring and adaptation. As organizations navigate the complexities of information security, adopting

a security-as-code approach becomes imperative to meet evolving standards and maintain a robust, compliant security posture.

References

[1] P. Anakhov, et al., Protecting Objects of Critical Information Infrastructure from Wartime Cyber Attacks by Decentralizing the Telecommunications Network, in: Workshop on Cybersecurity Providing in Information and Telecommunication Systems, vol. 3550 (2023) 240–245.

[2] H. Hulak, et al., Dynamic Model of Guarantee Capacity and Cyber Security Management in the Critical Automated System, in: 2nd International Conference on Conflict Management in Global Information Networks, vol. 3530 (2023) 102–111.

[3] V. Susukailo, I. Opirsky, O. Yaremko, Methodology of ISMS Establishment Against Modern Cybersecurity Threats, Future Intent-Based Networking, LNEE 831 (2022) 257–271. doi: 10.1007/978-3-030-92435-5_15.

[4] Y. Kurii, I. Opirskyy, Analysis and Comparison of the NIST SP 800-53 and ISO/IEC 27001:2013, in: : Cybersecurity Providing in Information and Telecommunication Systems Vol. 3288 (2021) 21–32.

[5] ISO/IEC 27002: Information Security, Cybersecurity and Privacy Protection—Information Security Controls (2022). URL: https://www.iso.org/standard/75652.html

[6] Global Cybersecurity Outlook (2022). URL: https://www.weforum.org/reports/global-cybersecurity-outlook-2022

[7] Which ISO Standards are the Most Popular—Analysis of ISO 2019 Survey. URL: https://advisera.com/articles/which-iso-standards-are-the-most-popular-analysis-of-iso-2019-survey/

[8] Y. Kurii, I. Opirskyy, L. Bortnik, ISO/IEC 27001:2022 – Analysis Of Changes And Compliance Features Of The New Version Of The Standard, IXth International Scientific and Technical Conference Information Protection and Information Systems Security (2023) 15–17.

[9] What Are The ISO 27001 Changes In 2022. URL: https://bestpractice.biz/what-are-the-iso-27001-changes-in-2022/

[10] ISO 27002:2022, Control 8.9–Configuration Management. URL: https://www.isms.online/iso-27002/control-8-9-configuration-management/

[11] M. TajDini, V. Sokolov, P. Skladannyi, Performing Sniffing and Spoofing Attack Against ADS-B and Mode S using Software Define Radio, in: IEEE International Conference on Information and Telecommunication Technologies and Radio Electronics (2021) 7–11. doi: 10.1109/UkrMiCo52950.2021.9716665.

[12] M. TajDini, V. Sokolov, V. Buriachok, Men-in-the-Middle Attack Simulation on Low Energy Wireless Devices using Software Define Radio, in: 8th International Conference on "Mathematics. Information Technologies. Education:" Modern Machine Learning Technologies and Data Science, vol. 2386 (2019) 287–296.

[13] R. Marusenko, V. Sokolov, V. Buriachok, Experimental Evaluation of Phishing Attack on High School Students, Advances in Computer Science for Engineering and Education III, vol. 1247 (2020) 668–680. doi:10.1007/978-3-030-55506-1_59

[14] CIS Critical Security Controls Version 8. URL: https://www.cisecurity.org/controls/v8

[15] CIS Controls v8 Mapping to ISO/IEC 27001:2022. URL: https://www.cisecurity.org/insights/white-papers/cis-controls-v8-mapping-to-iso-iec-27001-2022

[16] What are CIS Benchmarks?. URL: https://aws.amazon.com/what-is/cis-benchmarks/#:~:text=CIS%20Benchmarks%20from%20the%20Center,and%20manage%20their%20cybersecurity%20defenses

[17] R. Ferreira, Policy Design in the Age of Digital Adoption: Explore how PolicyOps Can Drive Policy as Code Adoption in an Organization's Digital Transformation, 1st Edition, Packt Publishing (2022).

[18] C. Adtani, et al., Security as Code: The best (and Maybe Only) Path to Securing Cloud Applications and Systems (2022).
[19] S. Das, Security as Code, 1st Edition (2023).
[20] X. Zhang, Cloud Governance and Compliance on AWS with Policy as Code (2021).
[21] X. Zhang, Compliance as Code and Auto-Remediation with Cloud Custodian (2020).
[22] O. Vakhula, I. Opirskyy, O. Mykhaylova, Research on Security Challenges in Cloud Environments and Solutions Based on the "Security-as-Code" Approach Cybersecurity Providing in Information and Telecommunication System Vol. 3550 (2023) 55–69.
[23] B. Lee, Using Open Policy Agent (OPA) to Apply Policy-as-Code to Infrastructure-as-Code (2022).
[24] CIS Amazon Web Services Foundations Benchmark v2.0.0 (2023).
[25] A. Sukianto, Common Cloud Misconfigurations and How to Avoid Them (2023). URL: https://www.upguard.com/blog/cloud-misconfiguration
[26] Policy Language Documentation. URL: https://www.openpolicyagent.org/docs/latest/policy-language/
[27] Guest Expert on GitGuardian, What is Policy-as-Code? An Introduction to Open Policy Agent (2022). URL: https://blog.gitguardian.com/what-is-policy-as-code-an-introduction-to-open-policy-agent/