Intechno Journal :
Information Technology Journal
Vol. 8 No. 1 July 2026
Homepage: https://jurnal.amikom.ac.id/index.php/intechno
INTECHNO Journal
INFORMATION TECHNOLOGY JOURNAL
e-ISSN: 2655-1438
p-ISSN: 2655-1632

Flow-Based Encrypted Network Traffic Classification Using Random Forest for Network Access Control

Wahyu Isnan[1], Yulia Fatmi[2], Resmi Darni[3], Vikri Aulia[4]
[1,2,3,4]Universitas Negeri Padang, Jl. Prof. Dr. Hamka Kota Padang, 25171, Indonesia

Article Info

Article history:

Received: 10 July 2026
Revised: 20 July 2026
Accepted: 22 July 2026
Published: 31 July 2026

Keywords:

Encrypted Network Traffic
Flow-Based Features
Random Forest
Network Access Control
Firewall Automation

ABSTRACT

Encrypted network communications reduce the effectiveness of payload based traffic identification and complicate the translation of traffic analysis into network access control decisions. This study evaluates a payload independent workflow that connects flow based multiclass classification with administrator triggered, time limited firewall enforcement. The experiment used the public ISCXVPN2016 benchmark. After removing 18,719 duplicate records, 40,987 unique flows remained; all 23 available numerical flow features were retained without feature selection or normalization. A Random Forest classifier with 300 trees was selected using five-fold cross-validated grid search on a stratified 80% training partition and evaluated on an independent 8,198 sample test set covering 14 VPN and non-VPN traffic classes. The model achieved 88.01% accuracy, 88.00% weighted precision, 88.01% weighted recall, and 87.97% weighted F1-score. It exceeded the strongest reproduced baseline, K-Nearest Neighbors, by 16.09 percentage points in accuracy and 16.29 percentage points in weighted F1-score. Supplemental five-fold evaluation produced a mean accuracy of 88.05% with a 0.36 percentage point standard deviation. The trained classifier was integrated with a web application in which administrators review predicted flows and initiate temporary MikroTik RouterOS rules. All 30 temporary blocking entries observed in the evaluation database reached the Unblocked state with recorded timestamps, demonstrating rule lifecycle traceability at the database level. The findings show that Random Forest can provide competitive flow based classification while supporting an auditable, human controlled access control workflow; however, device level reliability and cross dataset generalizability require further validation.

Corresponding Author:

Wahyu Isnan
Email: wahyuisnan002@gmail.com

1. INTRODUCTION

The rapid expansion of Internet services, cloud computing, mobile applications, and the Internet of Things (IoT) has substantially increased the volume and diversity of encrypted network traffic. Contemporary protocols such as QUIC secured with TLS [1] and HTTP/3 [2] protect the confidentiality and integrity of communications, but they also reduce the application level visibility available to network administrators. Recent research consequently identifies encrypted traffic classification as an important requirement for network monitoring, resource management, and security operations [3].

Traditional traffic identification techniques rely primarily on transport layer port numbers or Deep Packet Inspection (DPI). Port based identification becomes unreliable when applications use dynamic or shared ports,

whereas DPI depends on payload signatures that are inaccessible when traffic is encrypted. DPI also introduces computational overhead, requires continuous signature maintenance, and may conflict with privacy requirements. Azab et al. [4] describe these limitations across contemporary traffic classification techniques, while Wickramasinghe et al. [5] show that encrypted traffic classifiers must also be evaluated carefully to avoid results that do not generalize beyond a specific dataset or experimental configuration.

Flow based classification provides an alternative by representing communication through observable statistical characteristics rather than decrypted payload content. Features such as flow duration, packet and byte rates, packet inter arrival times, and forward/backward timing statistics remain available even when the application payload is encrypted. Recent surveys identify these statistical representations as a major approach to encrypted traffic classification [6]. Public benchmarks such as the ISCX VPN-nonVPN dataset also enable reproducible evaluation of application categories under VPN and non-VPN conditions, although class distribution, preprocessing, and evaluation protocols must be reported explicitly to support valid comparison [7].

Recent encrypted traffic studies have investigated classical machine learning, gradient boosting, and deep learning models. Elmaghraby et al. [8] combined neural representations with Random Forest, support vector machine, k-nearest neighbor, and other classifiers on the ISCX VPN-nonVPN dataset. Bozkır et al. [7] reported strong performance from XGBoost and LightGBM in an end-to-end traffic classification platform, whereas Luo et al. [9] emphasized that classical machine learning models remain relevant when interpretability and computational cost are important deployment constraints. Random Forest is therefore investigated in this study as a deployment-oriented classifier because it supports nonlinear multiclass decision boundaries, is robust to heterogeneous statistical features, provides feature importance information, and can be integrated without the training and inference complexity of deeper architectures. Its effectiveness must nevertheless be established through controlled experimental comparison rather than assumed from algorithmic characteristics alone.

A second challenge concerns how classification results are used after prediction. Salau and Beyene [10] demonstrated that machine learning traffic classifiers can be integrated with software defined networking to support centralized network management decisions. From a complementary perspective, Lee et al. [11] applied machine learning to firewall rule refinement and showed the operational importance of reducing erroneous policy changes. These studies indicate growing interest in connecting intelligent traffic analysis with network control; however, the reliable translation of encrypted application class predictions into auditable, time limited firewall policies remains insufficiently examined. Table 1 summarizes recent studies that define this research gap.

Table 1. State of the Art of Previous Studies

No	Authors	Main Focus	Method	Limitation
1.	Bozkır et al. (2023)	End-to-end encrypted network traffic classification platform	NFStream, GBTree, LightGBM, and XGBoost	Supports the classification lifecycle but does not translate predictions into firewall policy enforcement.
2.	Luo et al. (2023)	Lightweight packet-level encrypted traffic classification	BITization-based feature engineering with classical machine learning models	Focuses on classification and feature engineering without network access control integration.
3.	Elmaghraby et al. (2024)	Encrypted VPN and non-VPN traffic classification	Neural and BiLSTM representations combined with RF, SVM, LR, and KNN	Provides comparative classification results but no operational firewall enforcement.
4.	Salau and Beyene (2024)	Machine learning traffic classification in an SDN environment	Supervised and unsupervised classifiers with Mininet and an SDN controller	Supports centralized network management but does not evaluate time limited, application class driven firewall rules.
5.	Lee et al. (2024)	Machine learning based firewall rule refinement	Classification of real firewall and security event records	Targets firewall policy errors rather than encrypted application flow classification.

As shown in Table 1, recent research has advanced different parts of the problem but has not fully addressed their operational integration. Bozkır et al. [7] provided an end-to-end machine learning lifecycle for

encrypted traffic classification, but the resulting predictions were not converted into firewall policies. Salau and Beyene [10] connected traffic classification with centralized SDN management, but did not examine application-class-driven, time-limited firewall enforcement. Conversely, Lee et al. [11] investigated firewall-policy refinement using security-event data rather than encrypted application-flow classification. Consequently, an empirical gap remains between flow-level encrypted traffic identification and policy enforcement with explicit rule-lifecycle management.

To address this gap, this study proposes a flow based encrypted traffic classification approach implemented through a web-based network-management application and integrated with MikroTik RouterOS. The study examines whether a Random Forest model trained on statistical flow features can provide a competitive accuracy efficiency trade off for fourteen VPN and non-VPN application classes, and whether its predictions can be translated reliably into administrator-triggered firewall actions with automatic rule expiration. The research contribution is therefore not the web application itself, but the experimentally evaluated workflow that connects payload-independent traffic classification, operational policy enforcement, rule-expiration handling, and auditable access-control records in a single framework.

2. METHOD

2.1. Dataset and Flow Extraction

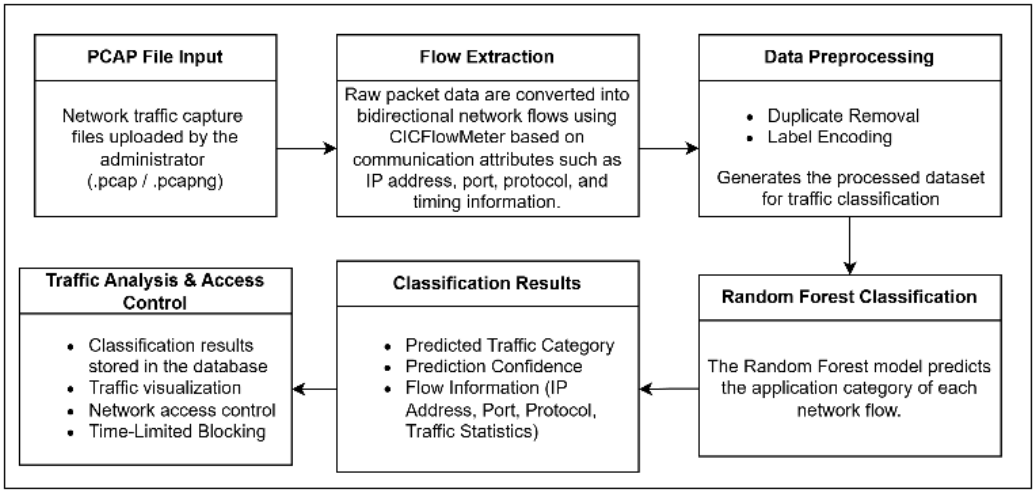


Figure 1. Flow-Based Classification and Access Control Workflow

This study used the publicly available ISCXVPN2016 VPN-nonVPN benchmark dataset released by the Canadian Institute for Cybersecurity, University of New Brunswick [12]. The working CSV contained 59,706 labeled flow records and 23 numerical predictors. After 18,719 duplicate records were removed, 40,987 unique samples remained; no missing values were detected. The 14 classes and their cleaned sample counts are BROWSING (6,459), CHAT (2,127), FT (2,146), MAIL (828), P2P (1,950), STREAMING (749), VOIP (6,344), VPN-BROWSING (7,317), VPN-CHAT (2,178), VPN-FT (3,336), VPN-MAIL (591), VPN-P2P (792), VPN-STREAMING (835), and VPN-VOIP (5,335). This class profile complements the dataset summary presented in Table 2.

The benchmark provides labeled packet captures and flow files generated with ISCXFlowMeter [12]. For model development, this study used the consolidated flow records rather than packet payloads. For operational inference, an uploaded PCAP or PCAPNG file is converted into bidirectional flows by the Python CICFlowMeter implementation, after which the extracted columns are mapped to the model's fixed feature order. Figure 1 shows the implemented sequence from packet-capture input through flow extraction and preprocessing to Random Forest classification and the resulting traffic-analysis and access-control output. Because only header-derived timing and volume statistics are used, encrypted payloads are neither decrypted nor inspected.

All 23 predictors available in the working dataset were retained; feature selection and normalization were not applied. The predictors comprise flow duration; total, minimum, maximum, and mean forward and

backward inter-arrival times; packets per second and bytes per second; minimum, maximum, mean, and standard deviation of overall flow inter arrival time; and minimum, mean, maximum, and standard deviation of active and idle periods. The categorical traffic label was encoded as an integer target only after duplicate removal. This fixed feature set was used consistently during training, evaluation, and operational inference.

2.2. Random Forest Classification

Random Forest was selected as the primary classifier because it can model nonlinear interactions among heterogeneous flow statistics, is insensitive to feature scaling, supports multiclass prediction directly, and exposes feature importance, while its independent trees can be trained and evaluated in parallel. In contrast, boosting methods such as XGBoost, LightGBM, and CatBoost build trees sequentially and introduce additional tuning complexity, whereas deep models generally impose greater computational and interpretability costs. Random Forest was therefore selected for the deployed classifier because its accuracy efficiency and interpretability properties fit the operational constraints; Elmaghraby et al. [8] also report competitive Random Forest results for VPN and non-VPN traffic under flow based settings. The implementation used scikit-learn's RandomForestClassifier [13].

Each tree is trained from a bootstrap sample, and a random subset of features is considered at each split. Split quality is measured using Gini impurity, defined as

$$Gini(D) = 1 - \sum_{i=1}^{C} p_i^2 \quad (1)$$

where D denotes the sample set at the current node, C = 14 is the number of traffic classes, and p_i is the proportion of samples in D that belong to class i.

During inference, the final class is obtained by aggregating the predictions of all decision trees,

$$\hat{y} = \text{mode}\{T_1, T_2, \ldots, T_n\} \quad (2)$$

Where ŷ is the final predicted class, T_i is the class predicted by the i-th decision tree, and n is the total number of trees in the forest.

Before training, 18,719 exact duplicate rows were removed; no imputation was required because the dataset contained no missing values. Class labels were encoded using LabelEncoder, and the cleaned data were partitioned by stratified sampling into 32,789 training samples (80%) and 8,198 independent test samples (20%) with random_state = 42. No normalization or standardization was applied because the classifier uses threshold-based tree splits. The held-out test set was not used during hyperparameter selection.

Hyperparameters were selected only on the training partition using GridSearchCV with stratified five-fold cross-validation and accuracy as the selection score [14]. The grid covered n_estimators in {100, 200, 300}, max_depth in {None, 20, 30}, min_samples_split in {2, 5}, and min_samples_leaf in {1, 2}, giving 36 configurations and 180 fits. The selected model used 300 trees, max_depth = None, min_samples_split = 2, min_samples_leaf = 1, Gini criterion, max_features = 'sqrt', bootstrap sampling, random_state = 42, and n_jobs = -1, as summarized in Table 3. This protocol separates model selection from the final held-out evaluation and makes the experiment reproducible.

2.3. Proposed System

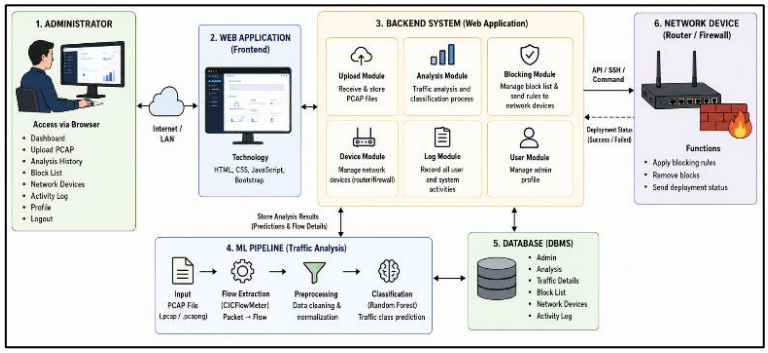


Figure 2. Proposed System Architecture

Figure 2 presents the implementation architecture used to validate the proposed classification to control workflow. The web application receives a PCAP or PCAPNG file, invokes flow extraction and feature mapping, applies the trained Random Forest model, stores predicted application classes and flow metadata, and exposes the results to the administrator. The application is therefore an experimental implementation layer for testing the method; the contribution evaluated here is the application-class-informed, administrator-triggered translation of a classified flow into a time-limited network access rule.

After inspecting a classified flow, the administrator selects the target IP address, blocking direction (source or destination), and duration. The backend sends the command to the configured MikroTik RouterOS device through its API and creates or updates an IP firewall address-list entry with a native RouterOS timeout [15]. The system stores the target, reason, activation time, expiry time, and device response so that the control action can be traced to the corresponding classification result.

Two complementary expiry mechanisms enforce time-limited blocking. RouterOS removes an address-list entry when its native timeout elapses, while a background scheduler queries the local database every 60 s, calls the unblock operation for expired entries, updates their status, and records an automatic-unblock event. These safeguards reduce the persistence of stale rules if either the device timeout or the application process is interrupted. Classifier quality is evaluated independently from this implementation behavior, as described in Section 2.4.

2.4. Performance Evaluation

Performance was evaluated on the independent 8,198-sample test set using four classification metrics: accuracy, support-weighted precision, support-weighted recall, and support-weighted F1-score. In the multiclass setting, precision, recall, and F1-score were computed one-versus-rest for each class and then averaged in proportion to class support [16]. Five-fold cross-validation was treated as a validation protocol, not as a fifth performance metric.

Accuracy measures the proportion of test samples assigned to the correct traffic class,

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad (3)$$

For each traffic class, precision measures the proportion of predicted positives that are correct,

$$Precision = \frac{TP}{TP+FP} \quad (4)$$

For each traffic class, recall measures the proportion of actual positives that are identified,

$$Recall = \frac{TP}{TP+FN} \quad (5)$$

and the F1-score is the harmonic mean of precision and recall,

$$F1\text{-}Score = \frac{2\times Precision\times Recall}{Precision+Recall} \quad (6)$$

where TP, TN, FP, and FN denote true-positive, true-negative, false-positive, and false-negative counts, respectively, for a one-versus-rest class evaluation.

Two complementary validation stages were used. First, stratified five-fold cross-validation within the training partition selected the Random Forest hyperparameters; the independent test partition was then used once for the primary metrics and confusion matrix. Second, a supplemental StratifiedKFold analysis with five folds, shuffling, and random_state = 42 was run on the 40,987 cleaned samples to estimate score stability across partitions. For a controlled comparison, Logistic Regression, five-neighbor K-Nearest Neighbors, and an RBF-kernel Support Vector Machine were trained and evaluated on the same 32,789/8,198 split. StandardScaler was fitted only on the training partition for these scale-sensitive baselines. The baseline choices follow classifiers evaluated in prior ISCXVPN2016 research [8].

Three evaluation hypotheses were assessed. H1 states that the tuned Random Forest achieves higher held-out accuracy and weighted F1-score than all reproduced baselines. H2 states that the mean five-fold accuracy differs from the held-out accuracy by less than one percentage point and that the fold-to-fold standard deviation remains below one percentage point. H3 states that every time-limited blocking entry in the evaluation database reaches an Unblocked state with a recorded unblock timestamp. H3 evaluates database-observable rule-lifecycle completion; it does not measure router latency or production-scale reliability.

3. RESULTS AND DISCUSSIONS

3.1. Dataset and Model Performance

The experiment began with 59,706 ISCXVPN2016 flow records [12]. Removing 18,719 exact duplicates produced 40,987 unique records, which were divided by stratified sampling into 32,789 training samples and 8,198 test samples. As summarized in Table 2, all 23 numerical predictors were retained and no missing values were present. The cleaned dataset was imbalanced: VPN-BROWSING contained 7,317 samples, whereas VPN-MAIL contained 591, a 12.38:1 ratio. This imbalance motivated reporting macro and support-weighted results in addition to overall accuracy.

Duplicate removal and label encoding were completed before the split, while feature scaling was not applied to Random Forest. Contrary to the previous wording, no feature-selection stage reduced the predictors; all 23 available flow features were retained. The same training and test indices were used for Random Forest and the three reproduced baselines, ensuring that differences in Table 4 are attributable to model behavior rather than different test samples.

Table 2. Summary of the Final Dataset

Item	Value
Dataset	VPN and Non-VPN Application Traffic (CIC-VPN2016)
Initial Samples	59,706
Duplicate Records Removed	18,719
Final Unique Samples	40,987
Training Samples	32,789 (80%)
Test Samples	8,198 (20%)
Retained Features	23
Traffic Classes	14
Missing Values	0

GridSearchCV evaluated 36 hyperparameter configurations through 180 training fits. The best mean cross-validation accuracy on the training partition was 87.43%. The selected configuration, reported in Table 3, used 300 trees, Gini impurity, max_depth = None, max_features = 'sqrt', min_samples_split = 2, min_samples_leaf = 1, bootstrap sampling, random_state = 42, and parallel processing across available logical cores [14].

Table 3. Random Forest Hyperparameters

Parameter	Value
Algorithm	Random Forest Classifier
n_estimators	300
criterion	Gini
max_depth	None
max_features	sqrt
min_samples_split	2
min_samples_leaf	1
bootstrap	True
random_state	42
n_jobs	-1

On the independent test set, Random Forest achieved 88.01% accuracy, 88.00% weighted precision, 88.01% weighted recall, and 87.97% weighted F1-score. The supplemental five-fold analysis produced a mean accuracy of 88.05% with a standard deviation of 0.0036, equivalent to 0.36 percentage points. Table 4 compares these results with three baselines reproduced using the same split and with the Technique 1 accuracy reported by Elmaghraby et al. [8] on ISCXVPN2016.

Table 4. Controlled and Literature-Based Comparison on ISCXVPN2016

Model/Study	Accuracy	Weighted Precision	Weighted Recall	Weighted F1-score
Logistic Regression (reproduced)	39.94%	40.85%	39.94%	33.82%
K-Nearest Neighbors (reproduced)	71.92%	71.93%	71.92%	71.68%
SVM-RBF (reproduced)	43.72%	49.22%	43.72%	38.24%
Random Forest (proposed)	88.01%	88.00%	88.01%	87.97%
Elmaghraby et al. [8], Technique 1	96.80%	NR	NR	NR

Under the controlled protocol, K-Nearest Neighbors was the strongest baseline at 71.92% accuracy and 71.68% weighted F1-score. Random Forest improved these values by 16.09 and 16.29 percentage points, respectively, supporting H1. The 96.80% accuracy reported by Elmaghraby et al. [8] is included for literature context, but it was obtained from 33,286 packets and four application classes using a neural-network ensemble. It is therefore not treated as a direct superiority test against the present 14-class flow-level experiment. NR in Table 4 denotes a metric not reported as a comparable weighted 14-class aggregate.

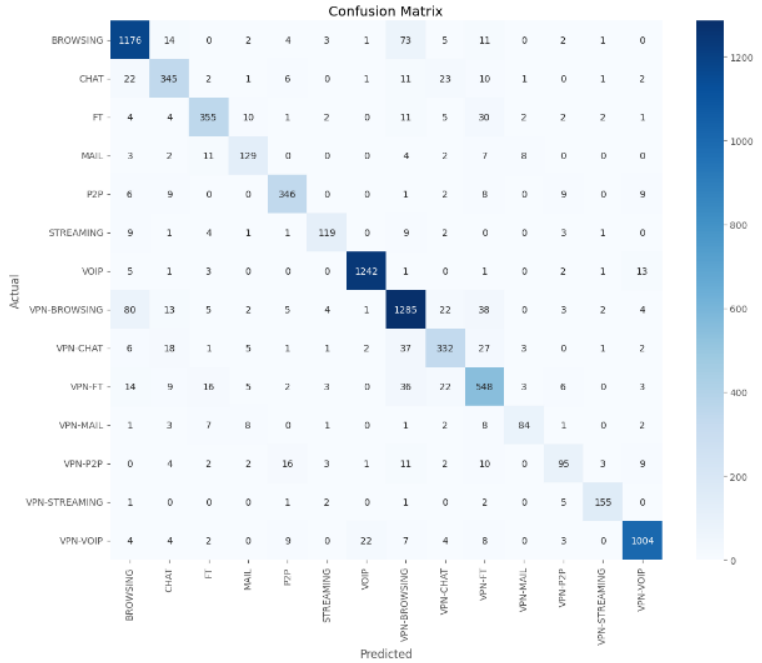
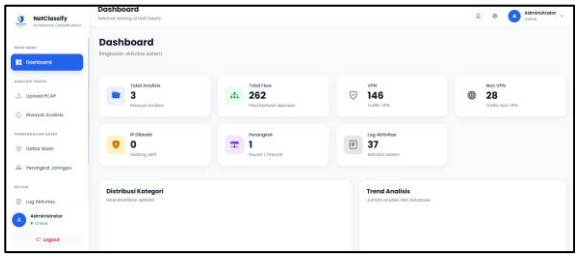


Figure 3. Confusion Matrix of the Random Forest classifier

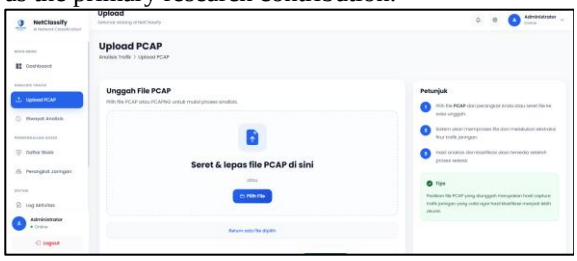
Figure 3 confirms a dominant diagonal but also identifies systematic class-level errors. The largest confusion was VPN-BROWSING predicted as BROWSING (80 flows), followed by BROWSING predicted as VPN-BROWSING (73 flows). Other notable errors were VPN-BROWSING predicted as VPN-FT (38) and VPN-CHAT predicted as VPN-BROWSING (37). VOIP achieved the highest class F1-score at 97.83%, whereas VPN-P2P achieved the lowest at 65.74%, with recall of 60.13%. Thus, the weighted F1-score of 87.97% should not be interpreted as uniform performance across all classes.

3.2. System Implementation

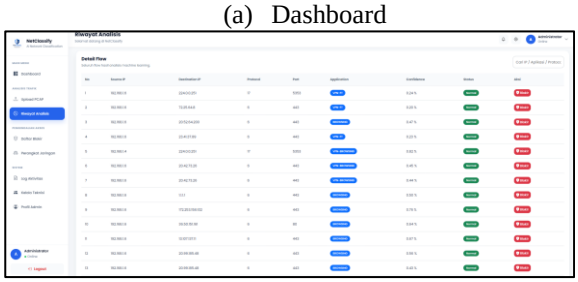
Figure 4 summarizes the implementation used as a validation environment for the classification-to-control workflow. Its role in the experiment was limited to receiving PCAP/PCAPNG files, invoking flow extraction and feature mapping, executing the trained classifier, storing results, and exposing classified flows for an administrator decision. The interface itself is not treated as the primary research contribution.



(a) Dashboard



(b) PCAP Upload



(c) Traffic Classification Result

(d) Analysis History

Figure 4. Implementation of the proposed web-based application

For each uploaded capture, the backend generated flow records and passed the same ordered 23-feature representation to the deployed Random Forest model. The implementation then stored the predicted class, confidence score, source and destination addresses, ports, protocol, duration, and byte count. This persistence layer allowed a prediction to be traced to the flow later selected for access control.

The analysis view presented both aggregate class distributions and individual flow predictions. These displays were used to inspect model output and select a target; they did not modify the classifier or contribute additional predictive features. Consequently, classification performance remained determined solely by the held-out and cross-validation experiments reported in Section 3.1.

Analysis history and activity logs retained the relationship between an uploaded capture, its classified flows, and subsequent control actions. This traceability is relevant to method validation because it separates the model prediction from the administrator's policy decision and from the RouterOS execution result.

Accordingly, the implementation demonstrates technical feasibility of the proposed workflow, but screenshots and interface functions are not used as evidence of classifier effectiveness. The scientific evidence is provided by the controlled model comparison, cross-validation stability, class-level analysis, and observable rule lifecycle.

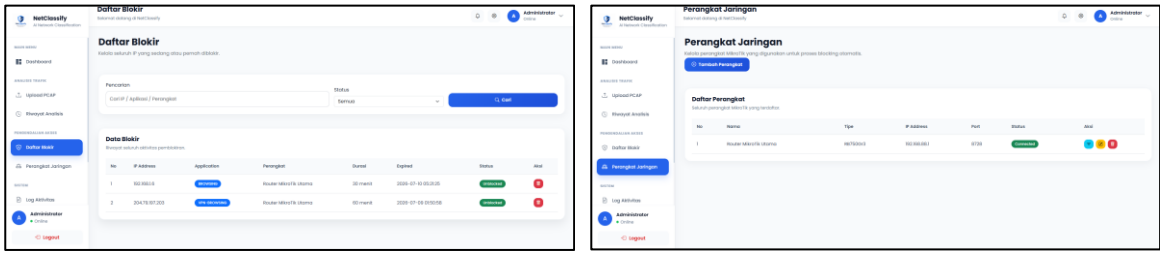
3.3. Network Access Control

The access-control stage is administrator-triggered rather than fully autonomous. After reviewing a classified flow, the administrator selects a source address, destination address, or application-category target and assigns a blocking duration. This design keeps the policy decision under human control while automating command generation, device submission, expiration handling, and audit recording.

Figure 5 shows the two access-control interfaces used in validation. The backend translates the selected target into a MikroTik RouterOS address-list entry and ensures that the corresponding drop rule is active. For temporary rules, the requested duration is also sent as a native RouterOS timeout [15]. The database stores the target type, activation time, duration, expiry time, status, and unblock time.

The evaluation database snapshot contained 30 time-limited blocking records: 28 category-level targets, one source target, and one destination target. All 30 records had reached the Unblocked state and contained a non-null unblock timestamp. This result supports H3 at the database-lifecycle level by showing that the recorded temporary rules did not remain indefinitely active after expiration.

The background scheduler checked expired entries every 60 s and recorded automatic-unblock events, while the activity log separately retained successful additions, failed blocking attempts, manual removals, and automatic removals. Because one category-level action can produce multiple database rows and repeated scheduler log events, the available records do not support a valid command-level success-rate or latency calculation. Those measures are therefore not claimed.



(a) Blocking Management Interface (b) Network Device Management Interface

Figure 5. Network Access Control using Time-Limited Blocking

These observations demonstrate feasibility and traceability of translating a selected classification result into a time-limited RouterOS rule. They do not establish production-scale availability or fully autonomous policy accuracy. The access-control implementation should therefore be interpreted as validation of the proposed method's operational pathway rather than as an independent software contribution.

3.4. Discussion

The controlled comparison in Table 4 provides direct empirical support for the Random Forest choice. Its 88.01% accuracy exceeded Logistic Regression, SVM-RBF, and K-Nearest Neighbors by 48.07, 44.29, and 16.09 percentage points, respectively. The weighted F1-score showed the same ordering. H1 is therefore supported under the common 80:20 split, although this conclusion applies to the stated baseline configurations rather than to every possible tuned implementation of those algorithms.

The held-out accuracy of 88.01% differed from the supplemental five-fold mean of 88.05% by only 0.04 percentage points, and the cross-validation standard deviation was 0.36 percentage points. Both values satisfy the thresholds defined for H2. Nevertheless, the macro F1-score was 84.04%, 3.93 percentage points below the

weighted F1-score, showing that class imbalance affected minority-class performance. In particular, VPN-P2P recall was 60.13%, whereas VOIP recall reached 97.87%.

The confusion matrix explains where the aggregate performance is lost. Bidirectional confusion between BROWSING and VPN-BROWSING accounted for 153 errors, and additional errors were concentrated among VPN-BROWSING, VPN-CHAT, and VPN-FT. These classes can exhibit overlapping timing and volume patterns, so the result identifies a limitation of the current 23-feature representation rather than evidence that encrypted payload inspection is required. Feature enrichment and validation across independent capture environments are needed to determine whether these confusions generalize [4], [5].

The literature comparison must be interpreted cautiously. Elmaghraby et al. [8] reported 96.80% for a neural-network ensemble on the same dataset family, but their experiment used four classes and packet-level inputs; the present experiment uses 14 classes and flow-level statistics. Accordingly, the contribution claimed here is not state-of-the-art accuracy. It is the empirically evaluated connection between payload-independent multiclass classification, administrator-reviewed policy selection, RouterOS enforcement, expiration handling, and auditable state transitions, addressing the operational gap identified in earlier classification and network-control work [7], [10], [11].

Several limitations remain. Only one public benchmark and one stratified random split were used; records originating from related capture sessions may therefore appear in both partitions. The reproduced baselines were not subjected to the same extensive hyperparameter search as Random Forest, and XGBoost, LightGBM, CatBoost, and deep models were not reimplemented. Operational validation was limited to one RouterOS integration and database-observable rule states, without controlled latency, throughput, failure-recovery, or multi-device testing. A secondary benchmark, capture-session-aware splitting, equally tuned model comparisons, and repeated access-control trials are required before broader generalization or production reliability can be claimed [5].

4. CONCLUSION

This study evaluated whether payload-independent flow statistics can support encrypted-traffic classification and an operational access-control workflow. Using 40,987 unique ISCXVPN2016 records, 23 retained flow features, and an independent 8,198-sample test set, the tuned Random Forest achieved 88.01% accuracy and a weighted F1-score of 87.97%. Under the same split, it exceeded the strongest reproduced baseline, K-Nearest Neighbors, by 16.09 percentage points in accuracy and 16.29 percentage points in weighted F1-score, supporting H1. The five-fold mean accuracy of 88.05%, its 0.36-percentage-point standard deviation, and the 0.04-percentage-point difference from the held-out result support H2. However, the macro F1-score of 84.04% and the VPN-P2P F1-score of 65.74% show that the aggregate result does not represent uniform performance across all classes.

The implementation further showed that administrator-reviewed classification results can be translated into time-limited MikroTik RouterOS rules with auditable lifecycle records. All 30 temporary blocking entries in the evaluation database reached the Unblocked state with recorded unblock timestamps, supporting H3 at the database-lifecycle level; this finding does not establish device-level latency, fault tolerance, or production reliability. The contribution is therefore the evaluated connection between payload-independent multiclass classification, human-controlled policy selection, RouterOS enforcement, automatic expiration, and auditability, rather than the web interface itself. Future work should validate the method on a secondary benchmark and independent capture sessions, compare equally tuned Random Forest, XGBoost, LightGBM, CatBoost, and deep models, and measure access-control latency, throughput, failure recovery, and multi-device behavior.

ACKNOWLEDGEMENTS

The authors would like to express their sincere gratitude to the Department of Electronic Engineering, Universitas Negeri Padang, for providing academic guidance and research facilities that supported the completion of this study. The authors also appreciate the valuable suggestions and constructive feedback provided by the supervisors and examiners throughout the research process.

CREDIT AUTHORSHIP CONTRIBUTION STATEMENT

Wahyu Isnan: Conceptualization, Methodology, Software, Writing. Yulia Fatmi: Validation. Resmi Darni: Validation. Vikri Aulia: Validation.

DECLARATION OF COMPETING INTERESTS

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

DATA AVAILABILITY

The dataset used in this study is the VPN and Non-VPN Application Traffic (CIC-VPN2016) dataset developed by the Canadian Institute for Cybersecurity (CIC), University of New Brunswick (UNB). The dataset is publicly available through the official CIC website and its Kaggle mirror:
Canadian Institute for Cybersecurity (CIC), University of New Brunswick (UNB):
https://www.unb.ca/cic/datasets/vpn.html
Kaggle:
https://www.kaggle.com/datasets/noobbcoder2/vpn-and-non-vpn-application-traffic-cic-vpn2016

REFERENCES

[1] M. Thomson and S. Turner, "RFC 9001: Using TLS to Secure QUIC," 2021. [Online]. Available: https://www.rfc-editor.org/info/rfc9001

[2] M. Bishop, "RFC 9114: HTTP/3," 2022. [Online]. Available: https://www.rfc-editor.org/info/rfc9114

[3] A. Sharma and A. H. Lashkari, "A survey on encrypted network traffic: A comprehensive survey of identification/classification techniques, challenges, and future directions," Feb. 01, 2025, *Elsevier B.V.* doi: 10.1016/j.comnet.2024.110984.

[4] A. Azab, M. Khasawneh, S. Alrabaee, K. K. R. Choo, and M. Sarsour, "Network traffic classification: Techniques, datasets, and challenges," *Digital Communications and Networks*, vol. 10, no. 3, pp. 676–692, Jun. 2024, doi: 10.1016/j.dcan.2022.09.009.

[5] N. Wickramasinghe, A. Shaghaghi, G. Tsudik, and S. Jha, " SoK: Decoding the Enigma of Encrypted Network Traffic Classifiers ," in *2025 IEEE Symposium on Security and Privacy (SP)* , Los Alamitos, CA, USA: IEEE Computer Society, May 2025, pp. 1825–1843. doi: 10.1109/SP61157.2025.00165.

[6] Z. Wang, Y. Yang, and Y. Wang, "A Survey of Encrypted Traffic Classification: Datasets, Representation, Approaches and Future Thinking," in *2024 IEEE/ACIS 24th International Conference on Computer and Information Science (ICIS)*, 2024, pp. 113–120. doi: 10.1109/ICIS61260.2024.10778376.

[7] R. Bozkır, M. Cicioğlu, A. Çalhan, and C. Toğay, "A new platform for machine-learning-based network traffic classification," *Comput. Commun.*, vol. 208, pp. 1–14, 2023, doi: https://doi.org/10.1016/j.comcom.2023.05.010.

[8] R. T. Elmaghraby, N. M. Abdel Aziem, M. A. Sobh, and A. M. Bahaa-Eldin, "Encrypted network traffic classification based on machine learning," *Ain Shams Engineering Journal*, vol. 15, no. 2, p. 102361, Feb. 2024, doi: 10.1016/J.ASEJ.2023.102361.

[9] P. Luo, J. Chu, and G. Yang, "IP packet-level encrypted traffic classification using machine learning with a light weight feature engineering method," *Journal of Information Security and Applications*, vol. 75, p. 103519, 2023, doi: https://doi.org/10.1016/j.jisa.2023.103519.

[10] A. O. Salau and M. M. Beyene, "Software defined networking based network traffic classification using machine learning techniques," *Sci. Rep.*, vol. 14, no. 1, p. 20060, 2024, doi: 10.1038/s41598-024-70983-6.

[11] J.-K. Lee, T. Hong, and G. Lee, "AI-Based Approach to Firewall Rule Refinement on High-Performance Computing Service Network," *Applied Sciences*, vol. 14, no. 11, 2024, doi: 10.3390/app14114373.

[12] UNB Canadian Institute for Cybersecurity, "VPN-nonVPN dataset (ISCXVPN2016)," University of New Brunswick. Accessed: Apr. 12, 2026. [Online]. Available: https://www.unb.ca/cic/datasets/vpn.html

[13] Scikit-learn Developers, "RandomForestClassifier," Scikit-learn Developers. Accessed: Apr. 20, 2026. [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html

[14] "GridSearchCV — scikit-learn 1.9.0 documentation." Accessed: Jul. 22, 2026. [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html

[15] MikroTik, "API | RouterOS Manual." Accessed: Jul. 10, 2026. [Online]. Available: https://manual.mikrotik.com/docs/developer-guides/api/

[16] "classification_report — scikit-learn 1.9.0 documentation." Accessed: Jul. 22, 2026. [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.metrics.classification_report.html