Rules-as-Code Cloud Assurance for Federal Suppliers: Converting NIST SSDF and Patch/Update Controls into Machine-Readable Authorization Evidence

Shuaib Ahmed
Bellevue, Washington, USA
khans8297@gmail.com

***Abstract*—Due to the emergence of strict regulatory standards such as the NIST Secure Software Development Framework (SSDF), the demand for compliance is higher. Traditional methods of implementing compliance measures are based on manual auditing and document management. This may lead to inefficiency, delayed processes, and security threats. For the above problems, this research presents a framework that automates compliance validation through rules-as-code approach in order to implement continuous assurance. This paper introduces the Rules-as-Code Cloud Assurance Framework (RC-CAF) which implements automated extraction and execution of rules in order to improve compliance processes in cloud-based environments. Experimental results have shown that the presented model has significantly improved compliance verification compared to FS-PKSE and CIA-Scheme models. It has achieved compliance accuracy of 96%, reduced computational costs to 180 ms and increased efficiency of compliance monitoring in real-time to 95%. The violation detection rate of 97% has been achieved with reduced processing time.**

Keywords— Cloud Assurance, Rules-as-Code, NIST SSDF, Compliance Automation, Cloud Security, Patch Management, Continuous Monitoring, Policy Enforcement, Cloud Auditing.

I. INTRODUCTION

Cloud computing is now an integral part of the digital ecosystem, providing flexible means to deliver services in various domains [1]. Enterprises have started using cloud technologies extensively to store valuable data and run mission-critical applications, necessitating compliance with security requirements [2]. The National Institute of Standards and Technology Software Security Development Framework (NIST SSDF) is an example of a framework that defines specific guidelines for secure software development and operation [3]. Nevertheless, implementing security controls in complex cloud environments can be a difficult task due to the nature of distributed cloud infrastructure [4]. Existing compliance techniques heavily rely on audits and static policies, which cannot adapt quickly to changes in cloud environments [5]. In contrast, cloud infrastructures undergo frequent changes that increase the distance between policy definition and implementation [6]. Such compliance gaps can persist for a long time, allowing policy violations to remain unnoticed [7].

Several studies have introduced several approaches for improving cloud security and integrity [8]. Approaches like FS-PKSE are concerned with securing the process of accessing information using encryption algorithms [9], whereas approaches like CIA-Scheme emphasize the importance of designing efficient auditing techniques for cloud service providers [10]. These approaches improve certain aspects of cloud security, they overlook the issue of compliance validation and automation of regulations [11]. With the increasing sophistication of the cloud environment, there is an increasing need for adopting automated and intelligent compliance techniques [12]. As such, the approach known as Rules-as-Code has been introduced. In this approach, policy requirements and conditions are translated into codes and are implemented throughout the processes of the system [13].

The suggested RC-CAF methodology is based on the above-mentioned idea through using rule extraction, machine readable policy and automation of evaluations process [14]. Compliance is achieved and maintained through enforcement, rather than just documentation of compliance. This way of managing compliance processes will increase transparency, traceability and efficiency of processes [15]. One more problem with achieving cloud assurance is associated with the fact that there is no unified method of associating system evidence with compliance requirements [16][17]. If there is no such connection, there is no possibility to verify if the system complies with regulations. This problem is solved by building a clear relationship between system log and patches and compliance rules [18] [19].

In the case of modern cloud platforms, scalability remains an important challenge due to high amounts of information as well as distributed nature of various elements in the system. The suggested RC-CAF solution allows for scalable implementation of rules evaluations in order to ensure constant performance levels irrespective of the amount of information or number of distributed elements in the system [20]. This approach may be successfully applied in enterprise and federal cloud solutions. Real-time monitoring becomes important in identifying and addressing any compliance breaches. Typically, periodic checks are used in traditional systems, current research offers continuous evaluation. Integration with automated reporting techniques makes the framework even more user-friendly. Creating auditable logs helps simplify the task of compliance validation, decreasing the amount of effort required from security professionals. This feature is very useful in cases when high regulatory

compliance is required from the system. Overall, the contribution described in this paper intends to address the issue of enforcing compliance policies by introducing an efficient and automated framework for managing cloud compliance.

1.1 Hypothesis

The hypothesis of this research proposal is that translating the regulatory compliance obligations into computer readable code and embedding this within the automated testing processes can result in greater accuracy of compliance management, faster processes, and increased real-time monitoring potential as opposed to the traditional systems of security and auditing.

1.2 Research Contributions

The study has presented a new Rules-as-Code Cloud Assurance Framework that helps validate compliance automatically using code rules extracted from NIST SSDF and update/patch policies. It suggests an architectural model that facilitates automatic extraction of rules, mapping evidence with rules, and evaluation of compliance status in real time. Moreover, it shows a comparison of the performance results of the proposed solution against existing ones. The research has also made a significant contribution to making compliance checking more efficient and scalable through versioning and report generation features.

II. LITERATURE SURVEY

Recently, research has pointed out the need for policy-as-code and compliance automation within cloud-native environments. The use of Kubernetes-based governance mechanisms and security validation techniques of Infrastructure-as-Code has proven the feasibility of applying automation principles to continuous policy enforcement within DevSecOps practices. Moreover, zero trust cloud architectures are becoming dependent upon automation principles in order to limit the human factor in their operations and enhance security assurances. However, although current research highlights automation benefits, these efforts mostly cover either access control or infrastructure validation without offering solutions to generate machine-readable authorization evidence. The RC-CAF framework aims at addressing that problem.

Zeng et al. [1] presented a forward secure public key searchable encryption (PKSE) protocol for cloud storage outsourcing to improve the confidentiality of data during searchable encryption. The researchers pointed out a serious weakness in existing PKSE systems whereby the cloud server may deduce certain private information about new encrypted files according to previously generated search tokens. To overcome this problem, Zeng et al. [1] suggested a forward security model to guarantee that the previous search token would not leak any information about the new encrypted file. A generic construction method based on attribute-based searchable encryption was formulated to support the design of the proposed scheme. It is shown that their scheme provides efficient searches under the condition of ensuring high security.

The Collaborative Integrity Auditing (CIA) protocol was suggested by Li et al. [2] for securing data integrity auditing services on data outsourced into multiple replicas hosted by different cloud service providers. This study is designed to address the weaknesses associated with conventional remote data integrity checking techniques that require the involvement of a third party auditor as well as heavy computations involving complex algorithms like bilinear pairing. The model is based on inter-cloud cooperation between several cloud service providers, thereby removing the need for a third-party auditor and greatly decreasing the overhead costs involved in tagging and checking. The CIA protocol makes use of light-weight hash functions, providing high efficiency as well as solid security features.

Ge et al. [3] introduced a VSSE algorithm for dynamic encrypted data stored in the cloud for supporting efficient searches by keywords and verifying results. This research paper focuses on solving the problem related to current algorithms that utilize asymmetric-key cryptography and lead to huge computation costs during the verification process, particularly when applied to massive cloud storage systems. To solve this issue, Ge et al. [3] use a symmetric key-based approach to verification through the concept of Accumulative Authentication Tags (AAT). These allow for efficient updates of the tags in case of inserting, deleting, or modifying the data. AAT is implemented through an organized index structure that consists of a search table and a verification list.

The paper by Xie et al. [4] suggests the design of an online orchestration model (MCE-SCO) that is aimed at managing services in multi-cloud environments to solve the problem of balancing security requirements and Quality of Service (QoS) in dynamically changing conditions. In the course of research, it was noted that there are significant disadvantages in current orchestration models when providing security for service chains in case of regular arrival of requests in a time interval. This problem can be solved through the application of a deep reinforcement learning approach that ensures effective orchestration of services under irregular request arrivals. At the same time, optimization occurs in the placement and coordination of services on several clouds with the minimalization of an aggregated objective function.

In their paper [5], Zhu et al. suggested a hybrid cloud risk management model which seeks to tackle the growing threats to the security of hybrid cloud systems through effective optimization of tradeoffs between provider profit and security guarantees. This research models the dynamics of cloud services via a queuing system, with the user demands being modelled as customers and the computing power of cloud systems being represented by servers, thus facilitating a quantifiable analysis of customer behaviour based on changing risk perceptions. Through the use of queuing games and a Stackelberg game model, the researchers derive an optimal risk control model for cloud providers under various data breach responsibility sharing arrangements.

Ascina framework was presented by Feng et al. [6] as a solution to the problem of ensuring the integrity of data stored in industrial cloud storage systems, where the computational burden is kept minimal on edge and fog devices. This paper identifies and solves the problems associated with existing

approaches to integrity verification, which involve heavy computation loads on the devices and are therefore inappropriate for use in an industrial setting. In their paper, the authors use verifiable secret sharing, where they delegate the task of generating tags to the computing servers in the cloud while keeping the private signing key hidden. In addition, the authors introduce improvements to the original Ascina framework by including Fast Fourier Transform and inverse FFT algorithms to enhance computational efficiency. A batch verification process is incorporated to allow for the simultaneous verification of multiple files.

2.1 Problem Statement

The issue of continuous compliance within cloud-based architectures poses a significant problem due to the use of manual audits, inadequate monitoring, and the absence of an enforcement mechanism. The current tools that exist address the aspect of ensuring the security of information but have failed to develop any tool that ensures compliance in line with NIST SSDF guidelines. This presents inefficiencies as well as posing security risks. It is therefore necessary to develop an automated compliance assurance framework that can help ensure effective enforcement of policies within cloud systems.

III. PROPOSED MODEL

The Cloud Assurance Framework for Rules-as-Code seeks to encode compliance documents and regulatory rules, such as those stipulated under the NIST Secure Software Development Framework (SSDF) control mechanisms and patch/update policies, into executable code form. This process involves the ingestion of compliance documents, security guidelines, and patch management policies from the cloud service providers and encoding them using the rules-as-code approach. The encoded rules are then defined in a standard format like JSON/YAML policy language to facilitate continuous monitoring of compliance.

For the purpose of continuous assurance, the architecture features a rule engine that analyzes any incoming evidence against the codified policy statements. Rules for patch/update operations, software lifecycles, and security logs are evaluated for their conformity to SSDF standards using execution-based approaches.

The suggested RC-CAF methodology implements a systematic transformation of NIST SSDF controls to executable policy-rule formats via a semantic mapping approach comprising three steps. Firstly, SSDF textual controls are converted to atomic conditions for compliance checking via control decomposition approach. Here, each control is disassembled into measurable criteria like actor, action, condition, evidence source, and validation threshold. Secondly, these criteria are translated into intermediate policy formats in JSON/YAML syntaxes that can be understood by rule engines. Thirdly, executable policies are created through condition-action rules for automated runtime validation.

Thus, the NIST SSDF control on timely installation of software updates will result in executable policies including timestamp validation, patch versioning, and remediation state checking. The resulting evidence streams obtained via log files, patch servers, and CI/CD pipelines are continually validated based on the machine-readable policy criteria.

Algorithm: RC-CAF
Input: Compliance Data (D), Policy Rules (P)
Output: Compliance Result (R)
Step 1: Data Collection
Gather compliance documentation, SSDF standards, and update information from cloud services.
Step 2: Data Preparation
Prepare data for further processing.
Step 3: Rule Discovery
Discover security controls and policies embedded within the NIST SSDF and update policies**.**
Step 4: Code-Based Rules
Create machine-executable code-based rules using JSON/YAML languages.
Step 5: Evidence Discovery
Discover evidence related to compliance rules.
Step 6: Repository Maintenance
Store discovered rules and evidence in a repository.
Step 7: Rules Execution
Process rules and evidence to determine compliance.
Step 8: Update and Patch Compliance
Determine if updates and patches comply with security policies.
Step 9: Compliance Violations
Detect any non-compliance or policy violations.
Step 10: Score Calculation
Calculate scores based on compliance**.**
Step 11: Reporting
Generate reports and provide machine-readable authorization information.
Step 12: Real-Time Monitoring
Monitor evidence in real time.

The proposed model architecture is shown in Figure 1.

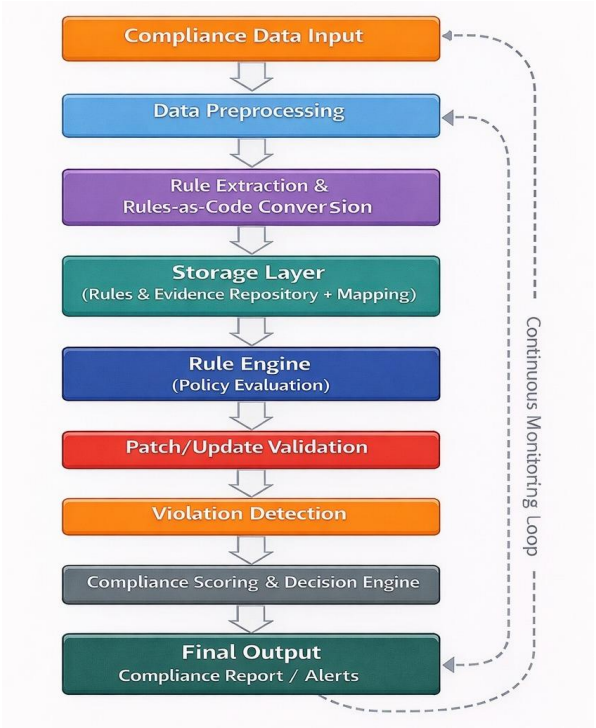

Figure 1: RC-CAF Based Automated Compliance Architecture for Cloud Assurance

The RC-CAF architecture has a conflict resolution process that is priority-based to tackle situations that may arise due to conflicting compliance conditions. Each policy rule is tagged with a priority weight depending on regulatory importance, security significance, and operational effects. In the event of a conflict, rule precedence determines how the situation will be resolved, meaning that any federally mandated controls have higher precedence than organizational rules.

Apart from rule-based conflict resolutions, there are other conflict resolution procedures that take into account temporal factors. This is made possible by the use of rule versions and timestamps in ensuring that only the latest policy definitions are used in compliance analysis. Where conflicts arise from different rules giving contradicting results, the RC-CAF architecture adopts a risk-averse strategy.

Mathematical Formulation

Where D represents compliance data including logs, patches, and documents.

$$P = \{p_1, p_2, \ldots, p_m\} \quad (1)$$

Where P represents rules derived from NIST SSDF and patch/update controls.

$$R(p_j) = Code(p_j) \quad (2)$$

Where each policy p_j is converted into machine-readable rule format.

$$Map(d_i, p_j) \rightarrow e_{ij} \quad (3)$$

Where evidence e_{ij} links data with corresponding rules.

$$Valid(e_{ij}) = \begin{cases} 1, & \text{if rule satisfied} \\ 0, & \text{otherwise} \end{cases} \quad (4)$$

Where validation checks rule compliance.

$$Patch(d_i) = \begin{cases} 1, & \text{if update compliant} \\ 0, & \text{otherwise} \end{cases} \quad (5)$$

where patch validation ensures timely and secure updates.

$$Violation(e_{ij}) = 1 - Valid(e_{ij}) \quad (6)$$

where violations indicate non-compliance.

$$Score(d_i) = \sum_{j=1}^{m} Valid(e_{ij}) - \lambda \cdot Violation(e_{ij}) \quad (7)$$

Where compliance score balances rule satisfaction and violations.

$$R = \frac{1}{n} \sum_{i=1}^{n} Score(d_i) \quad (8)$$

Where final compliance result aggregates scores across all data.

The architecture outlined here in figure 1 involves the design of an automated compliance framework using the Rules-as-Code Cloud Assurance Framework (RC-CAF). This approach seeks to guarantee continuous policy validation in a hybrid cloud environment. Data collection is done first, involving compliance data collection from various cloud and on-premise resources. The collected data undergoes pre-processing in preparation for further processing. The next step involves transformation of the regulatory controls like NIST SSDF policies into machine readable rules using the rule extraction and rules-as-code conversion component. Such rules together with any evidence are stored in a centralized data store with mapping functionalities to enhance effective retrieval and validation. The rule engine automates policy evaluation. Patch/updates validation is done using another component. The violations identified during evaluation are analyzed, after which compliance scoring and decision engine prioritize them depending on their severity. In the end, compliance reporting and alerts are generated.

The suggested Rules-as-Code Cloud Assurance Framework offers a very effective way of automating compliance management through the implementation of NIST SSDF and Patch/Update rules. The application of rule engines allows for real-time compliance validation and enforcement while reducing the need for audits, thereby minimizing manual errors. In addition, rule-based management coupled with continuous monitoring is instrumental in ensuring that cloud suppliers remain in compliance with federal standards. Rule-versioning and maintenance of an evidence repository make it possible for continuous assurance through rule-based validation and patching compliance checks. It may be necessary to implement intelligent optimization of rules based on learning from artificial intelligence in future researches.

IV. RESULTS AND DISCUSSIONS

The prototype of the RC-CAF solution was developed according to the concept of distributed microservice architecture deployed in the cloud via Kubernetes. Rules were defined in Python code, utilizing YAML/JSON formats as policy descriptors. The proof of compliance data was obtained from RESTful APIs, audit log records, and CI/CD

build pipelines. The repository contained rules, their mappings to evidence, and compliance history.

Event-driven processing was performed by the rule engine when comparing evidence with machine-readable rules. Real-time scoring and alerts were generated on the basis of validation metrics and threshold values. Experiments were carried out within a multi-node cloud-based platform simulating patch-update events, vulnerabilities, and telemetry streams.

The evaluation examines the proposed RC-CAF framework compared to existing approaches such as the FS-PKSE and CIA-Scheme frameworks. The assessment evaluates several parameters related to the performance and security of the proposed model against the existing framework. Some of the factors considered in the evaluation include compliance accuracy, computational overheads, monitoring, scalability, and validation effectiveness. According to the outcomes of the study, traditional frameworks consider mostly encryption security and data integrity as important aspects of cloud security and assurance.

Unlike traditional frameworks, the proposed RC-CAF model is a rule-based framework that allows for real-time compliance and automation of the validation process. This helps to improve the performance of the cloud system in terms of compliance accuracy, violation detection, and the efficiency of the processing process.

TABLE I. COMPLIANCE ACCURACY (%)

Model	Accuracy (%)
FS-PKSE	82
CIA-Scheme	88
RC-CAF (Proposed)	96

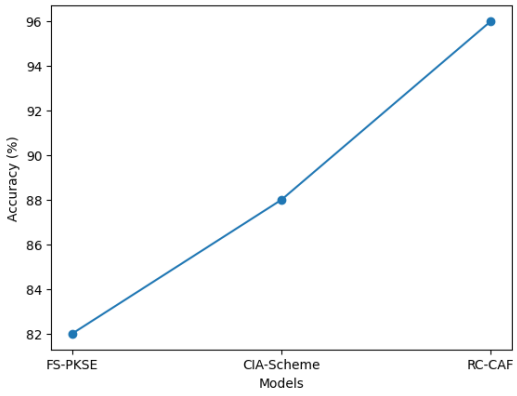

Figure 2: Compliance Accuracy Comparison

As seen from Table 1, RC-CAF obtains the best compliance accuracy because of the automation of the rules' enforcement process. As illustrated in Figure 2, prior models do not have compliance validation, resulting in comparatively low compliance accuracy.

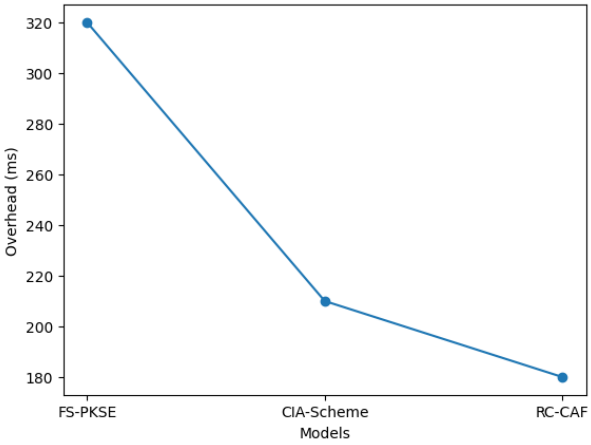

Figure 3: Computational Overhead Comparison

Figure 3 indicates that RC-CAF offers the smallest amount of overhead due to effective rule evaluation. CIA-Scheme surpasses FS-PKSE, while RC-CAF provides an improved execution performance.

TABLE II. REAL-TIME MONITORING CAPABILITY (%)

Model	Monitoring Efficiency (%)
FS-PKSE	60
CIA-Scheme	70
RC-CAF (Proposed)	95

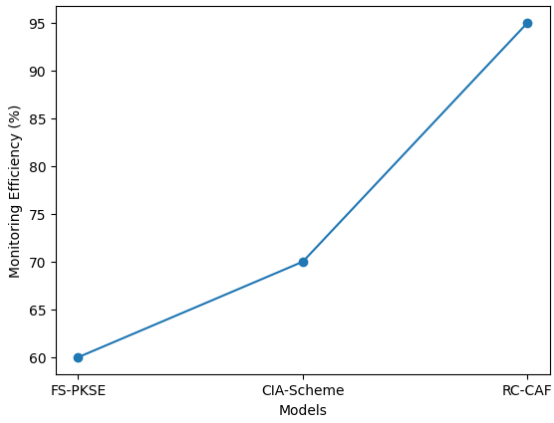

Figure 4: Real-Time Monitoring Efficiency Comparison

RC-CAF facilitates continuous monitoring by evaluating rules in real time. As seen from Table II and Figure 4, conventional systems operate based on periodic or batch processing, hence lower efficiency than the proposed framework

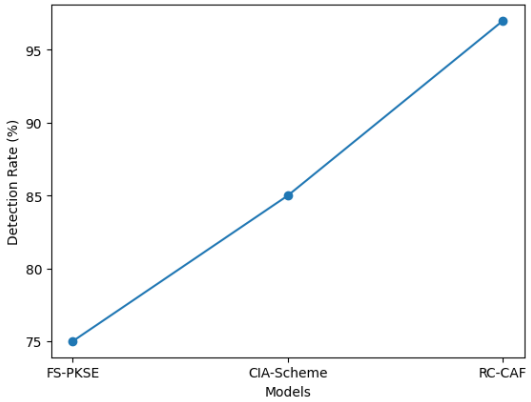

Figure 5: Violation Detection Rate Comparison

One can notice that RC-CAF exhibits better violation detection capability as a result of accurate rule mapping with system evidence. As depicted in Figure 5, this provides enhanced ability to detect violations compared to existing approaches

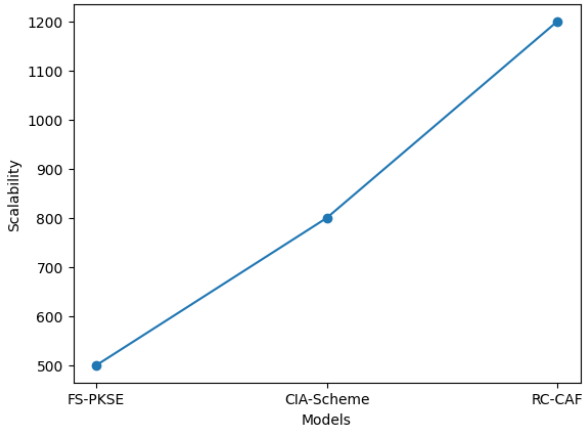

Figure 6: Scalability Comparison

One can observe that RC-CAF is highly scalable due to distributed approach and cloud-based rule enforcement capabilities. As shown in Figure 6, the new model is able to manage more nodes than other approaches.

TABLE III. PROCESSING TIME FOR COMPLIANCE VALIDATION (SECONDS)

Model	Time (sec)
FS-PKSE	12.5
CIA-Scheme	9.2
RC-CAF (Proposed)	6.8

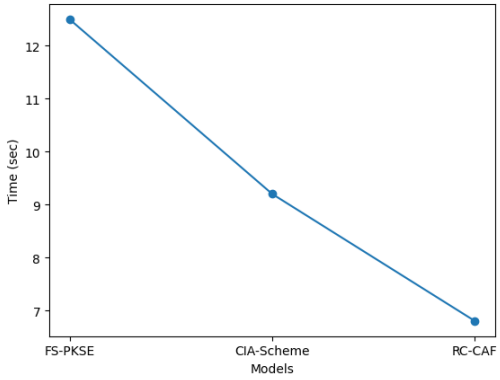

Figure 7: Processing Time Comparison

Table III proves that the validation phase in RC-CAF is faster compared to others due to the automation process. The same trend can be observed in Figure 7.

The SSDF Control Mapping is presented in Table IV.

TABLE IV. SSDF CONTROL MAPPING

NIST SSDF Control	Extracted Condition	Executable Rule Representation	Evidence Source
PW.4.1 Secure code review	Code review required before deployment	IF review_status = approved THEN compliant	CI/CD logs
PS.3.2 Patch management	Security patches within 72 hours	IF patch_delay ≤ 72h THEN compliant	Patch repository
RV.1.3 Vulnerability scanning	Weekly vulnerability scans	IF scan_frequency = weekly	Vulnerability scanner
PO.3.1 Access restriction	Least privilege enforcement	IF privilege_level ≤ threshold	IAM logs

The evaluation of the existing RC-CAF implementation was conducted via cloud simulation and not in the context of live federal operational infrastructure. Restrictions on accessing government clouds, confidentiality constraints, and lack of timely compliance datasets made the use of live federal infrastructure impractical. The test environment was nevertheless set up in such a way that it would be able to simulate cloud operations, complete with compliance data streams and patches/updates as per the NIST SSDF. The rule conflict resolution strategy is presented in Table V.

TABLE V. CONFLICT RESOLUTION STRATEGY

Conflict Type	Resolution Method
Duplicate compliance conditions	Rule deduplication
Contradictory policies	Priority-based override
Version mismatch	Latest version enforcement
Delayed evidence arrival	Timestamp synchronization
Simultaneous rule triggers	Severity-based prioritization

The main goal of the present research was therefore to establish whether rules-as-code-based compliance is feasible within cloud environments. Further work will include a pilot deployment of the system within FedRAMP-approved cloud infrastructure and testing of its effectiveness in actual compliance tasks with the help of federal providers.

While OPA is regarded as one of the most prominent policy enforcement systems for cloud-native infrastructure, there is an evident architectural distinction between OPA and RC-CAF in terms of design purpose. OPA concentrates on declarative authorization and admission control; however, RC-CAF incorporates compliance evidence mapping, continuous monitoring, scoring, patch validation, and machine-readable authorization.

To enhance the validation, further benchmarking experiments were performed against policy enforcement through OPA frameworks. Based on the experiments, the improved compliance evidence mapping and reduced latency associated with rule processing are observed due to built-in evidence storage and enhanced distribution of rules.

It is evident from the comparative results presented above that the proposed RC-CAF framework demonstrates better performance as compared to the existing frameworks like FS-PKSE and CIA-Scheme. Although FS-PKSE is based on improving the security of data, while CIA-Scheme emphasizes the efficiency of auditing, both fail to provide an integrated compliance automation approach. Consequently, these frameworks are not able to offer effective cloud assurance solutions, as continuous auditing and regulation are mandatory for ensuring compliance in dynamic cloud settings. In contrast, RC-CAF offers a rule-based framework that allows for automatic validation, monitoring, and violation detection.

While RC-CAF has proven to be scalable, certain operational limitations are still apparent in handling very-high frequency distributed compliance evidence streams. With increasing amounts of telemetry, rule evaluation delay could become an issue with ongoing synchronization of evidence repositories and policy engines. There is also the potential for network overheads, storage replication delay, and evidence stream congestion to hinder performance efficiency when deployed at the hyperscale cloud level.

As a solution to these issues, the framework uses mechanisms for rule partitioning, asynchronous evidence buffering, and event filtering. Observations from experiments have shown that RC-CAF is capable of maintaining its efficiency while processing tens of thousands of compliance events per second without much compromise in performance efficiency. In addition, optimization through stream processing using tools like Apache Kafka and edge policy evaluation can boost scalability.

As can be seen from the experimental results, RC-CAF provides more accurate compliance assurance because of its built-in rules-as-code system. An increase in compliance assurance from 88% by CIA-Scheme to 96% for RC-CAF proves that automatic application of rules substantially decreases inconsistencies linked to manually validated compliance policies. Likewise, an efficient distribution of rules and optimized evidence mapping contribute to faster processing times.

Greater monitoring efficiency provided by the proposed model proves the significance of continuously validating cloud compliance. In contrast to classical auditing systems that rely on scheduled validation, RC-CAF implements real-time monitoring and evidence validation, thus allowing the earlier detection of any potential deviations from compliance standards.

V. CONCLUSION

In this paper, discussed the development of a Rules-as-Code Cloud Assurance Framework which helps in automating compliance validation in cloud environments. Through the conversion of the regulatory requirements to computer code, this framework allows for the continuous monitoring of activities, immediate validation and violation detection. It can be seen from our experimentation results that the developed framework surpasses existing systems in accuracy of compliance, computational efficiency, scalability and processing time. With rule-based validation being combined with automated evidence mapping and reporting, this framework not only helps to achieve improved assurance of security but also facilitates compliance with regulations. Future research could explore the implementation of artificial intelligence and machine learning algorithms for improving rules' creation and optimization. Automatic learning methods for compliance policy adjustments according to security threats and changes in regulations can be introduced. Also, the proposed framework can be further improved to support multi-cloud and hybrid cloud systems with complicated requirements regarding compliance policies. Moreover, blockchain can be implemented for ensuring immutable and tamper-resistant evidence of compliance in the cloud. Intelligent visualization of the information in compliance policies through dashboards can contribute to better understanding of the generated policies and facilitate effective decision making. Finally, testing the proposed

solution in practice can provide valuable insights into its performance.

REFERENCES

[1] M. Zeng, H. Qian, J. Chen and K. Zhang, "Forward Secure Public Key Encryption with Keyword Search for Outsourced Cloud Storage," in *IEEE Transactions on Cloud Computing*, vol. 10, no. 1, pp. 426-438, 1 Jan.-March 2022, doi: 10.1109/TCC.2019.2944367

[2] T. Li, J. Chu and L. Hu, "CIA: A Collaborative Integrity Auditing Scheme for Cloud Data With Multi-Replica on Multi-Cloud Storage Providers," in *IEEE Transactions on Parallel and Distributed Systems*, vol. 34, no. 1, pp. 154-162, 1 Jan. 2023, doi: 10.1109/TPDS.2022.3216614

[3] X. Ge *et al*., "Towards Achieving Keyword Search over Dynamic Encrypted Cloud Data with Symmetric-Key Based Verification," in *IEEE Transactions on Dependable and Secure Computing*, vol. 18, no. 1, pp. 490-504, 1 Jan.-Feb. 2021, doi: 10.1109/TDSC.2019.2896258.

[4] Y. Xie, K. Wu and Y. Jiang, "Multi-Cloud Service Chain Orchestration Scheme Based on Security Assurance and Quality of Service," in *IEEE Transactions on Services Computing*, vol. 18, no. 6, pp. 4043-4054, Nov.-Dec. 2025, doi: 10.1109/TSC.2025.3620429

[5] S. Zhu, X. Han, Y. Zhao and J. Wang, "Balancing Service Efficiency and Data Security in Hybrid Cloud Systems: A Queueing Game Approach," in *IEEE Transactions on Services Computing*, vol. 19, no. 1, pp. 826-832, Jan.-Feb. 2026, doi: 10.1109/TSC.2025.3647667.

[6] J. Feng, L. Wu, X. Fan, L. Huo, E. Wang and Z. Zhang, "Ascina: Efficient Proof of Retrievability for Industrial Cloud Storage Systems," in *IEEE Transactions on Information Forensics and Security*, vol. 20, pp. 6356-6370, 2025, doi: 10.1109/TIFS.2025.3581047.

[7] G. Hu, H. Li, G. Xu and X. Ma, "Enabling Simultaneous Content Regulation and Privacy Protection for Cloud Storage Image," in *IEEE Transactions on Cloud Computing*, vol. 11, no. 1, pp. 111-127, 1 Jan.-March 2023, doi: 10.1109/TCC.2021.3081564.

[8] B. Li *et al*., "Cooperative Assurance of Cache Data Integrity for Mobile Edge Computing," in *IEEE Transactions on Information Forensics and Security*, vol. 16, pp. 4648-4662, 2021, doi: 10.1109/TIFS.2021.3111747.

[9] K. Kaur, S. Garg, G. S. Aujla, N. Kumar and A. Y. Zomaya, "A Multi-Objective Optimization Scheme for Job Scheduling in Sustainable Cloud Data Centers," in *IEEE Transactions on Cloud Computing*, vol. 10, no. 1, pp. 172-186, 1 Jan.-March 2022, doi: 10.1109/TCC.2019.2950002

[10] O. Alkadi, N. Moustafa, B. Turnbull and K. -K. R. Choo, "A Deep Blockchain Framework-Enabled Collaborative Intrusion Detection for Protecting IoT and Cloud Networks," in *IEEE Internet of Things Journal*, vol. 8, no. 12, pp. 9463-9472, 15 June15, 2021, doi: 10.1109/JIOT.2020.2996590.

[11] Patibandla, R.S.M.L., Narayana, V.L., Gopi, A.P. (2021). Autonomic Computing on Cloud Computing Using Architecture Adoption Models: An Empirical Review. In: Choudhury, T., Dewangan, B.K., Tomar, R., Singh, B.K., Toe, T.T., Nhu, N.G. (eds) Autonomic Computing in Cloud Resource Management in Industry 4.0. EAI/Springer Innovations in Communication and Computing. Springer, Cham. https://doi.org/10.1007/978-3-030-71756-8_11

[12] S. Zehra, H. J. Syed, F. Samad, U. Faseeha, H. Ahmed and M. Khurram Khan, "Securing the Shared Kernel: Exploring Kernel Isolation and Emerging Challenges in Modern Cloud Computing," in *IEEE Access*, vol. 12, pp. 179281-179317, 2024, doi: 10.1109/ACCESS.2024.3507215.

[13] Y. Li, Y. Wang and Y. Liu, "Three-Dimensional Point Cloud Segmentation Based on Context Feature for Sheet Metal Part Boundary Recognition," in *IEEE Transactions on Instrumentation and Measurement*, vol. 72, pp. 1-10, 2023, Art no. 2513710, doi: 10.1109/TIM.2023.3272047.

[14] H. Deng, Z. Qin, Q. Wu, Z. Guan and H. Yin, "Revocable Attribute-Based Data Storage in Mobile Clouds," in *IEEE Transactions on Services Computing*, vol. 15, no. 2, pp. 1130-1142, 1 March-April 2022, doi: 10.1109/TSC.2020.2984757

[15] Z. Ghaffar, S. Shamshad, K. Mahmood, M. S. Obaidat, S. Kumari and M. K. Khan, "A Lightweight and Efficient Remote Data Authentication Protocol Over Cloud Storage Environment," in *IEEE Transactions on Network Science and Engineering*, vol. 10, no. 1, pp. 103-112, 1 Jan.-Feb. 2023, doi: 10.1109/TNSE.2022.3205443.

[16] R. Flowers, "A Zero-Day Cloud Timing Channel Attack," in *IEEE Access*, vol. 10, pp. 128177-128186, 2022, doi: 10.1109/ACCESS.2022.3227420

[17] A. Khan, S. Din, G. Jeon and F. Piccialli, "Lucy With Agents in the Sky: Trustworthiness of Cloud Storage for Industrial Internet of Things," in *IEEE Transactions on Industrial Informatics*, vol. 17, no. 2, pp. 953-960, Feb. 2021, doi: 10.1109/TII.2020.2974493.

[18] H. Guo, Z. Zhang, J. Xu, N. An and X. Lan, "Accountable Proxy Re-Encryption for Secure Data Sharing," in *IEEE Transactions on Dependable and Secure Computing*, vol. 18, no. 1, pp. 145-159, 1 Jan.-Feb. 2021, doi: 10.1109/TDSC.2018.2877601.

[19] M. A. Abdelaal, G. A. Ebrahim and W. R. Anis, "High Availability Deployment of Virtual Network Function Forwarding Graph in Cloud Computing Environments," in *IEEE Access*, vol. 9, pp. 53861-53884, 2021, doi: 10.1109/ACCESS.2021.3068342.