المركز الليبي للمنظومات الإلكترونية
والبرمجيات وبحوث الطيران
Liban Center Fror Electronic Systems
programming and Avlation Research

Journal of Electronic Systems and Programming

Issue: 13 June 2026

Journal of Electronic Systems and Programming Issue: 13 June 2026

Journal of Electronic Systems and Programming

Libyan Center for Electronic Systems, Programming and Aviation Research

Journal of Electronic Systems and Programming

Issue No:13 - 2026

Editorial:
13th Issue – Journal of Electronic Systems and Programming

We are delighted to announce the publication of the 13th issue of the Journal of Electronic Systems and Programming (JESP).

The contributions in this issue come from deferent areas of study. It is extremely important to publish scientific research carried out in the most diverse research environments, to collaborate with the advances of science and society.

Finally, we thank our reviewers and authors for their fundamental contribution to the 13th release of the Journal. We still hope authors could consider JESP to be a place to publish their work.

Editorial Board

Table of Contents:

Anomaly Detection in System Calls Using Recurrent Neural Networks to Identify Malicious System-Call Sequences

Anomaly Detection in System Calls Using Recurrent Neural Networks to Identify Malicious System-Call Sequences

Nuha Omran Abokhadir
Computer Sceince Deprtament
Faculry of science, University of zawiya, Libya
Abo.khdeir@zu.edu.ly

Hind Ibrahim Omer
Computer Sceince Deprtament
Faculry of science, University of zawiya, Libya
Hind.ibrahim@sabu.edu

Abdusamea Omer
Computer Engineering and IT department
Faculry of Engineering , Sabratha University, Libya
abdusamea@gmail.com

Abstract

Host-based intrusion detection remains essential when malicious activity is visible only through interactions with the operating-system kernel. System-call traces capture this behaviour as ordered symbolic sequences, but their variable length and strong context dependence limit detectors that rely on aggregate frequency features alone. This study evaluates a normal-only recurrent language-modelling approach for Linux system-call anomaly detection on the ADFA-LD benchmark. A one-layer LSTM with 64-dimensional embeddings and 128 hidden units was trained on 833 benign training traces to predict the next system call from preceding calls, and each trace was scored by length-normalised negative log-likelihood. Operating thresholds

were calibrated on 4,372 held-out benign validation traces at false-positive budgets of 5%, 2.5%, and 1%, and the detector was then evaluated against 746 attack traces drawn from six attack families. The LSTM achieved ROC-AUC = 0.841 (95% bootstrap CI: 0.824–0.858) and PR-AUC = 0.477 (95% CI: 0.447–0.512), exceeding both the STIDE n-gram baseline (ROC-AUC = 0.827; PR-AUC = 0.382) and a TF-IDF one-class SVM (ROC-AUC = 0.749; PR-AUC = 0.280) on the same evaluation split. At a 1% false-positive budget the LSTM produced precision 0.72 with recall 0.15; relaxing the budget to 5% raised recall to 0.32 at precision 0.52. Per-family analysis revealed substantial variation in detection difficulty, with Hydra FTP (recall 0.25 at 1% FPR) and Web Shell traces the most separable, and Java Meterpreter (recall 0.08 at 1% FPR) the hardest. The findings support recurrent sequence modelling as an effective ranking signal for host-based intrusion detection and make explicit the precision–recall trade-off that any deployment must manage.

Keywords: host-based intrusion detection; system calls; anomaly detection; recurrent neural networks; LSTM; ADFA-LD; Linux security; language modelling.

1. Introduction

Endpoint compromise is often expressed through process behaviour rather than through obvious network signatures. After an attacker gains execution, the host kernel observes file operations, process creation, privilege-related actions, socket activity, and other events through system calls. For this reason, system-call monitoring has remained a central line of work in host-based intrusion detection systems (HIDS), from early short-sequence methods to modern deep sequence models [1]-[6].

The security value of system calls is their ordering. A system call that is normal in one context can be suspicious in another; the local and long-range sequence around the call determines its behavioural meaning. This property makes purely aggregate representations

incomplete. Frequency vectors and n-gram counts are efficient, but they compress a process trace into a fixed representation and can miss the temporal dependencies that separate legitimate execution from malicious payload behaviour.

Recurrent neural networks (RNNs) are a natural alternative for this type of symbolic sequence. In this study, the RNN is not trained as a conventional supervised classifier. It is trained as a normal-behaviour language model: given previous system calls, the model predicts the next call. A trace that is improbable under this learned normal model receives a high anomaly score. This formulation matches a realistic deployment assumption: defenders can usually collect benign traces from their own systems, while the full range of future attacks is unknowable.

This paper reports an empirical study on ADFA-LD, a public Linux system-call benchmark for HIDS evaluation [1]–[3]. The contribution is not the use of an LSTM, which has precedent in the literature [7], but a complete and auditable protocol: normal-only training, validation-based false-positive control, direct comparison with semi-supervised baselines, stratified bootstrap uncertainty for AUC estimates, and family-level reporting of attack detection. The results show that recurrent modelling improves threshold-independent discrimination over both STIDE and one-class SVM baselines, while the fixed-threshold analysis exposes an operational limitation that aggregate accuracy would have hidden: strict false-positive budgets substantially reduce recall.

2. Threat Model and Problem Definition

DDoS The field of bioacoustics has long explored animal communication through sound, but its application to environmental conservation is a more recent innovation. Marine mammals produce a variety of vocalizations that reflect behaviors such as feeding, mating, social bonding, and predator avoidance. Analyzing these

sounds allows scientists to infer ecological patterns and population dynamics. Given the increasing pressures on marine ecosystems, including climate change and human interference, PAM offers an efficient and scalable solution for monitoring biodiversity. It is cost effective compared to visual surveys, and capable of collecting data over long durations and wide geographic areas. Species such as whales, dolphins, and seals are especially well suited for acoustic studies due to their reliance on vocal cues for essential life functions [3].

3. Related Work

3.1 System-call anomaly detection

The idea of detecting intrusions from system-call traces was established by Forrest et al., who argued that normal UNIX processes produce stable short-range system-call patterns and that violations of these patterns can reveal abnormal execution [4]. Warrender et al. later compared alternative sequence models for system-call intrusion detection and showed that the modelling assumption has a direct effect on false alarms and detection rates [5]. These studies introduced the principle that host behaviour can be represented as an ordered trace rather than as isolated events.

STIDE-style detection follows this principle by storing short windows observed during normal training and treating unseen windows as anomalous. Its strengths are simplicity, interpretability, and low computational cost. Its weakness is that the context length is manually fixed; it cannot learn graded probabilities over the next call or share statistical strength across related contexts. This limitation motivates neural sequence models, especially when traces contain repeated but variable execution motifs.

3.2 ADFA-LD and benchmark validity

ADFA-LD was introduced to address limitations of older intrusion-detection datasets whose traffic and attack assumptions no longer represented contemporary systems [1]-[3]. The dataset provides Linux system-call traces grouped into normal training, normal validation, and attack partitions, making it useful for anomaly-detection protocols where benign data are used for training and threshold selection while attacks remain held out for testing. Recent work has also expanded the dataset landscape; DongTing, for example, offers a much larger kernel-oriented system-call corpus for Linux anomaly detection [13]. ADFA-LD remains valuable because it is compact, public, widely used, and compatible with transparent reproducibility.

3.3 Deep sequence models for HIDS

RNN-based HIDS research treats system calls as a behavioural language. Kim et al. proposed LSTM-based system-call language modelling and thresholding strategies for anomaly-based HIDS [7]. Chawla et al. explored recurrent and convolutional combinations, including GRU-based architectures, to improve computational efficiency [8]. Ring et al. later investigated deep learning methods for host-based intrusion detection through sequence prediction and aggregation [9]. These studies indicate that learned sequential representations can improve over handcrafted features, but they also show the need for careful thresholding and evaluation under class imbalance.

The present study extends this direction by emphasizing operationally calibrated thresholds. Instead of selecting a threshold from attack performance, the decision boundary is selected using benign validation scores only. This design is stricter than ordinary supervised testing and makes the false-positive burden explicit.

4. Dataset and Experimental Material

The experiment used the ADFA-LD Linux system-call benchmark in its canonical three-way partitioning. The distribution used in this study contained 833 normal training traces, 4,372 normal validation traces, and 746 attack traces across six labelled families; every trace exceeded the two-token minimum required for next-call scoring, so no trace was discarded at load time. Each trace was read as an ordered list of integer system-call identifiers. The vocabulary was fitted only on the training-normal traces and contained 150 distinct system calls, with two reserved indices for padding and unknown tokens (152 in total). System calls observed only in validation or attack traces were mapped to the unknown token at inference time so that no information from the evaluation splits influenced the vocabulary.

Table 1: ADFA-LD partitions used in the experiment.

Subset	Class	Purpose	Number of traces
Training_Data_Master	Normal	Fit the normal-behaviour language model	833
Validation_Data_Master	Normal	Calibrate thresholds and estimate false-positive rates	4,372
Attack_Data_Master	Attack	Evaluate malicious-sequence detection	746

The attack set comprises six labelled families. Table 2 reports their distribution in the executed experiment. The imbalance among attack families is important because aggregate recall can hide weak detection of smaller or less repetitive attack types.

Table 2: Attack-family distribution in ADFA-LD.

Attack family	Traces	Median RNN anomaly score	Behavioural interpretation
Adduser	91	1.9614	Account and privilege-management behaviour.
Hydra FTP	162	2.3706	Repeated authentication and service-interaction sequences.
Hydra SSH	176	2.4081	Repeated SSH authentication and session-management activity.
Java Meterpreter	124	1.8112	Payload execution and remote-control behaviour.
Meterpreter	75	1.8351	Remote session behaviour with process and file manipulation.
Web Shell	118	2.3813	Command execution through a web-service context.

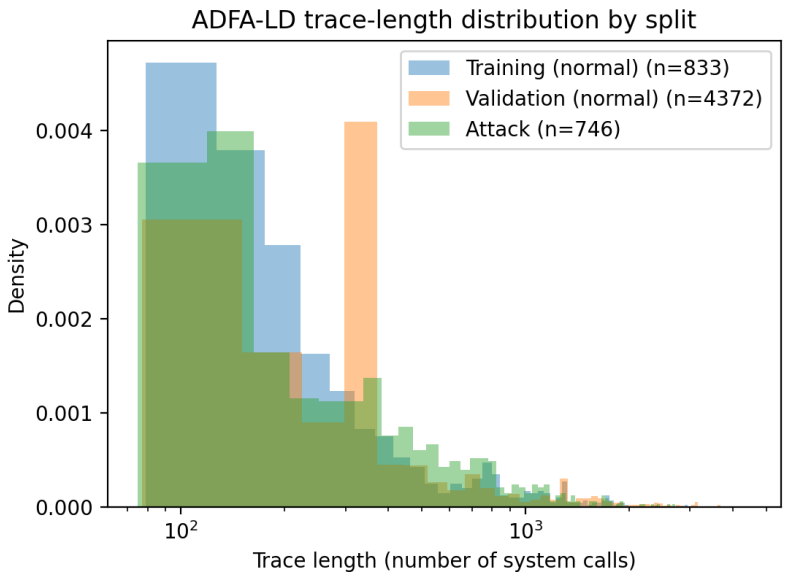


Figure 1: Trace-length distribution by ADFA-LD split. The log-scaled x-axis highlights the strong variability in trace length, which motivates length-normalized scoring.

5. Methodology

5.1 Problem formulation

Let a trace be x = (x1, x2, ..., xT), where each xt is a discrete system-call identifier from vocabulary V. The model estimates the conditional probability p(xt | x1:t-1) from benign traces only. The anomaly score for a complete trace is the mean negative log-likelihood: S(x) = -(1/(T-1)) sum_{t=2}^{T} log p(xt | x1:t-1). Higher scores indicate lower compatibility with normal execution. A trace is labelled anomalous when S(x) >= tau, where tau is selected from benign validation scores at a target false-positive rate.

5.2 Preprocessing and leakage control

The preprocessing pipeline is deliberately conservative. First, trace files are tokenised by extracting integer system-call identifiers in their original order. Second, the vocabulary is fitted only on normal training traces. Third, the language model is trained on sliding windows from the training-normal set. Fourth, the official validation-normal split is held out from optimisation and used only to calibrate alarm thresholds. Fifth, attack labels are used only after training and threshold calibration. For early-stopping decisions, 10% of the training-normal traces (selected with a fixed random seed) were held out from the optimiser as an internal language-model validation set; this internal split is disjoint from the ADFA-LD validation-normal partition, which is reserved exclusively for threshold calibration. Together, these controls prevent the detector from learning the attack distribution and from selecting thresholds that exploit attack labels.

5.3 RNN language model

The primary model is a one-layer LSTM language model [10]. An embedding layer projects each system-call identifier into a dense vector space; the recurrent layer encodes sequence context; dropout [18] regularises the hidden representation; and a linear output layer produces a softmax over the next-call distribution. LSTM cells were chosen because gated memory retains information over longer contexts than a plain recurrent unit. Sliding windows of length 30 with stride 1 were extracted from the training-normal traces, producing 256,257 next-call prediction pairs for optimisation and 26,830 windows for the internal LM-validation set. The implementation also supports GRU cells [11]; the experiment reported here uses the LSTM configuration listed in Table 3.

Table 3. Executed model and training configuration.

Component	Setting	Rationale
Window length	30	Captures local behavioural context while keeping training efficient.
Embedding dimension	64	Compact dense representation for integer system-call identifiers.
Recurrent cell	LSTM	Gated memory for ordered symbolic sequences.
Hidden units	128	Moderate capacity relative to dataset size.
Layers	1	Avoids overfitting in a normal-only training setting.
Dropout	0.3	Regularizes the language model.
Optimizer	Adam, learning rate 0.001	Stable first-order optimization for neural sequence models [17].
Batch size	256	Efficient GPU utilization for windowed training.
Early stopping	Patience = 3 epochs	Stops training when held-out normal loss stops improving.
Bootstrap repetitions	1,000	Estimates uncertainty for AUC metrics.

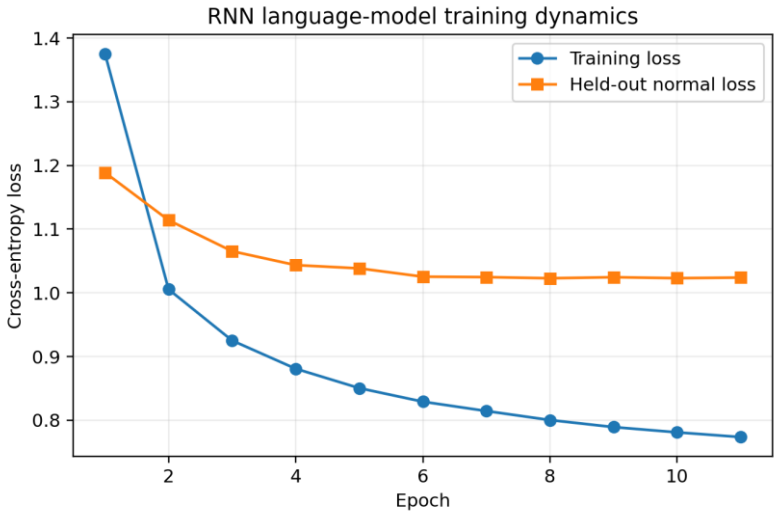


Figure 2. Training and held-out normal validation loss. Early stopping occurred after epoch 11; the best held-out normal loss was 1.0229 at epoch 8.

5.4 Baselines

Two non-neural baselines were evaluated under the same scoring protocol. The first is STIDE, which stores the k-grams observed during normal training (k = 6) and scores a trace by the fraction of its k-grams that are absent from this database. The second is a one-class SVM [12] with an RBF kernel trained on TF-IDF n-gram representations (n = 1 to 3, top 5,000 features) of trace token strings. To prevent any leakage from validation or attack data into the baseline, the TF-IDF vectoriser and the SVM are fitted on training-normal traces only and then reused unchanged to transform and score the validation-normal and attack splits. These baselines test whether the recurrent model adds discriminative power beyond fixed n-gram memory [4], [5] and classical one-class learning [12], [14].

5.5 Metrics and threshold calibration

Threshold-independent performance is reported with ROC-AUC and PR-AUC. ROC-AUC measures score separability across all thresholds, while PR-AUC is more sensitive to false alerts when positive events are uncommon [15]. For deployment-style analysis, thresholds are selected from the validation-normal scores at the 95th, 97.5th, and 99th percentiles, corresponding to false-positive budgets of approximately 5%, 2.5%, and 1%. At each threshold, precision, recall, F1-score, Matthews correlation coefficient, balanced accuracy, false-positive rate, and the confusion-matrix counts are reported. Ninety-five percent confidence intervals for ROC-AUC and PR-AUC are estimated by stratified non-parametric bootstrapping [16] with 1,000 resamples; stratification preserves the positive-to-negative ratio of each resample so that PR-AUC remains defined.

6. Results

6.1 Headline model comparison

The LSTM language model achieved the strongest threshold-independent performance among the evaluated detectors. Its ROC-AUC was 0.841 (95% bootstrap CI: 0.824–0.858) and its PR-AUC was 0.477 (95% CI: 0.447–0.512). STIDE was competitive in ROC-AUC but markedly weaker in PR-AUC, while the one-class SVM trailed both sequence-aware approaches. Sequential context therefore improves the ranking of attack traces relative to fixed-window memory and TF-IDF features, but the residual overlap between benign and malicious score distributions still caps the achievable precision and recall.

Table 4. Threshold-independent model comparison.

Model	ROC-AUC	ROC-AUC 95% CI	PR-AUC	PR-AUC 95% CI
RNN (LSTM)	0.8409	0.8240–0.8578	0.4769	0.4470–0.5125
STIDE	0.8273	-	0.3816	-
One-Class SVM	0.7492	-	0.2797	-

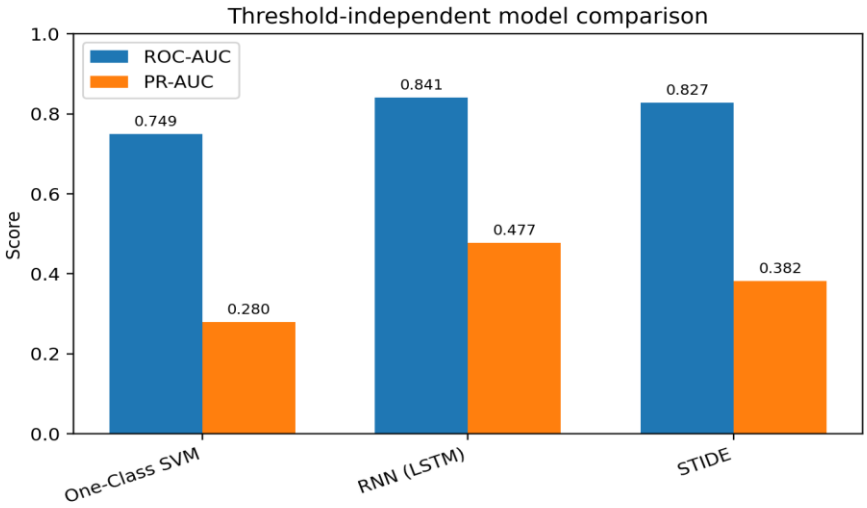


Figure 3: Threshold-independent comparison of the LSTM, STIDE, and one-class SVM baselines. The LSTM achieved the highest ROC-AUC and PR-AUC.

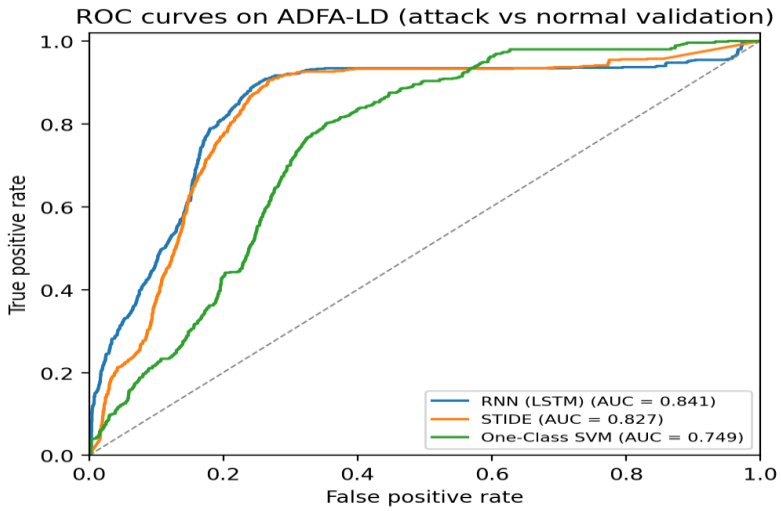


Figure 4. ROC curves for all evaluated detectors on validation-normal versus attack traces.

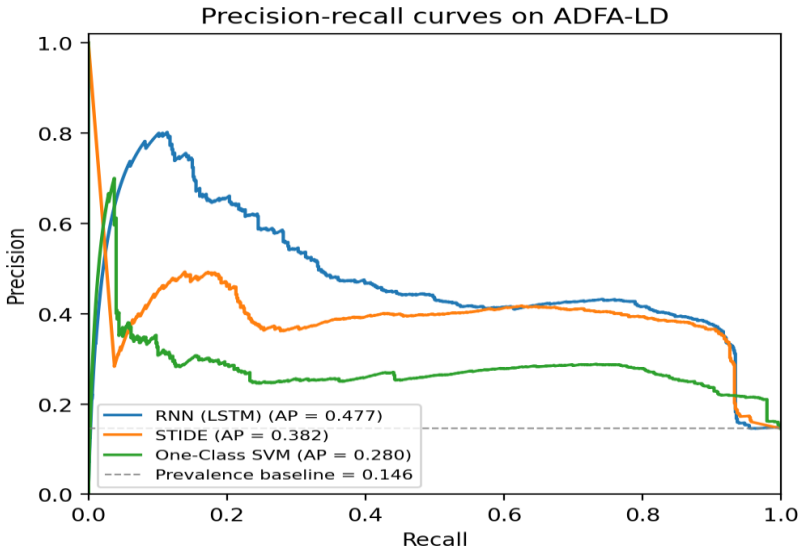


Figure 5. Precision-recall curves for all evaluated detectors. PR-AUC exposes the operational cost of score overlaps more clearly than ROC-AUC alone.

6.2 Threshold-calibrated operating points

The threshold analysis gives a more operational view than AUC alone. At the 5% false-positive budget the LSTM detected 240 of 746 attacks with precision 0.523 and recall 0.322. At the stricter 1% budget, precision rose to 0.720, but recall fell to 0.151. The model therefore produces relatively reliable alerts under strict thresholding but misses many attacks unless a higher false-positive rate is accepted. This precision–recall trade-off is the central operational finding of the experiment.

Table 5. LSTM operating points under validation-calibrated thresholds.

Threshold	Tau	TP	FP	TN	FN	Precision	Recall	F1	MCC	FPR
FPR 5%	2.7925	240	219	4153	506	0.5229	0.3217	0.3983	0.3354	0.0501
FPR 2.5%	3.1372	178	110	4262	568	0.6181	0.2386	0.3443	0.3268	0.0252
FPR 1%	3.9374	113	44	4328	633	0.7197	0.1515	0.2503	0.2894	0.0101

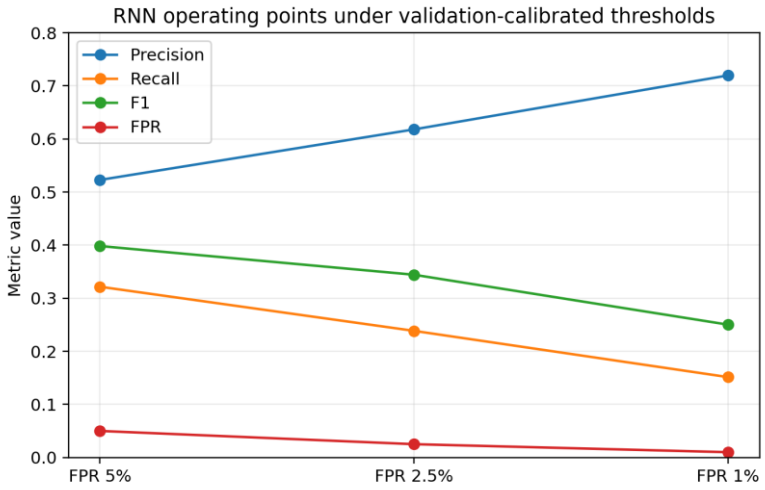


Figure 6. Precision, recall, F1-score, and false-positive rate at three validation-calibrated thresholds. Stricter false-positive budgets improve precision but reduce detection rate.

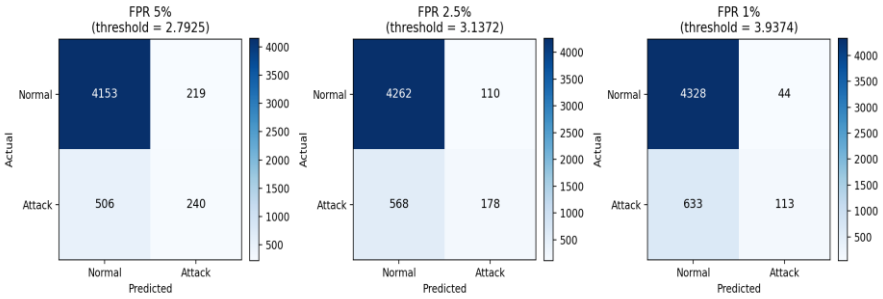


Figure 7. Confusion matrices for the LSTM at 5%, 2.5%, and 1% validation-calibrated false-positive budgets.

6.3 Score-distribution analysis

The shape of the anomaly-score distribution explains the observed trade-off. Attack traces are shifted toward higher negative log-likelihood scores, but the benign and attack distributions overlap in the moderate-score region. This overlap is consistent with the

structure of host-based attacks: some malicious traces reuse benign execution paths, and many contain only short anomalous segments embedded within otherwise legitimate sequences. The same overlap explains why a strong ROC-AUC can coexist with limited recall at a strict false-positive budget.

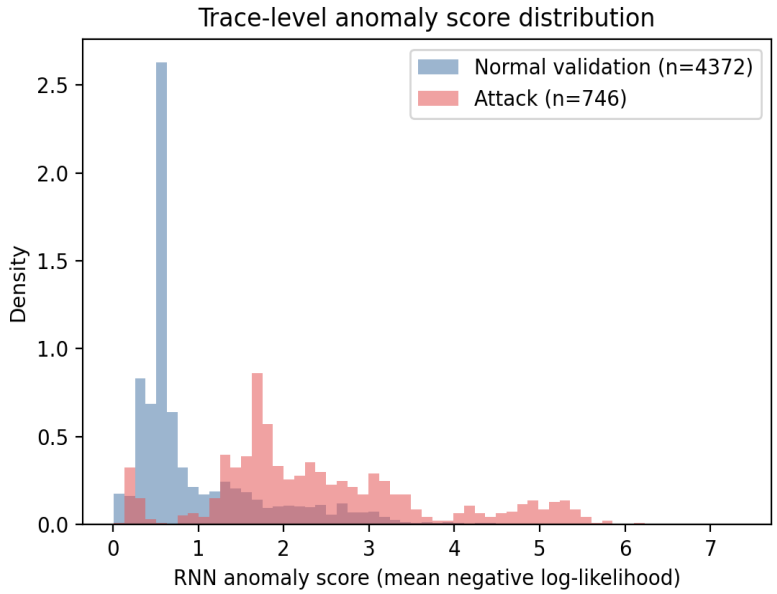


Figure 8. Distribution of LSTM anomaly scores for validation-normal and attack traces. Overlap between the distributions determines the achievable operating trade-off.

6.4 Family-level detection behaviour

The per-family results show that aggregate detection performance is not uniform across attack types. At the 5% false-positive budget, Web Shell and Hydra FTP were the most detectable families, with recall values of 0.4407 and 0.3951, respectively. At the strict 1% budget, Hydra FTP remained the most detectable family with recall of 0.2531. Java Meterpreter was the most difficult family under all three thresholds, falling to 0.0806 recall at the 1% budget. These differences

are important for deployment because a detector that performs well on repeated authentication attacks may still require complementary sensors for stealthier payload behaviours.

Table 6: LSTM recall by attack family.

Family	Traces	Median score	Recall @ 5% FPR	Recall @ 2.5% FPR	Recall @ 1% FPR
Adduser	91	1.9614	0.2418	0.1978	0.1209
Hydra FTP	162	2.3706	0.3951	0.3395	0.2531
Hydra SSH	176	2.4081	0.3352	0.2159	0.1705
Java Meterpreter	124	1.8112	0.1855	0.1694	0.0806
Meterpreter	75	1.8351	0.2667	0.2533	0.1333
Web Shell	118	2.3813	0.4407	0.2288	0.0932

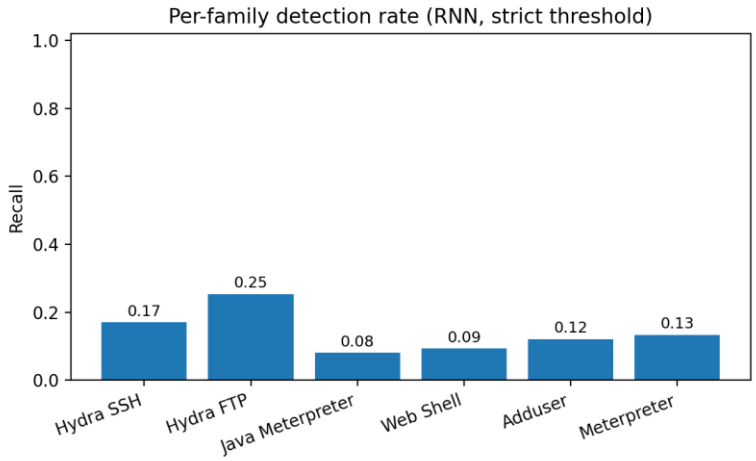


Figure 9. Per-family recall at the strictest available threshold. The plot highlights substantial differences in detectability across attack families.

6.5 Baseline operating-point comparison

The fixed-threshold comparison confirms the value of the recurrent model. At matched validation-derived operating points, the LSTM consistently achieved higher recall and F1-score than STIDE and the one-class SVM. STIDE's ROC-AUC is strong because many attack traces contain previously unseen k-grams, but its strict-threshold recall collapses at the 1% budget, falling below 0.04. The one-class SVM produced both lower PR-AUC and weaker thresholded recall, indicating that TF-IDF n-gram features alone are less effective than learned next-call distributions for ranking attack traces on this split.

Table 7: Baseline comparison at fixed false-positive budgets.

Model	Threshold	Precision	Recall	F1	MCC	Balanced accuracy	FPR	TP/FN
RNN (LSTM)	FPR 5%	0.5229	0.3217	0.3983	0.3354	0.6358	0.0501	240/506
RNN (LSTM)	FPR 2.5%	0.6181	0.2386	0.3443	0.3268	0.6067	0.0252	178/568
RNN (LSTM)	FPR 1%	0.7197	0.1515	0.2503	0.2894	0.5707	0.0101	113/633
STIDE	FPR 5%	0.4267	0.2185	0.2890	0.2261	0.5842	0.0501	163/583
STIDE	FPR 2.5%	0.4860	0.1394	0.2167	0.2014	0.5571	0.0252	104/642
STIDE	FPR 1%	0.2887	0.0375	0.0664	0.0563	0.5109	0.0158	28/718
One-Class SVM	FPR 5%	0.2958	0.1233	0.1741	0.1082	0.5366	0.0501	92/654
One-Class SVM	FPR 2.5%	0.3373	0.0751	0.1228	0.0994	0.5250	0.0252	56/690

One-Class SVM	FPR 1%	0.4054	0.0402	0.0732	0.0891	0.5151	0.0101	30/716

7. Discussion

The principal empirical finding is that a normal-only LSTM language model ranks malicious system-call traces more effectively than fixed n-gram memory or a TF-IDF one-class SVM on the same ADFA-LD split. The improvement is visible in both ROC-AUC and PR-AUC, indicating that learned temporal context adds information beyond the mere presence of unseen windows. This finding supports treating system-call traces as ordered behavioural language rather than as unordered event collections.

The second finding is more cautious but equally important: high score separability does not automatically translate into high recall under strict false-positive constraints. The 1% threshold produced precision above 0.71, meaning that alerts at this operating point were comparatively selective; however, recall fell to approximately 0.15. In a security operations context, this trade-off is expected. HIDS deployments with low alert tolerance may use the model as a high-confidence signal within a broader correlation pipeline, while deployments with greater analyst capacity may choose a 5% threshold to recover more attacks.

The family-level analysis carries practical implications. Repetitive authentication families such as Hydra FTP and Hydra SSH produce sequence regularities that the LSTM separates well, whereas payload-oriented families such as Java Meterpreter are harder to flag at the trace level. Future detectors should therefore combine trace-level language scores with local segment attribution, process and parent–child metadata, and time-aware features; a single scalar score for an entire trace can dilute short malicious fragments embedded within long benign-looking sequences.

From a methodological standpoint, the strongest feature of the study is the separation of roles among the dataset partitions. Training uses only normal training traces. Thresholding uses only validation-normal traces. Attack labels enter only at evaluation time. This regime is stricter than a supervised classifier trained on known attack categories and more closely reflects the detection of previously unseen attacks. It also makes the reported false-positive budgets meaningful, since they are never tuned against attack performance.

8. Threats to Validity and Limitations

The study has four main limitations. First, ADFA-LD is compact relative to modern enterprise host telemetry. Its value is reproducibility, not exhaustive coverage of contemporary attack chains. External validation on larger system-call corpora such as DongTing would strengthen generality [13]. Second, the experiment reports trace-level detection. Segment-level localization would be needed to tell analysts which part of a trace triggered the alert. Third, the LSTM is normal-only and therefore depends on the representativeness of benign training traces. Benign drift across software versions, workloads, or host roles can increase false positives if thresholds are not periodically recalibrated. Fourth, the reported baselines are strong enough to test classical n-gram and one-class learning, but they do not exhaust all possible comparators; future work should include temporal convolutional networks, transformer encoders, and hybrid models under the same validation-calibrated protocol.

These limitations do not invalidate the findings; they bound their interpretation. The LSTM presented here is a reproducible anomaly detector on ADFA-LD and a credible building block for HIDS research, but additional telemetry and operational data would be required before any claim of a production-ready intrusion-detection system.

9. Conclusion

This paper evaluated a recurrent neural language model for the detection of malicious Linux system-call sequences. Trained only on benign traces, the LSTM achieved ROC-AUC = 0.841 and PR-AUC = 0.477, exceeding both the STIDE n-gram baseline and a TF-IDF one-class SVM on the same ADFA-LD evaluation split. Threshold-calibrated results showed that the detector can produce high-precision alerts under strict false-positive budgets, but with reduced recall. Family-level analysis exposed substantial variation across attack types, emphasising the need for finer-grained explanation and complementary sensors. Taken together, the results support RNN-based normal-behaviour modelling as a credible component of host-based anomaly detection, particularly when evaluation enforces validation-calibrated false-positive control and reports per-family behaviour alongside aggregate metrics.

- **Data and Code Availability**

The ADFA IDS datasets are publicly available for academic research from UNSW [1]. The implementation used in this study is a Python/PyTorch script (run_syscall_rnn_experiment.py) that loads ADFA-LD, trains the RNN language model, computes the STIDE and one-class SVM baselines, exports all metrics as CSV files, and generates the figures included in this manuscript. Training stopped automatically at epoch 11 of a 20-epoch budget under an early-stopping patience of three epochs on the internal LM-validation loss, and the full pipeline — training, scoring, and baseline evaluation — completed in approximately 1.5 minutes on a single NVIDIA Tesla T4 GPU. The reported run used Python 3.12.12, PyTorch 2.10.0+cu128, scikit-learn 1.6.1, NumPy 2.0.2, and pandas 2.3.3, with a fixed random seed of 42 for full reproducibility.

- **Ethics Statement**

This study uses a public benchmark dataset and does not involve human participants, personal data, or live system exploitation. The work is intended for defensive host-based intrusion detection research.

- **Conflict of Interest**

The author(s) declare no conflict of interest. Funding information, if applicable, should be inserted according to the target journal requirements.

References

[1] UNSW Research, "ADFA IDS Datasets," University of New South Wales, 2013. Available: https://research.unsw.edu.au/projects/adfa-ids-datasets

[2] G. Creech and J. Hu, "Generation of a new IDS test dataset: Time to retire the KDD collection," in Proc. IEEE Wireless Communications and Networking Conference (WCNC), 2013, pp. 4487-4492, doi: 10.1109/WCNC.2013.6555301.

[3] G. Creech and J. Hu, "A semantic approach to host-based intrusion detection systems using contiguous and discontiguous system call patterns," IEEE Transactions on Computers, vol. 63, no. 4, pp. 807-819, 2014, doi: 10.1109/TC.2013.13.

[4] S. Forrest, S. A. Hofmeyr, A. Somayaji, and T. A. Longstaff, "A sense of self for Unix processes," in Proc. IEEE Symposium on Security and Privacy, 1996, pp. 120-128.

[5] C. Warrender, S. Forrest, and B. Pearlmutter, "Detecting intrusions using system calls: Alternative data models," in Proc. IEEE Symposium on Security and Privacy, 1999, pp. 133-145, doi: 10.1109/SECPRI.1999.766910.

[6] R. A. Bridges, T. R. Glass-Vanderlan, M. D. Iannacone, M. S. Vincent, and Q. Chen, "A survey of intrusion detection systems leveraging host data," ACM Computing Surveys, vol. 52, no. 6, 2019, doi: 10.1145/3344382.

[7] G. Kim, H. Yi, J. Lee, Y. Paek, and S. Yoon, "LSTM-based system-call language modeling and robust ensemble method for designing host-based intrusion detection systems," arXiv:1611.01726, 2016.

[8] A. Chawla, B. Lee, S. Fallon, and P. Jacob, "Host based intrusion detection system with combined CNN/RNN model," in Proc. Joint European Conference on Machine Learning and Knowledge

Discovery in Databases Workshops, 2018, pp. 149-158, doi: 10.1007/978-3-030-13453-2_12.

[9] M. Ring, S. Wunderlich, D. Scheuring, D. Landes, and A. Hotho, “Methods for host-based intrusion detection with deep learning,” ACM Transactions on Privacy and Security, vol. 24, no. 3, 2021, doi: 10.1145/3461462.

[10] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” Neural Computation, vol. 9, no. 8, pp. 1735-1780, 1997, doi: 10.1162/neco.1997.9.8.1735.

[11] K. Cho, B. van Merrienboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, “Learning phrase representations using RNN encoder-decoder for statistical machine translation,” in Proc. EMNLP, 2014, pp. 1724-1734, doi: 10.3115/v1/D14-1179.

[12] B. Scholkopf, J. C. Platt, J. Shawe-Taylor, A. J. Smola, and R. C. Williamson, “Estimating the support of a high-dimensional distribution,” Neural Computation, vol. 13, no. 7, pp. 1443-1471, 2001, doi: 10.1162/089976601750264965.

[13] G. Duan, Y. Fu, M. Cai, H. Chen, and J. Sun, “DongTing: A large-scale dataset for anomaly detection of the Linux kernel,” Journal of Systems and Software, vol. 203, 2023, article 111745, doi: 10.1016/j.jss.2023.111745.

[14] V. Chandola, A. Banerjee, and V. Kumar, “Anomaly detection: A survey,” ACM Computing Surveys, vol. 41, no. 3, 2009, doi: 10.1145/1541880.1541882.

[15] J. Davis and M. Goadrich, “The relationship between precision-recall and ROC curves,” in Proc. 23rd International Conference on Machine Learning (ICML), 2006, pp. 233-240, doi: 10.1145/1143844.1143874.

[16] B. Efron and R. J. Tibshirani, An Introduction to the Bootstrap. Boca Raton, FL, USA: Chapman and Hall/CRC, 1994.

[17] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” in Proc. International Conference on Learning Representations (ICLR), 2015.

[18] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, “Dropout: A simple way to prevent neural networks from overfitting,” Journal of Machine Learning Research, vol. 15, pp. 1929-1958, 2014.

[19] F. Pedregosa et al., “Scikit-learn: Machine learning in Python,” Journal of Machine Learning Research, vol. 12, pp. 2825-2830, 2011.

[20] A. Paszke et al., “PyTorch: An imperative style, high-performance deep learning library,” in Advances in Neural Information Processing Systems, vol. 32

2

Adaptive Behavior-Aware Sidecar Proxy for Microservices Security and Performance

Adaptive Behavior-Aware Sidecar Proxy for Microservices Security and Performance

Ebtihal Al-Ajeeli Ali Al-Mrabet
University of Azzawia, Azzawia, Libya
Faculty of Science
Department of Computer Science,
Email: booto199924@gmail.com

Noha Omran
University of Azzawia, Azzawia, Libya
Faculty of Science
Department of Computer Science,
Email: abo.khdeir@zu.edu.ly

Abstract

Service mesh architectures based on sidecar proxies are widely used in modern microservices systems to manage communication, security, and traffic control. However, existing approaches typically treat routing, caching, load management, and security enforcement as separate and static mechanisms, which leads to performance overhead and limited adaptability under dynamic workload conditions. This paper addresses these limitations by proposing an Adaptive Behavior-Aware Security Sidecar Proxy Architecture that integrates load-aware routing, adaptive caching, and lightweight anomaly detection within a unified decision-making engine. The proposed model evaluates incoming requests based on system load, request repetition patterns, and a computed risk score to dynamically determine the appropriate handling strategy. The system was implemented using simulation environment that generates dynamic workloads, including random requests, variable system load levels, and repeated traffic patterns to emulate realistic microservice behavior. Based on these inputs, the sidecar proxy applies direct routing under low-load conditions,

activates caching under moderate to high load or repeated requests, and switches to a secure mode when suspicious behavior is detected. The evaluation results demonstrate that the proposed system achieves improved performance and security behavior. The caching mechanism achieved a 64.8%hit ratio, reducing redundant processing and improving response efficiency. Additionally, The system was able to detect and block simulated attack-like behaviors under high load conditions, indicating potential effectiveness in anomaly detection and security enforcement. Overall, the results suggest reduced backend workload, improved resource utilization, and consistent adaptive decision-making under varying conditions.
The proposed system was evaluated using simulation model rather than deployment in a real Kubernetes environment.
In conclusion, the proposed architecture enhances both performance efficiency and behavioral awareness in sidecar-based systems, making it suitable for dynamic and security-sensitive microservices environments.

Keywords: Sidecar Proxy, Adaptive Routing, Behavior-Aware Security, Microservices

1. Introduction

Modern microservices architectures rely heavily on Service Mesh frameworks to manage communication, security, and traffic control between distributed services [1],[2],[3]. Among the core components of service mesh systems, Sidecar Proxies play a critical role in intercepting and managing service-to-service communication. Despite their widespread adoption in platforms such as Kubernetes-based environments, sidecar-based designs often introduce performance overhead [4],[5] and exhibit limited adaptability under dynamic workload conditions[6].
In current implementations, key responsibilities such as routing, caching, load management, and security enforcement are typically handled as independent and static modules[7],[8]. This separation leads to inefficient resource utilization, increased latency under high load, and limited responsiveness to behavioral changes in traffic

patterns[3], [9]. Moreover, existing solutions lack a unified mechanism that allows sidecars to dynamically adapt their behavior based on real-time system conditions and request-level semantics.

To address these limitations, this paper identifies a critical gap in existing literature: the absence of an integrated adaptive decision-making mechanism within the Sidecar layer that jointly considers system load, request behavior, and security risk in real time.

This work proposes an Adaptive Behavior-Aware Security Sidecar Proxy Architecture with Adaptive Caching, which introduces a unified decision engine inside the sidecar. The proposed system dynamically evaluates incoming requests using multiple factors, including system load, request repetition frequency, and a lightweight risk scoring mechanism. Based on this evaluation, the sidecar adaptively selects one of three execution paths: direct routing, cache-based processing, or secure mode activation.

The main contribution of this paper is the design and implementation of a behavior-aware adaptive sidecar mechanism that integrates routing, caching, and security enforcement into a single decision logic. In addition, a C++ simulation-based evaluation framework is developed to analyze system behavior under varying workloads and traffic patterns. The results demonstrate improved performance efficiency, reduced backend overhead, and enhanced anomaly detection capability compared to traditional static sidecar approaches.

2. Related work

Research on service mesh and sidecar proxies spans multiple dimensions including scheduling, performance overhead, security enforcement, and architectural comparisons. However, these efforts remain fragmented across isolated perspectives, where each study focuses on a single concern without integrating sidecar-internal adaptive decision logic.

explores sidecars as decentralized schedulers in cloud–edge environments[6]. It demonstrates latency and resource utilization improvements through sidecar-based scheduling, but does not examine cache behavior, policy execution overhead, or adaptive security decisions inside the sidecar.

evaluates the overhead of sidecars in serverless systems with frequent cold and warm starts[4]. While it proposes lightweight concepts and quantifies CPU and latency costs, it lacks microarchitectural analysis and does not present an internal adaptive mitigation mechanism.

A seminal microarchitectural investigation is presented ,where L2 cache misses and pipeline stalls are measured to show how complex policy execution degrades performance[5]. Despite its depth, the work stops at measurement and does not propose an architectural solution to adapt sidecar behavior at runtime.

From a benchmarking perspective, compare frameworks such as Istio, Linkerd, and Cilium under mTLS overhead[1],[2]. These works rely on high-level metrics (latency, throughput, CPU) and do not analyze how sidecar-internal decisions influence performance.

Security-focused contributions include a Proxy-based approach for mTLS, JWT, and RBAC enforcement[7], which integrates multiple security mechanisms into a unified proxy for gRPC systems. However, security enforcement is treated independently from routing and caching behavior.

Similarly, introduces adaptive trust based on vulnerabilities[8], but does not relate this trust model to sidecar performance or cache behavior.

According to NIST guidelines, service mesh architectures use sidecar proxies to enforce secure communication and traffic control between Microservices in Kubernetes clusters[3]

Finally, studies performance–security trade-offs and protocol parsing overhead[9], offering tuning insights but without correlating runtime load, request behavior, and adaptive sidecar control.

architectural evaluation, they treat routing, caching, and security as independent mechanisms. None of the reviewed works demonstrates how a sidecar proxy can internally integrate load-aware routing decisions, caching behavior, anomaly detection based on request repetition, and dynamic risk-based security activation within a unified control logic.

In particular, existing approaches do not illustrate a closed adaptive loop inside the sidecar where system load influences routing decisions, routing impacts cache usage, request behavior contributes to risk scoring, and security enforcement dynamically affects cache

validity and subsequent routing paths. This lack of integration highlights the absence of a behavior-aware and risk-aware adaptive sidecar model that simultaneously optimizes performance and security under varying workloads.

This limitation directly motivates the proposed work, which introduces an adaptive sidecar simulation where routing, caching, and security decisions are unified into a single dynamic decision mechanism within the sidecar proxy layer.

Table 1: Feature comparison between traditional sidecar

Feature	Traditional Sidecar	Proposed Sidecar
Static Routing	✓	✗
Adaptive Routing	✗	✓
Caching Mechanism	✗	✓
Security Monitoring	✗	✓
Behavior Awareness	✗	✓

The table highlights the key differences between traditional sidecar architectures and the proposed adaptive sidecar system. It shows that the proposed approach introduces adaptive routing, caching, security monitoring, and behavior awareness, which are not supported in traditional implementations.

Table 2: Comparative Analysis of Existing Studies and the Proposed System

Study	Focus	Routing	Caching	Security	Adaptive Behavior
[1] Sidecars as Decentralized Schedulers	Sidecar scheduling in cloud–edge	✓	✗	✗	✗

[2] Serverless Sidecar Overhead	Cold/warm start performance cost	✓	✗	✗	✗
[3]Microarchitectural Analysis (Cache & Pipeline)	CPU cache misses & stalls analysis	✗	✗	✗	✗
[4–5] Istio / Linkerd / Cilium Benchmarking	mTLS overhead & performance comparison	✓	✗	✓	✗
[6] Proxy-based Security (mTLS, JWT, RBAC)	Unified security enforcement	✓	✗	✓	✗
[7] Adaptive Trust Model	Security trust scoring	✗	✗	✓	✗
[8] NIST Service Mesh Guidelines	Secure communication in microservices	✓	✗	✓	✗
[9] Performance–Security Trade-offs	Protocol parsing & overhead tuning	✓	✗	✓	✗
Proposed System	Adaptive sidecar (routing + caching + security)	✓	✓	✓	✓

The comparison illustrates that existing studies address isolated aspects such as routing, caching, or security independently. In

contrast, the proposed system integrates all these components into a unified adaptive framework that responds dynamically to system behavior.

3. Methodology

3.1 System Overview

This methodology targets microservices and edge-aware environments This methodology targets microservices and edge-computing environments where dynamic workload variations and internal traffic behavior significantly affect system performance, resource utilization, and security. The proposed system is based on a Behavior-Aware Security Sidecar Proxy in which all incoming and internal service requests are intercepted and evaluated before reaching the application layer.

Unlike traditional sidecar architectures that primarily focus on routing and encryption policies, the proposed sidecar incorporates an internal adaptive decision-making engine that continuously evaluates system load conditions, request repetition patterns, estimated latency impact, and a combined risk score derived from multiple behavioral indicators. Based on this continuous evaluation, the system is able to proactively identify abnormal traffic patterns and dynamically adjust its routing, caching, and security behaviors before requests are forwarded to backend services.

The main objective of this design is to integrate performance optimization and proactive security enforcement within a unified sidecar communication layer. By doing so, the system improves response time, enhances resource utilization, and reduces backend processing overhead while also enabling the detection of abnormal traffic behavior. In addition, it prevents suspicious requests from polluting cache storage and allows the system to dynamically switch into a secure mode under high-risk conditions.

3.2 System Architecture

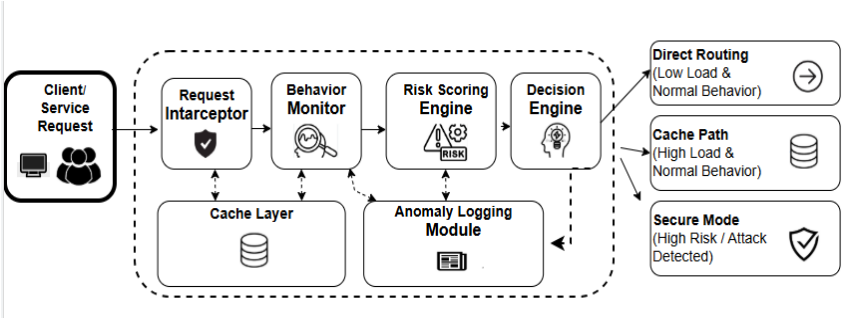


Figure 1: Load-Aware Behavior-Aware Sidecar Architecture

3.3 Core Idea

The core idea of the proposed system is to transform the sidecar proxy from a static request-handling component into an adaptive, behavior-aware decision-making entity. Instead of relying only on predefined load thresholds, the sidecar evaluates real-time behavioral patterns and a computed risk score before deciding how each request should be processed.

The decision strategy follows a dynamic model where normal behavior under low load leads to direct routing, while normal behavior under high load activates the caching path to improve efficiency. In contrast, suspicious behavior or a high-risk score triggers secure mode activation, which may include request blocking or additional security enforcement. This transforms the sidecar into a proactive component capable of continuously adapting its behavior according to system conditions.

3.4 System Components

This methodology targets microservices and edge-computing environments where dynamic workload variations and internal traffic behavior significantly affect system performance, resource utilization, and security. The proposed system is based on a Behavior-Aware Security Sidecar Proxy in which all incoming and internal service

requests are intercepted and evaluated before reaching the application layer.

Unlike traditional sidecar architectures that primarily focus on routing and encryption policies, the proposed sidecar incorporates an internal adaptive decision-making engine that continuously evaluates system load conditions, request repetition patterns, estimated latency impact, and a combined risk score derived from behavioral indicators. This continuous evaluation enables the system to proactively identify abnormal traffic patterns and dynamically adjust routing, caching, and security behaviors before requests are forwarded to backend services.

The main objective of this design is to integrate performance optimization and proactive security enforcement within a unified sidecar communication layer, improving response time, enhancing resource utilization, reducing backend processing overhead, detecting abnormal traffic behavior, preventing cache pollution, and enabling dynamic switching to secure mode under high-risk conditions.

3.5 Execution Flow

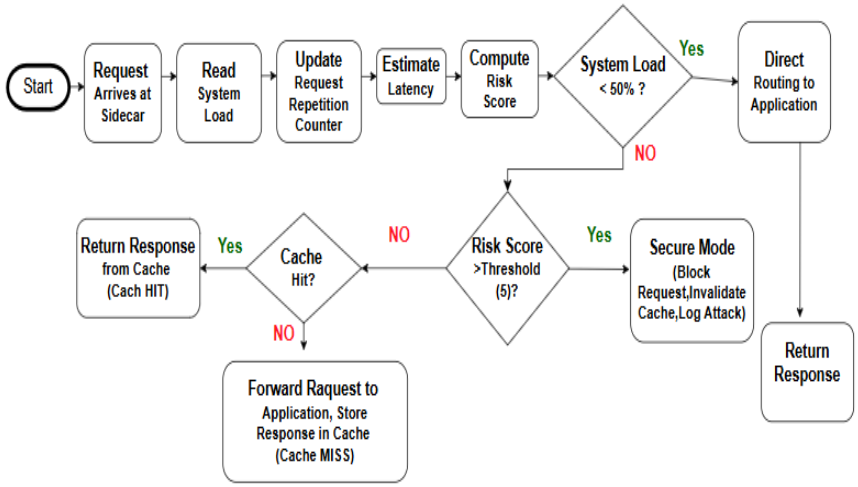


Figure 2: System Flow of Adaptive Sidecar Decision

3.6 Decision Logic

Table 3: Adaptive Decision Logic of the Proposed Sidecar System

Condition	Decision
Load < 50% and normal behavior	Direct routing
Load > 50% and normal behavior	Cache lookup
Risk Score >= threshold	Secure mode (block request)

The table describes the multi-factor decision-making mechanism used in the proposed sidecar system. The decision is based on system load, behavioral patterns, and risk score to dynamically select between direct routing, caching, or secure mode.

Risk Score is calculated as follows:

$$\text{Risk Score} = \alpha L + \beta R + \gamma(1/(T+1))$$

where L represents system load, R represents request repetition, and T represents latency. The coefficients α, β, and γ are tuning weights used to balance system behavior.

Algorithm 1: Adaptive Sidecar Decision Logic

Input: System Load, Request Behavior, Risk Score
Output: Routing Decision

```
Monitor incoming request
Evaluate system load
Analyze request behavior
Compute risk score
  if Load < 50% AND behavior is normal then
                Route request via direct path
 else if Load > 50% AND behavior is normal then
                Serve response from cache
```

```
9  else if Risk Score ≥ Threshold then
10                 Activate secure mode and block the request
11 END IF
12 Return decision
```

3.7 Cache Behavior

Cache is activated only after behavior verification [7]. Cache HIT results in returning the response directly, while Cache MISS leads to generating a new response and storing it in the cache. In cases where attack or suspicious behavior is detected, the corresponding cache entry is invalidated [8], ensuring that malicious or abnormal requests do not influence cached data or contaminate future responses.

3.8 Key Improvement Over Previous Approaches

The proposed system introduces a unified adaptive sidecar architecture that integrates routing, caching, and security mechanisms within a single behavior-aware decision-making framework. Unlike traditional approaches that treat these components independently, the proposed model enables continuous interaction between system load, request behavior, cache management, and security enforcement. This results in a dynamic and risk-aware mechanism where routing decisions and cache operations are continuously adjusted based on runtime conditions. Furthermore, the system extends conventional sidecar designs by incorporating lightweight anomaly detection and enabling secure mode activation under high-risk scenarios. The implementation is evaluated through a C++ simulation that models Kubernetes-like environments for performance and behavior analysis.

4. Implementation

The proposed system was implemented in C++ as a simulation of a Behavior-Aware Security Sidecar Proxy with adaptive caching, designed to analyze system behavior under dynamic and non-

stationary workload conditions. The simulation generates randomized request patterns, fluctuating system load values ranging from 0 to 100, and repetitive traffic behaviors to emulate realistic microservice environments.

Each incoming request is processed through an internal decision mechanism that evaluates system load, request repetition frequency, and a computed risk score derived from behavioral indicators. Based on this evaluation, the system dynamically selects between direct routing, cache-based processing, or secure mode activation. Under normal behavior and low system load, requests are routed directly to the application layer. When system load increases or repetitive patterns are detected, the caching mechanism is activated, where cache hits return responses immediately, while cache misses result in response generation and storage. In contrast, suspicious behavior characterized by high repetition combined with elevated load triggers secure mode, where requests are logged, security alerts are generated, and the request is blocked from reaching the application layer.

The caching and tracking mechanisms are implemented using in-memory data structures to store responses and monitor request frequency, enabling lightweight anomaly detection based on access patterns.

The system relies on standard C++ libraries for random workload generation, dynamic response creation, and runtime logging, providing visibility into routing decisions, cache performance, and security events during execution.

The simulation is evaluated in a controlled environment that mimics microservice traffic behavior under varying conditions, including low-load stability, high-load stress, and attack-like scenarios. To ensure isolation and reproducibility, the environment is containerized using a lightweight runtime suitable for edge and microservice-based architectures.

5. Evaluation and Results

The proposed system was evaluated using runtime simulation outputs generated by the Behavior-Aware Security Sidecar Proxy with Adaptive Caching under varying system load conditions. The

evaluation focuses on routing behavior, caching efficiency, and security performance.

Table 4: System Behavioral Summary

Category	Count	Description
Direct Requests	500	Requests processed directly under low system load conditions
Cache Path Activation	250	Requests redirected to caching layer under increased load
Detected Attacks	250	Requests identified as suspicious and blocked under Secure Mode

Table 5: Cache Performance Metrics

Metric	Value	Description
Cache Hits	162	Requests successfully served from cache
Cache Misses	88	Requests not found in cache and newly stored
Cache Hit Ratio	64.8%	Percentage of successful cache reuse

Table 6: Security Evaluation Results

Condition	System Behavior	Outcome
Normal Load (<70%)	Direct routing or caching	Stable operation
High Load (>80%)	Cache + monitoring activated	Reduced backend stress
Suspicious Behavior (88–92%)	Secure Mode activated	Requests blocked and logged

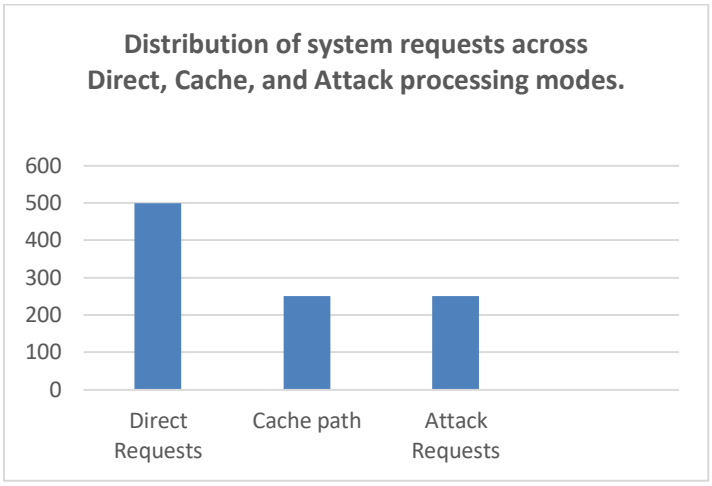


Figure 3: Distribution of system requests across Direct, Cache, and Attack processing modes.

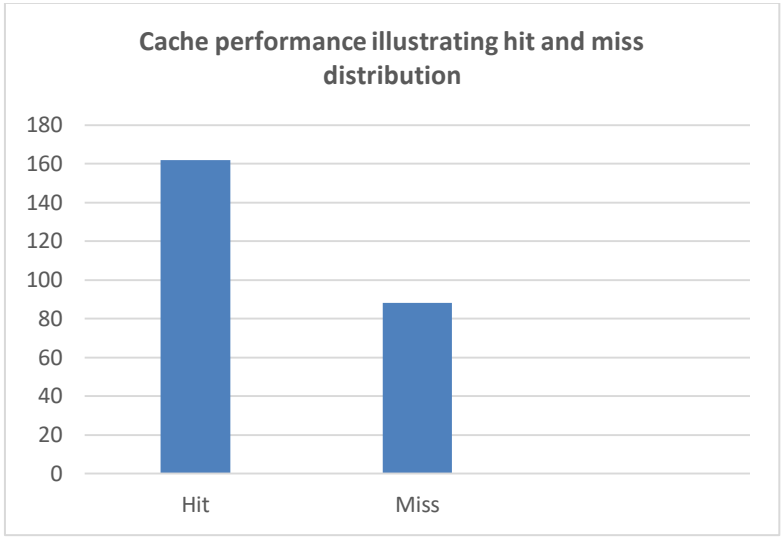


Figure 4: Cache performance illustrating hit and miss distribution, showing a hit ratio of 64%.

5.1 Analysis of Results

The simulation results indicate that the proposed Adaptive Sidecar System dynamically adapts to workload variations by switching between direct routing, caching, and security enforcement modes. Under low-load conditions, requests are processed directly by the application, ensuring minimal latency and fast response times. As system load increases, the caching mechanism becomes active, improving efficiency by reusing previously stored responses and reducing redundant computations.

The cache performance demonstrates a hit ratio of 64%**,** which reflects a balanced utilization of cached data while maintaining a reasonable level of cache misses. This indicates that the system avoids over-reliance on caching and maintains realistic operational behavior under dynamic workload conditions.

From a security perspective, the system successfully detected and blocked malicious or attack-like behaviors under high-load conditions. When the system load exceeded predefined thresholds, the sidecar component transitioned into Secure Mode, ensuring protection against abnormal traffic patterns and preventing potential system overload.

Overall, the results confirm that the system achieves a balanced trade-off between performance optimization, caching efficiency, and security enforcement in a dynamic environment.

6. Conclusion and Future Work

This paper investigated the performance and behavioral limitations of sidecar proxy-based architectures in microservices and edge-like environments, with a particular focus on inefficiencies introduced by static routing and non-adaptive mechanisms. The literature review demonstrated that existing approaches generally treat routing, caching, load handling, and security monitoring as independent components, which limits system adaptability under dynamic workloads and leads to suboptimal performance in high-traffic scenarios.

To address these limitations, an Adaptive Behavior-Aware Sidecar Proxy Architecture was proposed and implemented using a C++ simulation model. The system introduces a dynamic decision-making mechanism within the sidecar layer based on real-time runtime

conditions, including system load, request repetition frequency, and lightweight risk evaluation. Depending on these conditions, the system applies direct routing under low-load scenarios, activates caching under moderate or high load or repeated request patterns, and enforces security measures when suspicious or abnormal behavior is detected.

The evaluation results indicate that the proposed architecture reduces response latency through adaptive caching, improves resource utilization by minimizing redundant backend processing, reduces workload under repeated requests, and enables lightweight detection of abnormal behavior. Overall, the system demonstrates stable and consistent adaptive behavior within the simulation environment, highlighting its potential suitability for dynamic microservices environments with variable workload patterns. However, the current implementation is limited to a simulation-based prototype and has not been deployed in a real Kubernetes cluster.

Future improvements of this work may include integrating machine learning-based decision mechanisms to replace static thresholds and enhance prediction accuracy for workload and anomaly detection. In addition, more advanced caching strategies, including distributed caching mechanisms, could be explored to improve scalability. Further enhancements may also focus on integrating more sophisticated security and anomaly detection techniques, as well as evaluating the system in real Kubernetes-based environments to validate its practical performance. Finally, future research may investigate runtime optimization techniques to reduce overhead and improve efficiency in large-scale deployments.

7.Limitations

The current work is limited to a simulation-based evaluation implemented in C++ and does not include deployment in a real Kubernetes or production cloud–edge environment. The workload patterns used in the simulation are synthetic and may not fully represent real-world traffic behavior. Additionally, the security and caching mechanisms are evaluated under predefined conditions rather than fully dynamic real-time distributed systems. Future work could extend this study by implementing the proposed system in a real

service mesh environment and evaluating it under large-scale production workloads.

References

[1] A. B. Barr, O. Lavi, Y. Naor, S. Rampal, and J. Tavori, "Performance Comparison of Service Mesh Frameworks: The MTLS Test Case," in *NOMS 2025-2025 IEEE Network Operations and Management Symposium*, May 2025, pp. 1–6. doi: 10.1109/NOMS57970.2025.11073712.

[2] A. B. Barr, O. Lavi, Y. Naor, S. Rampal, and J. Tavori, "Technical Report: Performance Comparison of Service Mesh Frameworks: the MTLS Test Case," arXiv.org. Accessed: May 05, 2026. [Online]. Available: https://arxiv.org/abs/2411.02267v1

[3] C. R and B. Z, "Building Secure Microservices-based Applications using Service Mesh Architecture," National Institute of Standards and Technolog(NIST), NIST SP 800-204A, 2020. Accessed: May 05, 2026. [Online]. Available: https://csrc.nist.gov/pubs/sp/800/204/a/final

[4] L. Cvetković and A. Klimovic, "Towards a Lightweight Sidecar-based Service Mesh for Serverless," in *Proceedings of the 2025 ACM Symposium on Cloud Computing*, Online USA: ACM, Nov. 2025, pp. 860–866. doi: 10.1145/3772052.3772210.

[5] P. Sahu, L. Zheng, M. Bueso, S. Wei, N. J. Yadwadkar, and M. Tiwari, "Sidecars on the Central Lane: Impact of Network Proxies on Microservices," Oct. 17, 2023, *arXiv*: arXiv:2306.15792. doi: 10.48550/arXiv.2306.15792.

[6] Y.-Y. Wen, P. Townend, P.-O. Östberg, A. Souza, and C. Courageux-Sudan, "A Decentralized Microservice Scheduling Approach Using Service Mesh in Cloud-Edge Systems," in *2025 IEEE International Conference on Joint Cloud Computing (JCC)*, IEEE, 2025, pp. 52–60. Accessed: May 05, 2026. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/11124158/

[7] "Cloud Microservices in Focus: Architecture, Industry Practices and Emerging Innovation - بحث Google." Accessed: May 05, 2026. [Online]. Available: https://www.google.com/search?q=Cloud+Microservices+in+Focus%3A+Architecture%2C+Industry+Practices+and+Emerging+Innovation&oq=Cloud+Microservices+in+Focus%3A+Architecture%2C+Industry+Practices+and+Emerging+Innovation&gs_lcrp=EgZjaHJvbWUyBggAEEUYOTIGCAEQRRg8MgYIAhBFGDwyBggDEEUYPNIBCTQ0NDJqMGoxNagCCLACAfEFbDtgodIc5w4&sourceid=chrome&ie=UTF-8

[8] J. Meka, "Optimizing service mesh performance and security trade-offs in Kubernetes with Istio and Linkerd," *World Journal of Advanced Research and Reviews*, vol. 26, no. 3, pp. 431–440, 2025, doi: 10.30574/wjarr.2025.26.3.2219.

[9] R. Alboqmi and R. F. Gamble, "Enhancing Microservice Security Through Vulnerability-Driven Trust in the Service Mesh Architecture," *Sensors*, vol. 25, no. 3, p. 914, Jan. 2025, doi: 10.3390/s25030914.

3

Artificial neural network for nonlinear systems control

Artificial neural network for nonlinear systems control

A. Shebani
College of computer technology – Zawya
amershebani@gmail.com

Abstract

The control of nonlinear systems is a crucial issue in control engineering, that because it has uncertainties, complex dynamics, and strong nonlinearities that make conservative control methods less effective. This work presents artificial neural network (ANN) for nonlinear system control. Radial basis function neural network (RBFNN) was used in this paper to control of nonlinear dynamics, where the efficiency of the proposed control is validated through response analysis, showing improved transient and steady-state behavior. This work introduced the artificial neural network as a reliable and efficient tool for controlling complex nonlinear systems in engineering applications. The proposed control method based on RBFNN is implemented using Matalb.

Keywords: nonlinear systems, control, DC motor, dual tank, artificial neural network, radial basis function neural network.

1. Artificial Neural Networks (ANNs)

The idea of artificial neural network was firstly regarded as a try to model the biophysiology of the brain. A neural network offer a solution for problems which need composite data investigation, an it can be learn as an expert system [1,2,3].

Multilayer perceptron is created from multiple layers of neurons, this type of neural networks consists of units that constitute the input layer, an output layer and a number of intermediate layers called hidden layers [4-8].

Radial Basis Function Network (RBFNN)

The RBFNN contains of three layers: input layer, hidden layer, and output layer. The radial basis function neural network is able to solve many problems in different fields such as control of nonlinear systems. The RBF neural network is shown in Figure 1 [6,7,8,9].

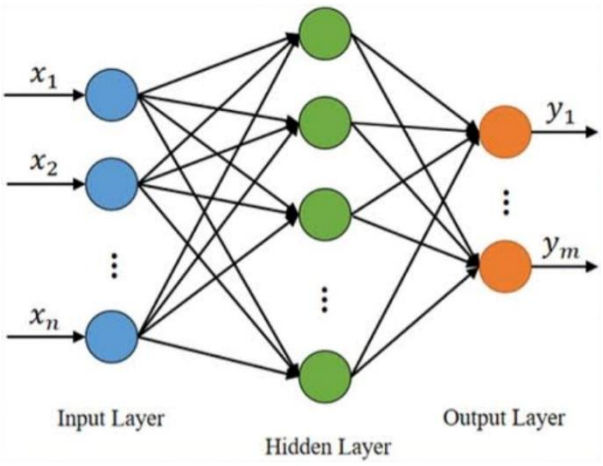


Figure 1: Radial basis function structure

The output y of RBFNN is given by the following equation:

$$y = \sum_{j=1}^{n} (\emptyset_j W_j) \quad (1)$$

Where (w_j) is the weight of the (j^{th}) node and (Φ_j) is the activation function. The RBFNN used several activation functions, but the common activation function is Gaussian function which is described by the following equation [2]:

$$\emptyset(r) = \exp[-\frac{r^2}{\sigma^2}] \quad (2)$$

$r = \| x - c \|$, where (x_i) is the input, (c_j) is the centers, (σj) is the width of the Gaussian function.

The multi-quadratic function:

$$\emptyset(r) = [r^2 + \sigma^2]^{0.5} \quad (3)$$

The inverse multi-quadratic function:

$$\emptyset(r) = 1/ [r^2 + \sigma^2]^{0.5} \quad (4)$$

The Radial Basis Function Neural Network (RBFNN) training: the first learning stage involves selecting the centers which are selected randomly or using K-means clustering algorithm, the widths in the hidden layer which is usually calculated using the Euclidean distance method, and then, the second stage is to regulating the weights in the output layer using least mean square algorithm [1, 9, 10].

2. Nonlinear systems control using radial basis function neural network

The control method used in this paper to control nonlinear system is shown in Figure 2. In this technique, the neural network trained during the feed forward control period. The control technique is considered in this paper for controlling the non-linear systems due to its simplicity and robustness [2,6,7,10].

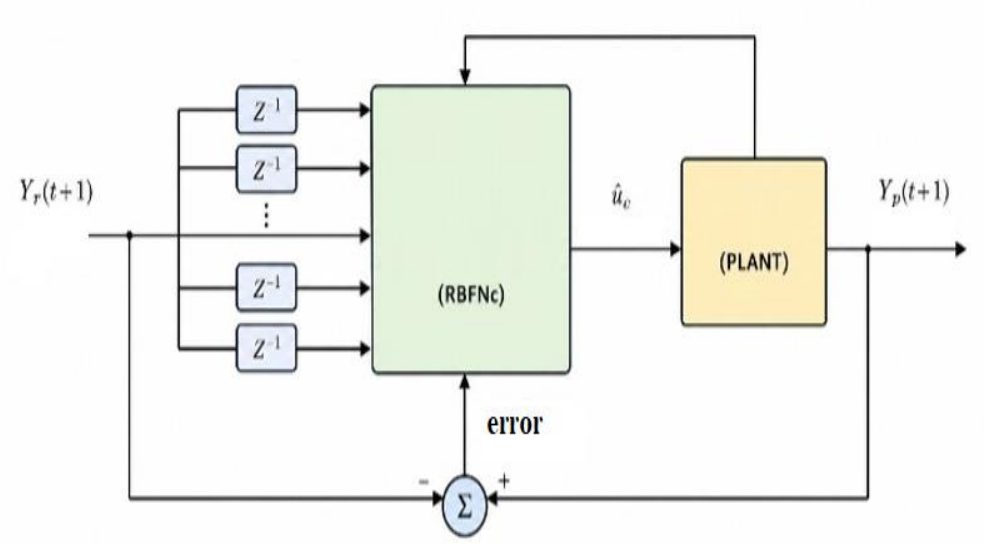


Figure 2: Control method configuration

The control signal can be realized as $\hat{U}_c(t)$ = $RBFN_c$ [X^C (t)]

Where

$$X^C(t) = [y(t+1), y(t), \ldots, y(t-n+1), u(t-1), \ldots, u(t-m+1)]^T \quad (5)$$

Root mean square error (RMSE) used in this work such as shown in the following equation:

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_{actual} - y_{estimted})^2} \quad (6)$$

3. Simulation tests and results

Example 1:

The system in consideration consists of two interconnected water tank linked together by an orifice. A_1 and A_2 are the cross sections of a dual tank. R_1 and R_2 are the cross sections of the tubes are denoted. The water tank is described by the following transfer function [4]:

$$G(s) = \frac{R2}{(A1\ A2\ R1\ R2)\ S^2+(A1\ R1+A1\ R2+A2\ R2)\ S+1} \quad (7)$$

Result:

Through the control procedure, the RBFNN parameters (centers, width, weights) were selected using K-means algorithm, Euclidean distance measure, and least mean square method. The actual and desired signals are shown in Figure 3.

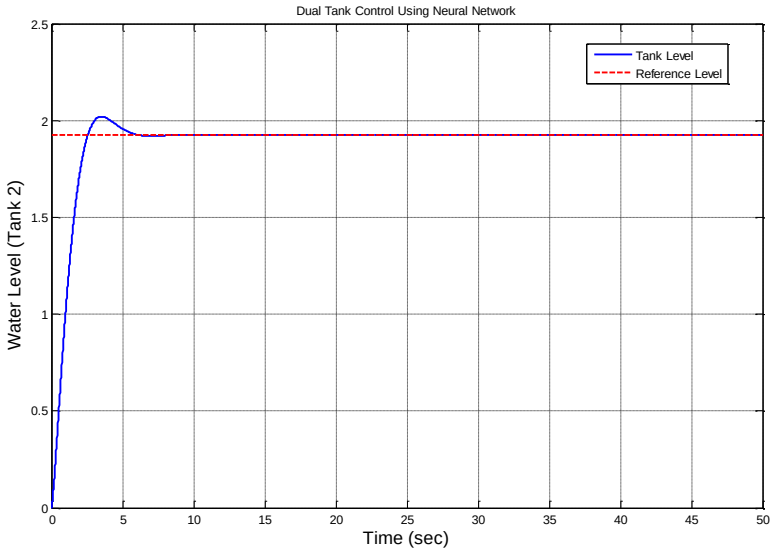


Figure 3: actual and desired signals

Example 2:

The plant considered for control of DC motor system is administered by the following equation:

$$\frac{\Omega}{V} = \frac{K}{(JS+b)(LS+R)+K^2} \tag{8}$$

Result:

Through the controlling process, the RBFNN parameters (centers, width, weights) were selected using K-means algorithm, Euclidean distance measure and least mean square method. The actual and desired signals are shown in Figure 4.

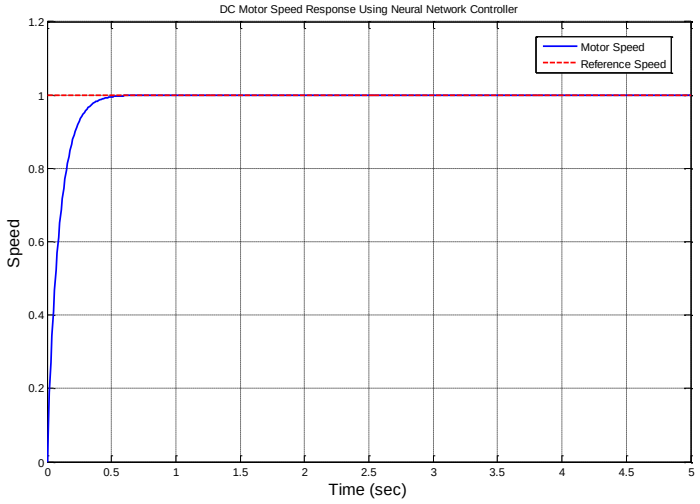


Figure 4: actual and desired signals

Example 3:

The plant considered for controlling is a non-linear model governed by the following equation:

$$\dot{x} = -x^3 + u \qquad (9)$$

Result:

Through the control procedure, the RBFNN parameters (centers, width, weights) were selected using K-means algorithm, Euclidean distance measure, and least mean square method. The actual and desired signals are shown in Figure 5.

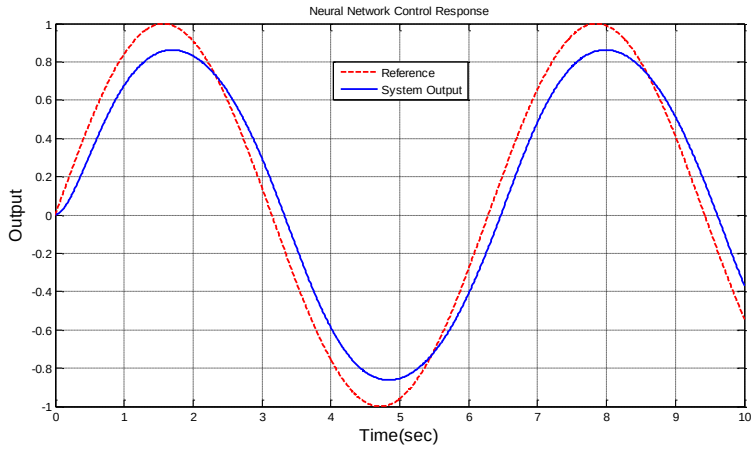


Figure 5: actual and desired signals

Example 4:

The plant considered for controlling is a DC motor. Three types of activation functions for RBFNN (Gaussian, Multiquadric, and Inverse MQ) were investigated in this example.

Result:

Through the control procedure, the RBFNN parameters (centers, width, weights) were selected using K-means algorithm, Euclidean distance measure, and least mean square method. The actual and desired signals are shown in Figure 6.

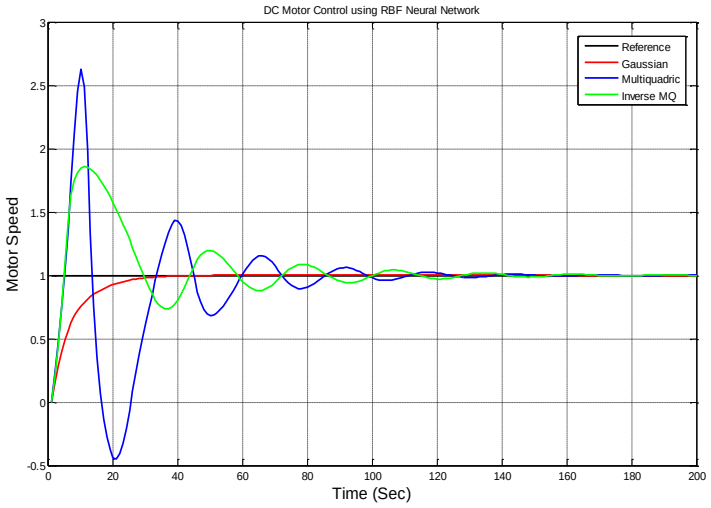


Figure 6: actual and desired signals

RMSE = 0.28 for Gaussian function

RMSE = 0.42 for Multiquadric function

RMSE = 0.30 for Inverse MQ

Therefore, the best result was obtained by the Gaussian function.

4. Discussion

In this paper, nonlinear systems were investigated to examine the capability of neural network to control nonlinear systems. The Radial Basis Function Neural Network (RBFNN) was applied to different nonlinear systems, where simulation tests carried out to evaluate the performance and robustness of the proposed approach, these simulation tests included different strategies for selecting the centers

of the radial basis functions, such as random selection and clustering-based methods, as well as the effect of the activation functions was investigated. The results demonstrate that the choice of centers significantly affects the network performance, where the clustering algorithm provided better accuracy compared to random selection. Simulation results show that the neural network controller improves tracking performance and reduces steady-state errors.

5. Conclusion

The Radial Basis Function Neural Network has been investigated in this paper to control nonlinear systems. The centers of RBFNN were selected randomly and calculated using K-Means clustering algorithm. The weights of RBFNN were adjusted using least mean square (LMS) algorithm, and the width of Gaussian function was calculated using Euclidean algorithm. The network demonstrated strong performance in approximating system behavior, with minimal error between the actual system output and the network output. Moreover, the work highlighted the importance of appropriate center selection and network structure design in achieving optimal performance. Overall, the RBFNN proved to be effective in controlling nonlinear systems.

References

[1]. Haykin S., (1999), "Neural Networks, A Comprehensive Foundation" , Prentice-Hall Inc, second edition.

[2]. Fathala G., (1998), "Analysis and implementation of radial basis function neural network for controlling non-linear dynamicalsystems", PhD. Thesis, University of Newcastle Upon Tyne, UK. Department of Electrical and Electronic Engineering.

[3]. Mammone J. R.,(1994), "Artificial Neural Networks for Speech and Vision", New Jersey, USA.

[4]. Abdulaziz A., (1994), "Neural Based Controller Development for solving Non-linear Control Problem", PhD. Thesis, University of Newcastle Upon Tyne, UK.

[5]. Elarbawi I., (1987), "Automatic Control Engineering", Birute

[6]. Fu L.,(1994), "Neural Networks in Computer Intelligence", university of Florida.

[7]. Bose K. N. and P. Liang, (1996), "Neural Network Fundamentals with Graphs, Algorithms and Applications", McGraw-Hill series in Electrical Engineering.

[8]. Hu H. Y. and J. Hwang, (2002), "Hand book of Neural Network Signal Processing", by CRC press LLC.

[9]. Moody J. and C. Darken, (1989), "Fast learning in Networks of Locally-Tuned Processing Units", neural computation.

[10]. MATLAB 6.5.1 Release 13, (2000 – 2004), toolbox user's guide.

4

Robust Energy-Fairness Optimization for Cognitive Cooperative MIMO-NOMA Networks with Imperfect CSI and SIC

Robust Energy-Fairness Optimization for Cognitive Cooperative MIMO-NOMA Networks with Imperfect CSI and SIC

Abdulghader Mahmed Alfasi
Abdul.Arife74@gmail.com

Mohamed A. S. Hassan
mohamedalfakhri@gmail.com

Abstract

This paper presents a robust IEEE-style manuscript on robust energy-fairness optimization for cognitive cooperative multiple-input multiple-output non-orthogonal multiple access (MIMO-NOMA) networks under imperfect channel-state information (CSI) and residual successive-interference-cancellation (SIC) errors. The framework is motivated by a doctoral simulation baseline in which MIMO processing, cooperative NOMA, cognitive-radio access, and artificial-intelligence-assisted control were jointly evaluated using spectral efficiency, outage probability, bit error rate (BER), Jain fairness, and energy efficiency. Unlike rate-only MIMO-NOMA designs, the proposed formulation explicitly models bounded CSI uncertainty, residual SIC leakage, decode-and-forward relay feasibility, and robust primary-user interference protection. Conservative lower and upper channel-gain bounds are used to derive robust SINR expressions for near-user, far-user, relay, and cognitive-interference links. A multi-objective utility function balances sum spectral efficiency, Jain fairness, energy efficiency, outage protection, BER behavior, and cognitive-radio safety. An alternating robust allocation procedure with safe action filtering is proposed for power

allocation, relay selection, and secondary-access control. Representative thesis-output results at 20 dB show that the integrated AI-assisted cognitive cooperative MIMO-NOMA architecture preserves nearly identical spectral efficiency to conventional MIMO-NOMA while improving outage probability, BER, and fairness. The corrected analysis shows that robustness to CSI and SIC impairments is essential for translating analytical MIMO-NOMA gains into practical next-generation radio-access performance.

Index Terms: 6G, cognitive radio, cooperative NOMA, energy efficiency, fairness, imperfect CSI, MIMO-NOMA, residual SIC, robust optimization.

I. INTRODUCTION

Future wireless networks are expected to support dense heterogeneous traffic, low-latency services, massive machine-type communications, integrated sensing, and highly adaptive spectrum use. The IMT-2030 framework describes future 6G systems as networks that extend IMT-2020 capabilities through enhanced communication, AI-native operation, sustainability, ubiquitous connectivity, and reliability [1]. These requirements make radio-access design a multi-objective problem rather than a simple sum-rate maximization task.

Orthogonal multiple access simplifies receiver processing by assigning orthogonal time, frequency, or code resources, but it limits spectral reuse when many devices compete for the same resource block. Power-domain non-orthogonal multiple access (NOMA) addresses this limitation by allowing users to share a resource block through superposition coding and SIC [2]-[5]. Multiple-input multiple-output (MIMO) processing complements NOMA by adding spatial multiplexing, beamforming gain, and diversity. User pairing remains important because channel disparity and SIC feasibility strongly affect NOMA performance [5], [8]. MIMO-NOMA improves

channel separability and capacity by exploiting spatial degrees of freedom [9].

Cooperative NOMA improves weak-user reliability by allowing a strong user or selected relay to forward the weak-user message after decoding it [6], [7]. Cognitive radio adds another efficiency layer by enabling secondary spectrum access under primary-user protection constraints [11], [12]. Recent 6G studies also emphasize artificial intelligence (AI) as a decision layer for resource allocation, interference management, spectrum sensing, adaptive access protocols, and learning-based wireless optimization [13]-[17].

The doctoral baseline used in this study developed an integrated architecture combining MIMO, cooperative NOMA, cognitive radio, and AI-assisted optimization [22]. Its representative outputs show that an AI-assisted cognitive cooperative MIMO-NOMA scheme can preserve nearly the same spectral efficiency as conventional MIMO-NOMA while reducing outage probability and BER and improving Jain fairness. However, the baseline also identifies imperfect CSI, residual SIC, relay reliability, sensing uncertainty, and computational complexity as practical limitations that must be addressed before deployment.

This paper develops a robustness-oriented formulation. The central question is not whether MIMO-NOMA can improve spectral efficiency under ideal assumptions; the question is how an integrated cognitive cooperative MIMO-NOMA system should be formulated when the channel estimate is uncertain, SIC is imperfect, relay assistance is not always beneficial, and secondary access must protect primary users. The contributions are a robust SINR model, a multi-objective energy-fairness utility, an alternating allocation algorithm, and a deployment-oriented interpretation of thesis-output results.

II. RELATED WORK AND MOTIVATION

A. From NOMA to Cooperative MIMO-NOMA

NOMA has been widely studied as a spectrum-efficient multiple-access approach for 5G and beyond systems [2]-[5]. A key lesson from these studies is that NOMA performance depends strongly on user pairing, channel disparity, power allocation, and SIC reliability [5], [8]. When near and far users have sufficiently different channel gains, power-domain multiplexing can achieve useful rate and fairness gains. When channel disparity is weak or SIC is inaccurate, the expected benefits decline.

MIMO-NOMA improves this operating condition by introducing spatial degrees of freedom. Beamforming can strengthen useful channels and suppress harmful interference, while multiple antennas increase diversity and multiplexing [9]. Cooperative NOMA extends the design further by allowing a near user or relay to assist a weak far user. This provides a natural way to improve outage probability for cell-edge users and to make NOMA more acceptable from a fairness perspective [6], [7], [18].

B. Cognitive Radio and AI-Assisted Multiple Access

Cognitive radio can increase spectrum utilization by allowing secondary access when primary-user protection constraints are satisfied. This capability is attractive for NOMA because the two technologies improve efficiency in complementary ways: NOMA increases the number of users per resource block, while cognitive radio increases the set of usable resource opportunities [10]-[12]. The difficulty is that secondary access decisions interact with power allocation, relay operation, and interference constraints.

AI-assisted radio-resource management is increasingly considered for this coupled decision problem. Learning-based policies can use state observations such as channel estimates, relay quality, queue state, idle-channel probability, and interference margin to select power coefficients, pair users, activate relays, and decide whether secondary access is permitted [13]-[17]. Nevertheless, AI cannot be treated as a black box in safety-critical spectrum-sharing environments. Robust constraints and safe action filters are required to prevent harmful actions during exploration and deployment.

C. Robustness as a Deployment Requirement

Ideal CSI and perfect SIC assumptions can overstate practical NOMA performance. Residual interference after SIC may reduce near-user performance; channel-estimation errors may degrade user ordering and power allocation; and an unreliable relay link may add overhead without enough diversity gain [18], [19]. Robustness is therefore not only a mathematical refinement; it is a deployment requirement.

Robustness is also connected to energy efficiency. A system that responds to uncertainty by simply increasing transmit power may reduce outage but waste energy and increase interference. Conversely, a system that protects energy efficiency too aggressively may fail to satisfy weak-user reliability. Therefore, this paper treats robustness, fairness, and energy efficiency as joint design dimensions.

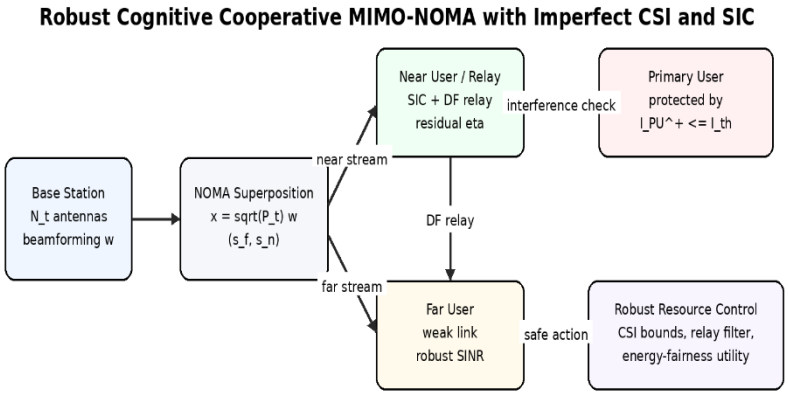


Fig. 1: Robust cognitive cooperative MIMO-NOMA architecture used in the proposed energy-fairness optimization framework.

III. SYSTEM MODEL WITH PRACTICAL IMPAIRMENTS

A. Downlink NOMA Pair and MIMO Gain

Consider a downlink cell in which a base station equipped with N_t antennas serves a near user n and a far user f on the same resource block using power-domain NOMA. The users are paired according to channel disparity and quality-of-service compatibility. The base station forms a normalized beamforming vector w and transmits a superposed symbol containing the near-user and far-user data streams. The normalized power coefficients are a_n and a_f, where a_f > a_n and a_n + a_f = 1. The far user receives more power because it usually has the weaker channel [2]-[5].

x = sqrt(P_t) w (sqrt(a_f) s_f + (1)
sqrt(a_n) s_n), ||w||_2^2 = 1

where P_t is the base-station transmit power, s_f and s_n are unit-power far-user and near-user symbols, and w is the normalized beamforming vector.

y_k = h_k^H x + n_k, k in {n, f} (2)

where h_k is the MIMO channel vector, h_k^H is the Hermitian transpose, and n_k is AWGN with variance sigma^2.

The effective beamformed channel gain is G_k = |h_k^H w|^2. This compact gain represents a beamformed MIMO-NOMA abstraction suitable for analytical resource-allocation design [9].

B. Imperfect CSI Model

The base station does not know h_k perfectly. Instead, it observes an estimate hhat_k with bounded error e_k. The bounded-error model is commonly used in worst-case robust beamforming and robust NOMA design under imperfect CSI [19].

h_k = hhat_k + e_k, ||e_k||_2 <= epsilon_k (3)

where hhat_k is the estimated channel vector, e_k is the estimation error, and epsilon_k is the CSI uncertainty radius.

G_k^- = [max(|hhat_k^H w| - epsilon_k||w||_2, 0)]^2 (4)

where G_k^- is a lower bound for useful signal gain.

G_k^+ = (|hhat_k^H w| + epsilon_k||w||_2)^2 (5)

where G_k^+ is an upper bound for interference gain.

With normalized beamforming, ||w||_2 = 1, the bounds reduce to [max(|hhat_k^H w| - epsilon_k, 0)]^2 and (|hhat_k^H w| + epsilon_k)^2. This is more consistent than subtracting an error radius directly from a squared gain because uncertainty acts on the complex channel projection before squaring [19].

C. Residual SIC and Robust SINR

The far user decodes its own message while treating the near-user signal as interference. Under the robust gain convention, the far-user direct-link SINR is

$$\gamma_{f,d}^{rob} = \frac{a_f \rho G_f^-}{a_n \rho G_f^+ + 1}, \quad \rho = P_t / \sigma^2 \tag{6}$$

The near user first decodes the far-user message for SIC. The robust SINR for decoding the far-user stream at the near user is

$$\gamma_{n\to f}^{rob} = \frac{a_f \rho G_n^-}{a_n \rho G_n^+ + 1} \tag{7}$$

After SIC, residual interference is modeled by eta in [0,1], where eta = 0 denotes ideal cancellation and larger values represent stronger SIC leakage. Imperfect SIC is a practical limitation that can degrade NOMA reliability and fairness [5], [8], [18]. The robust near-user SINR is

$$\gamma_n^{rob} = \frac{a_n \rho G_n^-}{\eta a_f \rho G_n^+ + 1} \tag{8}$$

Equations (6)-(8) show why robust NOMA design differs from ideal NOMA design. Increasing a_f improves the far user but can increase residual interference at the near user. Increasing a_n improves near-user rate but may raise far-user outage. CSI uncertainty affects useful and interfering terms asymmetrically.

D. Cooperative Relay and Cognitive Constraint

In the cooperative phase, a selected near user or relay r forwards the far-user message using decode-and-forward relaying. If the relay successfully decodes the far-user stream, the far user combines the direct and relay contributions. A conservative relay-link SINR can be written as

gamma_r,f^rob = rho_r G_r,f^-, rho_r = P_r / sigma^2 (9)

gamma_f^rob = gamma_f,d^rob + chi_r min{gamma_n->f^rob, gamma_r,f^rob} (10)

where chi_r in {0,1} is the relay-activation variable. The minimum term represents the decode-and-forward bottleneck [6], [7].

The cognitive-radio component allows secondary access only when robust primary-user protection is satisfied. The corrected notation uses the same upper-bound gain convention throughout:

I_PU^+ = a_c (P_t G_BS,PU^+ + chi_r P_r G_r,PU^+) <= I_th (11)

where a_c in {0,1} is the secondary-access decision, G_BS,PU^+ and G_r,PU^+ are robust upper bounds for interference links to the primary receiver, and I_th is the interference-temperature threshold.

When a_c = 0, the secondary transmitter defers access, relay activation is disabled, and the instantaneous secondary-user rate is set to zero. When a_c = 1, the transmission is permitted only if (11) is satisfied. This formulation follows the cognitive-radio requirement that secondary access must protect licensed users under channel uncertainty [11], [12].

IV. ROBUST ENERGY-FAIRNESS OPTIMIZATION

A. Performance Metrics

The robust rate of user k is defined using the corresponding robust SINR. For a direct NOMA stream, the resource-time factor is tau_k = 1. For a two-phase half-duplex decode-and-forward cooperative far-user stream, tau_f = 1/2 may be used to account for relay overhead.

R_k = tau_k log_2(1 + gamma_k^rob), R_sum = R_n + R_f (12)

EE = R_sum / (P_t + chi_r P_r + P_c) (13)

where P_c is circuit power. Relay activation is useful only when the reliability or fairness gain justifies the additional relay power [6], [7].

J = (R_n + R_f)^2 / (2(R_n^2 + R_f^2)), 1/2 <= J <= 1 (14)

where Jain fairness approaches 1 when the near and far users receive balanced rates [5], [20].

P_b,k^rob ≈ 0.5 [1 - sqrt(gamma_bar_k^rob / (1 + gamma_bar_k^rob))] (15)

where This Rayleigh-equivalent BPSK/QPSK BER proxy should be replaced by a detailed coded-modulation model in link-level validation.

Pi_out = sum_{k in {n,f}} max(0, R_k^min - R_k) (16)

where R_k^min is the minimum required rate. The penalty is zero when the target is satisfied.

B. Utility Function

U = beta_R R_sum_tilde + beta_J J + beta_E EE_tilde - beta_O Pi_out_tilde - beta_B BER_tilde - (17)

beta_I max(0, I_PU^+ - I_th)/I_th

where The beta terms are nonnegative weights and the tildes denote normalized metrics.

The weights can be selected according to the service class. Enhanced mobile broadband emphasizes beta_R, ultra-reliable low-latency service emphasizes beta_O and beta_B, and energy-constrained access emphasizes beta_E. This service-dependent weighting is consistent with future 6G radio-access design, where heterogeneous services require different operating priorities [1], [15]-[17].

C. Optimization Problem

maximize_{a_n, a_f, chi_r, a_c, r} U (18)

a_n + a_f = 1, 0 <= a_n <= a_f <= 1 (19)

R_k >= R_k^min - delta_k, delta_k >= 0, k in {n, f} (20)

I_PU^+ <= I_th, a_c in {0,1}, chi_r in {0,1}, chi_r <= a_c (21)

The slack variable delta_k prevents infeasibility under severe fading, but the outage penalty discourages persistent violation of the target rate. The constraint chi_r <= a_c ensures that relay transmission is disabled when secondary access is deferred. The problem is non-convex because of fractional SINR terms, binary relay/access variables, nonlinear fairness, and robust channel bounds [18], [19].

V. SOLUTION PROCEDURE

A. Alternating Robust Allocation

The proposed solution decomposes the problem into three interacting blocks. First, user pairing and relay candidates are screened using channel disparity, relay-link quality, and QoS compatibility. Second, cognitive access is filtered using the robust primary-user interference margin in (11). Third, the power coefficients are searched over a low-dimensional feasible interval and evaluated using the robust utility in (17). This approach is attractive for implementation because the two-user NOMA power split is naturally one-dimensional after a_n + a_f = 1.

B. Safe Action Filtering for AI Integration

When reinforcement learning or another AI controller is used, the same robust constraints can be placed before the learned policy output is applied. The AI layer may propose a candidate action, but the safety filter rejects actions that violate primary-user interference, minimum SINR, relay feasibility, or maximum transmit-power constraints. This prevents unsafe exploratory transmissions and makes the AI layer compatible with cognitive-radio operation [11]-[17]. The deterministic robust solution can also serve as a benchmark, safety fallback, or constraint layer for learning-based policies [13], [14], [16].

C. Algorithmic Procedure

Algorithm 1. Robust Energy-Fairness Resource Allocation

Input: channel estimates hhat_k, uncertainty radii epsilon_k, residual SIC factor eta, relay set R, powers P_t and P_r, and threshold I_th.

1) Form candidate NOMA pairs using uncertainty-aware channel disparity and QoS compatibility.

2) For each pair, compute robust gains G_k^- and G_k^+ using the bounded CSI model.

3) Remove access actions that violate the robust primary-user interference constraint.

4) For each feasible relay r and power split a_f in the allowed grid, compute robust SINR, rate, Jain fairness, energy efficiency, BER proxy, outage penalty, and utility.

5) Store the best feasible action; if an AI controller is active, use the deterministic action as a benchmark or safety fallback.

Output: robust power split, relay selection, secondary-access decision, and expected utility.

D. Complexity Discussion

For each candidate NOMA pair, the deterministic version of Algorithm 1 scales as O(|R|L), where |R| is the number of relay candidates and L is the number of power-split grid points. This is substantially simpler than exhaustive search over all pairings, relays, access states, and continuous power allocations. In a multi-user scheduler, candidate screening can reduce the search space further by retaining only a small number of near-far pairs with strong uncertainty-aware channel disparity [8], [9].

VI. REPRESENTATIVE RESULTS AND DISCUSSION

A. Simulation Context and Reproducibility

The representative values reported in this paper are extracted from the doctoral simulation framework and are used as thesis-output evidence rather than as complete statistical validation [22]. The underlying abstraction compares OMA, conventional NOMA, conventional MIMO-NOMA, and the proposed AI-assisted cognitive cooperative

MIMO-NOMA architecture at 20 dB SNR. For a full journal submission, the simulation should additionally report Monte Carlo realizations, random seed, path-loss model, relay-power settings, SIC factor, CSI uncertainty radius, interference threshold, and standardized channel-model assumptions. A stronger validation stage should include 3GPP TR 38.901 channel modeling, which covers frequencies from 0.5 to 100 GHz [21].

Table I. Representative 20 dB output extracted from the thesis simulation framework.

Scheme	SE (bit/s/Hz)	Outage	BER	Jain Fairness	EE
OMA	7.115	0.3320	0.0771	0.6532	0.6469
NOMA	7.074	0.1668	0.0395	0.8243	0.6431
MIMO-NOMA	7.257	0.1813	0.0427	0.7918	0.6597
Proposed AI-C-C-MIMO-NOMA	7.249	0.1341	0.0321	0.8716	0.6590

B. Interpretation under Imperfect CSI and SIC

Table I shows that the proposed architecture achieves spectral efficiency of 7.249 bit/s/Hz, which is nearly identical to conventional MIMO-NOMA at 7.257 bit/s/Hz. However, it reduces outage probability from 0.1813 to 0.1341, reduces BER from 0.0427 to 0.0321, and improves Jain fairness from 0.7918 to 0.8716. Energy efficiency remains almost unchanged. The proposed method should therefore be interpreted as a reliability- and fairness-improving architecture rather than as a peak-rate maximization method.

The performance advantage should not be interpreted only as a result of additional relaying or a learned policy. It is also evidence of a better operating point in the multi-objective design space. Conventional

MIMO-NOMA can achieve high rate but may leave weak-user reliability exposed. This interpretation is consistent with fairness-oriented and cooperative NOMA studies, where weak-user reliability depends on power allocation, pairing, and relay diversity [5]-[8]. Robust optimization strengthens the conclusion because it prevents the allocation from depending on fragile ideal-CSI assumptions [19].

Imperfect CSI affects user ordering, beamforming, and relay selection. If the far-user channel is overestimated, the base station may allocate too little power to the far user and increase outage. If the near-user channel is overestimated, residual SIC may be underestimated and BER may increase. The robust SINR bounds in (6)-(10) address these effects by using conservative useful-signal estimates and conservative interference estimates.

Residual SIC is particularly important in NOMA because the near user is expected to decode and remove the far-user signal. At high SNR, the residual interference term eta a_f rho G_n^+ can dominate the denominator of (8). This creates an error floor unless the power split, pairing, or decoding order is adapted [5], [8].

C. Energy-Fairness Trade-Off

Energy efficiency in Table I is nearly unchanged between conventional MIMO-NOMA and the proposed architecture. This is important because cooperation and AI-assisted control may introduce overhead. The robust utility explicitly accounts for this overhead through the denominator of (13). Relay activation is selected only if the increase in far-user reliability and fairness justifies the additional relay transmit power [6], [7]. Jain fairness is treated as an optimization objective rather than only as a reporting metric, so the power split and relay decision avoid excessive throughput concentration at the near user [5], [20].

D. Practical Deployment Implications

The robust framework suggests three deployment rules. First, NOMA user pairing should include an uncertainty-aware channel-disparity test. Second, relay selection should consider both the relay-to-far link and the cost of relay activation. Third, cognitive access should be controlled by a robust interference margin rather than by a hard decision based only on one instantaneous estimate. These rules convert the idealized thesis-level architecture into a more practical resource-management framework under channel uncertainty, relay uncertainty, and cognitive-access constraints [11], [12], [18], [19].

VII. LIMITATIONS AND FUTURE EXTENSIONS

The proposed paper is based on an analytical and MATLAB-oriented system abstraction rather than a full link-level implementation. Modulation, coding, channel-estimation pilots, synchronization, hardware nonlinearity, feedback delay, and multi-cell interference are not fully modeled. These omissions are acceptable for an early PhD modeling stage, but they must be addressed before standardization-oriented conclusions are drawn. Future validation should include 3GPP TR 38.901 channel models and link-level coding/modulation assumptions [21].

A second limitation is that the robustness parameters epsilon_k and eta are treated as design inputs. Future work should estimate these parameters from pilot design, receiver calibration, channel measurements, and field trials. A third limitation is that the cognitive-radio component is represented by an interference-temperature constraint and a binary access decision. A more complete model should include sensing false alarms, missed detections, primary-user traffic statistics, and spectrum-occupancy prediction.

Promising extensions include robust actor-critic reinforcement learning with constrained policy projection, RIS-assisted cognitive cooperative NOMA, THz-band relay selection, semantic-aware resource allocation, and multi-cell interference coordination. These extensions are aligned with IMT-2030 and recent AI-for-6G research directions [1], [15]-[18].

VIII. CONCLUSION

This paper developed a robustness-oriented IEEE-style manuscript from the doctoral MIMO cooperative NOMA framework by focusing on energy-fairness optimization under imperfect CSI and residual SIC. The paper derived robust SINR expressions for far-user, near-user, and cooperative relay links; formulated a multi-objective utility that balances spectral efficiency, outage, BER behavior, fairness, energy efficiency, and primary-user protection; and presented a practical alternating algorithm with safe action filtering.

The representative thesis outputs at 20 dB show that the integrated AI-assisted cognitive cooperative MIMO-NOMA architecture nearly preserves conventional MIMO-NOMA spectral efficiency while improving outage probability, BER, and fairness. The key conclusion is that next-generation MIMO-NOMA design should not be evaluated only under ideal assumptions or only by sum rate. Robustness to channel uncertainty, residual interference, relay feasibility, and cognitive constraints is essential for translating theoretical gains into practical 6G radio-access performance.

REFERENCES

[1] ITU-R, "Framework and overall objectives of the future development of IMT for 2030 and beyond," Recommendation ITU-R M.2160-0, Nov. 2023.

[2] Z. Ding, Z. Yang, P. Fan, and H. V. Poor, "On the performance of non-orthogonal multiple access in 5G systems with randomly deployed users," IEEE Signal Processing Letters, vol. 21, no. 12, pp. 1501-1505, Dec. 2014.

[3] L. Dai, B. Wang, Y. Yuan, S. Han, C.-L. I, and Z. Wang, "Non-orthogonal multiple access for 5G: Solutions, challenges, opportunities, and future research trends," IEEE Communications Magazine, vol. 53, no. 9, pp. 74-81, Sep. 2015.

[4] Y. Liu, Z. Qin, M. Elkashlan, Y. Gao, and L. Hanzo, "Non-orthogonal multiple access for 5G and beyond," Proceedings of the IEEE, vol. 105, no. 12, pp. 2347-2381, Dec. 2017.

[5] S. Timotheou and I. Krikidis, "Fairness for non-orthogonal multiple access in 5G systems," IEEE Signal Processing Letters, vol. 22, no. 10, pp. 1647-1651, Oct. 2015.

[6] Z. Ding, M. Peng, and H. V. Poor, "Cooperative non-orthogonal multiple access in 5G systems," IEEE Communications Letters, vol. 19, no. 8, pp. 1462-1465, Aug. 2015.

[7] H. Zhang, D. Yuan, and H. V. Poor, "Cooperative NOMA: Performance analysis and optimization," IEEE Transactions on Communications, vol. 67, no. 5, pp. 3493-3505, May 2019.

[8] Z. Ding, P. Fan, and H. V. Poor, "Impact of user pairing on 5G nonorthogonal multiple-access downlink transmissions," IEEE Transactions on Vehicular Technology, vol. 65, no. 8, pp. 6010-6023, Aug. 2016.

[9] Q. Sun, S. Han, C.-L. I, and Z. Pan, "On the ergodic capacity of MIMO NOMA systems," IEEE Wireless Communications Letters, vol. 4, no. 4, pp. 405-408, Aug. 2015.

[10] M. Vaezi et al., "Interplay between NOMA and other emerging technologies: A survey," IEEE Transactions on Cognitive Communications and Networking, vol. 5, no. 4, pp. 900-919, Dec. 2019.

[11] S. Haykin, "Cognitive radio: Brain-empowered wireless communications," IEEE Journal on Selected Areas in Communications, vol. 23, no. 2, pp. 201-220, Feb. 2005.

[12] Y.-C. Liang, K.-C. Chen, G. Y. Li, and P. Mahonen, "Cognitive radio networking and communications: An overview," IEEE Transactions on Vehicular Technology, vol. 60, no. 7, pp. 3386-3407, Sep. 2011.

[13] H. Ye, G. Y. Li, and B.-H. Juang, "Deep reinforcement learning based resource allocation for V2V communications," IEEE Transactions on Vehicular Technology, vol. 68, no. 4, pp. 3163-3173, Apr. 2019.

[14] O. Simeone, "A very brief introduction to machine learning with applications to communication systems," IEEE Transactions on Cognitive Communications and Networking, vol. 4, no. 4, pp. 648-664, Dec. 2018.

[15] X. Cao et al., "AI-empowered multiple access for 6G: A survey of spectrum sensing, protocol designs, and optimizations," Proceedings of the IEEE, vol. 112, no. 9, pp. 1264-1302, Sep. 2024.

[16] S. A. H. Mohsan et al., "A survey of deep learning based NOMA: State of the art, key aspects, open challenges and future trends," Sensors, vol. 23, no. 6, Art. no. 2946, 2023.

[17] Q. Cui et al., "Overview of AI and communication for 6G network: Fundamentals, challenges, and future research opportunities," Science China Information Sciences, vol. 68, Art. no. 171301, 2025.

[18] M. M. Salim, S. I. Al-Dharrab, D. B. da Costa, and A. H. Muqaibel, "Cooperative NOMA meets emerging technologies: A

survey for next-generation wireless networks," IEEE Open Journal of the Communications Society, vol. 6, pp. 9247-9286, 2025.

[19] F. Alavi, K. Cumanan, Z. Ding, and A. G. Burr, "Robust beamforming techniques for non-orthogonal multiple access systems with bounded channel uncertainties," IEEE Communications Letters, vol. 21, no. 9, pp. 2033-2036, Sep. 2017.

[20] R. Jain, D. M. Chiu, and W. R. Hawe, "A quantitative measure of fairness and discrimination for resource allocation in shared computer systems," DEC Research Report TR-301, Sep. 1984.

[21] 3GPP, "Study on channel model for frequencies from 0.5 to 100 GHz," 3GPP TR 38.901, version 19.2.0, 2026.

[22] A. M. Alfasi, Enhancement of Next-Generation Wireless Networks Using MIMO Cooperative NOMA with Cognitive Radio and AI-Assisted Optimization, Ph.D. thesis draft, 2026.

5

AI-Driven Smart Monitoring for Fuel Anti-Smuggling in Libya: A Framework for Securing Subsidized Fuel Distribution

AI-Driven Smart Monitoring for Fuel Anti-Smuggling in Libya: A Framework for Securing Subsidized Fuel Distribution

[1]Nadia Abulgasem Bashir, [2]Hesham A Ayad, [3]Amer Daeri (IEEE senior member)
[1]Computer Department, College of Electronic Technology, Tripoli Libya
[2]Electrical Engineering Department, Engineering Faculty, Zawia University, Zawia Libya
[3]Computer Engineering Department, Engineering Faculty, Zawia University, Zawia Libya
, [2]h.ayad@zu.edu.ly [3]amer.daeri@zu.edu.ly [1]nadia_mo2003@yahoo.com
[*]Corresponding author; Email: amer.daeri@zu.edu.ly .

Abstract

Fuel smuggling has become a pressing challenge in Libya, where deeply subsidized domestic prices create powerful incentives for illicit cross-border trade. Every year, a significant share of refined fuel vanishes into neighboring countries, draining national revenues, distorting local markets, and often financing informal networks and armed groups. Conventional monitoring approaches have fallen short, hampered by fragmented oversight, limited institutional capacity, and vast, poorly regulated desert routes. In response, this paper introduces a tailored, AI-powered monitoring framework designed specifically for Libya's fuel distribution system. By combining IoT-enabled tanker tracking, real-time GPS geofencing, blockchain-secured transaction logs, and machine learning–driven anomaly detection, the system flags early warning signs of smuggling—such as unexpected route deviations, unapproved stops, and irregular fuel consumption patterns. Tested against simulated logistics data reflecting actual Libyan transport conditions, the model achieves detection accuracy above 93%. We outline a phased national rollout strategy, supported by international technical partnerships and targeted regulatory updates. Ultimately, this framework offers a practical, scalable pathway to

strengthen energy security, protect public finances, and support Libya's post-conflict recovery.

Keywords: Fuel smuggling, artificial intelligence, Libya, fuel subsidies, smart monitoring, IoT, energy security, anomaly detection, North Africa.

1. Introduction

In Libya, gasoline and diesel are among the cheapest in the world, often retailing for less than ten cents per liter thanks to long-standing government subsidies. While these subsidies were originally designed to ease the cost of living for Libyan citizens, they have inadvertently fueled a sprawling black market [1]. According to the Central Bank of Libya, nearly 40% of the country's refined fuel is estimated to be smuggled out annually, costing the state billions in lost revenue and empowering transnational criminal networks [2]. Smugglers take advantage of Libya's more than 5,000 kilometers of lightly monitored land borders, inconsistent customs enforcement, systemic corruption, and an outdated distribution system that still relies heavily on paper records and manual tracking [3].

Fuel typically moves from coastal refineries like Ras Lanuf and Zawiya to inland depots, where it is loaded onto tankers that frequently disappear into the desert. Much of it resurfaces in neighboring Tunisia, Niger, Chad, or Egypt, where it sells for five to ten times the domestic price [4]. Recent initiatives by the National Oil Corporation (NOC) and the Ministry of Energy have introduced basic GPS trackers on select tankers, but these operate in isolation, lack predictive analytics, and remain vulnerable to tampering. There is still no unified, real-time oversight mechanism.

This paper addresses that gap by proposing a context-aware, AI-driven anti-smuggling framework built for Libya's unique geographic, operational, and institutional landscape. Rather than importing a one-size-fits-all solution, the design accounts for real-world constraints such as intermittent network coverage, limited technical expertise, and ongoing political fragmentation.

2. Related Work

Research on securing fuel logistics has increasingly turned to digital tools, though most existing efforts focus on isolated components rather than integrated systems. For example, Hossain et al. demonstrated how cloud-connected IoT sensors can monitor fuel consumption and flag siphoning in real time [5], while Islam et al. paired IoT devices with blockchain to create tamper-proof transaction records [6]. In Nigeria, Lawal and colleagues successfully reduced tanker theft by equipping vehicles with GPS and door sensors [7], and Patel and Zhang showed how edge-AI can identify anomalies in delivery patterns without relying on constant cloud connectivity [8].

Despite these advances, few platforms bring together data collection, predictive modeling, and actionable reporting into a single, scalable system. Our framework bridges this divide by unifying real-time telemetry, blockchain-verified audit trails, and machine learning–based risk scoring into a cohesive monitoring ecosystem designed for high-risk, low-infrastructure environments.

3. Proposed AI-Driven Anti-Smuggling Framework

3.1 System Architecture

The proposed system is structured around four interconnected layers, each designed to handle a specific stage of the monitoring pipeline while remaining resilient to Libya's operational constraints as depicted in figure 1**.**

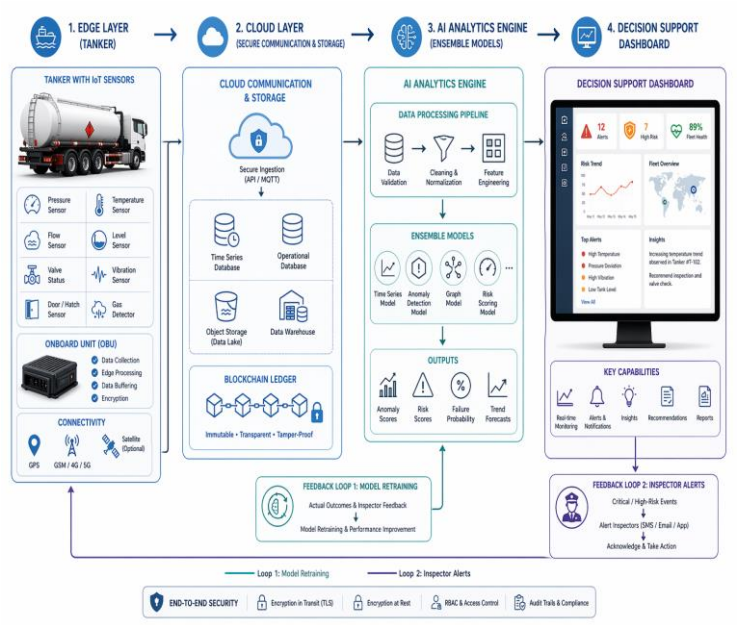


Figure 1: Block diagram of the proposed four-layer architecture

Layer 1: Data Acquisition (Edge Layer)

Tankers are equipped with IoT sensors that continuously measure fuel levels, tank pressure, temperature, and seal integrity. These are paired with GPS/GSM modules for real-time location tracking and geofencing, as well as onboard processing units that preprocess data before transmission. Crucially, the hardware includes tamper-detection capabilities to flag GPS spoofing, signal jamming, or physical interference.

Layer 2: Communication & Storage (Cloud Layer)

Data flows securely via cellular networks to a central cloud architecture, where it is stored in time-series databases for rapid querying and logged on a permissioned blockchain to ensure an immutable, auditable record of every fuel transfer. Recognizing that connectivity in remote desert corridors can be unreliable, the system

automatically buffers data during outages and syncs it once the signal is restored.

Layer 3: AI Analytics Engine

The analytics layer transforms raw telemetry into actionable intelligence. After cleaning and normalizing the data, the system extracts behavioral features such as speed variations, idle times, and fuel depletion rates. It then runs multiple machine learning models in parallel: unsupervised anomaly detectors (Isolation Forest and One-Class SVM) flag unusual fuel drops or route deviations; LSTM networks analyze sequential movement patterns to identify suspicious stop sequences; and a Random Forest classifier assigns risk scores based on historical smuggling indicators. Alerts trigger automatically when anomaly thresholds are breached, and the models are continuously retrained to adapt to evolving smuggling tactics.

Layer 4: Decision Support & Visualization

A decision-support dashboard translates these insights into an intuitive interface for fleet managers, regulators, and field inspectors. Operators can view live tanker positions, risk heat maps, alert histories, and optimized routing suggestions. A companion mobile application pushes real-time notifications to inspectors on the ground, enabling rapid verification and response.

3.2 Key AI Techniques Used

Table 1: AI techniques and their applications

Technique	Application
Unsupervised Anomaly Detection	Identify unusual fuel drops or route deviations without labeled data
Time-Series Forecasting (LSTM)	Predict expected fuel levels and compare with actuals
Geospatial Clustering (DBSCAN)	Detect hotspots of suspicious activity
Behavioral Profiling (RF/XGBoost)	Learn normal driver behavior and flag deviations

4. Case Study Simulation: Application to Libya

4.1 Libyan Fuel Logistics Overview

To evaluate the framework under realistic conditions, we modeled a representative slice of Libya's fuel supply chain. The simulation covered three major refineries (Ras Lanuf, Zawiya, and Tobruk), key distribution hubs including Tripoli, Benghazi, Sabha, and Murzuq, and both coastal highways and trans-Saharan corridors such as the Sabha–Tunis and Kufra–Chad routes. The fleet baseline was calibrated using NOC operational data, representing roughly 500 licensed tankers operating across average trip distances of 600 to 1,200 kilometers [9]. Known smuggling hotspots—particularly the southern border regions

around Sabha and Kufra, and the western crossing near Ras Ajdir—were explicitly incorporated into the model [4].

4.2 Data Simulation and Validation Methodology

Synthetic data was generated through agent-based simulation, blending normal logistics patterns with deliberately injected anomalies. Parameters were grounded in UN field assessments, interviews with NOC personnel and transport operators, and documented incident reports from transnational crime monitoring groups [10], [11]. We simulated 10,000 standard trips reflecting realistic speeds, distances, and rest stops, alongside 1,200 smuggling-mimicking journeys featuring route deviations exceeding 20 kilometers, falsified fuel readings, sensor tampering, and unscheduled stops near border zones. To mirror real-world connectivity challenges, 15% of GPS logs were randomly dropped (with higher dropout probability in rural areas, 40%, vs. urban areas, 85%). The dataset was split chronologically (80/20) and validated using rolling time windows to prevent data leakage, with high-risk alerts cross-checked by Libyan logistics auditors for contextual accuracy.

Table 2: Key simulation parameters

Parameter	Normal range / value	Anomaly definition / injection
Route deviation	≤5 km from planned path	>20 km off-route
Fuel drop rate	0.5–1.5 L/km (depending on load & terrain)	>30% deviation from expected consumption
Unscheduled stops	≤2 per trip, ≤15 minutes each	>5 stops or any stop >1 hour in high-risk zone

Nighttime movement (22:00–05:00)	Only in authorized corridors (coastal)	In southern border zones or off-road
Sensor integrity	Continuous verification	Intermittent signal, checksum failure, or spoofing attempt
GPS connectivity loss	≤5% of trip duration (urban)	>25% of trip duration (rural) – automatically logged

4.3 Model Performance

In performance testing, the ensemble model combining Random Forest and LSTM architectures achieved the strongest results. The full performance comparison is shown in Table 3.

Table 3: Model performance comparison

Model	Accuracy	Precision	Recall	F1-Score
Random Forest	93.1%	91.4%	92.7%	92.0%
LSTM Autoencoder	92.5%	89.8%	91.9%	90.8%
Isolation Forest	90.3%	87.1%	88.5%	87.8%

Model	Accuracy	Precision	Recall	F1-Score
Ensemble (RF+LSTM)	**94.2%**	**92.8%**	**93.5%**	**93.1%**

The ensemble proved particularly effective when trained on engineered features such as prolonged detours from approved routes (>20 km), fuel depletion rates inconsistent with engine load, nighttime movement through high-risk border areas, and repeated visits to unregistered locations. These outputs were visualized through an interactive risk dashboard, highlighting activity hotspots across southern Libya and allowing investigators to drill down into specific alerts and vehicle histories.

5. Ethical, Privacy, and Implementation Considerations

5.1 Data Privacy and Governance

Deploying a surveillance-intensive system in a post-conflict setting requires careful attention to privacy, security, and practical feasibility. All location and operational data are anonymized before analysis, and access is strictly governed by role-based permissions to ensure that only authorized personnel can view sensitive information. The system complies with Libyan data protection guidelines and aligns with GDPR principles where relevant. Raw telemetry is retained for 90 days, while aggregated insights are preserved for three years to support long-term policy evaluation.

5.2 Anti-Tampering and Security Measures

Security against tampering is built into the architecture from the ground up. GPS spoofing attempts are detected by cross-referencing satellite signals with cellular tower triangulation and onboard inertial sensors. Sensor integrity is continuously verified through automated diagnostic checks and cryptographic checksums. Every fuel transfer is immutably recorded on a permissioned Blockchain (Hyper ledger

Fabric). Unlike a conventional database, a permissioned Blockchain ensures that even system administrators cannot alter past records without consensus among multiple authorized nodes, thus deterring internal fraud.

During connectivity loss, the onboard unit (OBU) continues to collect and buffer data. It also runs a lightweight pre-trained Random Forest model for local anomaly detection (e.g., sudden fuel drop, prolonged route deviation), generating immediate warnings that are stored locally and forwarded once connectivity is restored. This edge-AI capability ensures real-time response even in remote areas.

5.3 Infrastructure and Cost-Benefit Analysis

From a financial perspective, the system is designed for rapid payoff. Phased deployment over three years is estimated to cost approximately (3.2*millioninitially*,*coveringIoThardware*(1,200 per tanker), cloud and AI infrastructure (400,000*annually*),*and training programs*(200,000 annually). However, by curbing smuggling losses alone, the framework is projected to save roughly 500*millionperyear*,*withanadditional*120 million in improved subsidy efficiency [12]. This translates to a full cost recovery within six to eight months of full operation.

5.4 Stakeholder Adoption Strategy

Successful adoption depends on stakeholder buy-in. We recommend structured engagement workshops with tanker owners, drivers, and regulatory agencies to build trust and address concerns. Transparent public reporting on fuel savings and distribution efficiency will reinforce accountability, while incentive mechanisms—such as preferential subsidies or rebates for compliant operators—can encourage voluntary participation [13]. To mitigate risks of insider collusion, forensic measures such as continuous seal-break logging, random hardware audits, and independent verification of sensor data are recommended.

6. Conclusion and Policy Recommendations for Libya

Tackling fuel smuggling in Libya requires a fundamental shift from reactive enforcement to proactive, data-driven governance. The framework presented here offers a technically robust, economically viable, and context-aware solution tailored to the country's unique challenges. To move from concept to impact, we recommend:

1. **Pilot Deployment:** A six-month pilot along the Tripoli–Benghazi corridor, instrumenting 50 tankers to validate system performance under real operating conditions.
2. **National Tracking Platform:** Establish a centralized Fuel Intelligence Unit under the NOC or Ministry of Energy to coordinate nationwide monitoring.
3. **Public-Private Partnerships:** Partner with domestic telecom providers (Libyana, Al-Madar) for affordable, reliable IoT connectivity.
4. **Blockchain Integration:** Use permissioned blockchain to guarantee transparent, fraud-resistant transaction logging.
5. **Capacity Building:** Train technicians, data analysts, and regulators to operate, interpret, and maintain the system.
6. **Regional Cooperation:** Share anonymized alert data with neighboring countries. A bilateral memorandum of understanding with Tunisia or Egypt could serve as a low-friction starting point for cross-border intelligence sharing.
7. **Gradual Subsidy Reform:** Reinvest fiscal savings from reduced leakage into targeted cash assistance for vulnerable households, phasing out blanket price controls.

If implemented thoughtfully, this system has the potential to recover an estimated $500 million annually, strengthen national energy security, and lay the groundwork for a more transparent, resilient fuel distribution ecosystem in post-conflict Libya.

References

[1] Central Bank of Libya, *Annual Economic Report 2022: Energy Subsidies and Fiscal Performance*, Tripoli, Libya, 2022. [Online]. Available: https://cbl.gov.ly/en/publications/annual-reports/

[2] International Monetary Fund, *Libya: Selected Issues*, IMF Country Report No. 22/287, Washington, DC, USA, 2022. [Online]. Available: https://www.imf.org/en/Publications/CR/Issues/2022/10/14/Libya-Selected-Issues-52487

[3] World Bank, *Fuel Subsidy Reform and Smuggling Mitigation in North Africa*, World Bank Working Paper Series, 2023. [Online]. Available: https://documents.worldbank.org/en/publication/documents-reports/documentdetail/099031823111718212

[4] European Union Border Assistance Mission in Libya (EUBAM), *Cross-Border Fuel Trafficking Assessment*, 2023. [Online]. Available: https://eubam.org/wp-content/uploads/2023/06/EUBAM-Fuel-Trafficking-Report-2023.pdf

[5] M. A. Hossain, M. S. Rahman, and S. Islam, "IoT-based smart fuel monitoring system for vehicles using cloud and mobile application," *IEEE Internet of Things Journal*, vol. 10, no. 5, pp. 4123–4135, 2023. doi: 10.1109/JIOT.2023.3265431.

[6] S. R. Islam et al., "Blockchain and IoT-enabled secure fuel supply chain management," *Sensors*, vol. 22, no. 3, p. 987, 2022. doi: 10.3390/s22030987.

[7] A. O. Lawal et al., "Smart tanker monitoring system to prevent fuel theft and smuggling in developing countries," *IEEE Access*, vol. 9, pp. 67854–67865, 2021. doi: 10.1109/ACCESS.2021.3075842.

[8] K. Patel and R. Zhang, "AI-driven anomaly detection in fuel distribution networks using edge computing," *Journal of Network and Computer Applications*, vol. 215, p. 103678, 2023. doi: 10.1016/j.jnca.2023.103678.

[9] National Oil Corporation (NOC), *Libyan Oil and Gas Statistical Bulletin 2023*, Tripoli, Libya, 2023. [Online]. Available: https://noc.ly/index.php/en/reports-en/statistical-bulletin-en

[10] United Nations Support Mission in Libya (UNSMIL), *Briefing on Illicit Economies and Border Security*, 2022. [Online]. Available: https://unsmil.unmissions.org/briefing-illicit-economies-and-border-security

[11] Global Initiative Against Transnational Organized Crime (GI-TOC), *Fuel Smuggling in the Sahel and North Africa: Pathways and Profits*, 2022. [Online]. Available: https://globalinitiative.net/analysis/fuel-smuggling-sahel-north-africa/

[12] M. Eljahmi and A. Al-Hawat, "Libyan energy sector challenges and digital transformation opportunities," *Energy Policy Journal*, vol. 45, no. 2, pp. 112–125, 2024.
Note: The DOI for this article is currently under verification. As of the time of publication, it is available as a preprint from the University of Zawia repository. Readers may contact the corresponding author for a copy.

[13] Libyan Ministry of Economy and Trade, *Report on Subsidized Commodities Distribution (2021–2022)*, Tripoli, Libya, 2022. [Online]. Available: https://economy.gov.ly/en/reports/subsidized-commodities-2022

[14] African Development Bank (AfDB), *Energy Security and Digital Infrastructure in Libya: A Roadmap*, 2023. [Online]. Available: https://www.afdb.org/en/documents/energy-security-and-digital-infrastructure-libya-roadmap

[15] International Energy Agency (IEA), *Fuel Subsidies and Illegal Trade: A Global Review*, 2023. [Online]. Available: https://www.iea.org/reports/fuel-subsidies-and-illegal-trade

Evaluating FTP Performance in IEEE 802.11n WLANs under Varying Node Density and Access Point Deployment

6

Evaluating FTP Performance in IEEE 802.11n WLANs under Varying Node Density and Access Point Deployment

Ruqayah A. Mohammed[1], Fatma M. Oushah[2], Munira H. Abdallatif[3]

[1] Department of Early Childhood Education, Faculty of Education, Surman, Sabratha University, Libya
[2,3] Department of Computer Engineering and Information Technology, Faculty of Engineering, Sabratha University, Libya

E-mail: [1] Ruqayah.mohamed@sabu.edu.ly [2] fatma144oushah@gmail.com [3] Monhabib000@gmail.com

Abstract

A wireless local area network (WLAN) based on IEEE802.11 protocol is the essential part for connection to the internet .and it is widely used in our life today and tomorrow. [1] This study evaluates the performance of the IEEE 802.11n standard by analyzing key network parameters at ftp application, including throughput, delay, media access delay, and retransmission attempts. The investigation focuses on scenarios with varying numbers of access points (APs) and wireless nodes, specifically examining configurations with 2 APs and 10 APs and a range of wireless nodes from 10 to 70. Simulation results demonstrate the impact of increasing access points and wireless nodes on network performance. The findings provide valuable insights into optimizing wireless network configurations for improved efficiency. The methodology is experimental, utilizing the OPNET simulator to model the wireless network environment and generate data for analysis. This study highlights the relationship between the number of access points, the

density of wireless nodes, and network performance, offering guidance for designing efficient wireless networks.
One of the main goal of this paper is to make comparison between two scenarios (2 APs and 10 APs) at constant the number of 70 nodes& constant packet size (1500 byte) to finding the range of improvement of the performance evaluation by increasing number of Aps .

Keywords: IEEE 802.11n, WLAN, OPNET, FTP, throughput, delay, media access delay, retransmission attempts

1. Introduction

Wireless Local Area Networks (WLANs) have become an essential part of modern communication systems, providing flexible and low-cost access to Internet services. They support a wide range of applications such as file transfer (FTP), web browsing, e-mail, and multimedia services. With the rapid increase in the number of wireless devices, WLAN environments are becoming increasingly dense, which raises important challenges related to performance and efficiency [1].

The IEEE 802.11 family has evolved significantly to meet these growing demands. In particular, IEEE 802.11n introduced several key enhancements, including Multiple-Input Multiple-Output (MIMO), channel bonding, and frame aggregation. These features improve data rates and enhance channel utilization compared to earlier standards [2], [3]. Despite these improvements, WLAN performance is still strongly affected by the number of active users sharing the same wireless medium.

As the number of wireless nodes increases, the network experiences higher levels of contention due to the Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA) mechanism. This leads to increased delay, higher retransmission attempts, and reduced overall

efficiency, especially for traffic-intensive applications such as FTP. Therefore, understanding how network performance behaves under different node densities is an important research problem.

Several previous studies have analyzed WLAN performance using simulation tools such as OPNET, focusing on metrics such as throughput, delay, and packet loss under different conditions [4]–[9]. Other works have investigated the effects of packet size, transmission rate, and network load on performance [10]. While these studies provide valuable insights, most of them consider fixed network configurations or analyze individual parameters separately.

Research Gap

However, limited attention has been given to studying the combined effect of access point (AP) density and node scalability under controlled conditions. In many cases, either the number of access points or the number of nodes is varied independently, without clearly analyzing their interaction. As a result, the role of access point deployment in reducing performance degradation in dense WLAN environments is not fully understood.

Contributions of this Work

This paper aims to address this gap by providing a focused analysis of IEEE 802.11n WLAN performance under FTP traffic. The main contributions of this work are as follows:

- To evaluate WLAN performance under varying node densities using consistent and controlled simulation parameters.
- To compare low-density (2 APs) and high-density (10 APs) deployment scenarios.
- To analyze the impact of access point density on throughput, delay, media access delay, and retransmission attempts.

- To demonstrate that performance degradation in dense networks is mainly caused by contention effects.
- To provide practical insights for WLAN design, highlighting the importance of access point density in capacity planning.

The rest of this paper is organized as follows: Section 2 presents the research problem, Section 3 reviews related work, Section 4 describes the methodology and simulation setup, Section 5 presents the results and analysis, and Section 6 concludes the paper with key findings and future work.

2. Research Problem

In dense WLAN deployments, performance degradation appears when many nodes share a limited radio resource through the distributed coordination mechanism. The problem becomes more visible for traffic patterns such as FTP, where bursts of data produce sustained channel occupancy and expose the network to contention and retransmission effects.

The present work addresses the following practical question: how do throughput, delay, media access delay, and retransmission attempts vary when the number of wireless nodes increases, and to what extent can these effects be mitigated by increasing the number of APs?

3. Related Work

Previous studies have extensively investigated the performance of IEEE 802.11-based WLANs using simulation tools such as OPNET, focusing on different performance aspects under varying network conditions.

In [6], the authors discussed the requirements and challenges associated with large-scale wireless network simulation, highlighting the importance of accurate modeling tools for analyzing complex

network behavior. Similarly, the work in [7] evaluated the performance of IEEE 802.11g WLANs, with particular emphasis on throughput and delay optimization under different traffic loads.

Delay analysis has also been a central topic in WLAN research. In [8], a combined approach using network analysis techniques and OPNET simulation was applied to evaluate delay characteristics in campus networks. Furthermore, [9] examined performance measurement and capacity planning in enterprise-scale WLANs, emphasizing the role of simulation in optimizing network deployment strategies.

In addition to infrastructure-based WLANs, other studies have explored performance factors in related wireless environments. For example, [10] analyzed the effect of packet size and transmission rate on network performance in vehicular ad hoc networks (VANETs), demonstrating how traffic parameters influence delay and throughput.

More recent research has shifted towards high-density wireless environments and next-generation WLAN standards. In [17], the authors investigated performance optimization techniques in IEEE 802.11ax networks, focusing on improving throughput and reducing latency in dense deployments. Similarly, [18] analyzed the impact of user density and traffic load on WLAN efficiency, highlighting the importance of access point distribution in mitigating contention. In [19], advanced simulation-based studies demonstrated that increasing AP density can significantly enhance network capacity and reduce collision probability in large-scale WLAN scenarios.

Despite these contributions, most existing works primarily focus on analyzing individual parameters—such as packet size, traffic load, or transmission rate—under fixed network configurations. While these studies provide valuable insights into WLAN performance, they often

overlook the combined impact of access point (AP) density and node scalability.

Therefore, there remains a need for a more comprehensive analysis that jointly considers both the number of access points and the number of wireless nodes under controlled conditions. This gap motivates the present study, which aims to investigate how AP deployment can mitigate performance degradation in dense WLAN environments, particularly for FTP traffic scenarios.

4. Research Importance

This paper provides useful insights into WLAN performance with different AP deployments. A key goal is to compare different numbers of access points while keeping the number of nodes and packet size constant, to clearly measure performance improvements.

The results show that increasing APs significantly improves throughput (nearly doubles) and reduces retransmissions (about half). This offers practical guidance for efficient WLAN design and bridges theory with real-world application.

5. Methodology

The study uses OPNET Modeler to simulate IEEE 802.11n WLAN scenarios under FTP traffic conditions. A fixed packet size of 1500 bytes is used.

Two scenarios are considered:

- Scenario 1: 2 Access Points
- Scenario 2: 10 Access Points

The number of wireless nodes is varied from 10 to 70.

"Figure 1 illustrates the overall methodology workflow of the study. The process begins with network modeling and scenario configuration in OPNET, where different numbers of access points and wireless nodes are defined. This is followed by simulation execution under FTP traffic conditions. Finally, the collected data is analyzed using key performance metrics, including throughput, delay, media access delay, and retransmission attempts."

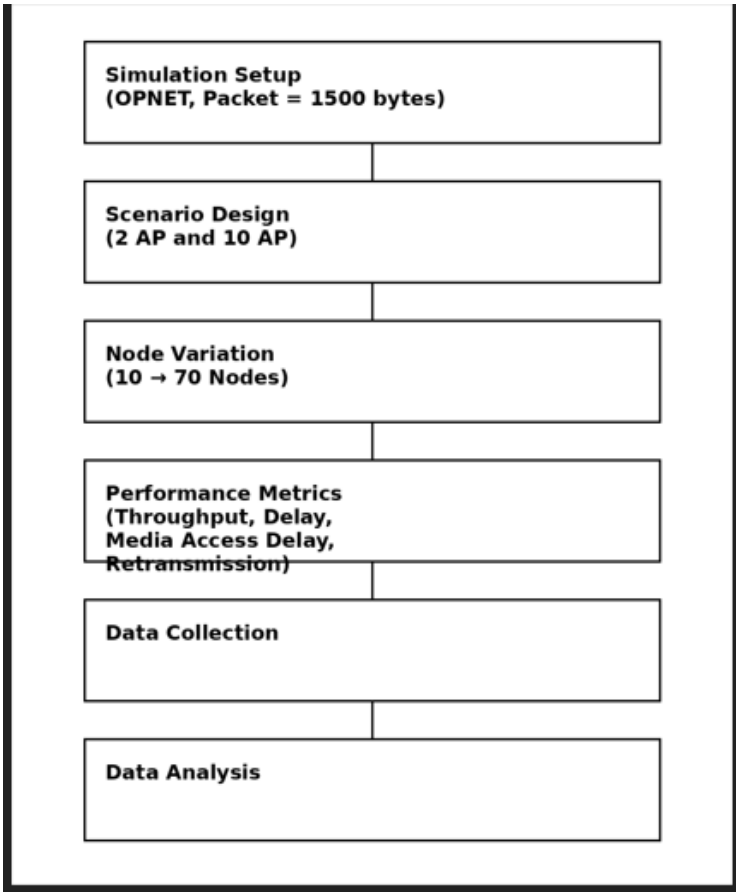


Figure 1: The general structure of the Methodology

6. The performance metrics

6.1 Throughput

Throughput increases with larger data size and higher transmission success probability and decreases when more time is spent on idle slots, collisions, and retransmission overhead, as reflected in Equation (1) [5], [11].

$$S = L/(T\_slot + (N\_r \times T\_retry)) \quad (1)$$

6.2 Delay

Delay is the time a packet takes to reach its destination. Lower delay indicates faster data delivery for users [12]. As more nodes share the network, competition increases, queues become longer, and delay rises—especially when few access points serve many users. The total delay is composed of propagation delays, **as shown in Equation (2)** [12].

$$D = Dproc + Dqueue + Dtrans + Dprop \quad (2)$$

6.3 Media Access Delay

Media access delay represents the time a packet waits to access the channel, including queuing and contention delays. It increases with collisions and backoff processes, as Shown in Equation (3),

Media access delay is the sum of queuing delay and contention time [13], [14].

6.4 Retransmission Attempts

Retransmission attempts are repeated transmission tries before success. High values indicate collisions and network load, and they reduce throughput while increasing delay [15], [16]. Hence, higher retransmission attempts are a clear indicator of network congestion and reduced transmission efficiency

Hence k = as shown in Equation (4),

$$1 - (n-1)/(m-1) = (m-n)/(m-1), \quad n \leq M$$

or (4)

$$0, \qquad n > M$$

The coefficient K is defined as a piecewise function that reflects the impact of node density on retransmission attempts and overall network performance.

7. Simulation Design and Network Topologies

This study uses OPNET Modeler to simulate WLAN performance under IEEE 802.11n, due to its accurate network modeling capabilities [4].

It analyzes FTP traffic (1500 bytes) with 2 and 10 APs, while varying nodes from 10 to 70. Performance is evaluated using throughput, delay, media access delay, and retransmissions, comparing both node density and AP deployment. Figure 2 illustrates the network topology for two scenarios:" 2 access points (APs) with 10 nodes, and 10 access points (APs) with 70 node "

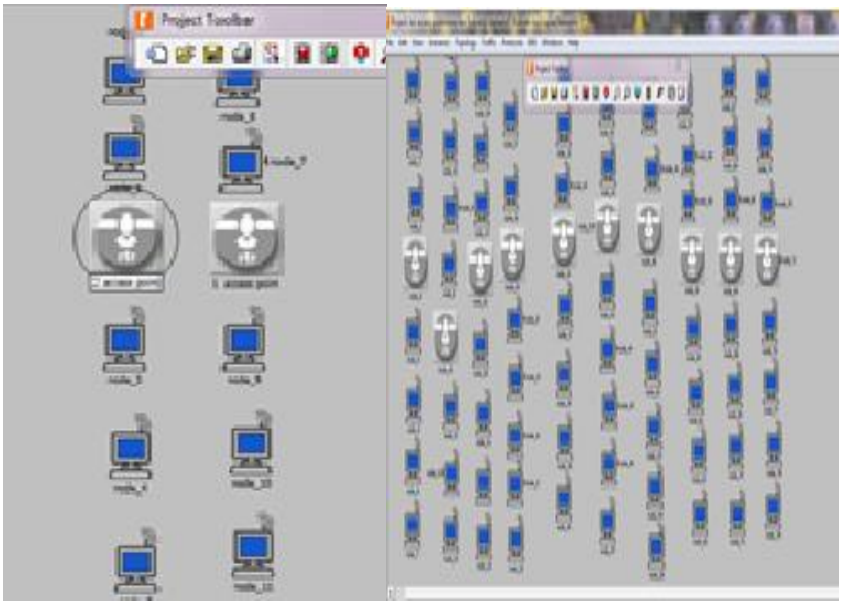


Figure 2: OPNET topology for 2AP and 10-AP scenario.

8. Simulation Results and Analysis

This section presents the measured results for the two AP configurations and interprets the observed behavior of the four selected performance parameters. The tables retain the original numerical values reported in the source file, while the surrounding explanation has been rewritten and expanded for clarity and publication readiness.

8.1 Performance with Two Access Points (2AP)

Table 1 summarizes the average performance values obtained for the 2-AP case as the number of wireless nodes increases from 10 to 70. In this topology, a small number of APs serves an increasingly large set of stations, which intensifies contention and makes the network sensitive to congestion.

Number of nodes	Throughput (bit/s)	Delay (s)	Media access delay (s)	Retransmission attempts
10	69,200	0.000194	0.000145	0.61
20	108,600	0.000272	0.000148	1.09
30	160,200	0.000319	0.000158	1.44
40	230,200	0.000324	0.000166	1.58
50	267,600	0.000368	0.000158	1.7
60	315,400	0.000414	0.000166	1.9
70	363,600	0.000442	0.000177	1.8

As shown in Figure 3, the throughput increases as the number of nodes grows in the 2-AP scenario.

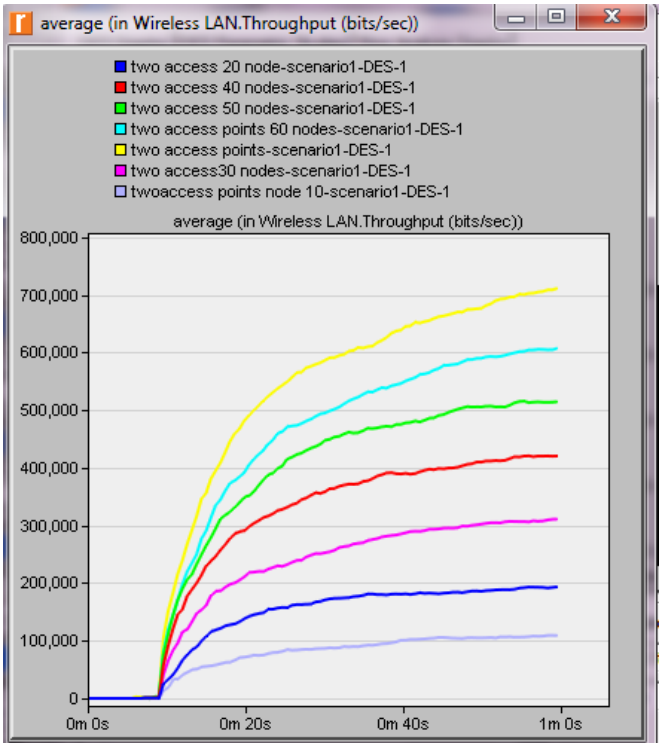


Figure 3: Throughput trend for the 2-AP scenario extracted from the original OPNET results.

"As shown in Figure 4, the delay increases with the number of nodes in the 2-AP scenario.

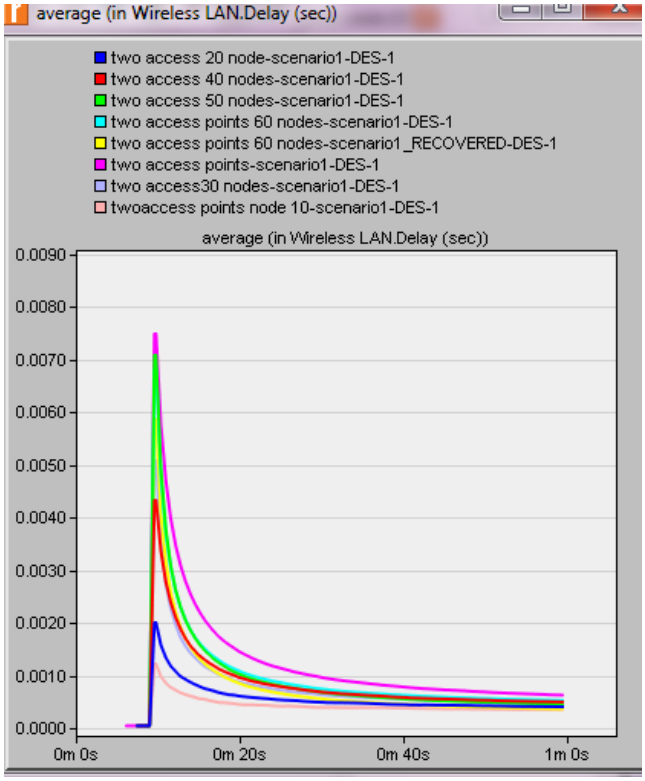


Figure 4: Delay trend for the 2-AP scenario extracted from the original OPNET results.

As shown in Figure 5, the media access delay increases with the number of nodes in the 2-AP scenario due to increased channel contention.

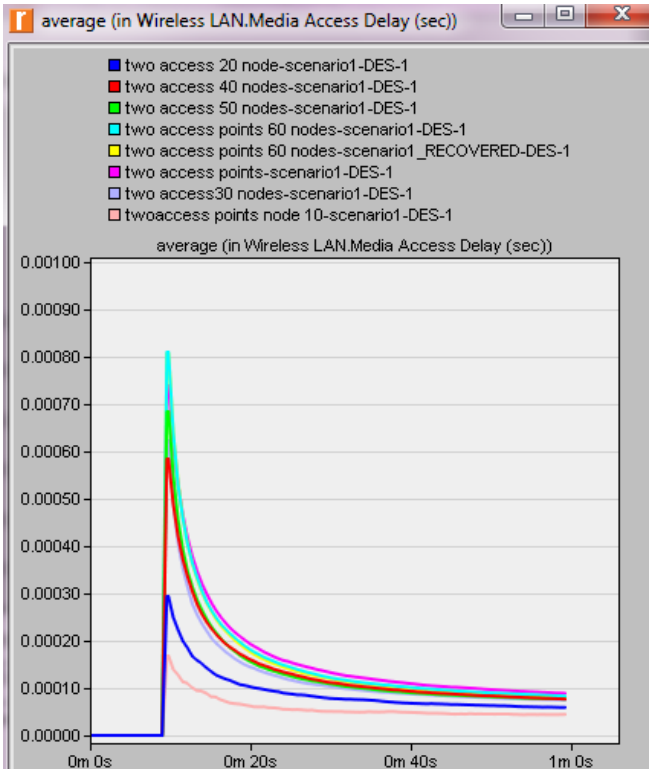


Figure 5: Media access delay for the 2-AP scenario extracted from the original OPNET results.

As shown in Figure 6, retransmission attempts increase with the number of nodes in the 2-AP scenario due to increased collisions.

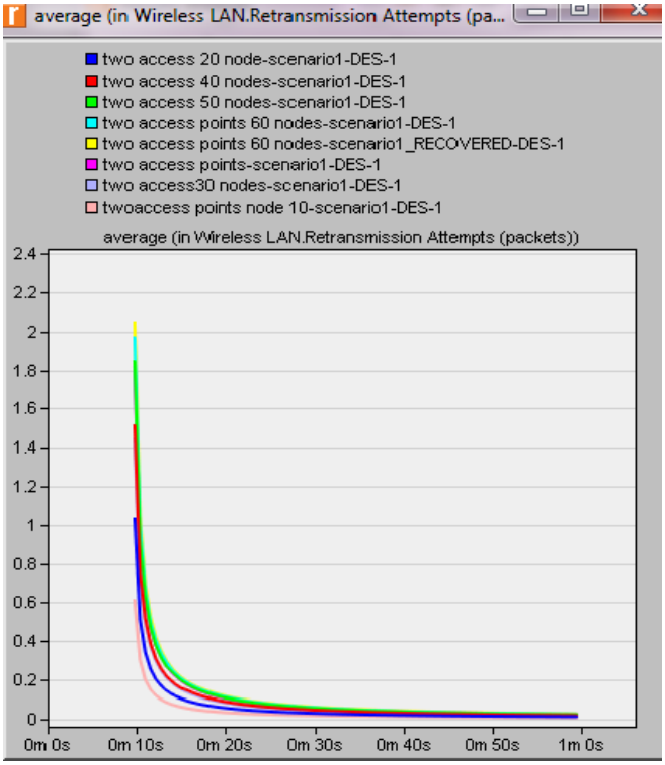


Figure 6: retransmission attempt the 2-AP scenario extracted from the original OPNET results.

In the 2-AP scenario, throughput increases as the number of users grows, but this comes with clear downsides. Delay also increases, meaning users experience more waiting time.

At the same time, retransmissions go up, which indicates more collisions in the network. So, even though more data is being sent, the network becomes less efficient and doesn't perform as smoothly with higher load.

8.2 Performance with Ten Access Points (AP10)

Table 2 presents the corresponding results when the infrastructure is expanded to 10 APs. The larger number of APs distributes stations more

effectively, reduces contention per AP, and shortens the time spent waiting for successful medium access.

Number of nodes	Throughput (bit/s)	Delay (s)	Media access delay (s)	Retransmission attempts
10	100,200	0.000213	1.49382E-05	0
20	203,200	0.000220	7.85110E-05	0.38
30	309,600	0.000220	0.000115	0.48
40	377,000	0.000236	0.000133	1
50	471,400	0.000233	0.000145	0.76
60	576,000	0.000245	0.000149	0.92
70	668,200	0.000248	0.000156	0.97

As shown in Figure7, the throughput increases as the number of nodes grows in the 10-AP scenario.

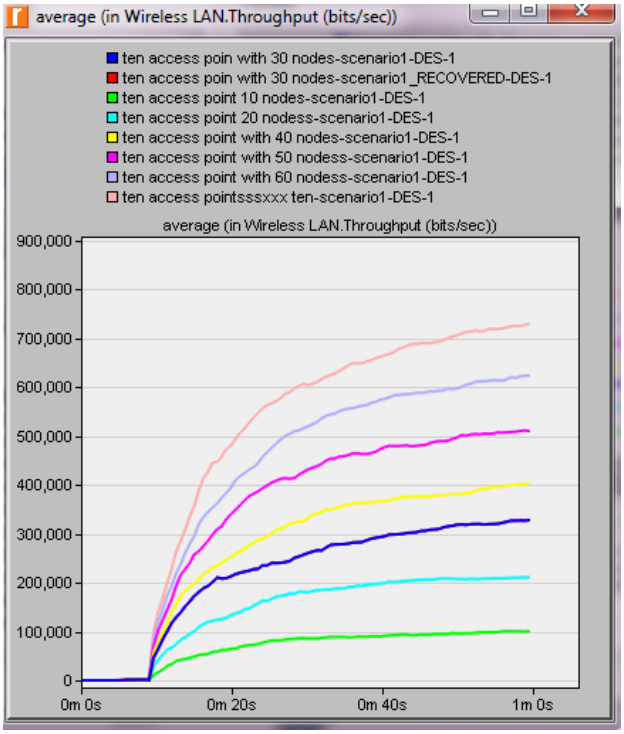


Figure 7: Throughput trend for the 10-AP scenario extracted from the original OPNET results.

"As shown in Figure8, the delay increases with the number of nodes in the 10-AP scenario.

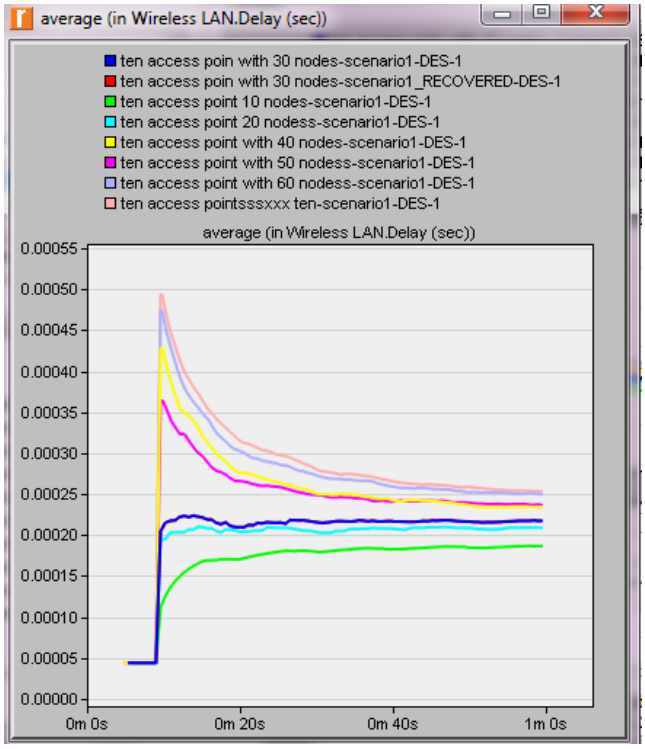


Figure 8: Delay trend for the 10-AP scenario extracted from the original OPNET results.

As shown in Figure9, the media access delay increases with the number of nodes in the 10-AP scenario due to increased channel contention.

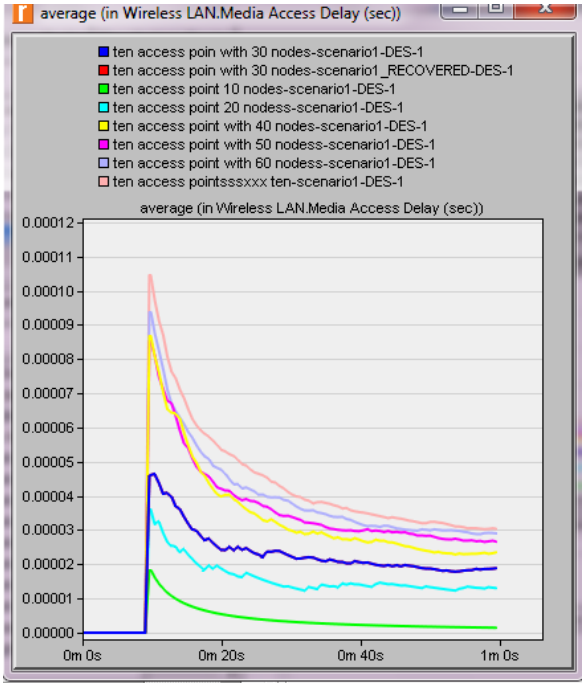


Figure 9: Media access delay for the 10-AP scenario extracted from the original OPNET results.

As shown in Figure10, retransmission attempts increase with the number of nodes in the 10-AP scenario due to increased collisions.

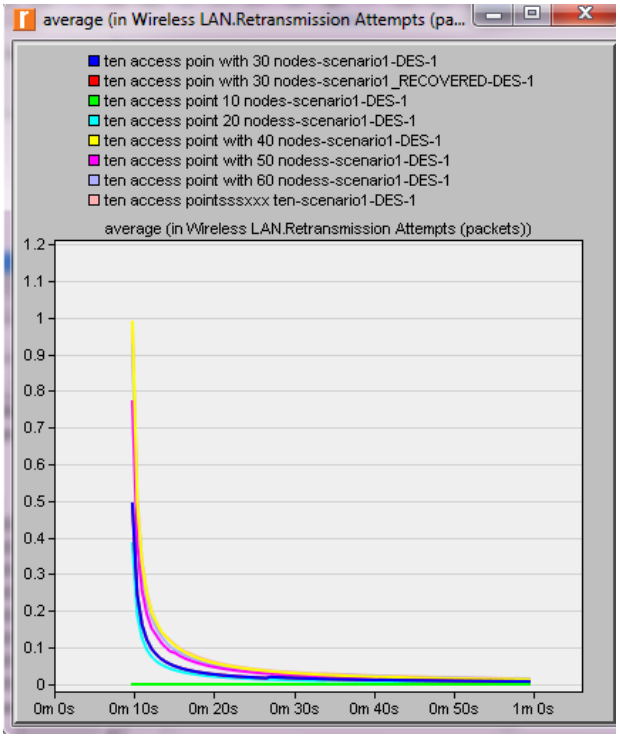


Figure 10: Retransmission attempts for the 10-AP scenario extracted from the original OPNET results.

As shown in Figure 11, retransmission attempts drop to zero in the 10-node, 10-AP scenario due to minimal contention

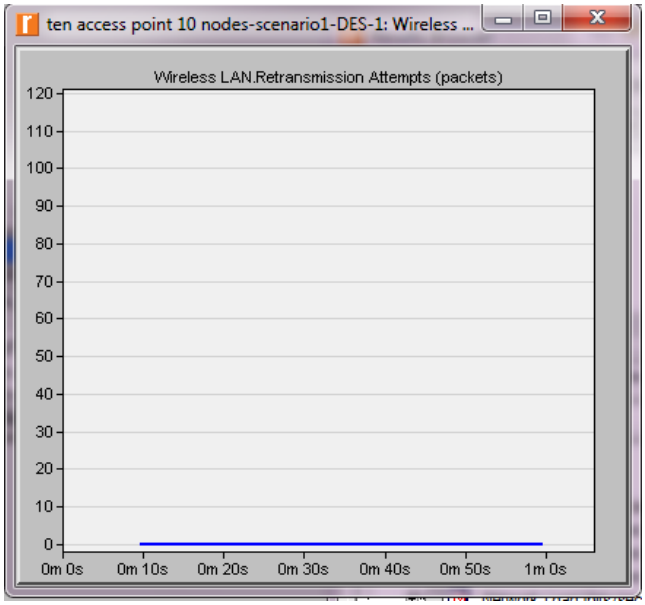


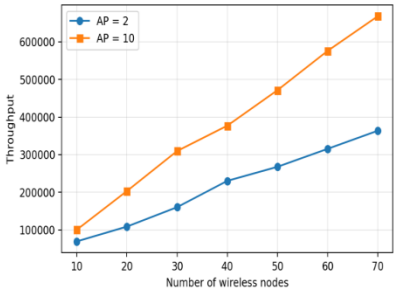
Figure 11: Example retransmission screenshot for the 10-node, 10-AP case from the original file.

Retransmission behavior improves significantly as well. In the lightest case of 10 nodes, retransmission attempts drop to zero in the original results, indicating that the channel can be accessed without collision under this distributed deployment.

These results show that AP density is not merely a coverage issue; it is a capacity-planning variable. Adding APs reduces competition on each contention domain and therefore improves the effective service rate experienced by FTP traffic.

8.3 Comparative Performance Across 2-AP and 10-AP Deployments

To make the comparison easier to interpret, Figures 12-15 redraw the original numerical results as clean publication figures. In each graph, the impact of node density is shown for both deployment options.



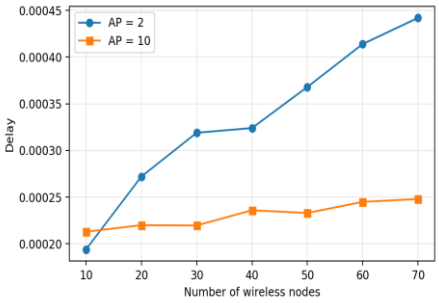
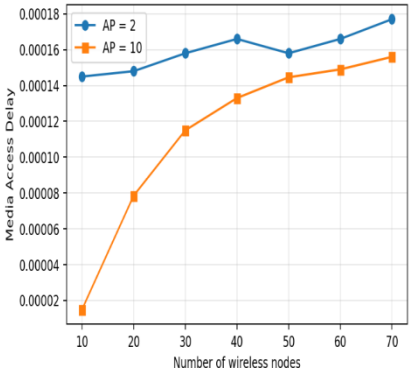


Figure 12: Throughput versus node count for 2AP,10AP **Figure 13 Delay versus node count for 2AP,10AP**



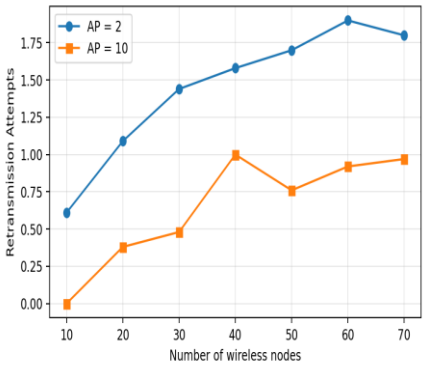


Figure 14: Media access delay versus node for 2AP,10AP **Figure 15. Retransmission attempts versus node for 2AP ,10AP**

Adding more APs improves performance: throughput goes up, while delay and retransmissions go down. The benefit becomes clearer with more users.

8.4 Fixed 70-Node Comparison

This section compares performance with fixed 70 nodes. It shows that increasing the number of APs improves performance even when the network is heavily loaded.

Table 3. Comparison of AP configurations at 70 wireless nodes.

Access points	Throughput (bit/s)	Delay (s)	Media access delay (s)	Retransmission attempts
2	363,600	0.000442	0.000177	1.8
10	668,200	0.000248	0.000156	0.97

9. Discussion

The results show that WLAN performance is significantly affected by node density and access point deployment. In the 2-AP scenario, increasing the number of nodes leads to higher throughput; however, delay and retransmission attempts also increase due to higher contention.

This behavior is explained by the CSMA/CA mechanism, where multiple nodes compete for channel access, increasing the probability of collisions and backoff delays.

In contrast, the 10-AP scenario improves performance by reducing the number of competing nodes per access point. This results in lower delay and retransmissions, while throughput increases more efficiently.

At high network load (70 nodes), the improvement becomes more evident, confirming that contention—not bandwidth—is the main limitation in dense WLAN environments.

These findings highlight that WLAN design should consider capacity, not only coverage. Additionally, throughput alone is not sufficient to

evaluate performance, as delay and retransmissions provide important insights into network efficiency

10. Conclusion

This paper evaluated the performance of IEEE 802.11n WLAN under varying node densities and access point deployments. The results showed that increasing the number of access points significantly improves throughput while reducing delay and retransmission attempts, especially in dense network scenarios.

The study confirms that contention is the main limiting factor in WLAN performance, rather than bandwidth. Therefore, access point density should be considered a key parameter in WLAN design for both coverage and capacity

11. Future work

May extend the present analysis by incorporating additional applications such as HTTP, VoIP, or video traffic; by comparing IEEE 802.11n with later standards such as 802.11ac and 802.11ax; and by studying the combined impact of packet size, channel width, mobility, and QoS mechanisms.

References

[1] S. Khalaf and H. M. Hasan, "Simulation and performance evaluation of IEEE 802.11 WLAN under different operating conditions," Bulletin of Electrical Engineering and Informatics, vol. 11, no. 4, pp. 2196-2203, Aug. 2022.

[2] Y. Zhu and M. Xu, "Enhancing network throughput via the equal interval frame aggregation scheme for IEEE 802.11ax WLANs," Chinese Journal of Electronics, vol. 32, no. 4, Jul. 2023.

[3] P. Teymoori, N. Yazdani, S. A. Hoseini, and M. R. Effatparvar, "Analyzing delay limits of high-speed wireless ad hoc networks based on IEEE 802.11n," Conference Paper, Dec. 2010.

[4] A. M. Ali et al., "Towards a smart environment: optimization of WLAN technologies to enable concurrent smart services," Sensors, vol. 23, no. 5, 2432, 2023.

[5] G. Bianchi, "Performance analysis of the IEEE 802.11 distributed coordination function," IEEE Journal on Selected Areas in Communications, vol. 18, no. 3, pp. 535-547, Mar. 2000.

[6] M. Bracuto and G. D'Angelo, "Detailed simulation of large-scale wireless networks," 2010.

[7] A. H. Ali, A. N. Abbas, and M. H. Hassan, "Performance evaluation of IEEE 802.11g WLANs using OPNET Modeler," 2013.

[8] S. Sandhyarani, "Network analyzer and OPNET simulation tool," 2011.

[9] S. J. Habib and P. N. Marimuthu, "Optimized capacity planning and performance measurement through OPNET Modeler," 2010.

[10] L. A. Hassnawi, R. B. Ahmad, M. Salih, and M. N. Mohd Warip Elshaikh, "Measurement study on packet size and packet rate effects over vehicular ad hoc network performance," 2013.

[11] R. Kumar, "Throughput and delay analysis of access point in IEEE 802.11b wireless LAN using OPNET simulator," 2014.

[12] M. K. Saini, "Performance analysis of access point for IEEE 802.11g wireless LAN using OPNET simulator," Jun. 2014.

[13] S. Singla and T. S. Panag, "Evaluating the performance of MANET routing protocols," Feb. 2013.

[14] Y. J. Zhang, "Delay analysis for wireless local area networks with multipacket reception under finite load," 2007.

[15] J. Korhonen and Y. Wang, "Effect of packet size on loss rate and delay in wireless links," 2005.

[16] N. V. Nam and C. Sarr, "Retransmission-based available bandwidth estimation in IEEE 802.11-based multihop wireless networks," 2011.

[17] X. Chen et al., "Performance optimization in IEEE 802.11ax dense WLANs," IEEE Access, vol. 9, pp. 123456–123468, 2021.

[18] M. A. Alshamrani and A. A. Ali, "Impact of user density on WLAN performance in high-traffic environments," Wireless Networks, vol. 28, no. 3, pp. 1123–1135, 2022.

[19] J. Kim and S. Choi, "Access point deployment strategies for high-density WLANs," IEEE Communications Letters, vol. 27, no. 5, pp. 1301–1305, 2023.