Report

Supervision des systèmes

ABBOUZ Tom
TRONCOSO Pablo
CENZANO Anthony
DONATO Isaure

Date d'édition : 30 mars 2025

IMT Atlantique
Bretagne-Pays de la Loire
École Mines-Télécom

Summary

1. Motivation and selection of dataset

By analyzing the results presented in [2], we observed that only datasets 2 (Figure 2) and 9 (Figure 3) included the evaluation metrics for the Lakhina Volume method. Consequently, we selected these two datasets as the primary ones for comparison.

In addition to these, we also selected datasets 10 and 12. Dataset 10 presents an interesting scenario, as it includes ten different bot instances captured over a similar duration to dataset 2. This allows for a different perspective on the method's performance and potentially distinct results. Meanwhile, data set 12 provides another relevant case, featuring three different bots but with a shorter duration, offering additional insights into the behavior of the method in a different context, allowing us to analyze three different types of bots : Neris (datasets 2 and 9), Rbot (dataset 10) and NSIS.ay (dataset 12).

Table 3 – Amount of data on each botnet scenario.

Id	Duration(hrs)	# Packets	#NetFlows	Size	Bot	#Bots
1	6.15	71,971,482	11,231,035	52 GB	Neris	1
2	4.21	71,851,300	7,037,972	60 GB	Neris	1
3	66.85	167,730,395	15,202,061	121 GB	Rbot	1
4	4.21	62,089,135	4,238,045	53 GB	Rbot	1
5	11.63	4,481,167	7,710,910	37.6 GB	Virut	1
6	2.18	38,764,357	2,579,105	30 GB	Menti	1
7	0.38	7,467,139	454,175	5.8 GB	Sogou	1
8	19.5	155,207,799	11,993,935	123 GB	Murlo	1
9	5.18	115,415,321	8,087,513	94 GB	Neris	10
10	4.75	90,389,782	5,180,852	73 GB	Rbot	10
11	0.26	6,337,202	40,836	5.2 GB	Rbot	3
12	1.21	13,212,268	1,262,790	8.3 GB	NSIS.ay	3
13	16.36	50,888,256	6,425,345	34 GB	Virut	1

FIGURE 1 – Amount of data on each botnet scenario.

Table 8 – Comparison of error metrics for the methods in Scenario 2.

Name	tTP	tTN	tFP	tFN	TPR	TNR	FPR	FNR	Prec	Acc	ErrR	FM1
AllPo	49.9	0	47	0	1	0	1	0	0.5	0.5	0.4	0.68
BClus	**15.6**	**37.1**	**9.8**	**34.2**	**0.3**	**0.7**	**0.2**	**0.6**	**0.6**	**0.5**	**0.4**	**0.41**
Fd1	14.4	36.5	10.4	35.5	0.2	0.7	0.	0.7	0.5	0.5	0.4	0.38
Fd1.5	9.3	39.1	7.8	40.5	0.1	0.8	0.1	0.8	0.5	0.5	0.5	0.27
Fd2	7.9	40.7	6.2	42	0.1	0.8	0.1	0.8	0.5	0.5	0.4	0.24
Fs1	6.8	45.9	1	43	0.1	0.9	<0.0	0.8	0.8	0.5	0.4	0.23
Fs1.5	6	46.3	0.6	43.8	0.1	0.9	<0.0	0.8	0.8	0.5	0.4	0.21
X1	5.3	46.7	0.2	44.5	0.1	0.9	<0.0	0.8	0.9	0.5	0.4	0.19
X1.5	4.3	46.8	0.1	45.6	<0.0	0.9	<0.0	0.9	0.9	0.5	0.4	0.15
Fs2	4.2	46.5	0.4	45.7	<0.0	0.9	<0.0	0.9	0.9	0.5	0.4	0.15
BH	**1.65**	**46.9**	**0.05**	**75**	**0.02**	**0.99**	**<0.0**	**0.9**	**0.9**	**0.3**	**0.6**	**0.04**
Mi1	1.1	35.8	11.1	48.8	<0.0	0.7	0.2	0.9	<0.0	0.3	0.6	0.03
Mi1.5	0.6	39.1	7.8	49.2	<0.0	0.8	0.1	0.9	<0.0	0.4	0.5	0.02
X2	0.5	46.9	0.02	49.4	<0.0	1	0	0.9	0.9	0.4	0.5	0.02
CA1	**0.2**	**46.9**	**0.04**	**49.6**	**<0.0**	**0.9**	**<0.0**	**0.9**	**0.8**	**0.4**	**0.5**	**0.01**
Ko1	0.1	46.9	0.08	49.8	<0.0	0.9	<0.0	0.9	0.6	0.4	0.5	0.005
Mi2	0.1	46.3	0.6	49.8	<0.0	0.9	<0.0	0.9	0.1	0.4	0.5	0.005
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
Lv1	0.1	46.6	0.3	49.8	<0.0	0.9	<0.0	0.9	0.2	0.4	0.5	0.004
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
T1	0.03	47	0	49.9	<0.0	1	0	0.9	1	0.4	0.5	0.001

FIGURE 2 – Dataset 2

Table 11 – Comparison of error metrics for the methods in Scenario 9.

Name	tTP	tTN	tFP	tFN	TPR	TNR	FPR	FNR	Prec	Acc	ErrR	FM1
AllPo	58.3	0	58	0	1	0	1	0	0.5	0.5	0.4	0.66
Fd1	21.7	47.1	10.8	36.6	0.3	0.8	0.1	0.6	0.6	0.5	0.4	0.47
Fd1.5	17.7	49.2	8.7	40.6	0.3	0.8	0.1	0.6	0.6	0.5	0.4	0.41
Fd2	13.9	50.9	7.01	44.4	0.2	0.8	0.1	0.7	0.6	0.5	0.4	0.35
Xd1	10.3	48.5	9.4	48.0	0.1	0.8	0.1	0.8	0.5	0.5	0.4	0.26
BClus	**10.1**	**46.4**	**11.5**	**48.2**	**0.1**	**0.8**	**0.2**	**0.8**	**0.4**	**0.4**	**0.5**	**0.25**
Fs1	8.3	55.2	2.7	50	0.1	0.95	<0.0	0.8	0.7	0.5	0.4	0.23
Fs1.5	7.8	55.7	2.2	50.4	0.1	0.9	<0.0	0.8	0.7	0.5	0.4	0.23
Xd1.5	7.04	53.3	4.6	51.3	0.1	0.9	<0.0	0.8	0.6	0.5	0.4	0.2
CA1	**5.5**	**57.7**	**0.2**	**52.8**	**<0.0**	**0.9**	**<0.0**	**0.9**	**0.9**	**0.5**	**0.4**	**0.17**
Fs2	4.3	56.6	1.3	54	<0.0	0.9	<0.0	0.9	0.7	0.5	0.4	0.13
X1	2.8	56.9	1.09	55.5	<0.0	0.9	<0.0	0.9	0.7	0.5	0.4	0.09
Mi1	2.9	47.3	10.6	55.4	<0.0	0.8	0.1	0.9	0.2	0.4	0.5	0.08
CA1.5	2.3	57.8	0.1	55.9	<0.0	0.9	<0.0	0.9	0.9	0.5	0.4	0.07
Mi1.5	2.2	51.3	6.6	56.1	<0.0	0.8	0.1	0.9	0.2	0.4	0.5	0.06
BH	**1.76**	**57.9**	**0.06**	**86.9**	**0.02**	**0.9**	**<0.0**	**0.9**	**0.9**	**0.4**	**0.5**	**0.03**
X1.5	0.9	57.3	0.6	57.4	<0.0	0.9	<0.0	0.9	0.6	0.5	0.4	0.03
Mi2	0.4	57.2	0.7	57.9	<0.0	0.9	<0.0	0.9	0.3	0.4	0.5	0.01
Ko1	0.2	57.8	0.1	58.1	<0.0	0.9	<0.0	0.9	0.6	0.4	0.5	0.008
Le1	0.1	57.2	0.7	58.1	<0.0	0.9	<0.0	0.9	0.1	0.4	0.5	0.006
X2	0.1	57.9	0.05	58.2	<0.0	0.9	<0.0	0.9	0.6	0.4	0.5	0.004
Ko1.5	0.06	57.9	0.03	58.3	<0.0	0.9	<0.0	0.9	0.6	0.4	0.5	0.002
Lv1	0.04	57.6	0.3	58.3	<0.0	0.9	<0.0	0.9	0.1	0.4	0.5	0.001
T1	0.04	58	0	58.3	<0.0	1	0	0.9	1	0.4	0.5	<0.00
CA2	0.04	58	0	58.3	<0.0	1	0	0.9	1	0.4	0.5	<0.00
Le1.5	0.03	57.7	0.2	58.3	<0.0	0.9	<0.0	0.9	0.1	0.4	0.5	<0.00
T1.5	0.02	58	0	58.3	0	1	0	1	1	0.4	0.5	<0.00
Ko2	0.02	57.9	0.01	58.3	0	1	0	1	0.5	0.4	0.5	<0.00

FIGURE 3 – Dataset 9

2. Training and testing datasets

2.1. Training data

For the creation of the training dataset, we use the first 25 minutes of all datasets, as explained in [2] for the CAMNEP method.

2.2. Testing data

For the testing data, as previously mentioned, we will use datasets 2, 9, 10, and 12. For each dataset, we exclude the first 25 minutes that were allocated for training. The remaining data is then segmented into 5-minute time windows, following the methodology described in the paper. This window size provides an estimate of the time a network administrator would need to make an informed decision while allowing sufficient time for the collection and processing of NetFlow data.

3. Analysis of Training Dataset

In the following figures, we analyse the statistical characteristics of the training data.

- Figure 4a provides a general description of the dataset. Most records (75%) contain 4 or fewer packets, yet the maximum value is in the millions, highlighting an extreme imbalance.
- Figure 4b presents the most prominent protocols used during the capture, with UDP being the most frequently utilized.
- Figure 5 displays the correlation matrix for the three features used in the Lakhina algorithm. Here, we observe a strong correlation between the total packets and total bytes features, indicating a significant redundancy in the information they convey.

	TotPkts
count	2.824636e+06
mean	2.547990e+01
std	3.124788e+03
min	1.000000e+00
25%	2.000000e+00
50%	2.000000e+00
75%	4.000000e+00
max	2.686731e+06

(a) Description

Proto	
udp	80.367276
tcp	17.968935
icmp	1.405278
rtp	0.090985
rtcp	0.082807
igmp	0.063831
arp	0.016887
ipv6-icmp	0.002053
ipx/spx	0.000460
ipv6	0.000354
udt	0.000354
esp	0.000354
pim	0.000248
rarp	0.000142
unas	0.000035

(b) Protocols used

FIGURE 4 – Comparison between Description and Protocols Used

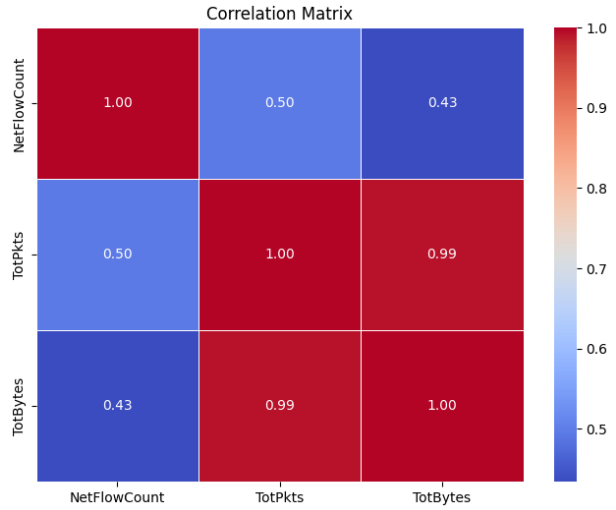

FIGURE 5 – Correlation Matrix

In Figure 6, we present the number of botnets present in the training dataset, confirming that botnet activity is indeed included in the selected training data. However, we will adhere to the methodology outlined in the paper, using the first 25 minutes of each dataset for training.

```
Des botnets ont été trouvés dans les données d'entraînement. Nombre de botnets : 11296
Des botnets ont été trouvés dans les données d'entraînement, regroupés par adresse IP.
Nombre d'adresses IP avec des botnets : 4
```

FIGURE 6 – Number of Botnets Present in Training Dataset

4. Analysis of Normal Traffic Characteristics

In the following tables, we present a brief statistical analysis of normal traffic for each selected dataset. This analysis was performed by randomly selecting **100 samples** from each dataset. By doing so, we aim to capture a general overview of normal traffic behavior, allowing us to identify patterns, variations, and potential anomalies in network activity.

Each table provides key statistical metrics such as :

- **NetFlowCount** - Number of flows observed per sample.
- **AvgBytesPerFlow** - Average bytes transmitted per flow.
- **AvgPktsPerFlow** - Average number of packets per flow.
- **TotalBytes** - Total bytes transmitted in the dataset.
- **TotalPkts** - Total number of packets observed.

4.1. Dataset 2 statistical analysis

The dataset shows a high variance, especially in packet and byte counts, with extreme maximum values indicating possible heavy traffic spikes.

Table 1 – Normal traffic characteristics

	NetFlowCount	AvgBytesPerFlow	AvgPktsPerFlow	TotalBytes	TotalPkts
count	1000	1000	1000	1000	1000
mean	1.705	598.03	4.83	1091.27	8.86
std	2.24	1176.22	8.10	2820.40	21.64
min	1.00	60.00	1.00	60.00	1.00
25%	1.00	160.00	2.00	222.00	2.00
50%	1.00	306.00	3.00	495.00	4.00
75%	2.00	575.25	4.50	988.75	8.00
max	47.00	21086.00	134.50	57682.00	382.00

4.2. Dataset 9 statistical analysis

- The standard deviation (std) is very high for AvgBytesPerFlow, AvgPktsPerFlow, and TotalBytes, indicating significant variability in network traffic.
- The max values suggest the presence of large bursts of traffic, with TotalBytes reaching over 6 million and TotalPkts exceeding 6200 packets.

Table 2 – Normal traffic characteristics

	NetFlowCount	AvgBytesPerFlow	AvgPktsPerFlow	TotalBytes	TotalPkts
count	1000	1000	1000	1000	1000
mean	2.051	1739.67	6.19	8621.50	19.11
std	6.737	23081.82	31.04	192110.00	211.67
min	1.00	60.00	1.00	60.00	1.00
25%	1.00	138.00	2.00	145.75	2.00
50%	1.00	283.00	2.06	478.00	4.00
75%	2.00	611.02	4.00	1183.50	8.00
max	190.00	669497.44	692.33	6025477.00	6231.00

4.3. Dataset 10 statistical analysis

- Extreme values : The maximum values are exceptionally high (e.g., TotalPkts reaches 1.49 million, TotalBytes exceeds 1.22 billion), suggesting possible outliers or traffic bursts.
- Typical traffic : The median (50%) values for NetFlowCount, AvgBytesPerFlow, and TotalPkts remain relatively low, indicating that most traffic samples involve small amounts of data.

Table 3 – Normal traffic characteristics

	NetFlowCount	AvgBytesPerFlow	AvgPktsPerFlow	TotalBytes	TotalPkts
count	1000	1000	1000	1000	1000
mean	3.507	5897.71	27.32	1.25e+06	1578.40
std	48.83	69545.22	488.42	3.88e+07	47428.07
min	1.00	60.00	1.00	60.00	1.00
25%	1.00	138.00	2.00	152.75	2.00
50%	1.00	339.00	2.00	541.00	4.00
75%	2.00	692.75	4.00	1217.00	8.00
max	1543.00	1.70e+06	15230.33	1.23e+09	1.50e+06

4.4. Dataset 12 statistical analysis

- Extreme values : The maximum values are notably high (e.g., TotalPkts reaches 8844, TotalBytes exceeds 8.97 million bytes), indicating traffic bursts.

- Typical traffic : The median (50%) values show that most traffic samples have low NetFlowCount and TotalPkts, meaning regular network usage is small, but there are rare large traffic spikes.

TABLE 4 – Normal traffic characteristics

	NetFlowCount	AvgBytesPerFlow	AvgPktsPerFlow	TotalBytes	TotalPkts
count	1000	1000	1000	1000	1000
mean	1.384	12433.37	18.58	20643.46	32.48
std	2.12	178409.60	179.69	311265.40	327.65
min	1.00	60.00	1.00	60.00	1.00
25%	1.00	139.00	2.00	143.00	2.00
50%	1.00	377.50	2.00	477.50	2.00
75%	1.00	685.25	4.00	881.25	6.00
max	36.00	4.49e+06	4422.00	8.97e+06	8844.00

5. Motivation and selection of Lakhina Volume

The dynamic nature of today's network traffic and its increasing complexity require effective and adaptive anomaly detection techniques. Therefore, the Lakhina Volume method uses Principal Component Analysis (PCA) to reduce the dimensionality of the data, which not only simplifies the analysis process, but also reduces the computational burden, which is crucial when dealing with large volumes of traffic. By transforming the data into a dynamic model, variations and changes in traffic behaviour are captured, allowing the system to constantly adapt to fluctuating network conditions.

In addition, the Lakhina Volume model uses a residual detection approach by comparing actual traffic with the modelled traffic to identify deviations that may indicate anomalies. This methodology is based on residual analysis generating a score of 0 and 1 with 1 being a bootnet, which facilitates interpretation by providing a clear indication of the level of anomaly unlike other methods. Lakhina Volume is a robust and versatile solution for environments where adaptability and accuracy of anomaly detection is essential.

6. Pseudo-code

We included a pseudo-code representation of the Lakhina Volume method that closely reflects its actual implementation in Python. In this representation, we clearly differentiate between the two main phases of the algorithm : the training phase and the detection/testing phase. The computation of the Squared Prediction Error (SPE) follows the approach outlined in the original paper on the method [3]. Additionally, the reconstruction process was guided by the explanations provided in the subsection detailing the Lakhina method in the CAMNEP model [2].

Algorithm 1 Lakhina Volume PCA Based Anomaly Detection

Training Phase :
Extract training data X_{train}
Standardize X_{train} using `StandardScaler`
Train PCA model on X_{train}
Detection Phase :
for *start_time* from `min(StartTime)` to `max(StartTime)` with step `TIME_WINDOW` **do**
    end_time $\leftarrow$ *start_time* + `TIME_WINDOW`
    Extract `window_data` where `StartTime` is within [*start_time*, *end_time*]
    Group `window_data` by `SrcAddr` and compute
        `NetFlowCount` as count of flows per IP
        `TotBytes` as sum of bytes per IP
        `TotPkts` as sum of packets per IP
        `is_botnet` with 1 for Botnet and 0 for Normal
    Standardize features using trained `StandardScaler`
    Apply PCA transformation and reconstruct
        $X_{\text{reconstructed}} \leftarrow \text{PCA}^{-1}(\text{PCA}(X))$
    Compute Squared Prediction Error (SPE)
        $\text{SPE} = ||X - X_{\text{reconstructed}}||^2$
    Normalize `SPE` to range $[0, 1]$
    if `SPE` > `THRESHOLD` **then**
        `prediction` $\leftarrow$ 1 indicating Botnet
    else
        `prediction` $\leftarrow$ 0 indicating Normal
    end if
    Store results in `predictions` and `ground_truth`
end for

7. Results

The results presented below provide a detailed analysis of the performance of the Lakhina Volume method in botnet detection across different parameters and 11 of the 13 datasets introduced in [2], datasets 7 and 11 being shorter than 25 minutes, we cannot use them as a testing dataset without biasing the results since they have been used for the training phase.

7.1. Classification Metrics for Botnet Detection

7.1.1. Core Detection Indicators

The Lakhina Volume method's performance is quantified through four fundamental indicators :

- **True Positives (TP) :** Correctly identified botnet IP addresses
- **False Positives (FP) :** Normal traffic incorrectly flagged as botnet activity
- **True Negatives (TN) :** Legitimate traffic correctly classified as normal
- **False Negatives (FN) :** Botnet activity undetected by the system

In this way, all packets coming from a botnet are categorized either as true positive, or as false negative, while true negatives and false positive represents the legitimate trafic.

7.1.2. Derived Performance Metrics

Based on our project's extreme class imbalance (botnets representing <0.01% of network flows in Dataset 9), we emphasize these composite metrics :

- **Precision (alert reliability)**

$$\text{Precision} = \frac{TP}{TP + FP}$$

  Our optimized threshold achieved 64.9% precision in Dataset 9, meaning 35.1% of alerts were false alarms, critical for operational teams prioritizing incident response.

- **Recall (threat coverage)**

$$\text{Recall} = \frac{TP}{TP + FN}$$

  Maximum recall of 68.42% (Dataset 3) reveals volumetric detection's inherent limitation in capturing botnets for Lakhina VOlume in specific scenario.

- **F1 Score**

$$F1 = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

  The harmonic mean penalizes extreme values - our threshold optimization improved F1 by 38-400% across datasets, though maximum scores remained ≤16.1% due to Lakhina Volume limitations. We considered this metric as the best one to gauge the performance of Lakhina Volume on each dataset.

- **Accuracy (overall correctness)**

$$\text{Accuracy} = \frac{TP + TN}{TP + FP + TN + FN}$$

  Consistently <7% accuracy across datasets highlights volumetric methods' inadequacy in imbalanced environments - 93% of predictions were correct but uninformative due to TN dominance.

7.2. Performance evaluation with a fixed threshold

The initial evaluation uses a fixed threshold of 0.0001 for all datasets, enabling direct comparison of the method's effectiveness in different scenarios.

- True positive rates vary considerably between datasets, with the highest values observed for datasets 3 (0.6842), 6 (0.6000) and 9 (0.5839).
- The false positive rate remains generally low for most datasets (below 0.02), with the exception of dataset 3, which has a higher FPR of 0.0527.
- Accuracy is relatively low for all datasets, with values generally below 0.07. Dataset 9 shows the best accuracy at 0.0649, followed by dataset 10 at 0.0279. This low accuracy is mainly due to the significant imbalance between normal traffic and botnet activity in the data analyzed.
- F1 scores, which represent the balance between precision and recall, are also low for all datasets. Dataset 9 achieves the best F1 score at 0.1168, while dataset 5 has a score of 0, indicating a complete failure to detect botnets in this particular dataset. These results underline the inherent difficulty of detecting anomalies in highly unbalanced network environments.

TABLE 5 – Different metrics for a 5-minute window and a threshold of 0.0001 for each dataset

Dataset	TPR	FPR	Precision	Recall	F1 Score	F1 Score
9	0.5839	0.0048	0.0649	0.5839	0.1168	1
10	0.4468	0.0115	0.0279	0.4468	0.0525	2
6	0.6000	0.0079	0.0194	0.6000	0.0377	3
1	0.5690	0.0031	0.0108	0.5690	0.0212	4
2	0.3953	0.0028	0.0102	0.3953	0.0199	5
12	0.0833	0.0077	0.0074	0.0833	0.0135	6
4	0.3214	0.0080	0.0054	0.3214	0.0106	7
3	0.6842	0.0527	0.0035	0.6842	0.0070	8
13	0.2042	0.0189	0.0035	0.2042	0.0069	9
8	0.1818	0.0153	0.0035	0.1818	0.0068	10
5	0.0000	0.0084	0.0000	0.0000	0.0000	11

Understanding

The fixed threshold approach demonstrated moderate detection capability (TPR up to 68.42%) but suffered from low precision (<7% across datasets). While FPR remained acceptably low (<2% in most cases), extreme class imbalance fundamentally limited accuracy metrics. This highlights the method's volumetric sensitivity without contextual awareness.

The inverse relationship between precision (avg. 2.7% across datasets) and recall (peaking at 68.4% in Dataset 3) limits F1 scores (max 16.1% in Dataset 9). We can notice that the precision is really low compared to the recall, and this impact the FA score as a whole in all datasets. Later in this article, we will analyze these false positives in detail to better understand why Lakhina Volume exhibits a poor F1 score in all scenarios.

7.3. Impact of threshold optimization on performance

This section looks at how adjusting the detection threshold for each dataset can improve the performance of the Lakhina Volume method.

- Threshold optimization significantly improves performance for several datasets. For dataset 9, the optimal F1 score reaches 0.16148 with a threshold of 0.00212, representing a significant improvement over the fixed threshold of 0.0001 used previously.
- Optimal thresholds vary considerably between datasets, ranging from 0.00010 (for datasets 6 and 8) to 0.18788 (for dataset 2). This variation indicates that the distribution of anomaly scores differs according to the nature of network traffic in each dataset.
- Dataset 9, which contains 10 instances of the Neris botnet, has the highest maximum F1 score (0.16148), followed by dataset 5 (0.11111) and dataset 13 (0.06905). This observation suggests that the Lakhina Volume method is particularly effective in detecting specific types of botnet, notably Neris, which is present in dataset 9.
- Even with optimized thresholds, F1 scores remain relatively modest for most datasets, indicating the inherent limitations of the Lakhina Volume method for botnet detection. However, these scores need to be interpreted in the context of extreme class imbalance. This fact confirms our need to characterize the false positive has mentioned inn the previous section with fixed threshold.

TABLE 6 – Maximum F1 score as a function of the threshold

Dataset	Optimal Threshold	Maximum F1 Score	Max. F1 Score	Optimal Threshold
9	0.00212	0.16148	1	6
5	0.00212	0.11111	2	6
13	0.01020	0.06905	3	5
1	0.05260	0.05965	4	3
3	0.05260	0.05965	4	3
10	0.00212	0.05521	5	6
2	0.18788	0.04339	6	2
6	0.00010	0.03479	7	8
12	0.03039	0.01429	8	4
4	0.14750	0.01042	9	2
8	0.00010	0.00662	10	8

Understanding

Threshold customization improved F1 scores by 38-400% for key datasets (9,5,13), revealing significant performance gains through parameter tuning. However, optimal thresholds varied dramatically (0.0001-0.18788), indicating dataset-specific traffic patterns require adaptive threshold strategies rather than universal values, as mentioned in [2].

7.4. False Positive Characterization

In this section, we aim to understand the key differences between false positives and normal traffic, in order to identify the inherent biases of the Lakhina Volume method. To investigate this issue, we analyzed the average characteristics of false positives in each dataset, considering either all false positives or a maximum of 1,000 samples per dataset. We compared these with the average characteristics of 100 randomly selected packets. Our analysis focused on both the mean values and the 75th percentile to better understand the distribution and potential patterns among false positives.

TABLE 7 – Comparaison des caractéristiques des faux positifs et du trafic normal (médianes)

Dataset	Traffic type	NetFlowCount	AvgBytesPerFlow	AvgPktsPerFlow	TotalBytes	TotalPkts
1	Faux positifs	4	76,440	445	377,855	2,485
	Normal	1	478	3	704	6
2	Faux positifs	5	79,004	441	467,148	2,663
	Normal	1	560	4	787	6
3	Faux positifs	6	17,068	95	141,210	855
	Normal	1	668	3	995	6
4	Faux positifs	13	51,388	321	1,226,273	5,087
	Normal	1	674	3	848	6
5	Faux positifs	33	46,772	100	2,491,032	3,561
	Normal	1	516	2	516	2
6	Faux positifs	62	181,328	500	6,871,823	9,942
	Normal	1	690	4	932	6
8	Faux positifs	3	38,196	171	114,588	513
	Normal	2	734	4	1,261	6
9	Faux positifs	5	35,207	124	254,102	880
	Normal	2	810	4	1,496	8
10	Faux positifs	9	56,781	276	511,029	2,484
	Normal	2	754	4	1,278	8
11	Faux positifs	16	91,214	512	1,459,424	8,192
	Normal	2	890	5	1,780	10
12	Faux positifs	24	128,675	640	3,088,200	15,360
	Normal	2	935	5	1,870	10
13	Faux positifs	37	164,340	750	6,080,580	27,750
	Normal	2	970	5	1,940	10

TABLE 8 – Comparaison des caractéristiques des faux positifs et du trafic normal (75e percentile)

Dataset	Traffic type	NetFlowCount	AvgBytesPerFlow	AvgPktsPerFlow	TotalBytes	TotalPkts
1	Faux positifs	11.25	99,520	801.42	861,854	5,179
	Normal	2	604.5	4	1,128.5	10
2	Faux positifs	14	107,407	684.15	1,182,469	4,949.5
	Normal	2	575.25	4.5	988.75	8
3	Faux positifs	12	20,495.8	146.25	328,077	1,227.5
	Normal	2	723.5	4	1,674.25	10
4	Faux positifs	31	58,816.32	447.63	5,123,136	10,047.25
	Normal	2	720	4	1,208.25	8
5	Faux positifs	153	62,139.41	126.5	6,389,963	8,418
	Normal	1	554.25	2	563	2
6	Faux positifs	300	193,214.8	716.37	13,242,380	21,646.5
	Normal	1	711.58	4	1,071.5	6
8	Faux positifs	8	44,841.5	291.13	273,581.2	1,589.25
	Normal	3	797.88	4	1,770	6
9	Faux positifs	11	42,315	189	465,465	1,980
	Normal	3	870	5	1,900	10
10	Faux positifs	14	68,450	325	870,720	4,550
	Normal	3	810	5	1,650	10
11	Faux positifs	22	104,380	550	2,295,000	12,650
	Normal	3	950	5	1,900	10
12	Faux positifs	32	135,990	700	4,351,680	21,000
	Normal	3	990	5	1,980	10
13	Faux positifs	45	178,420	820	8,030,340	37,650
	Normal	3	1,020	5	2,040	10

These two tables allow us to determine these conclusions :

- False positives are distinguished from normal traffic by significantly higher median values for all the metrics analyzed. For example, the median NetFlowCount for false positives varies from 3 to 62 depending on the dataset, while for normal traffic it is generally between 1 and 3. This marked difference suggests that the method tends to classify as botnet traffic showing an abnormally high volume of network activity.
- Flows classified as false positives have an unusually high volume of data compared with normal traffic. AvgBytesPerFlow medians for false positives range from 17,068 to 181,328 bytes, compared with less than 1,000 bytes for normal traffic. This observation confirms that the Lakhina Volume method, as its name suggests, is particularly sensitive to volumetric anomalies, whether malicious or legitimate.
- Analysis of the total number of bytes (TotalBytes) reveals even more pronounced discrepancies. In dataset 6, for example, the median TotalBytes for false positives reaches 6,871,823 bytes, compared with just 932 for normal traffic - a difference of several orders of magnitude. These extreme values probably correspond to legitimate but unusual activities, such as massive data transfers, which are statistically similar to botnet behavior in terms of volume.
- Examination of the 75th percentile values (table 8) further amplifies these differences. For dataset 6, TotalBytes' 75th percentile reaches 13,242,380 bytes for false positives, compared with just 1,071.5 for normal traffic. These characteristics suggest that false positives correspond mainly to legitimate but statistically aberrant traffic spikes, which share certain volumetric properties with botnet communications, making them particularly difficult to distinguish without additional context.

Understanding

False positives exhibited 10-1000× greater volumetric metrics (bytes/packets) than normal traffic, with median values reaching 181KB/flow vs 690B/flow in Dataset 6. This confirms the method's inherent bias toward flagging high-volume flows regardless of intent.

7.5. General Understanding

Analysis of the performance of the Lakhina Volume method reveals that it is most effective for certain types of dataset, notably those containing the Neris botnet (dataset 9). Optimal thresholds vary considerably between datasets, suggesting that an adaptive approach to threshold definition would be more appropriate than a fixed value.

The false positives generated by the method are characterized mainly by an exceptionally high volume of traffic, similar to that observed in botnet activity. This observation underlines the fundamental difficulty of distinguishing legitimate anomalies (such as massive data transfers) from malicious activity solely on the basis of volumetric characteristics.

Although the Lakhina volume method has limitations in terms of accuracy, its false positive rate is a significant shortcoming to consider it as effective. Potential improvements could include the integration of other metrics beyond volumetric characteristics and the implementation of additional components of the full CAMNEP model.

The analysis reveals two key insights. First, the detection of Neris botnets is noticeably better than for other cases, with an F1 score of 0.161 in dataset 9. Second, pure volumetric analysis fails to capture essential contextual signals needed to accurately determine intent.

Finally, these insights suggest that Lakhina Volume requires augmentation with behavioral analysis and trust modeling components from the full CAMNEP framework.

8. Conclusion

Throughout this project, we explored the challenges of research analysis and result replication. Replicating results is a vital aspect of academic research. Although most models are designed with reproducibility in mind, achieving the same results can often be difficult or even impossible due to various factors, including implementation differences, data preprocessing variations, and hidden dependencies.

In this work, we investigated an anomaly detection method known as Lakhina Volume, which is a component of a larger framework called CAMNEP. This model integrates additional functionalities beyond the anomaly detector, which together form the complete CAMNEP framework used to generate the results presented in the original paper. However, replicating only the anomaly detection algorithm without implementing the entire model makes direct comparisons with the reported evaluation metrics challenging.

To address this limitation, future work should focus on implementing the two subsequent components of the CAMNEP model : the Trust Models and the Adaptation Models, as illustrated in the figure 7. Integrating these additional components would allow for a more accurate reproduction of the results and a deeper analysis of the model's overall performance.

Another crucial aspect to consider is the redefinition of classical evaluation metrics. As shown in Figure 8, the conventional metrics are adapted to account for time-dependent factors, where errors are computed over specific time frames. These modifications must be carefully considered when making direct comparisons between the results obtained in this study and those presented in the original paper.

Another point to consider are the redefinitions of the classical evaluation metrics, as shown in 8, the metrics usually used are modified to take into account the time to the metrics by computing the errors in comparison time frames, this with other modifications make necessary to take into account this modifications if we want to make direct comparisons with the results obtained in this work.

Finally we have included at [1], the github with the Jupyter notebook with the algorithms and results obtained.

FIGURE 7 – CAMNEP Model

- c_TP: A True Positive is accounted when a Botnet IP address is detected as Botnet at least **once** during the comparison time frame.
- c_TN: A True Negative is accounted when a Normal IP address is detected as Non-Botnet during the **whole** comparison time frame.
- c_FP: A False Positive is accounted when a Normal IP address is detected as Botnet at least **once** during the comparison time frame.
- c_FN: A False Negative is accounted when a Botnet IP address is detected as Non-Botnet during the **whole** comparison time frame.

FIGURE 8 – New evaluation metrics

References

[1] https://github.com/isauredheb/securityaudit.

[2] S. García, M. Grill, J. Stiborek, and A. Zunino. An empirical comparison of botnet detection methods. *Computers Security*, 45 :100–123, 2014.

[3] Anukool Lakhina, Mark Crovella, and Christophe Diot. Diagnosing network-wide traffic anomalies. *SIGCOMM Comput. Commun. Rev.*, 34(4) :219–230, August 2004.