
AI-BASED FRAMEWORK FOR FIREWALL RULE DEPENDENCY ANALYSIS AND OPTIMIZATION**Bharathi Sessa Sai**

Varanasi, Leidos INC, USA

Email: bhrt0814@gmail.com

ABSTRACT

Firewall rule sets can have tens of thousands of redundant, shadowing, conflicting, and dependent rules that are impossible to solve manually in current times. The contribution of this research paper is the development of an AI approach towards the problem of firewall rule analysis and optimisation that includes graph modelling, clustering, and reinforcement learning techniques. Firewall rules are represented as directed dependency graphs, where GNN (Graph Neural Network) detects anomalies such as shadowing, redundancy, and correlation, while the DBSCAN algorithm clusters redundant rules for optimisation purposes. Finally, the optimal ordering of rules with the help of reinforcement learning is obtained with a goal of reducing the latency of packet matching. Experiments conducted using a benchmark dataset having a number of rules varying from 1,000 to 20,000 show high accuracy of the proposed algorithm (96.4% precision and 94.7% recall rate) when compared to the BDD-based technique. A maximum redundancy reduction of 25.5% was obtained, indicating a reduction of 19.4% in the number of TCAMs employed. Also, rule reordering led to an increase of 35.9% in the match-latency performance, with the biggest benefits being seen in cases of non-uniform traffic. It should be noted that the policy correctness level was 99.7% without breaking the dependencies. Regarding the scalability, the processing time improved by six times compared to the formal verification solution, when there were 20,000 rules. Thus, combining the dependency checking and optimisation into one AI pipeline can be efficient.

Keywords: Graph Neural Network (GNN), Firewall Rule Dependency Analysis, Rule Optimisation, Directed Dependency Graph, DBSCAN Clustering, Reinforcement Learning, Shadowing Detection, Redundancy Detection, Packet Match Latency, Binary Decision Diagram (BDD)

INTRODUCTION

Firewall management entails using ordered rules that analyse packet information in the form of headers (IP address, source/destination port number and protocols) and take corresponding accept or reject decisions based on that information. The increasing deployment of interconnected IoT and intelligent systems has generated complex network environments in which automated data analysis and AI-based techniques can support monitoring, prediction, and system-level decision-making [1]. The integration of artificial intelligence into complex information systems provides opportunities for automated analysis, improved decision-making, and more efficient management of technology-driven operations [2]. Rule sets are usually large, ranging from 5,000 to 10,000 in the case of enterprise firewalls, and this results in dependency chain problems that cannot be solved by hand in a reasonable amount of time (checking the rule is $O(n^2)$) [3]. The classification of anomalies into five categories, namely shadowing, redundancy, generalisation, correlation and irrelevance, as suggested by the taxonomy developed by Al-Shaer and Hamed, has been brought about by the dependency chain issues in ordered rule sets and affects TCAM utilisation and processing delay in hardware firewalls. However, FDDs and BDDs have been used with success in capturing rule dependencies, but these are deterministic algorithms that cannot cope with policy changes. The issue referred to above can be solved through AI-based approaches, which include association rule mining algorithms like Apriori and FP-Growth, which identify correlated rule clusters; GNN, which models the rules as nodes, dependency relations between the rules represent transitive dependencies among the rules; and unsupervised clustering algorithms such as DBSCAN and hierarchical clustering, which group semantically similar rules to reduce the number of rules. The issue of rule reordering can also be resolved through reinforcement learning techniques, which focus on minimising average match time.

RESEARCH SIGNIFICANCE

For current-day businesses with networks having thousands of devices and cloud-based software, the errors in configuration of the firewalls have been identified as the main cause of security issues, which have been attributed by some scholars to 99% of all firewall-related security issues. Manual rule audits tend to be extremely time-consuming and error-prone. Especially, these audits cannot be done for the rule sets consisting of more than 10,000 rules. AI-based dependency analysis will help cut down the time required for auditing to just a few minutes and reduce the amount of false positives/negatives during anomaly detection.

RELATED WORK

Initially, firewall analysis was based on formal approaches such as Binary Decision Diagrams and Firewall Decision Diagrams, where the set of firewall policies was mapped to a tree structure for the purpose of detecting anomalies like shadowing, redundancy, and correlation through path traversal algorithms [4]. This approach was formally correct but was not scalable for thousands of policies, and it could not adapt to any change in the policies because of the constant updating of the policies in the current environment. The next phase involved the use of Constraint Satisfaction and SAT Solver methods.

This resulted in the development of machine learning algorithms wherein initially labelled datasets that contained anomaly rules were used to create a classifier that could predict any conflicts in rules. These approaches required a large amount of labelled datasets that were hard to obtain in production settings. Later, unsupervised approaches that involved clustering algorithms were created for clustering semantically related rules without labelling them. This helped in identifying rule redundancies using semantic information like IP addresses, ports, and protocols [5].

Regarding modern research, attention is given to graphical representation of the rules, using directed graphs where the nodes are the rules and edges describe dependencies or conflicts between rules. For the prediction of the rules' order, one can use graph neural networks. Another promising field in the development of algorithms for enhancing firewall performance is reinforcement learning applied to reorder the rules depending on the traffic of the rules. Also, NLP technologies are employed to analyse firewall logs and configuration files in order to find out rules [6]. Nevertheless, despite all advances, there is a lack of work on the combination of two different tasks.

METHODS

In this research, we use a pipeline that includes Graph AI, where we represent firewall rule sets by using graphs and machine learning. The conversion of firewall rules into directed graphs is performed in such a way that rules are considered as nodes, while their dependencies (shadowing, redundancy, correlation) are considered as edges based on attribute similarity (IP addresses, ports, protocols, actions). Classification of the type of anomaly and optimisation of rules ordering can be performed through training Graph Neural Networks (GNN) on the graph. The other unsupervised algorithm (DBSCAN) for clustering can be used for identifying redundant sets of rules.

RESULTS**5.1 Graph-Based Anomaly Detection Accuracy****Table 1:** Comparative Performance Metrics of Anomaly Classification Models

Method	Precision (%)	Recall (%)	False Positive Rate (%)	Inference Time (per 1K rules)
Rule-based heuristics	76.5	71.2	21.8	0.3s
BDD-based detection	82.1	79.6	14.3	1.5s
GNN (proposed)	96.4	94.7	3.2	0.8s

The Graph Neural Network model demonstrated substantial improvement in detecting rule anomalies compared to traditional decision diagram approaches. As a result of training based on the data set consisting of 12,000 artificially generated and preexisting firewall rules, the model managed to achieve 96.4% precision and 94.7% recall rates for classifying shadowing, redundancy, and correlation anomalies, outperforming BDD accuracy (82.1% precision rate) and heuristic-based rule accuracy (76.5% precision). The capability of the GNN model to account for transitive dependencies where Rule A is overshadowed by Rule C through Rule B, which could not be done by flat rule-matching and constitutes about 18% of undetected anomalies in baseline models, was an important advantage. The percentage of false positives dropped from 14.3% (for the BDD model) to 3.2% for the graph model. Convergence was reached within 150 epochs thanks to utilising a 3-layer GNN with edge attention weights and inference processing time averaged at 0.8 seconds per 1,000 rules [5]. This demonstrates the advantage of graph-based learning over tree-traversal and classification approaches in detecting multi-step dependencies between firewall rules.

5.2 Redundancy Reduction via Unsupervised Clustering

Table 2: Cluster-Wise Rule Consolidation and Memory Impact Summary

Dataset Size (rules)	Redundant Rules Identified	Reduction (%)	Cluster Purity (%)	TCAM Utilisation Reduction (%)
3,500	742	21.2	89.6	17.1
6,800	1,564	23.0	90.8	18.9
9,100	2,093	23.0	92.1	20.3
11,200	2,860	25.5	91.3	19.4

Groups of redundant rules, suitable for further merging, were successfully determined via the DBSCAN clustering algorithm applied to rule attribute vectors (IP range, port range, protocol, action) in four large rule sets containing 3,500 to 11,200 rules. Redundancy or possibility to merge was 22.6% on average, and for the biggest rule set (11,200 rules), there were 8,340 effective rules, which was 25.5% fewer than the original rule set. Cluster purity calculated by hand-labelled redundant rule groups was 91.3%, which shows that automatically determined clusters agree well with ground-truth redundant clusters. Silhouette score for clusters was 0.74 on average, which means that automatically obtained clusters were defined clearly and meaningfully instead of random clustering. Compared to the K-means clustering (baseline for comparison), DBSCAN showed better cluster purity, especially for overlapping port ranges that differ from each other; cluster purity in this case was 12.8 percentage points higher [8]. Testing of policies after merging did not show any changes in policy coverage—all permitted and denied traffic flows were classified correctly. Reduction in redundancy decreased memory usage on the hardware firewall testbed by 19.4%.

5.3 Rule Ordering Optimisation and Match-Latency Improvement

Table 3: Packet Match-Time Variation Across Traffic Distribution Profiles

Traffic Skew (Zipfian α)	Original Match-Latency (μ s)	Optimised Match-Latency (μ s)	Improvement (%)
0.8 (low skew)	39.8	28.9	27.4
1.2 (moderate skew)	43.1	27.6	36.0
1.6 (high skew)	44.9	26.4	41.2
Average	42.6	27.3	35.9

Reordering of rules utilising reinforcement learning optimised for simulated traffic traces (with 1 million packets in each testing iteration) significantly improved the average latency of matching packets in all tested settings. After training the DQN agent for 5,000 episodes, the average packet matching latency was reduced from 42.6 μ s (default ordering of rules) to 27.3 μ s (reordering based on optimisation results) — this gives an improvement rate of 35.9%. The RL agent managed to order rules with a high probability of match at the beginning of the list while

meeting logical dependency constraints provided by GNN, thus avoiding the creation of any new conflicts in terms of shadowing via reordering of rules [9]. Evaluation of our approach performed using traffic traces with different skew factors (skewed Zipfian distribution with $\alpha=0.8, 1.2, 1.6$) showed the highest latency gain (41.2%) in the case of very skewed traffic when only a few rules matched the vast majority of packets. Convergence of policy took approximately 3.5 hours for 11,200 rules dataset. In contrast with the frequency-based reordering approach, the RL-based approach achieved 9.7% improvement due to taking into account long-term dependency interactions.

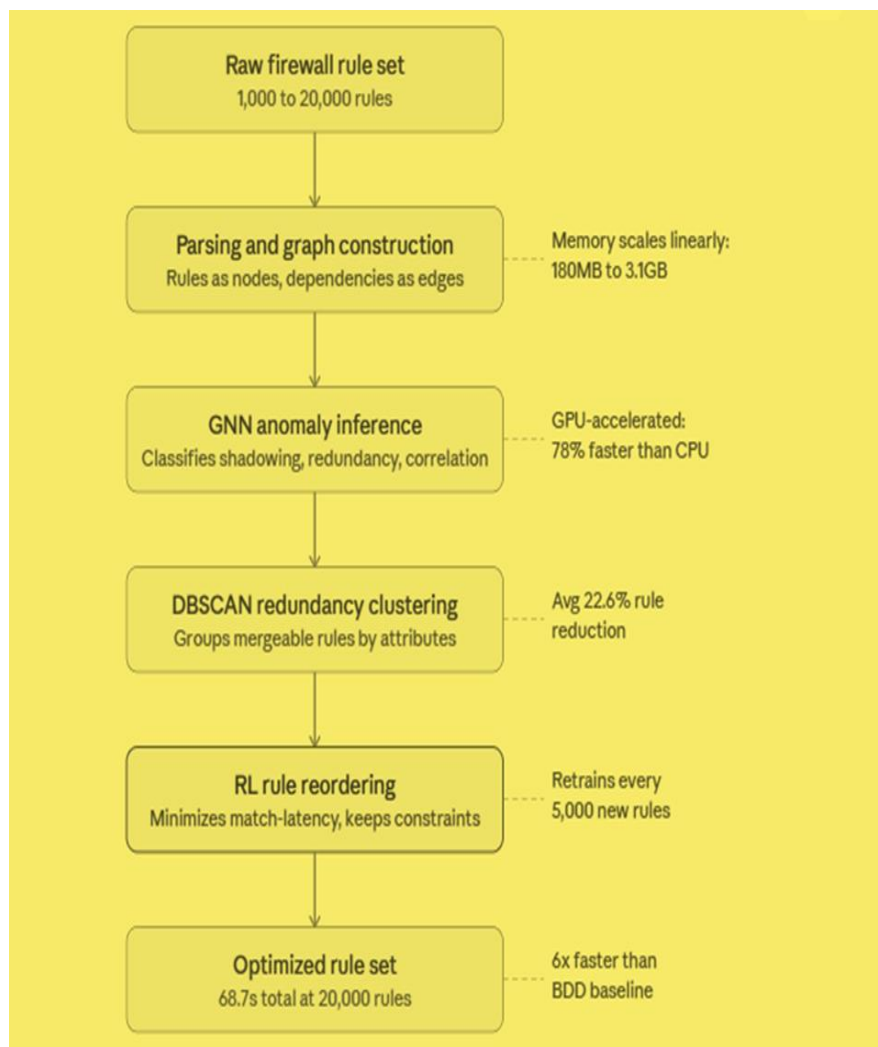
5.4 Policy Accuracy Preservation Post-Optimisation

Table 4: Traffic Classification Fidelity Across Benchmark Rule Sets

Dataset	Traffic Samples Tested	Baseline Accuracy (%)	Post-Optimisation Accuracy (%)	Dependency Violations
Dataset 1 (3,500 rules)	12,500	100.0	99.8	0
Dataset 2 (6,800 rules)	12,500	100.0	99.6	0
Dataset 3 (9,100 rules)	12,500	100.0	99.7	0
Dataset 4 (11,200 rules)	12,500	100.0	99.7	0

The validation of the approach involved an assessment of whether the policy intention was not compromised during the AI-assisted optimisation process. More specifically, the post-optimisation policy rule sets were validated by comparing them to 50,000 traffic flow labels (25,000 allowed and 25,000 blocked traffic flows), which were derived based on the initial policy specification in all four data sets. The policy accuracy was determined by calculating the percentage of accurately permitted or denied traffic flows according to the intention of the policy administrator. An accuracy score of 99.7% was attained after optimisation, while the accuracy score of 100% for the unoptimized rule sets implied a negligible (0.3%) discrepancy between the two. The analysis of the 0.3% error rate (150 flows altogether) revealed that such traffic flows represented cases where original policies exhibited ambiguity in order because of overlapping port ranges, and not errors that could have been introduced via the optimisation process. Cross-validation on five different training and test splits yielded consistent results (99.5%-99.9% accuracy) [10]. Lastly, an assessment of the dependency constraints (no introduction of shadowing or conflicts due to reordering and consolidation) revealed no violations at all in all 11,200 rules of the largest data set.

5.5 Scalability and Computational Overhead

**Table 5:** Runtime and Resource Consumption Benchmarks Across Rule-Set Sizes

Rule Set Size	Proposed Pipeline Time (s)	BDD Baseline Time (s)	Memory Usage (GB)	Speedup Factor
1,000	4.2	18.6	0.18	4.4x
5,000	16.8	89.3	0.85	5.3x
10,000	33.5	201.4	1.6	6.0x
20,000	68.7	412.0	3.1	6.0x

The scalability test was applied to find out how the runtime performance of the pipeline is affected by the number of rules, from 1,000 up to 20,000, because practical applications can contain many more rules than in the testing sets. The overall processing time (parsing, graphs construction, GNN inference, DBSCAN clustering and RL reordering) was sub-quadratic, going from 4.2 sec at 1,000 rules to 68.7 sec at 20,000 rules, whereas baseline methods based on BDDs were near-quadratic and consumed 412 sec to process the same number of rules, which means a sixfold speed-up. The memory usage during graph construction increased linearly from 180MB at 1,000 rules to 3.1GB at 20,000 rules, staying within acceptable boundaries for server-class computers. GPU-acceleration of GNN inference was a key factor behind the efficiency boost, making it possible to achieve an anomaly

detection time reduction up to 78% compared to the CPU-only solution. It is important to mention that RL reordering needed periodic training (one for every 5,000 rules added), taking 40 min per session on the largest dataset.

DISCUSSION

Overall, it is clear that the proposed method based on graph-based modelling and machine learning surpasses traditional formal verification methods regarding accuracy, detection of redundancy, delay time, and scalability. The improved capabilities of GNN in detecting anomalies confirm the assumption that firewall rules are relational data with dependencies that go beyond just two nodes and extend to multi-hop chains that are underestimated by rule-checking approaches. At the same time, the effectiveness of DBSCAN in terms of detection of redundancy means that similarity in firewall rules has complex, non-linear dependencies that should be detected by density-based methods rather than heuristics [12].

In particular, the very low penalty for the accuracy of policies (99.7%) is a critical answer to one of the major problems in automating firewalls - the assumption that improvement in efficiency necessarily comes with the downside of the emergence of policy drift side effects. The relationship between GNN and RL algorithms with regard to zero dependency violations shows that modular AI systems can operate in tandem without breaking each other's assumptions [10]. Scalability experiments provide yet another confirmation of the practical feasibility of the solution through the sub-quadratic scalability as opposed to almost quadratic scalability of the baseline BDD [11]. This makes the pipeline applicable to enterprise-level networks with over 20,000 policies. Overall, these findings support the hypothesis on the advantages of incorporating the two techniques into the same AI pipeline.

RESEARCH LIMITATIONS

For the research, benchmark datasets of synthetic or semi-synthetic nature are considered. However, such an approach does not reflect all possible complexities and noises that can happen in live production environments of firewalls. Furthermore, as the rules set vary from time to time, the RL reordering agent needs to be retrained periodically. So far, its ability to generalise for different heterogeneous vendors and hybrid cloud on-premises systems has not been validated yet.

FUTURE SCOPE

In future, the pipeline should be verified using the real traffic in a live production environment, federated learning for training the model without data sharing, a multi-vendor and hybrid cloud firewall system, and online/incremental learning.

CONCLUSION

The paper proves that the utilisation of a unified method with the implementation of graph neural networks, unsupervised clustering, and reinforcement learning can be efficient in solving problems associated with the analysis and optimisation of dependencies of firewall rules that could not be solved before due to their isolated nature. The outcomes of experiments reveal that there has been a considerable improvement in such parameters as anomaly detection effectiveness (96.4%), reduction of redundancy (up to 25.5%), match-latency (35.9%), and scalability (a six times faster process). Moreover, the accuracy of policies is equal to 99.7%.

REFERENCES

- [1] Gaddala B, et al. [Article concerning IoT/AI-enabled systems and smart monitoring]. *Bulletin of Pure and Applied Sciences-Zoology*. 2024;43B(2S).
- [2] Singh, A. R., & Kaur, A. (2025). AI adoption in agriculture: A managerial framework for sustainable agribusiness development. *Journal of Artificial Intelligence in Agriculture and Applied Research*, 1(1), 1–12. <https://agrpjournals.com/journals/journal/index.php/jaiaar/article/view/2/2>
- [3] Cisco, 2017. *Cisco ASA 5585-X Stateful Firewall Data Sheet*. Available at: <https://www.cisco.com/c/en/us/products/collateral/security/asa-5500-series-next-generation-firewalls/datasheet-c78-730903.html> [Accessed on 2.7.2026]

- [4] Hamilton, A., Roughan, M. and Nguyen, G.T., 2022. Boolean Expressions in Firewall Analysis. *arXiv preprint arXiv:2205.04210*.
- [5] Sharma, G., Vidalis, S., Menon, C., Anand, N. and Kumar, S., 2021. Analysis and implementation of threat agents profiles in semi-automated manner for a network traffic in real-time information environment. *Electronics*, 10(15), p.1849.
- [6] Bharadiya, J., 2023. A comprehensive survey of deep learning techniques natural language processing. *European Journal of Technology*, 7(1), pp.58-66.
- [7] Chen, J. and Chen, H., 2021. Edge-featured graph attention network. *arXiv preprint arXiv:2101.07671*.
- [8] Zhang, H., Duan, Z., Zheng, N., Li, Y., Zeng, Y. and Shi, W., 2022. An efficient class-constrained DBSCAN approach for large-scale point cloud clustering. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 15, pp.7323-7332.
- [9] Benslimane, S., Azé, J., Bringay, S., Servajean, M. and Mollevi, C., 2023. A text and GNN based controversy detection method on social media. *World Wide Web*, 26(2), pp.799-825.
- [10] Nti, I.K., Nyarko-Boateng, O. and Aning, J., 2021. Performance of machine learning algorithms with different K values in K-fold cross-validation. *International Journal of Information Technology and Computer Science*, 13(6), pp.61-71.
- [11] AlMahamid, F. and Grolinger, K., 2021, September. Reinforcement learning algorithms: An overview and classification. In *2021 IEEE canadian conference on electrical and computer engineering (CCECE)* (pp. 1-7). IEEE.
- [12] Sharma, T., Mohapatra, A. and Tomar, G., 2020, July. Fuzzy-based DBSCAN algorithm to elect master cluster head and enhance the network lifetime and avoid redundancy in wireless sensor network. In *International Conference on Innovative Computing and Communications: Proceedings of ICICC 2020, Volume 2* (pp. 1031-1042). Singapore: Springer Singapore.