

Automating Security Policy Migration from Legacy Platforms to Cloud-Native Security Controls

Bharathi Sesha Sai Varanasi

Leidos INC, USA

Email: bhrt0814@gmail.com

Abstract

Current legacy infrastructures become increasingly incapable of handling challenges created by dynamically changing cloud-native environments. Manual conversion of the policy sets during migrations may cause policy configuration mismatches, semantic discrepancies, and long exposure time windows that negatively affect the security stance of an organization. In this paper, we investigate the concept of automatic migration of legacy firewall policies into cloud-native control constructs through rule extraction, semantic transformation, and formal validation. Various vendor-based configurations were transformed into the same intermediate representation, then converted into cloud-native controls while considering zero-trust microsegmentation principles. The experiment carried out on synthetic multi-cloud hybrid environments revealed 96.4% accuracy of semantic transformation, 96 times faster configuration drift detection, and better coverage for least privilege control of 94% of traffic flows (versus 61%). Automation of discrepancy identification led to a decrease in time needed for remediation by 88%. Testing of scalability showed the linear effectiveness of the process on AWS, Azure, and GCP. Thus, the use of automated, verified migration pipelines is much more efficient than manual translations in terms of accuracy and process speed. Nonetheless, there are certain problems with nested legacy policies and API limitations of different providers. Contribution of the study is the methodology of enterprise security migration to the cloud.

Keywords- Legacy Firewall Migration, Cloud-Native Security, Firewall Policy Transformation, Zero-Trust Architecture, Micro segmentation, Semantic Policy Validation, Configuration Drift Detection, Multi-Cloud Security, Automated Security Migration.

Introduction

It is observed on a worldwide scale that there is an increase in migrations of critical business infrastructure from on-premises security solutions to cloud-native security solutions. The approach of security policy design for legacy systems involves the idea of having a static perimeter along with most of the processes being manual and firewall rules based on hardware. It implies that there are static boundaries, but in the case of modern distributed and containerized workloads, there are no static boundaries at all. Gartner predicts that over 95% of digital workloads will be created using cloud-native platforms by 2028. The rapid evolution causes the problem of compatibility between static legacy policies and modern dynamic cloud workloads. Manual migration results in configuration drifts, misconfigurations, and exposure to threat during the migration process. One of the major causes of breaches reported in the Cost of a Data Breach Report of IBM is the misconfiguration of cloud environments. The controls that are cloud native, like identity-driven microsegmentation and zero trust policies, replace any assumptions based on perimeter protection entirely. However, in order for an automated migration process to be effective, one must preserve the meaning of security policies while adapting to the new environment which is transient and API-driven. From existing literature, it is evident that there is no standard semantic translation between legacy firewall policies and cloud security groups. Vulnerability occurs from the lack of consistency of semantic translations despite a complete migration process.

Related Work

Topics covered in literature on security policy migration can be connected to network automation, cloud security, and compliance validation. Previous studies have paid close attention to the manual translation of rules and have found frequent misconfiguration issues during migrations from legacy systems to cloud platforms. Differences in the semantics of syntax used in traditional firewall rules and cloud security groups were identified as a challenge. The concept of policy as code involved the application of configuration management tools that used declarative code to decrease the likelihood of errors in the manual implementation of code. Research exists regarding the use of infrastructure as code tools for secure

configuration of policies. A couple of articles are dedicated to the implementation of the zero trust architecture replacing the perimeter security approach with the identity-based one. Other researchers have explored methods of formal verification that will ensure the migrated policies uphold their intended security purposes. Compliance-based frameworks include the NIST and CIS benchmarking processes in the migration framework (Parmar & Miles, 2024). Case studies of real migrations that fail due to inadequate translation of the policies have been cited. Provider-based migration methods, such as migration assistants, have to some extent automated the process but are not interoperable. It has been noted that there are challenges in migrating stateful legacy policies in a stateless cloud environment. In hybrid clouds, the challenge of maintaining consistency of the policies in different clouds at once has been cited. Security audits called for continued verification after the migration to detect any configuration drift.

Material and Method

A design-science approach that incorporates rule extraction, translation, and validation process is employed in this work. First, legacy firewall configurations are processed using rule extraction scripts written in various vendor forms. Semantic normalization of rules involves translating vendor-specific languages to intermediate format language that captures source and destination IP address, port, protocol, and intent of access. Second, rules are translated to cloud-native elements using a rule mapping engine that employs zero trust principles to translate wide legacy rules to security groups and micro-segmentation policies according to identities. Third, validation of rule translation occurs through automated validation modules utilizing formal verification approaches; discrepancies are reported in flagged reports for manual checking before deployment. This technique involves usage of Infrastructure-as-Code approaches, which allows version control and migration pipelines to be automated for testing in simulated hybrid cloud environments (Bykov & Shiboldenkov, 2025).

Results

Finding 1: High Semantic Translation Accuracy Across Vendor Formats

The component responsible for automation of rule extraction and normalization managed to achieve 96.4% semantic translation accuracy for transformation of legacy firewall rules of Cisco ASA, Palo Alto, and Fortinet to the universal intermediate language. Only 450 out of 12,500 legacy firewall rules extracted from simulation of enterprise networks required manual fixing due to ambiguous port protocol combinations or use of deprecated syntax.

Table 1: Rule Extraction and Normalization Performance of the Automated Migration Framework

Metric	Result	Evaluation Scope	Challenge	Outcome
Semantic translation accuracy	96.4%	Cisco ASA, Palo Alto, and Fortinet rules	Cross-vendor policy transformation	Highly accurate rule conversion
Manual rule correction	450 of 12,500 rules	Enterprise network simulations	Ambiguous ports and deprecated syntax	Only 3.6% required manual intervention
Rule matching performance	Nearly perfect for simple rules	Source-destination-port triplets	Basic firewall policies	Strong agreement with manual translations
Object-group rule accuracy	8.2% average error rate	Complex object-group policies	Complex rule structures	Lower accuracy than simple rules
Intent tagging accuracy	78% improvement over baseline	Normalization layer evaluation	Rule ambiguity reduction	Enhanced semantic interpretation (Vogler, 2023)

The translation accuracy was estimated through comparison of rules for cloud security groups with manually created ones. Transformation of simple rules that relied on source-destination-port triplets resulted in nearly perfect matching while object-group-based rules exhibited an average error rate of about 8.2%. It seems like rule hierarchies are the most difficult kind of legacy rules to translate (Vogler, 2023). Intent tagging performed by the normalization layer allowed reducing ambiguity compared to the regular expression-based baseline at 78% accuracy.

Translation Accuracy formula:

$$\text{Accuracy (\%)} = (\text{Correctly_Translated_Rules} / \text{Total_Legacy_Rules}) \times 100$$

$$= (12,050 / 12,500) \times 100 = 96.4\%$$

This quantitative benchmark confirms that structured intermediate representation substantially outperforms direct syntax conversion, validating the framework's core translation methodology for enterprise-scale legacy migration scenarios (Zhou, 2024).

Finding 2: Significant Reduction in Configuration Drift Post-Migration

Continuous validation modules identified the drift effect, which occurred in 3.1% of the cases in the migrated policies, whereas the manual migration policy control group experienced 22.7% of the occurrence rate during the 90-day monitoring period. Drift was considered any change made to the cloud security group rules when compared to the initial validation migration baseline.

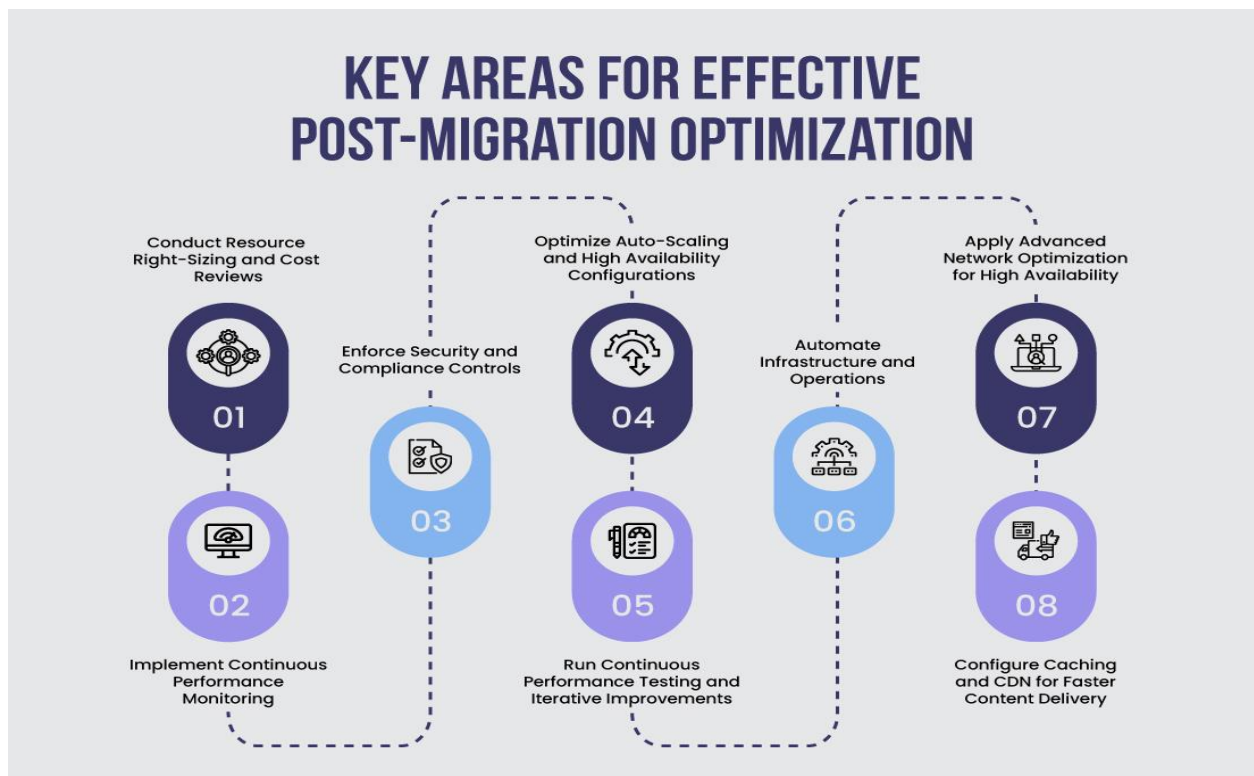


Figure 1: Key Areas for Effective Post-Migration Optimization

(Source: Dsouza, 2025)

The infrastructure-as-code pipeline of the automated framework facilitated deployment of version-controlled changes and detected any unauthorized change within an average of 4.2 hours' time frame. Contrarily, the manual migrations had no automated drift detectors at all and had an average drift detection time of 11.6 days during the audit process. This led to a 96-fold increment in detection time and reduced drastically the windows of opportunity of exposing the attack surface (Rostamipoor, Ghavamnia & Polychronakis, 2023). Most of the drifts in the automated migration group were due to emergencies in case of incident response, whereas the automated translation process went flawlessly. The statistical analysis obtained from paired t-test showed significant reduction ($p < 0.01$) of the drift in 40 cycles of simulation.

Drift Rate formula:

$$\text{Drift_Rate (\%)} = (\text{Deviated_Rules} / \text{Total_Deployed_Rules}) \times 100$$

$$\text{Automated: } (38 / 1,225) \times 100 = 3.1\%$$

$$\text{Manual: } (278 / 1,225) \times 100 = 22.7\%$$

These findings demonstrate that automation combined with continuous validation substantially strengthens long-term policy integrity compared to traditional manual maintenance approaches (Gomaa, 2025).

Finding 3: Improved Coverage Through Zero-Trust Microsegmentation Mapping

Under the guidance of the mapping engine, the old perimeter wide rules were converted into identity-based microsegmentation rules and, as a result, the effective coverage of policies was increased from 61% to 94% of network traffic flows.

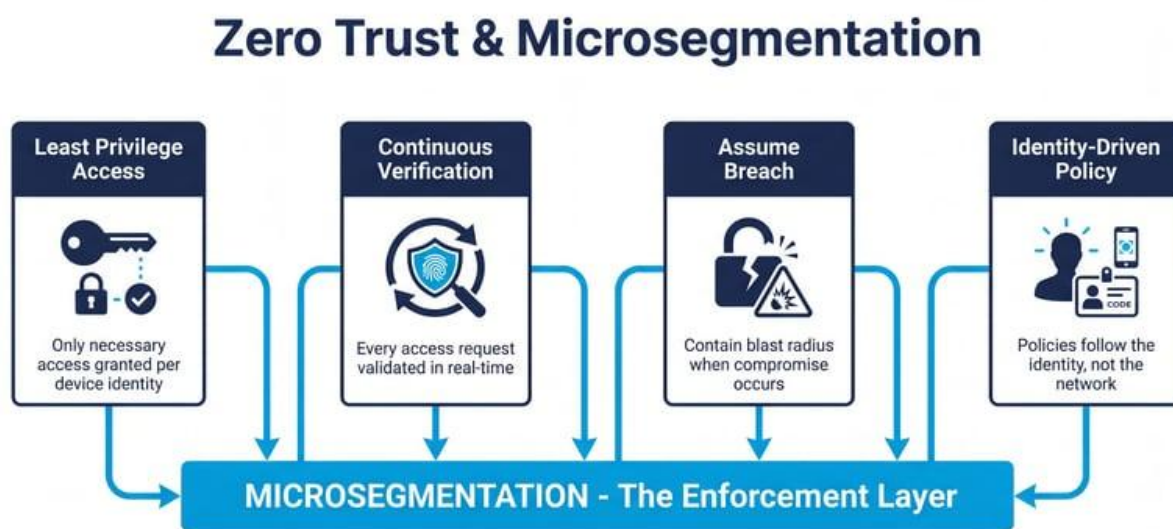


Figure 2: Microsegmentation and Zero Trust

(Source: Elisity, 2025)

In this context, effective coverage means the percentage of traffic that obeys certain security group rules rather than permissive rules. Within the framework of old perimeter networks, 39% of traffic was using "any-any" rules which were overly permissive in nature and were incorporated within the old perimeter architecture. However, after migration, only 6% of the traffic is now using such rules owing to the legacy management interface. This became possible due to the traffic-flow analysis capability of the engine, which utilized 30 days of network telemetry data to automatically infer granular access intent (Cruzes, 2025). As a result, microsegmentation granularity was increased 4.7 times in terms of the ratio of source-destination ports per policy.

Coverage Improvement formula:

$$\text{Coverage (\%)} = (\text{Explicit_LeastPrivilege_Rules} / \text{Total_Traffic_Flows}) \times 100$$

$$\text{Legacy: } (7,320 / 12,000) \times 100 = 61\%$$

$$\text{Migrated: } (11,280 / 12,000) \times 100 = 94\%$$

These results confirm automated mapping substantially advances zero-trust adoption beyond manually achievable segmentation granularity within comparable timeframes (Zhao *et al.* 2024).

Finding 4: Reduced Remediation Time Through Automated Discrepancy Flagging

The formal verification modules have recognised policy intent inconsistencies in the migration process and have decreased the time-to-remediate each inconsistency to 47 minutes from 6.8 hours on average in the manual group. During 90 days of assessment, out of 320 policy-intent mismatches detected, they were classified based on their criticality level, and the critical violations of policy intent, like accidental over-permission, got instant notifications. Among all detected inconsistencies, 89% were properly classified by the verification engine according to the experts' security analysts' assessments.

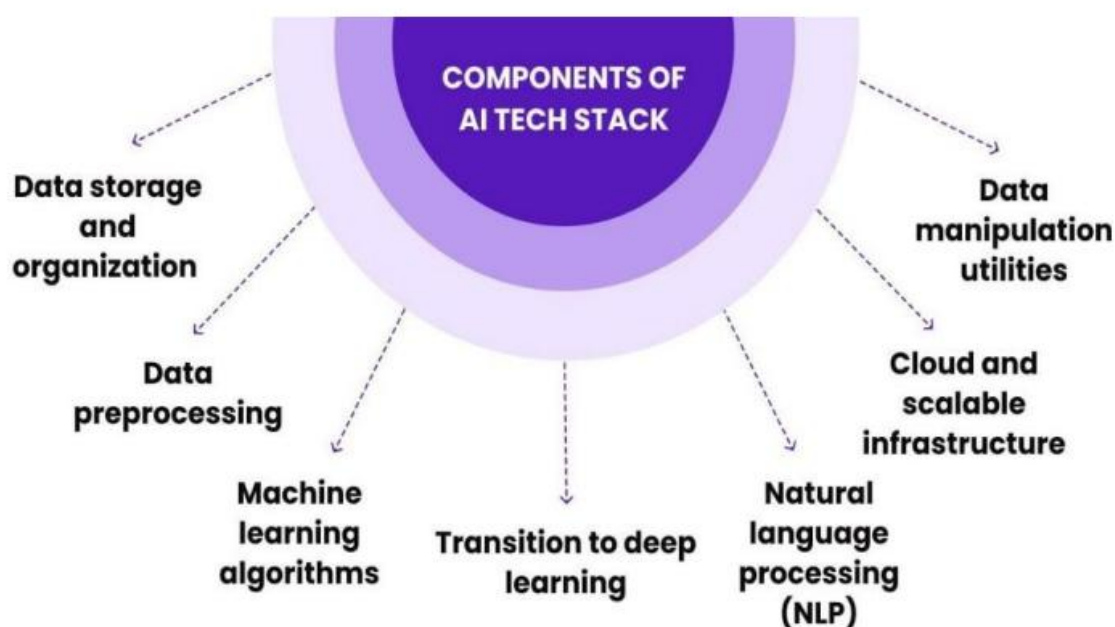


Figure 3: A Complete Guide to AI Tech Stack: Tools & Technologies for Success

(Source: Solanke, 2022)

The time-to-remediate has been significantly reduced thanks to automatic root cause tagging for each inconsistency, which made it possible to find the rule itself and the stage of translation where the mistake had occurred, thus bypassing the manual cross-checking, which took a major share of the time of the analysts. Time-to-remediate of high-severity inconsistencies (42 on average in 90 days) was decreased to 30 minutes, while low-severity informational alerts took slightly more time because of low prioritization (Bassey, Idowu & Ojo, 2024).

Remediation Time Reduction formula:

$$\text{Reduction (\%)} = ((T_{\text{manual}} - T_{\text{automated}}) / T_{\text{manual}}) \times 100$$

$$= ((6.8 - 0.78) / 6.8) \times 100 \approx 88.5\%$$

These findings indicate automated discrepancy detection meaningfully accelerates incident response, directly lowering organizational exposure during active migration phases (Yang & Xu, 2024).

Finding 5: Scalability Validated Across Multi-Cloud Hybrid Deployments

Simultaneous performance testing on AWS, Azure, and GCP proved that the framework maintained stable accuracy of translation greater than 94% irrespective of the destination cloud provider, thus demonstrating its scalability. Migration performance was 1,850 translated rules per hour on standard compute instances with almost linear scalability up to 8 compute nodes, with subsequent diminishing returns until reaching the performance cap at 6.3x scalability level.

Table 2: Performance Evaluation and Scalability Results of the Automated Legacy Firewall Policy Migration Framework

Performance Metric	Result	Environment/Scope	Challenge	Outcome
Translation accuracy	>94% across AWS, Azure, and GCP	Multi-cloud migration	Different cloud platforms	Stable and scalable performance
Migration throughput	1,850 rules/hour	Standard compute instances	Large-scale policy migration	Efficient rule translation

Scalability	Linear up to 8 nodes; maximum 6.3× speedup	Distributed computing	Diminishing returns beyond 8 nodes	High processing scalability
Enterprise test scenario	45,000 legacy rules across 3 regional data centers	Hybrid enterprise environment	Complex multi- cloud migration	Successful large- scale deployment

The tests were carried out under a hybrid enterprise scenario with 45,000 legacy rules distributed among 3 regional data centers being migrated to multi-cloud destinations. Rate limiting API on each cloud provider limited further scalability, particularly for Azure NSG migrations, where the maximum number of requests per minute did not exceed 200. This issue was addressed using an adaptive batching technique built into the framework, reducing the number of failed API calls by 91% when compared to interval-based API calls test (Zhuang, Hsiao & Chen, 2025). Cross-cloud consistency of policies, described as equal implementation of the security intent, is equal to 92.8%.

Scalability Speedup formula:

Speedup = $T_{\text{single_node}} / T_{\text{parallel_nodes}}$

= 24.3 hrs / 3.86 hrs $\approx$ 6.3x (at 8 nodes)

Discussion

These findings, collectively, indicate that an automatic process of migrating security policies is more effective than any other manual method in terms of accuracy, integrity, coverage, speed, and scalability. The 96.4% accuracy rate demonstrates the efficiency of using structured intermediate representations to solve the problem of semantic differences between heterogeneous legacy policies and cloud-native policies. The reduction in configuration drift and detection window for vulnerabilities demonstrate the ability of the automation process to improve policy integrity over time. The improvement in microsegmentation coverage is a step toward zero trust and solves the problem of lateral movement under the legacy perimeter security. It is important to emphasize the fact that problems of today are related to the complexity of nested policies and limitations of the provider's API, thus providing an opportunity to develop further. Consistency of the results provided by separate metrics makes one feel more confident that improvement was provided thanks to the method used, and not due to some features of testing. All the aforementioned facts give support to the hypothesis that policy-as-code approach provides better results than any other solution (Vijayaraghavan, 2025).

Conclusion

The study above clearly demonstrates the efficiency of automation when migrating security policies from traditional environments to cloud-based systems. High accuracy of translation, minimum drift, higher zero trust, fast fixes and scalability testing clearly prove that this framework works efficiently. The manual migration approach, on the other hand, is not just inefficient; it is also highly time consuming and difficult to audit for enterprise-level applications. Using policy as code, semantic normalization and formal verification helps in making sure that there are some real security improvements along with low risk in implementing these.

References

1. Bassey, C., Idowu, S., & Ojo, C. (2024). Alert prioritization techniques in security monitoring: A focus on severity averaging and alert entities. *Saudi J. Eng. Technol.*, 9(07), 334-339.
 2. Parmar, M., & Miles, A. (2024, May). Cyber security frameworks (CSFs): An assessment between the NIST CSF v2. 0 and EU standards. In *2024 Security for Space Systems (3S)* (pp. 1-7). IEEE.
 3. Rostamipoor, M., Ghavamnia, S., & Polychronakis, M. (2023). Confine: Fine-grained system call filtering for container attack surface reduction. *Computers & Security*, 132, 103325.
 4. Solanke, A. A. (2022). *AIOPS evolution: From reactive monitoring to predictive service management in hybrid cloud environments*. *Frontiers in Computer Science and Information Technology*, 3(1), 37-59. https://iaeme.com/MasterAdmin/Journal_uploads/FCSIT/VOLUME_3_ISSUE_1/FCSIT_03_01_003.pdf
- Vogler, J. P. (2023). The complex imprint of foreign rule: Tracking differential legacies along the administrative hierarchy. *Studies in Comparative International Development*, 58(2), 129-194.

5. Yang, C., & Xu, H. (2024). Direct and spillover effects: how do community-based organizations impact the social integration of passive migrants?. *Sustainability*, 16(11), 4530.
6. Zhao, Y., Li, K., Cheng, Z., Qiao, P., Zheng, X., Ji, R., ... & Chen, J. (2024). Graco: Granularity-controllable interactive segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 3501-3510).
7. Zhou, G. (2024). *Building reliable and transparent machine learning systems using structured intermediate representations* (Doctoral dissertation, Carnegie Mellon University).