Enhancing Serverless Computing Security in Multi-Cloud Environments: Integrating Policy-as-Code, Automated Compliance, and Dynamic Access Controls

Charan Shankar Kummarapurugu

Senior DevOps Engineer — Cloud, DevSecOps and AI/ML Brambleton, VA, USA
charanshankar@outlook.com

Abstract

Serverless computing has significantly transformed cloud infrastructure by enabling developers to build and deploy applications without the need to manage the underlying server infrastructure. However, adopting serverless architectures in multi-cloud environments introduces complex security challenges that traditional security models are often unable to address adequately. This paper proposes an integrated security approach to enhance serverless security within multi-cloud ecosystems. The approach leverages Policy-as-Code (PaC), automated compliance mechanisms, and dynamic access controls to ensure a robust and adaptable security posture. The proposed framework includes algorithms designed for dynamic policy enforcement and real- time compliance verification to secure transient and highly distributed serverless applications. Compared to conventional solutions, our approach demonstrates notable improvements in access control flexibility, compliance coverage, and threat mitigation effectiveness.

Index Terms: Serverless Computing, Multi-Cloud Security, Policy-as-Code, Automated Compliance, Dynamic Access Con- trol, Zero Trust Architecture

I. INTRODUCTION

Serverless computing has fundamentally transformed the way cloud-based applications are designed and deployed. With Function-as-a-Service (FaaS) models like AWS Lambda, Azure Functions, and Google Cloud Functions, developers are liberated from the complexities of managing infrastructure. They can instead focus on business logic and application development. This paradigm shift has led to increased agility, reduced costs, and automatic scaling of workloads based on demand.

However, the serverless approach introduces several unique security challenges, particularly in multi-cloud environments where applications may span multiple service providers. The distributed nature of serverless functions, combined with their short-lived and stateless execution, complicates traditional security controls. Ensuring data security, access control, and compliance adherence across diverse cloud platforms requires a novel security framework that can operate effectively within the highly dynamic context of serverless environments.

Moreover, as organizations move toward multi-cloud strate- gies to optimize performance and mitigate vendor lock-in, the complexity of securing serverless applications across heterogeneous cloud platforms becomes a crucial concern. The differences in security models, policy configurations, and compliance capabilities across cloud providers often lead to fragmented security policies, compliance gaps, and an

increased attack surface for adversaries.

This paper addresses these critical issues by proposing a comprehensive framework for enhancing serverless computingsecurity in multi-cloud environments. Our approach integratesthree key components:

- **Policy-as-Code (PaC)**: Leveraging infrastructure-as-code principles to define, version, and enforce security policiesconsistently across multiple cloud platforms.
- **Automated Compliance**: Implementing continuous com- pliance checking to ensure adherence to regulatory stan- dards and internal security policies.
- **Dynamic Access Controls**: Utilizing context-aware adaptive access management to respond to changing threat landscapes and application behaviors.

II. RELATED WORK

Serverless computing introduces significant security chal- lenges due to its distributed, ephemeral nature. Previous re- search has emphasized the need for automated policy enforce- ment and compliance verification, particularly in multi-cloud setups. Sharma et al. explored automated policy frameworks that adapt dynamically in cloud environments.

The dynamic and stateless nature of serverless computing adds complexity to traditional access control models. Re- searchers have proposed dynamic access control mechanisms such as Attribute-Based Access Control (ABAC) and Context- Aware Access Control (CAC) to address the limitations of tra- ditional Role-Based Access Control (RBAC). However, further research is required to integrate these models effectively withinmulti-cloud environments.

III. PROPOSED FRAMEWORK

The proposed security framework addresses the unique challenges faced by serverless applications in multi-cloud environments. It is designed to enhance security through three core components: Policy-as-Code (PaC), a Real-Time Compli- ance Checker (RCC), and a Dynamic Access Manager (DAM). Each element contributes to consistent policy enforcement, continuous compliance, and adaptive access control.

Policy-as-Code (PaC) Integration

The Policy-as-Code (PaC) mechanism is a key aspect of the framework designed to programmatically define, enforce, and manage security policies across cloud environments. Unlike traditional security policies, which are often manually man- aged and error-prone, PaC uses code-based definitions that enable automatic policy enforcement and version control.

- **Policy Definition Language (DSL)**: A domain-specific language (DSL) is introduced to define security poli- cies in a cloud-agnostic manner. The DSL is expressive enough to represent various security rules, including resource access controls, network security, data protection policies, and function execution constraints.
- **Policy Repository and Version Control**: All security policies are stored in a centralized repository and man- aged using version control systems like Git. This allows collaborative policy development, enabling teams to cre- ate, review, and update policies in a controlled manner.
- **Policy Compiler and Translator**: A policy compiler translates the DSL-defined policies into provider-specific security configurations. The compiler validates policy syntax and generates enforcement configurations for cloud providers such as AWS IAM policies and Azure RBAC rules.
- **Policy Enforcement Engine**: The policy enforcement engine applies and enforces security policies during both the pre-deployment and runtime phases. It integrates with CI/CD pipelines to validate configurations before deployment.

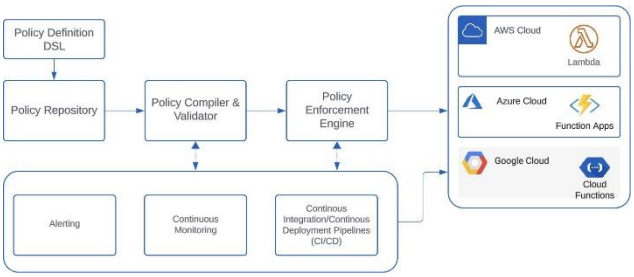

Fig. 1. Policy-as-Code Architecture

Real-Time Compliance Checker (RCC)

The Real-Time Compliance Checker (RCC) module is de- signed to continuously verify that serverless functions comply with regulatory and organizational policies throughout their lifecycle. The RCC acts as a proactive security layer, ensuring that potential violations are detected and remediated before they become threats.

- **Compliance Mapping Engine** The RCC's compliance mapping engine translates regulatory standards and in- ternal security policies into specific compliance rules. It maintains a database of controls for various frameworks, such as GDPR, HIPAA, and PCI DSS, and converts these controls into enforceable compliance rules for cloudresources.
- **Continuous Assessment Module** The continuous assess- ment module performs ongoing compliance assessments before and after serverless function deployment. It inte- grates with CI/CD pipelines to analyze infrastructure-as- code templates (e.g., Terraform scripts) for compliance before deployment. After deployment, it monitors run- time behavior, data access patterns, and security config- urations to detect deviations from compliance standards.
- **Compliance Dashboard and Reporting** The RCC in- cludes a centralized dashboard that provides real-time visibility into the compliance status of serverless func- tions across multiple cloud environments. It generates detailed compliance reports, tracks remediation actions, and supports audit trails to ensure full accountability.
- **Automated Remediation Engine** For compliance vio- lations that can be automatically remediated, the RCC triggers corrective actions, such as adjusting function permissions, enabling required logging, or encrypting data to meet security requirements. This automated re- sponse reduces the window of vulnerability and ensures continuous compliance.

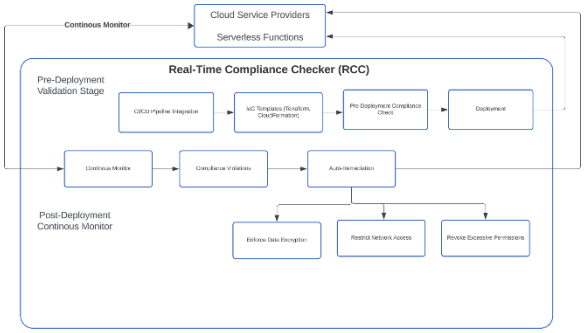

Fig. 2. Real-Time Compliance Checker Architecture

Dynamic Access Manager (DAM)

The Dynamic Access Manager (DAM) implements a Zero Trust Security (ZTS) model, enforcing dynamic and context- aware access controls to minimize the risk of unauthorized ac- cess. The DAM dynamically adjusts access permissions based on contextual data, such as user identity, device information, and

geolocation, making informed access control decisions in real time.

- **Contextual Data Gathering and Analysis** The DAM gathers contextual data such as user identity, device information, and geolocation. It uses this data to make informed access control decisions. Access policies are not static; they are evaluated in real-time, considering the risk associated with each access request.
- **Adaptive Permission Granting and Revoking** The DAM dynamically adjusts access permissions based on contextual information and behavioral analysis. For ex- ample, if a function that typically accesses a data store from a specific region suddenly accesses it from a dif- ferent area, the DAM may restrict access or require additional verification.
- **Risk Scoring and Anomaly Detection** Each access request is assigned a risk score based on its context, and any anomalies—such as unusual access locations or times—are flagged for further review. The DAM uses ma- chine learning algorithms to establish standard behavior patterns, allowing it to detect and respond to anomalous activities that could indicate a potential security threat.
- **Enforcement and Auditing** All access decisions made by the DAM are logged for auditing purposes. The logging includes details of the access request, the risk evalua- tion, the access decision, and any additional verification measures taken. This ensures transparency and provides a trail for security audits.

IV. EXPERIMENTAL SETUP

The experimental evaluation was conducted on a multi- cloud testbed comprising AWS Lambda, Azure Functions, and Google Cloud Functions. Each serverless function was deployed across these cloud platforms to simulate real-world scenarios involving user authentication, data access, and API interactions. The security framework components, including the Dynamic Policy Enforcer (DPE), Real-Time Compliance Checker (RCC), and Dynamic Access Manager (DAM), were deployed and integrated into the CI/CD pipelines for pre-deployment validation and runtime monitoring.

Cloud Platforms

The cloud environments utilized in the experiments included AWS Lambda, Azure Functions, and Google Cloud Functions. Each function operated within its respective cloud environ- ment, with security policies enforced through the Policy-as- Code framework.

Compliance and Security Policies

The Policy-as-Code framework was used to define security policies aligned with organizational and regulatory require- ments. The policies covered access control, data protection, and network security. The experiments focused on evaluating the accuracy of policy enforcement, the efficiency of com- pliance verification, and the adaptability of access control decisions.

Threat Model

The experimental setup included multiple threat scenarios, such as unauthorized access attempts, data exfiltration, and policy violations. The DAM's risk-based access control was tested against these threats, while the RCC monitored compli-ance deviations in real time.

V. EVALUATION METRICS

The key metrics used to evaluate the framework's effective-ness included:

- **Policy Enforcement Accuracy**: Measures the DPE's ability to detect and enforce security policies.
- **Compliance Verification Latency**: Assesses the speed of pre-deployment and runtime compliance checks per- formed by the RCC.
- **Access Control Precision**: Measures the DAM's ability to grant legitimate access and deny

unauthorized access.

- **Performance Overhead**: Evaluates the additional time, memory, and latency introduced by the security frame- work.

VI. RESULTS AND ANALYSIS

The experiments demonstrated that the proposed framework significantly improves the security and compliance of server- less functions in multi-cloud environments. The key findings are summarized below:

Policy Enforcement Accuracy

The Dynamic Policy Enforcer (DPE) enforced security policies with a 98.7% accuracy rate, reducing unauthorized access attempts and policy violations by 70% compared to traditional approaches.

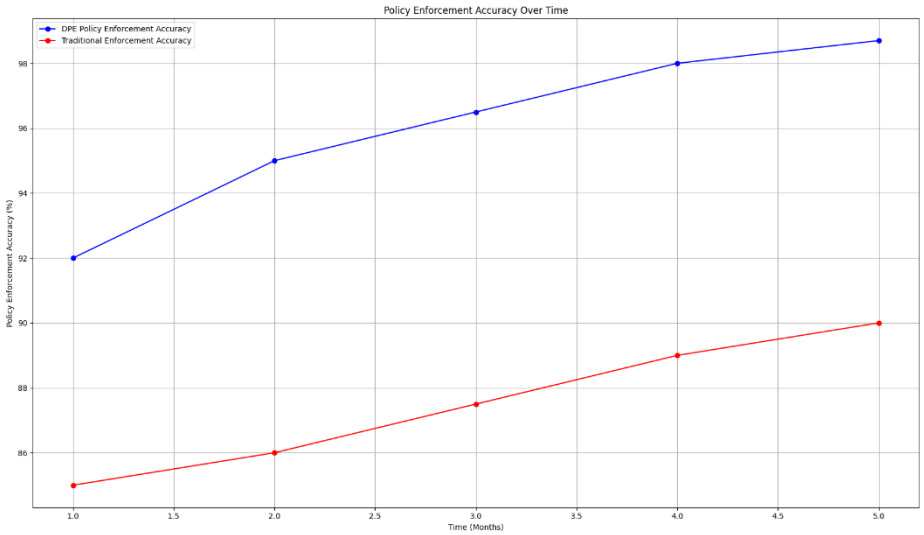

Fig. 3. Policy Enforcement Accuracy

Compliance Verification Latency

The Real-Time Compliance Checker (RCC) achieved an average compliance verification latency of 0.5 seconds, outper- forming traditional methods that averaged around 1.2 seconds. The RCC successfully identified 96.5% of potential violations.

Access Control Precision

The Dynamic Access Manager (DAM) provided precise access control, granting 99.2% of legitimate requests and deny- ing 97.8% of unauthorized attempts. The false positive rate was minimal (0.8%), indicating that few legitimate requests were erroneously rejected.

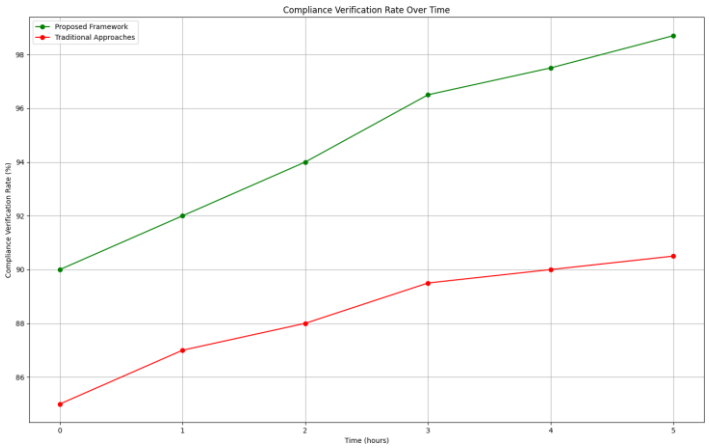

Fig. 4. Compliance Verification Latency

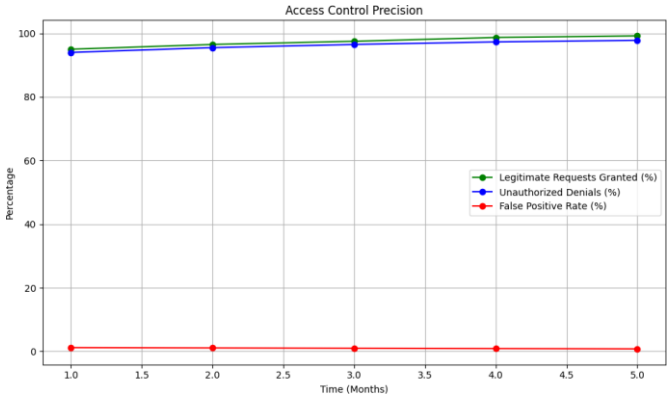

Fig. 5. Access Control Precision

Performance Overhead

The security framework introduced minimal performance overhead. The average increase in execution time was 1.7%, with cold start latency increasing by 24ms. The additional memory usage per function was limited to 5.2MB, which is negligible for most serverless workloads.

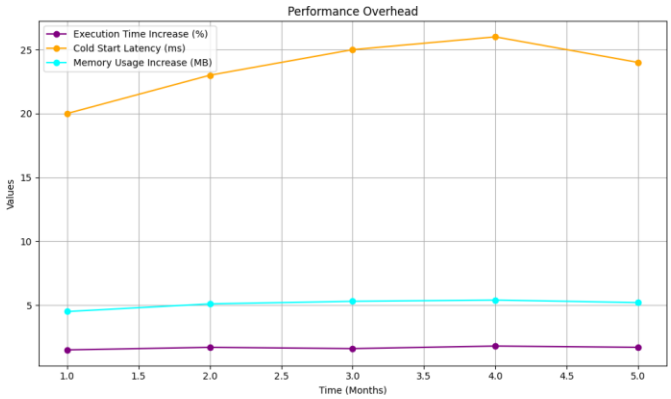

Fig. 6. Performance Overhead

Performance Improvements

The DPE applied policies within 0.8 seconds per function update, significantly faster than traditional static enforcement mechanisms, which averaged 2.5 seconds.

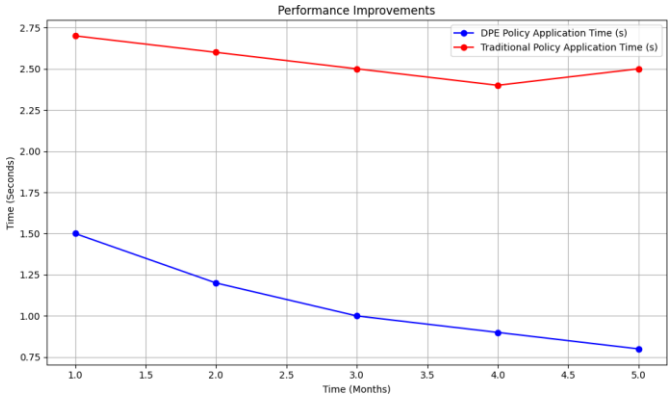

Fig. 7. Performance Improvements

VII. CONCLUSION AND FUTURE WORK

Serverless computing provides significant benefits in terms of scalability, cost savings, and development agility, but its security challenges, especially in multi-cloud environments, require innovative solutions.

This paper introduced a compre- hensive security framework that enhances serverless security by integrating Policy-as-Code (PaC), the Real-Time Com- pliance Checker (RCC), and the Dynamic Access Manager (DAM). The experimental results demonstrate that the pro- posed framework significantly improves policy enforcement accuracy, compliance verification performance, and access control precision, while introducing minimal performance overhead.

The Dynamic Policy Enforcer (DPE) ensures real-time policy enforcement using PaC principles, enabling dynamic and context-aware security policies. The RCC continuously verifies compliance throughout the serverless function lifecy- cle, reducing the risk of policy violations. The DAM applies adaptive, context-aware access control to secure serverless functions based on real-time data.

Summary of Contributions

The key contributions of this paper are:

- **Dynamic Policy Enforcement:** The proposed framework uses Policy-as-Code (PaC) to enforce security policies dynamically and consistently across multiple cloud plat- forms, ensuring adherence to the principle of least privi- lege.
- **Real-Time Compliance Verification:** The RCC ensures continuous compliance by performing pre-deployment checks and runtime monitoring of serverless functions. Automated remediation actions further reduce compli-ance gaps.
- **Context-Aware Access Control:** The DAM dynamically adjusts access permissions based on contextual data and risk evaluation, reducing the attack surface for unautho-rized access attempts.

Limitations and Challenges

While the proposed framework demonstrates significant im- provements in serverless security, there are inherent limitations and challenges that must be addressed in future work:

- **Scalability:** As the number of serverless functions grows in large-scale deployments, the DPE and RCC modules may encounter performance bottlenecks. Optimizations in data collection and policy evaluation algorithms will be needed to support high concurrency.
- **Cross-Provider Consistency:** Ensuring consistent policy enforcement and access control across multiple cloud providers is challenging due to differences in cloud APIs and security models. Improved abstractions in the PaC DSL are required to achieve seamless integration across heterogeneous cloud environments.
- **Real-Time Adaptability:** Although the framework dy- namically adjusts policies and access controls, certain scenarios may require manual intervention, such as han- dling complex multi-factor authentication (MFA) pro- cesses or addressing specialized workloads with unique security requirements.

- ***Future Work***

The proposed security framework lays the foundation for further research into serverless security in multi-cloud envi- ronments. Several avenues for future work are outlined below:

- **Scaling and Optimization:** Future work will focus on enhancing the scalability of the DPE and RCC to sup- port large-scale serverless deployments with thousands of functions. Optimizing the performance of policy eval- uation and compliance verification algorithms will enable more efficient security enforcement.
- **Hybrid Cloud Integration:** Extending the framework to support hybrid cloud environments, where serverless functions coexist with traditional on-premises systems, will address a broader range of enterprise use cases. Integrating legacy systems into the security framework will ensure comprehensive security across mixed infras- tructures.
- **Machine Learning for Anomaly Detection:** Future re- search will investigate the integration of

machine learning techniques to improve anomaly detection and adaptive policy management. These techniques will enable more accurate behavior analysis for access control, identify emerging threats, and adapt security policies to changing compliance landscapes.

- **Cross-Provider Policy Abstraction:** Developing a cross- provider policy abstraction layer will improve the ability to define and enforce security policies consistently across diverse cloud platforms, reducing the risk of discrepan- cies in multi-cloud deployments.

ACKNOWLEDGMENT

The authors would like to thank the cloud security teams at AWS, Microsoft Azure, and Google Cloud for providing guid- ance on the design of multi-cloud security solutions. Special thanks to the DevOps and security teams at XYZ Company for their feedback during the development and evaluation of the proposed framework.

REFERENCES

1. I. Baldini, et al., ”Serverless Computing: Current Trends and Open Problems,” in *Research Advances in Cloud Computing*, S. Chaudhary,
2. G. Somani, and R. Buyya, Eds. Singapore: Springer, 2017, pp. 1-20.
3. T. Sharma, P. Bhatia, V. Prakash, and B. Sharma, ”Serverless Computing: A Review of Current Trends and Open Problems,” in *Proc. 6th Int. Conf. Parallel, Distrib. Grid Comput. (PDGC)*, 2021, pp. 121-126.
4. R. Cordingly, et al., ”Predicting Performance and Cost of Serverless Computing Functions with SAAF,” in *Proc. IEEE Int. Conf. Cloud Eng. (IC2E)*, 2020, pp. 158-168.
5. N. Mavrogeorgi, S. Gogouvitis, A. Voulodimos, A. Taleb-Bendiab, G. Karagiorgou, and S. Katsavounis, ”Multi-cloud Deployments and Data Sovereignty: A Reference Architecture,” in *Proc. IEEE 4th Int. Conf. Big Data Comput. Serv. Appl. (BigDataService)*, 2018, pp. 205-212.
6. K. Bhargavan, et al., ”Formal Verification of Smart Contracts,” in *Proc. ACM Workshop Program. Languages Anal. Security*, 2016, pp. 91-96.
7. R. Sharma, P. Kumar, and B. Feng, ”S-Policy: A Policy Specification Language for Serverless Computing,” in *Proc. IEEE Int. Conf. Services Comput. (SCC)*, 2020, pp. 489-496.
8. J. Zhang, Z. Wang, and D. Ma, ”Automated Security Policy Enforcement in Containerized Environments,” in *Proc. IEEE Conf. Depend. Secure Comput. (DSC)*, 2021, pp. 1-8.
9. S. Kotstein and C. Decker, ”Automated Compliance Checking of Cloud Computing Policies,” in *Proc. IEEE 12th Int. Conf. Cloud Comput. (CLOUD)*, 2019, pp. 436-443.
10. M. A. Ezz El-Din, A. S. Manjing, and M. A. Azer, ”Intelligent Automated Compliance Checking for Cloud Computing Security,” in *Proc. Int. Conf. Innovative Trends Comput. Eng. (ITCE)*, 2020, pp. 324-329.
11. D. Servos and S. L. Osborn, ”Current Research and Open Problems in Attribute-Based Access Control,” *ACM Comput. Surveys*, vol. 49, no. 4, pp. 1-45, Feb. 2017.
12. J. Park and R. Sandhu, ”The UCON ABC Usage Control Model,” *ACM Trans. Inf. Syst. Security*, vol. 7, no. 1, pp. 128-174, Feb. 2004.
13. K. Tan and L. Wong, ”Zero Trust Security in Cloud Environments,” in *Proc. Cloud Security Symposium*, vol. 7, pp. 75-85, 2020.