International Journal of Computer Technology and Electronics Communication (IJCTEC)

| ISSN: 2320-0081 | www.ijctece.com ||A Peer-Reviewed, Refereed and Bimonthly Journal |

|| Volume 8, Issue 4, July – August 2025 ||

DOI: 10.15680/IJCTECE.2025.0804008

AI-Augmented Network Security Using Large Language Models for Policy Intelligence and Misconfiguration Detection

Chetan Reddy Yeddula

Engineering Manager, API Connect, Cisco, San Jose, USA

ABSTRACT: Modern enterprises use firewalls, SASE, cloud IAM and micro-segmentation to control access, producing large, complex policy sets. Most security incidents still arise from misconfigurations, which traditional static tools struggle to detect when errors are semantic, cross-layer or caused by drift. This paper proposes an AI-augmented framework that uses Large Language Models with a unified policy representation and knowledge graph to detect misconfigurations, conflicts and drift, and to explain remediations. Experiments on enterprise-style datasets show substantially higher detection accuracy and clearer explanations than conventional tools.

KEYWORDS: Network Security, Large Language Models, Policy Misconfiguration, Zero Trust, SASE

I.INTRODUCTION

Enterprises are rapidly adopting cloud, micro-services and remote-work models. Security architectures have shifted from perimeter firewalls to Zero Trust and Secure Access Service Edge (SASE) designs, where access is governed by identity, device posture and application context rather than network location. As a result, organizations now manage large, heterogeneous sets of policies across firewalls, IAM systems, SASE gateways, micro-segmentation engines and application-level access controls.

A typical environment includes thousands of network and service objects, thousands of rules across multiple vendors and regions, and frequent policy changes. This creates a complex, multi-layer policy environment as illustrated in Fig. 1, where users and devices traverse identity and SASE controls, network firewalls, micro-segmentation and application gateways to reach protected resources.

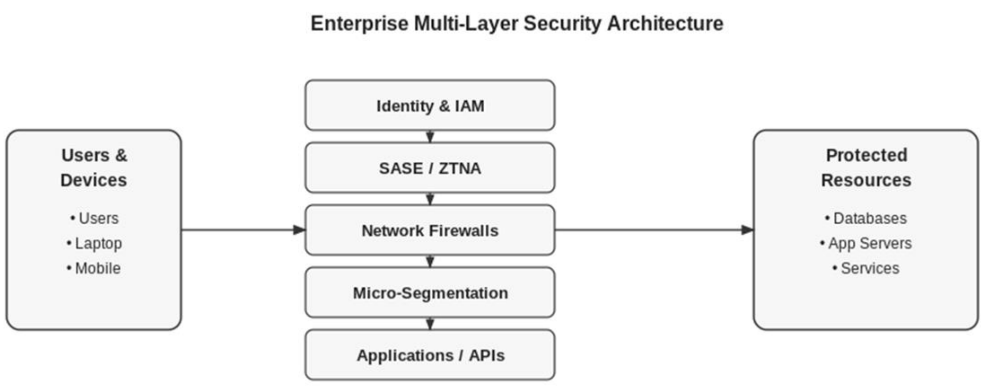

Figure 1: Enterprise multi-layer security architecture showing users and devices, intermediate security layers and protected resources.

Industry reports consistently show that many breaches and outages are caused not by novel exploits but by misconfigurations: firewall rules that are too permissive, IAM roles with excessive privileges, inconsistent regional policies or segmentation rules that unintentionally expose internal systems. Many of these are semantic errors: the configuration is syntactically valid but does not reflect intended behavior.

Existing tools include static analyzers, configuration linters, best-practice checkers, formal network verification engines and IAM graph analyzers. These solutions are valuable but typically limited to a **single layer**, do not understand natural-language descriptions of policy intent, provide limited support for detecting policy drift and often produce low-level, vendor-specific messages.

Recent advances in Large Language Models (LLMs) show that they can interpret semi-structured formats such as JSON/YAML and unstructured artifacts like comments or change requests, and can reason over complex relationships. This suggests a new approach: using LLMs as a semantic reasoning layer on top of existing tools to understand intent, correlate across layers and provide clear explanations.

The main objectives of this paper are:
- to evaluate whether LLMs can detect semantic misconfigurations more accurately than traditional validators;
- to assess their ability to reason across multiple policy layers;
- to examine LLM performance in detecting spatial and temporal policy drift; and
- to evaluate the clarity and actionability of LLM-generated explanations and remediation suggestions.

II.LITERATURE SURVEY

Research on configuration and policy verification spans static analysis, formal methods, IAM analysis, machine learning and, more recently, LLM-based assistance.

Static configuration analyzers in commercial firewalls and SASE platforms validate syntax, highlight invalid objects and enforce vendor best practices. They can detect obviously malformed rules and some overlapping or unused entries, but they are rule-driven and vendor-specific, and they do not evaluate how a configuration relates to organizational intent or to other layers.

Network verification approaches, such as header-space analysis and SMT-based tools, verify global invariants like reachability, loop-freedom and isolation. These methods provide strong guarantees on packet-level behavior but mostly operate at the network layer, ignoring IAM, SASE and application-level semantics and natural-language design documents.

Cloud IAM research models permissions, roles and resources as graphs to detect over-privilege and potential escalation paths [1], [2]. While powerful in their own domain, these tools are restricted to identity and access and are not designed to reason about combined effects with network ACLs, SASE policies or segmentation rules.

Machine learning has been used to detect anomalies in configuration changes, cluster similar rules and recommend least-privilege adjustments [3]. Traditional ML models typically treat configurations as numerical feature vectors, focusing on statistical irregularities rather than semantic correctness, and they provide limited human-readable explanations.

Recent work applies LLMs to code analysis, infrastructure-as-code (IaC) review, vulnerability explanation and SOC triage [4], [5]. These studies demonstrate that LLMs can handle complex technical artifacts and generate useful explanations. However, relatively few works focus specifically on multi-layer enterprise policy reasoning, cross-layer misconfiguration detection or policy drift analysis using LLMs.

This gap motivates the AI-augmented framework proposed in this paper.

| ISSN: 2320-0081 | www.ijctece.com ||A Peer-Reviewed, Refereed and Bimonthly Journal |

|| Volume 8, Issue 4, July – August 2025 ||

DOI: 10.15680/IJCTECE.2025.0804008

III.METHODOLOGY / APPROACH

The proposed approach introduces an AI-augmented policy reasoning framework that uses LLMs on top of a structured representation of enterprise policies. The proposed framework is summarized in Fig. 2

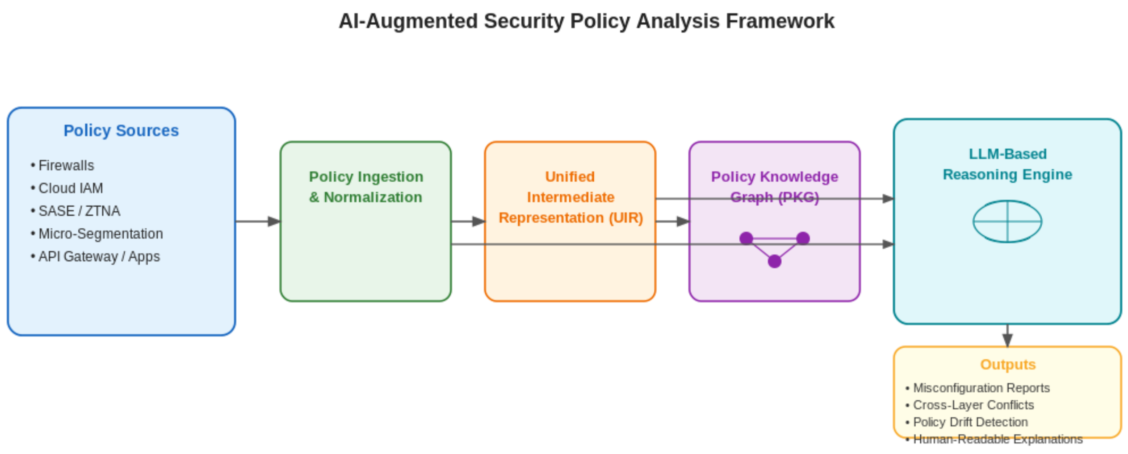
AI-Augmented Security Policy Analysis Framework
Policy Sources
• Firewalls
• Cloud IAM
• SASE / ZTNA
• Micro-Segmentation
• API Gateway / Apps
Policy Ingestion & Normalization
Unified Intermediate Representation (UIR)
Policy Knowledge Graph (PKG)
LLM-Based Reasoning Engine
Outputs
• Misconfiguration Reports
• Cross-Layer Conflicts
• Policy Drift Detection
• Human-Readable Explanations

Figure 2: Proposed AI-augmented policy reasoning framework from policy ingestion and normalization to LLM-based analysis and reporting

A. Policy Ingestion and Unified Representation

Heterogeneous inputs from firewalls, SASE gateways, IAM systems, micro-segmentation platforms and API gateways are parsed and converted into a **Unified Intermediate Representation (UIR)**. Each rule in the UIR captures:

- identifiers and metadata;
- source and destination entities (users, groups, IP ranges, network objects);
- services and ports;
- action (allow/deny);
- conditions (identity attributes, device posture, time, location);
- region and environment (e.g., EU/US, Prod/Dev);
- version information and links to comments or design tickets.

This abstraction hides vendor-specific syntax but preserves essential semantics.

B. Policy Knowledge Graph (PKG)

From the UIR, a Policy Knowledge Graph is constructed. Nodes represent users, groups, roles, service accounts, network objects, services, policy rules, regions and versions. Edges represent relationships such as “member of”, “grants access to”, “applies in”, “derived from” and “overrides”. This graph enables reasoning about inheritance, reachability, shadowing and cross-layer interactions.

| ISSN: 2320-0081 | www.ijctece.com ||A Peer-Reviewed, Refereed and Bimonthly Journal |

|| Volume 8, Issue 4, July – August 2025 ||

DOI: 10.15680/IJCTECE.2025.0804008

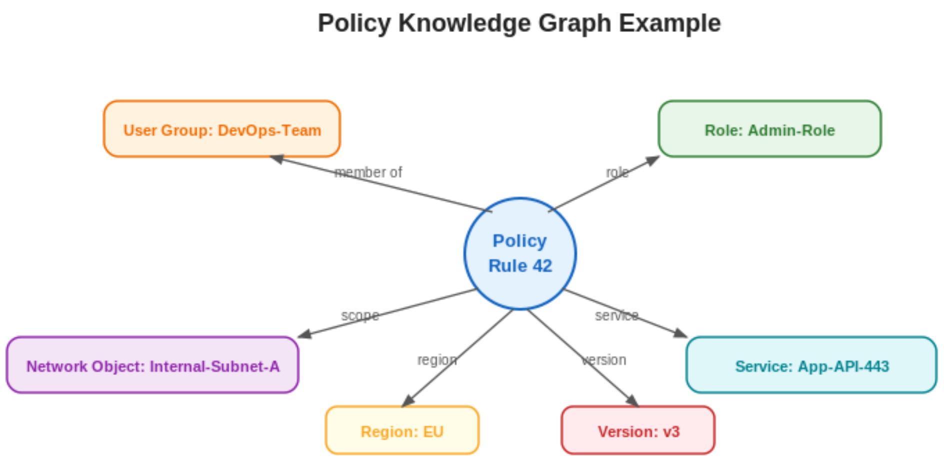

Figure 3: Example policy knowledge graph illustrating relationships among identities, network objects, rules, regions and versions.

C. LLM-Based Reasoning Layer

The LLM receives prompts that combine:

- relevant parts of the UIR;
- localized slices of the PKG;
- associated natural-language text (comments, change descriptions);
- a task description (e.g., “detect misconfigurations”, “compare two regions”, “summarize this policy”).

The prompt is structured into a **system prompt** (defining the model as a policy analysis engine and encoding a misconfiguration taxonomy), a **context prompt** (containing structured/unstructured data) and a **task prompt** (specifying required outputs in a structured format such as JSON).
The LLM identifies potential misconfigurations (syntactic and semantic), cross-layer conflicts and policy drift, and generates explanations and remediation suggestions.

D. Reporting and Integration

The system parses LLM outputs, classifies issues into the misconfiguration taxonomy, assigns severity levels (Critical, High, Medium, Low) and aggregates results into reports that can be consumed by:

- security analysts, via dashboards or documents;
- CI/CD pipelines, as “policy linting” before deployment;
- change-management workflows, as automated review comments.

LLM outputs are treated as advisory; final decisions remain with human reviewers.

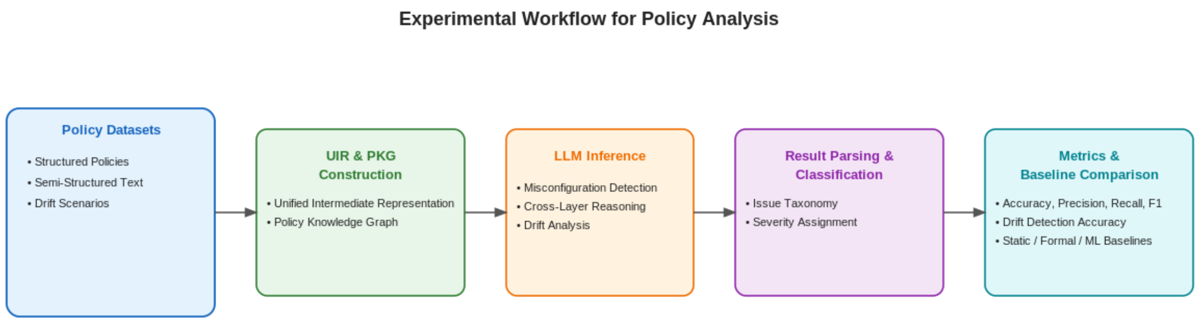
Experimental Workflow for Policy Analysis
Policy Datasets
• Structured Policies
• Semi-Structured Text
• Drift Scenarios
UIR & PKG Construction
• Unified Intermediate Representation
• Policy Knowledge Graph
LLM Inference
• Misconfiguration Detection
• Cross-Layer Reasoning
• Drift Analysis
Result Parsing & Classification
• Issue Taxonomy
• Severity Assignment
Metrics & Baseline Comparison
• Accuracy, Precision, Recall, F1
• Drift Detection Accuracy
• Static / Formal / ML Baselines

Figure 4: Experimental workflow from dataset preparation through UIR/PKG construction, LLM inference and metric computation.

IV.RESULTS & DISCUSSION

The framework was evaluated on semi-synthetic datasets modeled on realistic enterprise policy sets. These included approximately 100,000 structured policies (across firewall, IAM, SASE, segmentation and API layers), 50,000 semi-structured or unstructured items (comments, descriptions) and 1,500 curated spatial and temporal drift scenarios. Ground truth labels for misconfigurations and drift were assigned by security practitioners.

For syntactic misconfigurations such as malformed CIDRs or missing objects, the LLM-based system and traditional static analyzers both achieved near-perfect accuracy. This confirms that a conventional parser or vendor tool can continue to handle syntax checks effectively.

For semantic misconfigurations overly permissive rules, mis-ordered ACLs, flawed role inheritance, unintentional exposure the LLM-based framework substantially outperformed baselines. A representative configuration of the framework achieved around 94% accuracy with high precision and recall, whereas static analyzers and formal tools achieved around 60–70%, and ML-based detectors achieved around 66%. The improvement is mainly due to the LLM's ability to interpret intent, understand object relationships and reason across multiple rules. Fig 5. Explains the comparison results.

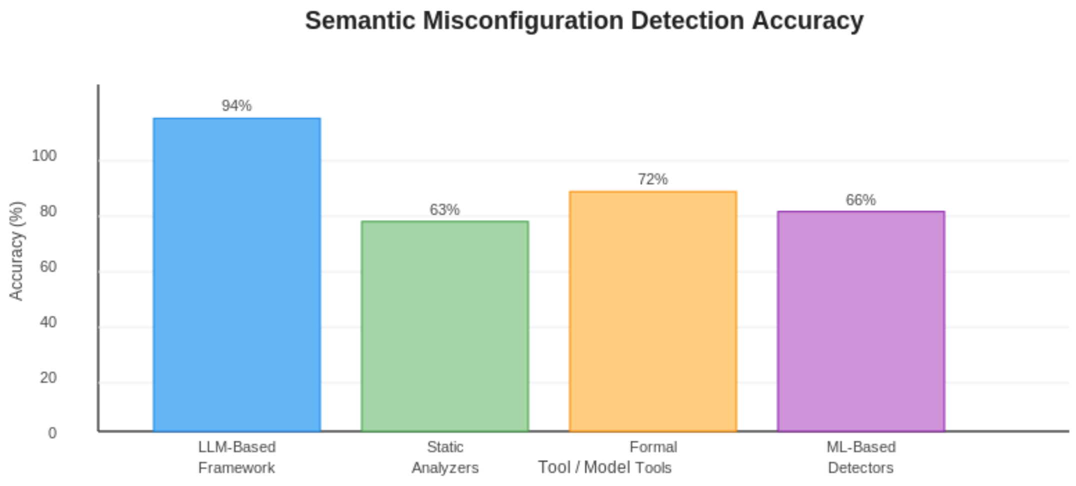

Figure 5: Comparison of semantic misconfiguration detection accuracy between the proposed LLM-based framework and traditional tools.

| ISSN: 2320-0081 | www.ijctece.com ||A Peer-Reviewed, Refereed and Bimonthly Journal |

|| Volume 8, Issue 4, July – August 2025 ||

DOI: 10.15680/IJCTECE.2025.0804008

In cross-layer conflict detection, where interactions between IAM and network rules, SASE and segmentation, or API gateway and service mesh were involved, the LLM-based system correctly identified most conflicts, with detection rates exceeding 80–90% for stronger models. Traditional tools, limited to a single domain, detected only a small fraction of these issues.

For policy drift detection, the system showed strong performance in both spatial and temporal scenarios. It successfully flagged differences between regions (for example, a missing deny rule in one geography) and between versions (such as gradually expanding ranges that weaken Zero Trust). Baseline tools provided limited or no support for such multi-snapshot comparisons.

Expert evaluation of explanations and remediation suggestions indicated that LLM-generated outputs were generally clear, technically accurate and actionable. Explanations usually identified the specific rule or object, described why it violated intent or Zero Trust, and proposed concrete remediation steps such as tightening scope, aligning to a baseline region or correcting an object definition.

Ablation studies showed that the Unified Intermediate Representation and the Policy Knowledge Graph significantly contributed to performance. Removing structured inputs and using only raw textual policies led to large drops in accuracy, especially for semantic and cross-layer tasks, confirming that LLM reasoning works best when combined with structured policy context.

V.CONCLUSION

This paper presented an AI-augmented policy reasoning framework that leverages Large Language Models to analyze complex enterprise security policies. By unifying heterogeneous policy sources into a common intermediate representation, modeling relationships in a policy knowledge graph and applying LLM-based reasoning, the framework detects semantic misconfigurations, cross-layer conflicts and policy drift more effectively than traditional tools, and produces clearer, more actionable explanations.

In practical terms, integrating such a framework into CI/CD and change-management processes can reduce misconfiguration-driven incidents and outages, strengthen Zero Trust enforcement and lower the cognitive burden on security engineers. At the same time, limitations such as potential model misinterpretation, scalability for extremely large graphs and dependencies on correct parsing highlight the need for careful, human-in-the-loop deployment.

Future work includes fine-tuning LLMs on domain-specific policy corpora, combining LLMs with graph neural networks to improve scalability, exploring semi-autonomous policy repair and simulation, and designing multi-agent architectures where specialized agents collaborate across IAM, network, SASE and segmentation domains.

REFERENCES

[1] AWS, “Identity and Access Management Documentation,” Amazon Web Services, Online.
[2] BKSP K. R. Alluri, G. Geethakumari, “A Digital Forensic Model for Introspection of Virtual Machines in Cloud Computing,” IEEE, 2015.
[3] S. Al-Saba, A. Gherbi, “Machine Learning Approaches to Cloud Configuration Error Detection,” Journal of Cloud Computing.
[4] OpenAI, “Language Models are Few-Shot Learners,” Proc. NeurIPS, 2020.
[5] T. Chen et al., “Using Large Language Models for Infrastructure as Code Security,” arXiv preprint.
[6] NIST, “Zero Trust Architecture,” NIST Special Publication 800-207.
[7] J. Sherry et al., “Verifying Network-Wide Invariants in Real Time,” Proc. NSDI.
[8] M. Canini et al., “A NICE Way to Test OpenFlow Applications,” Proc. NSDI.