Flow-Based Analysis of Campus WiFi Traffic Anomalies Using Machine Learning

Dody Roky Immanuel Nainggolan[1], T. Yudi Hadiwandra[2]
[1,2]Universitas Riau, Indonesia
Email: dodyroky1212@gmail.com

Abstract

Campus WiFi networks are exposed to attacks such as Distributed Denial of Service (DDoS), brute force, and port scanning, but flow-based machine learning models trained on public datasets may not recognize these attacks on a different network. This study evaluates, on campus WiFi traffic, two Random Forest models trained only on the corrected CNS2022 version of CICIDS2017. Model 1 used 55 flow features selected by variance and correlation filtering, while Model 2 used 34 flag, size, and contextual features computed over six-second windows, with the size features of DDoS training flows randomized. Both models were tested on 51 controlled attack recordings of DDoS, File Transfer Protocol (FTP) brute force, Secure Shell (SSH) brute force, and port scanning, and on 49,625 unlabeled operational flows. Both models achieved a macro F1 above 0.99 on the CICIDS2017 test set, but the mean recall across the four attack classes on the campus simulations fell to 0.383 for Model 1 and 0.691 for Model 2. FTP brute force remained recognizable because the server's authentication reply had the same size in both datasets, whereas Model 1 failed on DDoS by relying on packet sizes specific to the training data. Model 2 recognized DDoS and port scanning mainly through contextual features. SSH brute force was hardly recognized because 99.3% of its flows came from connections rejected by the server before authentication. Fewer than 0.2% of operational flows were predicted as attacks. Flow-based detection on a different network must therefore be evaluated per attack class.

Keywords: *CICIDS2017, Flow-Based Intrusion Detection, Generalization, Random Forest, WiFi Network.*

A. INTRODUCTION

Campus WiFi networks support academic services, access to learning resources, communication, and data exchange, and the resulting traffic changes over the day while still showing regular patterns (Meng et al., 2022). These networks are also exposed to attacks. Indonesia's National Cyber and Crypto Agency (BSSN) recorded more than 330 million traffic anomalies in 2024, and the advanced persistent threat group most frequently found in Indonesian cyberspace that year was known to target higher education organizations (Badan Siber dan Sandi Negara, 2025). Distributed Denial of Service (DDoS) attacks overwhelm a service with traffic from many sources (Sharafaldin et al., 2018), brute-force attacks repeatedly try credentials on services such as the File Transfer Protocol (FTP) and Secure Shell (SSH) (Ardiansyah et al., 2025), and port scanning probes many ports to find open services (Intan Sabila et al., 2025). Because traffic volume also rises during normal use, for example when many users download course materials at the same time, these attacks need to be distinguished from normal traffic by their communication patterns rather than by volume alone.

Signature-based detection relies on known attack patterns and is therefore limited when traffic does not match its rules (Rodríguez et al., 2022). Supervised machine learning instead learns patterns from labeled traffic. A common approach is flow-based analysis, which summarizes each communication between endpoints into attributes such as duration, packet count, and byte count without inspecting packet contents (Rodríguez et al., 2022). This study uses the corrected CNS2022 version of CICIDS2017 (Liu et al., 2022), which fixes errors in the feature extraction and labeling of the original dataset and includes benign traffic and the four attack classes considered here: DDoS, FTP brute force, SSH brute force, and port scanning.

However, a supervised model can recognize only patterns that appear in its training data or resemble them (Koumar et al., 2025). A classifier may therefore achieve high scores on a test split from the same dataset and still miss attacks on another network, because differences in devices, services, and attack tools can change the distribution of flow features (Cantone et al., 2024). High scores may also come from patterns that happen to co-occur with a label in the training data, such as characteristics of the recording environment, rather than from the attack itself (Arp et al., 2022). Evaluating models on traffic from a different network, together with the traffic attributes they rely on, is therefore needed to examine which attack classes remain recognizable and why errors occur.

This study compares two flow-based Random Forest models trained only on the corrected CICIDS2017 dataset. Random Forest was chosen because tree-based models remain strong on medium-sized tabular data (Grinsztajn et al., 2022) and the contribution of their features can be measured by permuting feature values (Abdelaziz et al., 2025). The first model (Model 1) uses flow features selected by variance and correlation filtering. The second model (Model 2) is designed from the literature on dataset-specific patterns (Arp et al., 2022) and time-window activity (Meng et al., 2022): it uses only flag and size flow features, adds contextual features that summarize the activity of each source address within a time window, and randomizes the size features of DDoS flows in the training data. Both models are evaluated on held-out CICIDS2017 data, on controlled attack simulations recorded on a campus WiFi network, and on unlabeled operational campus traffic. Accordingly, this study addresses two research questions: how the detection performance of flow-based models trained on CICIDS2017 changes for each attack class on a campus WiFi network, and which traffic characteristics and feature groups explain the recognition or failure of each class. The contributions of this study are an evaluation of flow-based models on campus WiFi traffic broken down by attack class, tool, and intensity, and an analysis that links each result to traffic characteristics, such as server reply sizes, connection rejection by the SSH server, and connection patterns within a time window.

B. LITERATURE REVIEW

Several studies have applied machine learning to flow-based intrusion detection on CICIDS2017. Rodríguez et al. (2022) compared ten algorithms and

found that tree-based methods, including Random Forest, performed best, with Random Forest achieving F1 scores of 0.999 for benign traffic and 0.997 for attack traffic in a binary experiment. Alduailij et al. (2022) selected features using mutual information and Random Forest feature importance for DDoS detection on CICIDS2017 and CICDDoS2019, and their Random Forest with 19 features misclassified only one attack flow on CICIDS2017. Abdelaziz et al. (2025) removed constant and highly correlated features, used permutation feature importance to select 26 features, and obtained a macro F1 of 0.9331 with class weights and a custom prediction function. These studies show that Random Forest performs well on CICIDS2017, but each evaluated its models on data drawn from the same dataset used for training, so their performance on other networks remains unknown.

The reliability of such results also depends on the dataset. Engelen et al. (2021) identified problems in the flow construction and labeling of CICIDS2017, and Liu et al. (2022) found errors in attack execution, feature construction, documentation, and labeling in CICIDS2017 and CSE-CIC-IDS2018, then released corrected versions with documented labeling rules. Flood et al. (2024) identified design issues in benchmark network intrusion detection system (NIDS) datasets, such as the low diversity of the denial of service (DoS) Hulk data in CICIDS2017 caused by fixed network conditions and target pages. Arp et al. (2022) warned that models can learn spurious correlations, that is, patterns that co-occur with a label in the training data but are unrelated to the attack itself, which leads to over-optimistic evaluation. These studies focus on the quality of the datasets themselves rather than on how a model trained on them behaves on traffic from another network.

Models trained on one dataset also do not transfer well to other data. Cantone et al. (2024) compared four classifiers across CICIDS2017, CSE-CIC-IDS2018, LycoS-IDS2017, and LycoS-Unicas-IDS2018 and found that performance was high within a single dataset but declined in many cross-dataset tests, partly because attack tools and settings changed the distribution of flow features. Their study examined generalization between benchmark datasets, rather than per attack class and tool on a real network where the attacks are known.

Other studies have described traffic beyond a single flow. Lee and Stolfo (2000) built time-based traffic features that summarize connections to the same host and the same service within the last two seconds, because many attacks consist of many connections. Meng et al. (2022) built seven statistics of Internet Protocol (IP) address activity within a time window, including the number of flows, distinct destination addresses, and distinct destination ports, combined them with an IP proximity matrix, and analyzed them using CUR matrix decomposition on the MAWI dataset and the CERNET campus network. In the campus deployment, these features were used to trace devices showing signs of scanning and DDoS activity. Their approach detects anomalies without labeled attacks, so it does not show how such features contribute to recognizing specific attack classes when used in a supervised classifier.

In summary, previous studies reported high performance within CICIDS2017, identified quality issues in benchmark datasets, and showed that models generalize poorly across datasets, mostly through evaluations between benchmark datasets. How a flow-based classifier trained on a public dataset recognizes each attack class, tool, and intensity on a real campus WiFi network, and which traffic characteristics explain its successes and failures, has not been examined in the studies above.

C. METHOD

1. Data Sources and Campus Data Collection

This study used three sources of network traffic. The corrected CNS2022 version of CICIDS2017 (Liu et al., 2022) provided the labeled flows used to train both Random Forest models and evaluate them on a held-out test set. The analysis covered five classes: benign traffic, DDoS, FTP brute force, SSH brute force, and port scanning. The FTP-Patator and SSH-Patator labels were kept as separate classes because they target different services. DoS labels were excluded because those attacks were launched from a single machine and therefore do not belong to the DDoS class (Sharafaldin et al., 2018), so the DDoS class in the training data contains only LOIC attacks. Flows labeled as Attempted were also excluded because they were generated by attack tools but contain no attack activity (Engelen et al., 2021). The excluded flows were not relabeled as benign, since they still originate from attack activity.

Figure 1 shows the research workflow, from data collection to model evaluation and feature analysis.

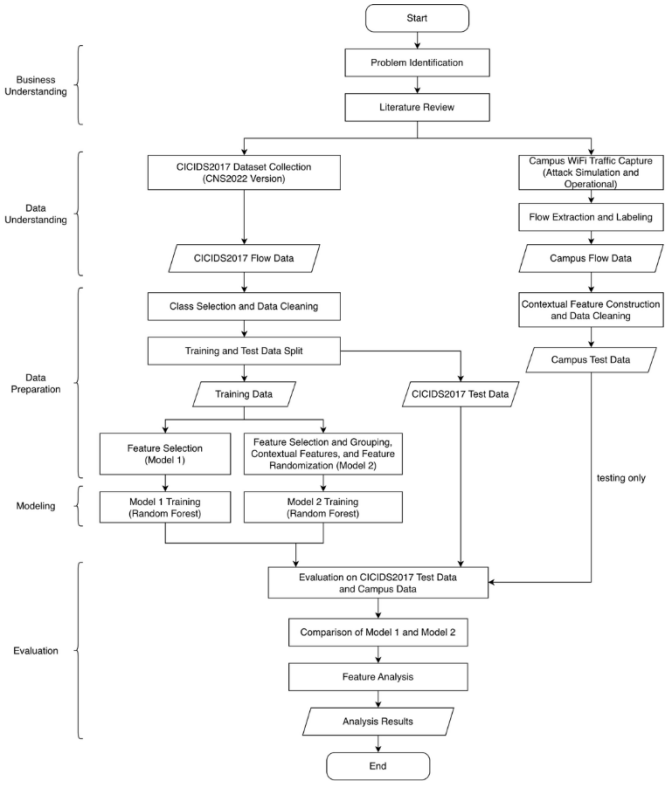


Figure 1. Research Workflow

The campus data were recorded on a university campus WiFi network and consisted of controlled attack simulations and operational traffic, neither of which was used for training. In the simulations, the attacker and target devices were both connected to the campus WiFi network, so the simulated traffic passed through the network under study. The target was a Kali Linux virtual machine running vsftpd on port 21, OpenSSH on port 22, and a Flask web application on port 8080, with one test account prepared for the brute-force attacks. Brute-force and port scanning attacks were launched from an Arch Linux device, while DDoS attacks were launched simultaneously from three Windows virtual machines with different IP addresses.

Each attack class was run with several tools at three intensity levels: low, medium, and high. Intensity was set through tool parameters, such as the number of threads, the number of connections, or the packet rate, using the flow rate of a single attacker address in CICIDS2017 as a reference. The medium level approximated that rate, the low level was lower, and the high level was higher. DDoS and port scanning attacks ran for 120 seconds, while each brute-force run tried the same 300 passwords from the rockyou wordlist, with the correct password removed so that every recording contained the same number of attempts. Traffic was captured on the target with tcpdump, filtered by the attacker address and target port, so that each capture contained one scenario. Some scenarios were recorded twice, and one scenario, SSH brute force with Hydra at the high level, produced no usable recording, leaving 51 recordings as listed in Table 1.

Table 1. Attack Simulation Scenarios

Attack Class	Tools	Recordings
DDoS	LOIC, HOIC, GoldenEye, Slowloris, aiohttp	16
FTP brute force	Hydra, Medusa, Ncrack, Patator	12
SSH brute force	Hydra, Medusa, Ncrack, Patator	13
Port scanning	Nmap, Masscan, RustScan	10
Total		**51**

Operational traffic was recorded on four weekdays in the morning, midday, and afternoon sessions. A MikroTik router was placed between the main campus network and one WiFi access point, and the traffic passing through it was captured with Wireshark, yielding 26 files of about 30 minutes each. Because user traffic was not inspected individually, the operational traffic had no verified labels, and its results were reported as the number and proportion of flows assigned to each class. Only flow-level features were extracted from the operational traffic, packet payloads were not analyzed, and the results are reported only in aggregate without identifying individual users.

2. Flow Extraction, Labeling, and Cleaning

Campus packet captures were processed with the corrected version of CICFlowMeter used to build the CNS2022 version of CICIDS2017 (Engelen, n.d.),

built from commit 13a0615 and run in a Docker container so that every capture was processed with the same extractor version. Captures with out-of-order packet timestamps were sorted before extraction, and packet counts were checked against the original files. Using the same extractor kept feature definitions consistent between the benchmark and campus data. The extraction yielded 727,193 flows from the attack simulations and 64,469 flows from operational traffic.

Each simulation capture contained one attack scenario. A flow was assigned the scenario's attack label if it connected the recorded attacker and target IP addresses in either direction and matched the target service port. FTP and SSH brute-force attempts were matched to ports 21 and 22, respectively. For port scanning, all target ports were considered. Flows that did not meet the labeling rules were excluded from the simulation evaluation.

Before training, the CICIDS2017 flows were cleaned by removing flows with infinite or missing values, flows with Protocol 0, which use neither the Transmission Control Protocol (TCP), the User Datagram Protocol (UDP), nor the Internet Control Message Protocol (ICMP), and duplicate flows with the same feature values and class, keeping only the first occurrence. Duplicates were removed so that identical flows would not appear in both the training and test sets, which can make the evaluation overly optimistic (Arp et al., 2022). Cleaning reduced the data from 1,843,709 to 1,526,145 flows, mainly because 98.94% of the port scanning flows were duplicates. The campus data were cleaned in a similar way, and flows containing only one packet were also removed because the CNS2022 version of CICIDS2017 contains no such flows. Unlabeled simulation flows were removed, while duplicates were kept because the campus data were used only for evaluation. After cleaning, 721,435 simulation flows and 49,625 operational flows remained.

3. Data Splitting and Model Development

The cleaned CICIDS2017 flows were divided into five folds using StratifiedGroupKFold (Pedregosa et al., 2011). Four folds were used for training and one was held out for testing, giving an approximately 80:20 split with 1,220,869 training flows and 305,276 test flows. Flows from the same six-second window on the same recording day were kept in the same fold, so that the contextual features, which are computed per window (Meng et al., 2022), contained no information from the test set. Of the 23,604 windows, 18,884 were assigned to the training set and 4,720 to the test set, with no window shared between them. The split was stratified so that about 20% of each class, including the small FTP and SSH brute-force classes, was placed in the test set. Both models used the same split, and feature selection was performed using only the training data.

Both Random Forest models were implemented in Python 3.14 with scikit-learn 1.9.0 (Pedregosa et al., 2011) using 100 trees, balanced class weights to account for differences in class frequency, and a random state of 42. Model 1 used 55 flow features selected from 82 candidates following Abdelaziz et al. (2025). One feature with zero standard deviation was removed, and one feature from each pair with an

absolute Pearson correlation of at least 0.95 was removed, which excluded 26 more features.

The selected flow features were divided into six groups according to the quantity they measure: time, rate, TCP, count, flag, and size. Time, rate, and count features are affected by network conditions, including link-layer retransmission on WiFi (Kurose & Ross, 2021), and TCP features such as the initial window size depend on the TCP/IP stack of the sending device (Graham, n.d.). Flag features follow protocol rules that are part of the attack mechanism, such as SYN and RST segments in port scanning (Kurose & Ross, 2021), while size features reflect application messages, such as the USER and PASS commands in FTP (Postel & Reynolds, 1985). Because a model should rely on attack behavior rather than on characteristics of the recording environment (Arp et al., 2022), Model 2 used only the 12 flag features and 12 size features that passed feature selection.

Model 2 also used ten contextual features computed for each source IP address within a six-second window, as listed in Table 2. The first seven, OD, NDDA, NDR, NDDP, NP, SNF, and LT, follow the IP activity features of Meng et al. (2022), while NSS, NSD, and RSS were adapted from the same host and same service features of Lee and Stolfo (2000) to distinguish connections to the same service from all connections of a source. The features were computed from all flows in each comma-separated values (CSV) file before cleaning, so that each window reflected all traffic in that interval. In total, Model 2 used 34 features.

Table 2. Contextual Features

Feature	Definition
OD	Number of flows from the source IP address
NDDA	Number of distinct destination IP addresses contacted by the source
NDR	Number of flows in which the source IP address is the destination
NDDP	Number of distinct destination ports contacted by the source
NP	Mean number of packets per flow of the source
SNF	Mean number of bytes per flow of the source
LT	Mean flow duration of the source
NSS	Number of flows from the source to the same destination IP address and port
NSD	Number of flows from all sources to the same destination IP address and port
RSS	Ratio of NSS to OD

Source: Adapted from Meng et al. (2022) and Lee and Stolfo (2000)

In the training data, all DDoS flows were recorded from one firewall address to one target address and port, and Fwd Packet Length Max had a single value of 20 bytes. Flood et al. (2024) reported a similar lack of diversity in the DoS Hulk data of CICIDS2017. The principle of domain randomization (Tobin et al., 2017) was therefore adapted at the feature level. In each DDoS training flow, the values of all size features and SNF were replaced with those of a randomly selected benign

training flow, and the DDoS label was retained. Other classes, the CICIDS2017 test set, and the campus data were not changed.

Model 2 was developed using feature processing approaches from prior studies, taking into account the differences between CICIDS2017 and campus WiFi traffic found in the analysis of Model 1. Because the campus simulation traffic had been examined during this development, the results of Model 2 on the simulations cannot be regarded as a test on entirely unseen data (Arp et al., 2022). The comparison between the two models was therefore used to analyze the role of feature groups in recognizing each attack class, rather than to determine the best model.

4. Evaluation and Feature Analysis

Macro F1 was the primary metric on the held-out CICIDS2017 test set because it gives equal weight to each class (Opitz, 2024). Both models were also evaluated using five-fold cross-validation, in which feature selection and feature randomization were repeated using only the training portion of each fold. Because Random Forest and the feature randomization both involve random sampling (Breiman, 2001), both models were trained with five random seeds, and the averages were used as the main results. For the campus attack simulations, recall was reported for each attack class and averaged across the four classes. Recall was also broken down by attack tool and intensity level.

For each class c, precision, recall, and F1 were computed from the true positives, false positives, and false negatives of that class, as in equations (1)–(3), and macro F1 was computed as their unweighted mean over all classes, as in equation (4). The mean recall across the four attack classes on the campus simulations was computed in the same way as equation (4), using recall instead of F1.

$$P_c = \frac{TP_c}{TP_c + FP_c} \tag{1}$$

$$R_c = \frac{TP_c}{TP_c + FN_c} \tag{2}$$

$$F1_c = \frac{2P_cR_c}{P_c + R_c} \tag{3}$$

$$\text{Macro F1} = \frac{1}{C}\sum_{c=1}^{C} F1_c \tag{4}$$

where P_c, R_c, and $F1_c$ are the precision, recall, and F1 of class c; TP_c, FP_c, and FN_c are the numbers of true positives, false positives, and false negatives of class c; and C is the number of classes.

For SSH brute force, recall was also broken down by connection stage. An OpenSSH server can reject new connections from a source address under penalty through the PerSourcePenalties mechanism (OpenBSD, n.d.), so not every

connection reaches the authentication stage. Each TCP connection to port 22 was therefore examined in the packet capture. A connection was classified as rejected if the server sent the message "Not allowed at this time" and then closed the connection, as accepted if the server sent a packet larger than 200 bytes, indicating the SSH key exchange, and as other if neither condition was met. Each flow was then matched to its connection by client IP address, client port, and flow start time.

Feature analysis began by comparing feature distributions between the training data and the campus data. For the ten features with the highest mean decrease in impurity (MDI) in Model 1 (scikit-learn developers, n.d.), the median and the range from the first to the third quartile were computed for each class in the training data, and the proportion of campus flows whose values fell within that range was calculated. This comparison shows whether the feature values relied on by the model fall outside the distribution learned from the training data, since a model can rely on characteristics that appear only in the recording environment of a dataset (Arp et al., 2022). For the DDoS class, the contextual feature values of each attack tool were also compared with those in the training data to explain differences between tools.

To examine which features Model 2 relied on, grouped permutation feature importance (Au et al., 2022) was measured on a sample of the campus simulation flows, with at most 5,000 flows for each combination of attack class and tool so that each tool had equal weight. The features in each group were shuffled together, and the importance of group g for class c was computed as the decrease in recall in equation (5). The analysis covered four groups: contextual IP activity (7 features), contextual service-level activity (3 features), flag (12 features), and size (12 features). Each group was shuffled K = 3 times (Abdelaziz et al., 2025).

$$\Delta R_{c,g} = R_c - \frac{1}{K}\sum_{r=1}^{K} R_c^{(r,g)} \tag{5}$$

where $\Delta R_{c,g}$ is the importance of group g for class c, R_c is the recall of class c on the original data, $R_c^{(r,g)}$ is the recall after the r-th shuffle of group g, and K is the number of shuffles.

D. RESULT AND DISCUSSION

1. Results on the CICIDS2017 Test Set

On the held-out CICIDS2017 test set, the macro F1 over five random seeds was 0.9939 ± 0.0003 for Model 1 and 0.9994 ± 0.0004 for Model 2 (mean ± standard deviation), and five-fold cross-validation gave 0.9967 ± 0.0020 and 0.9981 ± 0.0015. In the run with seed 42, the mean recall across the four attack classes was 0.9940 for Model 1 and 0.9981 for Model 2. In the same run, the lowest F1 of Model 1 was 0.9721 for port scanning, mainly because 13 benign flows were predicted as port scanning and 6 port scanning flows as benign, while Model 2 misclassified only 4 of the 305,276 test flows. Both models therefore recognized all five classes very well within the dataset used for training, and this mean recall provides a basis for

comparison with the campus attack simulations.

2. Results on the Campus Attack Simulations

Table 3 shows the recall of both models on the campus attack simulations as the mean ± standard deviation over five random seeds.

Table 3. Recall by Attack Class on the Campus WiFi Simulations

Attack Class	Flows	Model 1	Model 2
DDoS	462,972	0.0000 ± 0.0000	0.8298 ± 0.0182
FTP brute force	1,490	0.9514 ± 0.0158	0.9656 ± 0.0196
SSH brute force	51,980	0.0006 ± 0.0003	0.0017 ± 0.0001
Port scanning	204,993	0.5781 ± 0.0002	0.9682 ± 0.0348
Mean recall		0.3825 ± 0.0040	0.6913 ± 0.0099

For Model 1, mean recall across the four attack classes fell from 0.9940 on the CICIDS2017 test set to 0.3825 on the campus simulations. Model 1 did not recognize any DDoS flow, almost failed to recognize SSH brute force, and recognized only 57.81% of port scanning flows, while FTP brute force remained well recognized with a recall of 0.9514. For Model 2, mean recall fell from 0.9981 to 0.6913. Its recall increased to 0.8298 for DDoS and 0.9682 for port scanning, while SSH brute force remained almost unrecognized.

The confusion matrices from the run with seed 42 show where the errors occurred. Model 1 predicted 462,811 of the 462,972 DDoS flows as benign, most of its port scanning errors were also benign predictions, totaling 86,455 flows, and 29,819 SSH brute-force flows were predicted as port scanning. Model 2 reduced the benign predictions for DDoS and port scanning to 79,169 and 1,328 flows, respectively, but predicted 47,488 SSH brute-force flows as benign.

Intensity did not change the overall pattern of results, but it affected recall for some classes. Averaged over five seeds, port scanning recall for Model 1 ranged from 0.500 at the medium level to 0.715 at the high level, while DDoS recall for Model 2 increased from about 0.80 at the low and medium levels to 0.896 at the high level. The largest difference occurred for SSH brute force, whose recall at the low level reached 0.151 for Model 1 and 0.353 for Model 2 but was close to zero at the medium and high levels. At the low level, almost all SSH connections were accepted by the server, whereas at the medium and high levels most connections were rejected by the OpenSSH server.

3. Analysis by Attack Class

Results also differed between tools within the same attack class. Table 4 shows the recall of both models by attack tool, averaged over five seeds.

Table 4. Recall by Attack Tool on the Campus WiFi Simulations

Attack Class	Tool	Flows	Model 1	Model 2
DDoS	aiohttp	114,811	0.000	0.868
	GoldenEye	116,015	0.000	0.719

	HOIC	81,670	0.000	0.691
	LOIC	144,847	0.000	0.994
	Slowloris	5,629	0.000	0.120
FTP brute force	Hydra	197	0.960	0.975
	Medusa	708	0.954	0.976
	Ncrack	362	0.944	0.941
	Patator	223	0.947	0.964
SSH brute force	Hydra	55	0.000	0.076
	Medusa	174	0.000	0.000
	Ncrack	49,757	0.000	0.000
	Patator	1,994	0.015	0.042
Port scanning	Masscan	100,246	0.141	0.942
	Nmap	40,020	0.998	0.988
	RustScan	64,727	0.996	0.997

For DDoS, Model 1 recognized no flows from any tool, including LOIC, the tool used to generate the DDoS class in the training data. The packet sizes of DDoS in the training data were highly uniform because all attacks were launched with the same tool against one target. Fwd Packet Length Max was always 20 bytes in the DDoS training flows, whereas its median in the campus DDoS flows was 132 bytes. Among the ten features with the highest MDI in Model 1, four size features, namely Fwd Packet Length Max, Fwd Packet Length Mean, Bwd Packet Length Max, and Packet Length Variance, had no campus DDoS flow within the interquartile range of the training data. Model 1 therefore relied on packet sizes specific to the recording environment of CICIDS2017, which is the kind of spurious correlation described by Arp et al. (2022).

Model 2 recognized DDoS mainly from connection patterns, since the size features of DDoS had been randomized in the training data. In the training data, all DDoS flows were recorded from a single firewall address, so the number of connections from one source to the same service (NSS) equaled the number of connections from all sources to that service (NSD), with a median of 486. In the campus data, the three attacking machines appeared as three addresses, so NSD was about three times NSS for most tools. Both values nevertheless remained much larger than those of benign training traffic, who's median NSS was only 15, so the pattern of many connections to one service could still be recognized. Recall was highest for LOIC, at 0.994, followed by aiohttp at 0.868.

HOIC and GoldenEye had lower recall, 0.691 and 0.719, even though they opened more connections per source than the DDoS training data, with medians of 940 and 688. This difference cannot be explained by the number of connections alone and is likely influenced by other features, such as the flag features that also contribute to DDoS recognition. Slowloris had the lowest recall, 0.120, because it opens far fewer connections and keeps them open longer, with a median of 140 connections per source and a mean flow duration of about 62 seconds, compared

with 486 connections and about 7 seconds in the DDoS training data. In CICIDS2017, Slowloris belongs to the DoS attacks and was therefore not part of the DDoS training class.

For FTP brute force, both models achieved recall above 0.94 for all tools. This result is related to Bwd Packet Length Max, the largest packet size sent by the server, which was 34 bytes in all FTP brute-force training flows and in 1,482 of the 1,490 campus flows. This size equals the length of the reply "331 Please specify the password." sent by the vsftpd server on the target after it receives a user name to request the password (Postel & Reynolds, 1985). Because the same value appeared in all training flows, the FTP server in CICIDS2017 likely produced the same reply. FTP brute force was therefore recognized from the server's authentication reply, which depends on the FTP server software rather than on the attack tool.

In contrast, SSH brute-force recall remained close to zero for both models. To find the cause, each connection to port 22 was examined by its connection stage. Table 5 shows the recall for each stage, averaged over five seeds.

Table 5 Recall of SSH Brute Force by Connection Stage

Connection Stage	Flows	Model 1	Model 2
Accepted	332	0.089	0.265
Rejected	51,614	0.000	0.000
Other	34	0.000	0.000

Of the 51,980 flows, 51,614, or about 99.3%, came from connections rejected by the OpenSSH server through the PerSourcePenalties mechanism with the message "Not allowed at this time". The server log on the target recorded all of these rejections as penalties for failed authentication, and none was caused by MaxStartups. The rejections appeared at the medium and high levels after many failed logins attempts from the same address. Because PerSourcePenalties is enabled by default (OpenBSD, n.d.), this condition can also occur on other SSH servers with the default configuration. Some tools kept opening new connections after being rejected, so the number of flows far exceeded the number of passwords tried. Ncrack, for example, produced 49,757 flows from 300 passwords.

Rejected connections never reach the authentication stage, so their flows contain no login attempts. Model 1 mostly predicted these flows as port scanning and Model 2 as benign, and Model 2 also predicted 4,401 SSH brute-force flows as DDoS. The repeated connections produced a very large number of flows from one source in one window, with a median of 1,138 compared with 6 in the training data, which resembles DDoS more than SSH brute force in the training data. These rejected flows are similar to the Attempted category of the CNS2022 version of CICIDS2017, which was excluded from training (Engelen et al., 2021), so SSH brute-force recognition is better assessed on accepted connections, which contain actual login attempts. On these connections, recall was 0.089 for Model 1 and 0.265 for Model 2. In the run with seed 42, accepted connections from Patator were recognized by Model 2 with a recall of 0.918, whereas recall for Hydra, Medusa, and Ncrack was below 0.04. Patator is the same tool used to generate the SSH brute-force class in

CICIDS2017 (Sharafaldin et al., 2018), so the low recall was also caused by the dependence of the models on the tool used in the training data.

For port scanning, Model 1 recognized Nmap and RustScan with recall above 0.99 but only 14.1% of Masscan flows. This difference was not visible in individual feature values, because the medians of the ten most important Model 1 features for Masscan were almost the same as those for Nmap and RustScan. In the TCP group, the initial window size of both Masscan and Nmap was 1,024 bytes, far from the training median of 29,200 bytes, while the forward header length of Nmap was even further from the training data, at 24 bytes compared with 40 bytes for Masscan and 44 bytes in the training data. Nmap flows were nevertheless recognized, so the failure of Model 1 on Masscan was likely caused by a combination of features rather than a single feature. Model 2 recognized port scanning mainly from contextual features, such as the number of distinct destination ports within a window, and its recall for Masscan increased to 0.942.

4. Results on Operational Traffic

Most operational flows were predicted as benign by both models. The 49,625 analyzed flows came from 143 local IP addresses on the recorded access point. Averaged over five seeds, Model 1 predicted 0.189% of the flows as attacks and Model 2 predicted 0.031%. In the run with seed 42, Model 1 predicted 95 flows and Model 2 predicted 10 flows as attacks, all of them as port scanning.

The ten flows predicted as port scanning by Model 2 were examined individually, and all were short flows of two packets. Seven came from one HTTPS server that sent SYN-ACK segments to seven different ports of one user device within about 50 milliseconds without receiving a reply. Because CICFlowMeter determines the direction of a flow from its first packet (Habibi Lashkari, n.d.), the server was recorded as a source contacting many destination ports, so its NDDP value resembled port scanning. Two other flows were SYN-ACK segments from an HTTPS server that were answered with RST by a user device, and one was a connection from a user device to the gateway on port 80 that was rejected with RST, which resembles a closed-port probe in port scanning. These ten flows were therefore false positives arising from residual TCP connection establishment rather than from scanning activity.

Because the operational traffic has no ground truth, these proportions cannot be treated as exact error rates, but they can be viewed as an upper bound on false positives if all operational traffic is normal. A low proportion also does not necessarily mean that normal traffic is recognized well, since Model 1 did not predict any DDoS flow in the simulations either. The operational results should therefore be read together with the simulation results.

5. Feature Group Importance

Table 6 shows the grouped permutation feature importance of Model 2 for each attack class, calculated in the run with seed 42 on a sample of 48,713 simulation

flows. Because each tool had equal weight in the sample, the initial recall can differ from Table 3.

Table 6. Grouped Permutation Feature Importance of Model 2 by Attack Class

Attack Class	Initial Recall	IP Activity (7)	Service Level (3)	Flag (12)	Size (12)
DDoS	0.679	0.232	0.283	0.238	−0.034
FTP brute force	0.940	0.595	0.444	0.752	0.903
SSH brute force	0.013	0.007	0.004	0.013	0.013
Port scanning	0.992	0.684	0.677	0.438	0.317

The feature groups that mattered differed by attack class. For port scanning, the largest decreases occurred in the two contextual groups, 0.684 and 0.677, so port scanning was recognized mainly from connection patterns within a window, such as the number of destination ports contacted by one source. For DDoS, the three groups other than size had similar decreases between 0.232 and 0.283, while shuffling the size group had almost no effect because the DDoS size features had been randomized in the training data. DDoS was therefore recognized from a combination of connection patterns and flags rather than from packet sizes. In contrast, FTP brute force depended most on the size and flag groups, with decreases of 0.903 and 0.752, which is consistent with the identical Bwd Packet Length Max values in the training and campus data. For SSH brute force, all decreases were very small because the initial recall was only 0.013. Across all classes, the individual features with the largest importance were NSD and NSS, followed by NDDP and LT.

6. Comparison with Previous Studies and Limitations

The decline from the CICIDS2017 test set to the campus simulations agrees with the cross-dataset results of Cantone et al. (2024), who also found that models performing well within one dataset lost performance on other data. The present results add a breakdown by attack class and tool, which shows that the decline was not uniform. The failure of Model 1 on DDoS illustrates the spurious correlations described by Arp et al. (2022) and the low diversity of CICIDS2017 attack data reported by Flood et al. (2024), because the near-constant packet sizes of the single-tool DDoS training data were learned as a class signature. Randomizing these values at the feature level, following the principle of domain randomization (Tobin et al., 2017), allowed Model 2 to rely on connection patterns instead. The importance of time-window features for port scanning and DDoS is consistent with the use of IP activity features by Meng et al. (2022) to trace scanning and DDoS activity in a campus network, although in this study they were used within a supervised classifier. The very high within-dataset scores reported by Rodríguez et al. (2022), Alduailij et al. (2022), and Abdelaziz et al. (2025), which were also observed for both models in this study, therefore do not indicate how a model will perform on another network. For network administrators, these results suggest that a flow-based model trained on a public dataset should be validated for each attack class before deployment, and that SSH brute-force detection must take server-side defenses, such

as connection rejection, into account.

This study has several limitations. The campus simulation traffic was examined during the development of Model 2, so its results on the simulations are not a fully independent test. The simulation recordings contained only traffic between the attacker and target devices, so the contextual features were computed without traffic from other users, and their values may change when attacks are mixed with normal traffic. The recognition of FTP brute force relied on a server reply whose size matched the training data, and other FTP server software was not tested. Finally, the operational traffic had no ground truth, so its prediction errors could not be measured exactly.

E. CONCLUSION

Regarding the first research question, flow-based Random Forest models that perform almost perfectly on the CICIDS2017 test set do not keep that performance on a campus WiFi network, and the decline depends on the attack class. FTP brute force remained recognizable, port scanning and DDoS were recognized well only when the model relied on connection patterns rather than packet sizes, and SSH brute force was hardly recognized by either model.

Regarding the second research question, each result could be traced to specific traffic characteristics. FTP brute force was recognized through the size of the server's authentication reply, which was identical in both datasets. DDoS and port scanning depended on contextual features that describe connection patterns within a time window, whereas the uniform packet sizes of the single-tool DDoS training data caused the model with selected flow features to fail. SSH brute force was hidden by the server's rejection of repeated connections before authentication. A high score on a benchmark dataset therefore says little about performance on another network unless it is examined for each attack class together with the traffic characteristics behind it.

Based on these findings, network administrators are advised to validate a model trained on a public dataset on local traffic for each attack class before deployment and to take server-side defenses, such as connection rejection, into account when monitoring SSH. Future research should test the models on new campus recordings collected after all configurations are fixed, mix attack and normal traffic within the same time windows, use other server software, separate rejected SSH connections using server authentication logs, label part of the operational traffic, and combine the CNS2022 versions of CICIDS2017 and CSE-CIC-IDS2018 to reduce the dependence on a single recording environment.

REFERENCES

1. Abdelaziz, M. T., Radwan, A., Mamdouh, H., Saad, A. S., Abuzaid, A. S., AbdElhakeem, A. A., Zakzouk, S., Moussa, K., & Darweesh, M. S. (2025). Enhancing network threat detection with random forest-based NIDS and

permutation feature importance. *Journal of Network and Systems Management, 33*(1). https://doi.org/10.1007/s10922-024-09874-0

2. Alduailij, M., Khan, Q. W., Tahir, M., Sardaraz, M., Alduailij, M., & Malik, F. (2022). Machine-learning-based DDoS attack detection using mutual information and random forest feature importance method. *Symmetry, 14*(6), 1095. https://doi.org/10.3390/sym14061095
3. Ardiansyah, Suradi, A. A. M., & Saputra, W. (2025). Strategi keamanan router MikroTik: Deteksi dan mitigasi serangan brute force berbasis scripting. *JUKI: Jurnal Komputer dan Informatika, 7*(1), 12–19.
4. Arp, D., Quiring, E., Pendlebury, F., Warnecke, A., Pierazzi, F., Wressnegger, C., Cavallaro, L., & Rieck, K. (2022). Dos and don'ts of machine learning in computer security. *Proceedings of the 31st USENIX Security Symposium*, 3971–3988.
5. Au, Q., Herbinger, J., Stachl, C., Bischl, B., & Casalicchio, G. (2022). Grouped feature importance and combined features effect plot. *Data Mining and Knowledge Discovery, 36*(4), 1401–1450. https://doi.org/10.1007/s10618-022-00840-5
6. Badan Siber dan Sandi Negara. (2025). *Lanskap keamanan siber Indonesia 2024*. Id-SIRTII/CC – BSSN.
7. Breiman, L. (2001). Random forests. *Machine Learning, 45*(1), 5–32. https://doi.org/10.1023/A:1010933404324
8. Cantone, M., Marrocco, C., & Bria, A. (2024). Machine learning in network intrusion detection: A cross-dataset generalization study. *IEEE Access, 12*, 144489–144508. https://doi.org/10.1109/ACCESS.2024.3472907
9. Engelen, G. (n.d.). *CICFlowMeter (fixed version)*. Retrieved November 1, 2026, from https://github.com/GintsEngelen/CICFlowMeter
10. Engelen, G., Rimmer, V., & Joosen, W. (2021). Troubleshooting an intrusion detection dataset: The CICIDS2017 case study. *2021 IEEE Security and Privacy Workshops (SPW)*, 7–12. https://doi.org/10.1109/SPW53761.2021.00009
11. Flood, R., Engelen, G., Aspinall, D., & Desmet, L. (2024). Bad design smells in benchmark NIDS datasets. *2024 IEEE 9th European Symposium on Security and Privacy (EuroS&P)*, 658–675. https://doi.org/10.1109/EuroSP60621.2024.00042
12. Graham, R. D. (n.d.). *MASSCAN: Mass IP port scanner*. Retrieved November 1, 2026, from https://github.com/robertdavidgraham/masscan
13. Grinsztajn, L., Oyallon, E., & Varoquaux, G. (2022). Why do tree-based models still outperform deep learning on typical tabular data? *Advances in Neural Information Processing* Systems, *35*. https://doi.org/10.52202/068431-0037
14. Habibi Lashkari, A. (n.d.). *CICFlowMeter*. Retrieved November 1, 2026, from https://github.com/ahlashkari/CICFlowMeter
15. Intan Sabila, M., Tahir, M., Dwi Mardania, S., & Ilham Arifin, R. (2025). Implementasi Snort sebagai IDS dalam mendeteksi serangan port scanning Nmap pada simulasi jaringan virtual. *JATI (Jurnal Mahasiswa Teknik Informatika), 9*(4), 6944–6948. https://doi.org/10.36040/jati.v9i4.14340

16. Koumar, J., Hynek, K., Čejka, T., & Šiška, P. (2025). CESNET-TimeSeries24: Time series dataset for network traffic anomaly detection and forecasting. *Scientific Data, 12*, 338. https://doi.org/10.1038/s41597-025-04603-x
17. Kurose, J. F., & Ross, K. W. (2021). *Computer networking: A top-down approach* (8th ed.). Pearson.
18. Lee, W., & Stolfo, S. J. (2000). A framework for constructing features and models for intrusion detection systems. *ACM Transactions on Information and System Security, 3*(4), 227–261. https://doi.org/10.1145/382912.382914
19. Liu, L., Engelen, G., Lynar, T., Essam, D., & Joosen, W. (2022). Error prevalence in NIDS datasets: A case study on CIC-IDS-2017 and CSE-CIC-IDS-2018. *2022 IEEE Conference on Communications and Network Security (CNS)*, 254–263. https://doi.org/10.1109/CNS56114.2022.9947235
20. Meng, Y., Qin, T., Li, S., & Wang, P. (2022). Behavior pattern mining from traffic and its application to network anomaly detection. *Security and Communication Networks, 2022*. https://doi.org/10.1155/2022/9139321
21. OpenBSD. (n.d.). *sshd_config(5): OpenSSH daemon configuration file*. Retrieved October 31, 2026, from https://man.openbsd.org/sshd_config
22. Opitz, J. (2024). A closer look at classification evaluation metrics and a critical reflection of common evaluation practice. *Transactions of the Association for Computational Linguistics, 12*, 820–836. https://doi.org/10.1162/tacl_a_00675
23. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, É. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research, 12*, 2825–2830. https://jmlr.org/papers/v12/pedregosa11a.html
24. Postel, J., & Reynolds, J. (1985). *File Transfer Protocol* (RFC 959). https://doi.org/10.17487/RFC0959
25. Rodríguez, M., Alesanco, Á., Mehavilla, L., & García, J. (2022). Evaluation of machine learning techniques for traffic flow-based intrusion detection. *Sensors, 22*(23). https://doi.org/10.3390/s22239326
26. scikit-learn developers. (n.d.). *Ensembles: Gradient boosting, random forests, bagging, voting, stacking*. Retrieved November 2, 2026, from https://scikit-learn.org/stable/modules/ensemble.html
27. Sharafaldin, I., Lashkari, A. H., & Ghorbani, A. A. (2018). Toward generating a new intrusion detection dataset and intrusion traffic characterization. *Proceedings of the 4th International Conference on Information Systems Security and Privacy (ICISSP 2018)*, 108–116. https://doi.org/10.5220/0006639801080116
28. Tobin, J., Fong, R., Ray, A., Schneider, J., Zaremba, W., & Abbeel, P. (2017). Domain randomization for transferring deep neural networks from simulation to the real world. *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 23–30.