2023 15th International Conference on Cyber Conflict
Meeting Reality
T. Jančárková, D. Giovannelli, K. Podiņš, I. Winther
2023 © NATO CCDCOE Publications, Tallinn

Towards Generalizing Machine Learning Models to Detect Command and Control Attack Traffic

Lina Gehri
ETH Zurich
Department of Electrical Engineering and Information Technology
Zurich, Switzerland
lina.gehri@gmail.com

Roland Meier
Cyber-Defence Campus
armasuisse Science and Technology
Thun, Switzerland
roland.meier@ar.admin.ch

Daniel Hulliger
Cyber-Defence Campus
armasuisse Science and Technology
Thun, Switzerland
daniel.hulliger@ar.admin.ch

Vincent Lenders
Cyber-Defence Campus
armasuisse Science and Technology
Thun, Switzerland
vincent.lenders@ar.admin.ch

Abstract: Identifying compromised hosts from network traffic traces has become challenging because benign and malicious traffic is encrypted, and both use the same protocols and ports. Machine learning-based anomaly detection models have been proposed to address this challenge by classifying malicious traffic based on network flow features learned from historical patterns. Previous work has shown that such models successfully identify compromised hosts in the same network environment in which they were trained. However, cyber incidence response teams often have to look for intrusions in foreign networks, and we have found that learned models often fail to generalize to different network conditions. In this paper, we analyse the root cause of this problem using five network traces collected from different years and teams of Locked Shields, the world's largest live-fire cyber defence exercise. We then explore techniques to make machine learning models generalize better to unknown network environments and evaluate their accuracy.

Keywords: *machine learning, traffic classification, network security, command and control, Locked Shields*

1. INTRODUCTION

Despite many years of active research, detecting malicious communications from infected hosts in a network remains a challenge. Over the years, attackers have adapted their communication patterns to mimic the protocols and ports of benign traffic, making it difficult to differentiate them in deployed network intrusion detection systems [1]. At the same time, with the wide adoption of HTTPS, network traffic is almost entirely encrypted by default [2], and it is no longer possible for intrusion detection systems such as Suricata, Snort, Bro, or Zeek to analyse the contents in order to look for malicious signatures in the packets' payloads.

As a response, researchers have proposed anomaly detection techniques that use machine learning to identify malicious traffic based on network flow features (cf. surveys in [3,4]). These techniques do not require inspecting the packets' payloads and are thus well-suited for encrypted traffic. However, a major challenge is that available labelled datasets for training are scarce, especially those originating from real environments, because they contain information that the affected organizations do not want to share. Moreover, labelling traffic from real attacks is often impossible due to the lack of ground truth.

One solution to this problem is to use unsupervised learning techniques such as clustering. However, these solutions do not perform well on nonconvex data and are sensitive to initialization and clustering parameters [3]. Another approach is to share machine learning models across networks and use models trained in one environment in order to detect malicious activity in other environments. In this paper, we analyse the feasibility of this approach using five real-world datasets collected from Locked Shields, the largest cyber defence live-fire exercise in the world [5].

First, we analyse the detection performance of command and control (C2) attack flows using machine learning models trained and tested in different environments. We find that models that may work well for a particular environment typically fail to generalize to multiple environments. Second, we investigate the root cause for this effect by analysing which model features work best under which conditions. Then, we explore flow-based and host-based models that generalize to different environments. Our results show that it is possible to train generalized models by carefully selecting time-independent features that are not significantly affected by the environment. However, they also show that training such models is not trivial, and the models generally fail to achieve the same performance as those trained and tested in the same environment.

Our main contribution is the comparison and analysis of supervised machine learning models in five realistic network datasets that include millions of real attack flows generated by security professionals over the course of multiple days at different instances of the Locked Shields cyber defence exercises. Our work presents novel experimentation with new insights made possible using a systematic analysis of these datasets.

2. RELATED WORK

There have been many attempts to exploit machine learning methods for network intrusion detection systems (IDSs), and we refer here to surveys that summarize these techniques. Ahmad et al. [6] and Liu and Lang [3] compare machine learning methods for network-based IDSs and review recent papers on this topic. Khraisat et al. [7] review papers about various kinds of IDSs, and Da Costa et al. [8] concentrate on Internet of Things-related detection. Lastly, Lashkari et al. [9] outline botnet detection methods, including some machine learning-based methods, using various data sources.

Khraisat et al. [10] use the NSL-KDD dataset [11] to compare the classifiers C5, C4.5, SVM, and Naive Bayes. They find that the C5 classifier performs best, with an accuracy of 99.82% and few false positives. Alqahtani et al. [12] compare seven ML-based classification techniques for IDS development using the KDD'99 cup dataset [13]. They find that the random forest model performs best, with an accuracy of 94% and the highest precision and recall score. Jabbar et al. [14] combine a random forest classifier with an average one-dependence estimator to classify traffic. They use the Kyoto dataset [15], and the combined model achieves an accuracy of 90.51% with a false alarm rate of 0.14%.

In this paper, we focus on random forest models trained on Locked Shields datasets to detect malicious flows and hosts. The concept of cyber defence exercises such as Locked Shields is described in [16], and Max Smeets reviews the development and evaluates the achievements of Locked Shields until 2022 in [17]. Our work builds on the work by Känzig et al. [18], which also uses data from Locked Shields to train and test machine learning methods. While their models were developed and tested for only two years of the same team, we generalize the trained classifiers to different years and teams of Locked Shields.

Similar to our work, the authors of [19] and [20] investigate whether it is possible to circumvent detectors of C2 traffic. However, their focus is on the modification techniques that allow circumventing detectors, not on the impact of different environments.

3. LOCKED SHIELDS DATASETS

This section introduces the datasets used in this paper. Locked Shields is a live-fire cyber defence exercise based on realistic scenarios. It is organized once a year by the NATO Cooperative Cyber Defence Centre of Excellence (CCDCOE) [21]. The scenarios involve a cyber incident affecting a fictional country.

Each member nation of the CCDCOE can participate as a Blue Team that assumes the role of the defenders. Blue Teams are typically between 20 and 100 persons, with an average of 40 persons in 2021. The Blue Teams are challenged by a Red Team consisting of professional penetration testers and hackers. The Red Team's goal is to compromise the Blue Team's systems. Attacks include defacing websites, stealing data, denial of service, and compromising hosts by executing malicious payloads [22]. The Red Team uses standard exploitation tools such as Kali Linux [23], Metasploit [24], and Cobalt Strike [25]. The latter is used as the default C2 tool. Custom attacks can be launched if necessary. The whole exercise takes place in Gamenet, which consists of more than 5,000 virtual systems. Every Blue Team is responsible for protecting more than 150 systems over a period of three days. These systems include Linux and Windows machines as well as firewalls, routers, 5G services, drones, industrial control systems, and other systems. To create realistic traffic in the network of the Blue Teams, other teams act as users and use the Blue Teams' services during the whole exercise [22].

We have collected the Locked Shields network traffic of two countries (Country A and Country B) for different years in the form of PCAP files. The network traffic of Country A's Blue Team is from 2017, 2018, 2019, and 2021,[1] and the traffic of Country B's Blue Team is from 2021, resulting in five datasets, as shown in Table I. All datasets are highly imbalanced and heavily skewed towards normal traffic, especially the dataset for 2019, which includes only 0.006% (about 4,000) malicious flows.

In addition, we have auxiliary Red Team activity reports that allow us to label the malicious C2 flows from these PCAP files.

1 Locked Shields 2020 was cancelled due to COVID-19.

TABLE I: OVERVIEW OF LOCKED SHIELDS DATASETS

Dataset	Size PCAP files	Size CSV files	Number of flows	% malicious
LS17	109 GB	7.1 GB	14,094,546	10.7%
LS18	207 GB	10.7 GB	20,925,882	8.7%
LS19	1.4 TB	34.4 GB	62,955,546	0.006%
LS21A	1.7 TB	24.9 GB	51,699,619	0.5%
LS21B	1.1 TB	19.0 GB	39,903,036	1.1%

4. CHALLENGES OF TRANSFERRING MODELS TO DIFFERENT DATASETS

In this section, we analyse how well a machine learning model trained on Locked Shields data from one year/team performs in detecting malicious flows from another year/team. As a baseline, we consider the machine learning pipeline developed by Känzig et al. in [18].

Model Training

For reasons of space, we analyse only the performance of the best-performing machine learning model developed in [18]. It is a random forest model where the maximum tree depth is 10, and the number of trees is 128. The model was trained with the best 20 features, including custom features and per-flow features extracted using a modified version of CICFlowMeter [26]. (We describe the modifications of CICFlowMeter in Appendix A.) The extracted feature set includes time-based features, such as interarrival times between packets, including average, maximum, minimum, and standard deviation values, as well as time-independent features, such as the number of packets. The best 20 features were selected using a recursive feature elimination algorithm applied to the LS17 dataset. A complete list of the features is provided in [27], and we include a list of the 20 most important features in Appendix B. A flow is defined by its quintuple; it is bidirectional, and the first packet defines the direction.

We trained four separate models on a subsample of 7,000,000 flow instances from the datasets of Country A. To subsample, we randomly sample malicious and normal flows with a ratio of malicious flows as close to 10% as possible. To label the malicious C2 flows, we extract a list of malicious IPs with the help of the Cobalt Strike attack reports from the Red Teams, as suggested by Känzig et al. [18]. Then, we use this list

to label the flows extracted by CICFlowMeter. If the source or destination IP is in the list, the flow is labelled as malicious.

Evaluation

We evaluate the models using a fivefold cross-validation of each model to predict the flow labels for all datasets and compare the F1 score of predictions to the labels. The F1 score is calculated as follows:

$$F1 = 2\times \frac{precision \times recall}{precision + recall}$$

With $precision = \frac{TP}{TP + FP}$ and $recall = \frac{TP}{TP + FN}$

where TP, FN, and FP correspond respectively to the true positive, false negative, and false positive rates.

The results are shown in Table II. Training and testing interchangeably on the 2017 and 2018 datasets gave good results, in line with what Känzig et al. obtained. However, the LS17 and LS18 models are bad at classifying flows for Country A's 2019 and 2021 data. Generally, the 2019 dataset performed worst when used for training or testing. Finally, none of the models trained on Country A's data performed well on Country B's data.

TABLE II: F1 SCORES FOR DETECTING C2 MALICIOUS FLOWS

Test data / Training data	LS17	LS18	LS19	LS21A	LS21B
LS17	0.993	0.966	0.007	0.856	0.215
LS18	0.945	0.993	0.060	0.806	0.167
LS19	0.743	0.928	0.791	0.351	0.000
LS21A	0.952	0.918	0.038	0.986	0.158

Possible explanations for the bad performances include the fact that the features were selected using 2017 data only and might not be as relevant for the other years. In addition, the Locked Shields network infrastructure looks different each year, meaning time-dependent features can vary, leading to wrong classifications. Finally, the bad performance of the 2019 model might be because there are only about 4,000 malicious flows, amounting to only a few malicious training instances.

5. CROSS-DATASET FEATURE ANALYSIS AND RANKING

One way to enhance the badly transferable models from Section 4 is by selecting a more suitable set of features. In this section, we use feature elimination and ranking methods on Country A's datasets to select features that generalize better to the different datasets.

Feature Elimination

First, we eliminate irrelevant features across all datasets using the methods described below. A complete list of the eliminated features can be found in Appendix C.

Constant features: We remove the features that are constant over all datasets as they provide no information about whether a flow is malicious or normal.

Feature correlation: Any two highly correlated features contain approximately the same information about the label, and dropping one does not erase information. To remove only features that are inherently correlated and not just because the different gamenets are similar, we include the CIC-IDS2017 dataset [28] in the analysis.

We proceed as follows: First, we calculate the sample Pearson correlation coefficient r for each feature pair of each dataset of Country A and the CIC-IDS2017 dataset and take its absolute value. Then, we find the feature pairs with $|r| > 0.9$ for all datasets. Finally, for each pair, we discard the feature with the lowest relative mutual information (RMI) value with the label.

Relative mutual information: Next, we eliminate all features with less than 15% RMI in all datasets of Country A. The mutual information (MI) between two random variables X and Y measures how much information X contains about Y [29]. We use the sklearn.feature_selection.mutual_info_classif function [30] to calculate the MI between each feature and the discrete label for each dataset. If the feature is also discrete, the function uses the frequencies of the values x and y and the value pairs (x, y) to estimate the probability mass functions. If the feature is continuous, it estimates the MI from k-nearest neighbour statistics, according to [31]. The RMI corresponds to the percentage of uncertainty removed from X when Y is known. It is calculated by dividing the MI by X's entropy:

$$\mathrm{RMI(X;Y)} = \mathrm{MI(X;Y)/H(X)}$$

We calculate the RMI by dividing each MI score by the entropy of the corresponding year's labels. Finally, we remove the features with less than 15% RMI for all datasets.

Feature Ranking and Selection

Next, we rank the features according to their importance using recursive feature elimination (RFE) and single-feature cross-validation (SF-CV) for each of Country A's datasets.

Feature Ranking with Recursive Feature Elimination

We use RFE to rank the features for each year. We employ the default scikit-learn RFE function [32] on the subsampled LS datasets with 7,000,000 instances. RFE eliminates features one by one. The average ranks of each year's top 10 features can be found in Table III, which shows that features ranked very highly for one year need not rank highly in all the other years. Still, some features are highly ranked for all years; for example, numbers 1 through 10 are, with three exceptions, all ranked in the top 20 for all years. We use these average RFE ranks to choose the features for our random forest models.

TABLE III: RANKS OF THE TOP 10 FEATURES FOR EACH DATASET ACCORDING TO RFE, SORTED BY AVERAGE RANK NUMBER

	No	LS17	LS18	LS19	LS21A
Fwd Pkt Len Max	1	2	4	2	4
Bwd Pkt Len Std	2	5	7	5	6
TotLen Fwd Pkts	3	1	2	16	9
Bwd Pkt Len Max	4	20	6	3	7
Pkt Len Max	5	12	16	10	5
TotLen Bwd Pkts	6	24	3	1	15
Fwd Pkt Len Std	7	3	5	22	16
Pkt Len Mean	8	13	17	11	10
Bwd Pkt Len Mean	9	4	30	4	19
Fwd IAT Max	10	10	12	18	17

Feature Ranking with Single-Feature Cross-Validation

To get a clearer picture of how decisive each feature is, we use SF-CV F1 scores. To compute the scores, we use fivefold cross-validation on an RF model that uses only a

single feature and a 7,000,000-instance subsample of a dataset. The average F1 scores over all folds can be found in Table IV for the top 10 features, sorted by average score. For 2017, 2018, and 2021, there are many features with a score higher than 0.9, while for 2019, there is not a single one. Also, only two of the 20 features have a score over 0.1 for 2019.

This could explain the poor performance when using the 2019 data in Section 4.2.

TABLE IV: F1 SCORES OF THE TOP 10 FEATURES FOR EACH DATASET ACCORDING TO SF-CV, SORTED BY AVERAGE SCORE

	No	LS17	LS18	LS19	LS21A	Avg.
Bwd Pkt Len Std	1	0.98	0.99	0.73	0.95	0.91
Pkt Len Var	2	0.98	0.99	0.69	0.96	0.91
Bwd Seg Size Avg	3	0.97	0.99	0.69	0.94	0.90
Bwd Pkt Len Mean	4	0.97	0.99	0.69	0.94	0.90
TotLen Fwd Pkts	5	0.97	0.99	0.66	0.93	0.89
Pkt Len Max	6	0.97	0.98	0.57	0.96	0.87
Fwd Pkt Len Max	7	0.97	0.98	0.57	0.96	0.87
Bwd Pkt Len Max	8	0.98	0.99	0.56	0.94	0.87
Fwd Pkt Len Mean	9	0.97	0.98	0.60	0.89	0.86
Fwd Seg Size Avg	10	0.97	0.98	0.60	0.89	0.86

Time-Independent Features

The last feature selection method is to consider time-independent features only. This includes any features directly influenced by bandwidth changes or packet loss, such as packet interarrival times or byte rates. We expect that these features are most affected by network environment changes. In Appendix D, we provide a ranking of these features using RFE. These time-independent features also have dependencies on network conditions and time, but these are less direct than for time-dependent features.

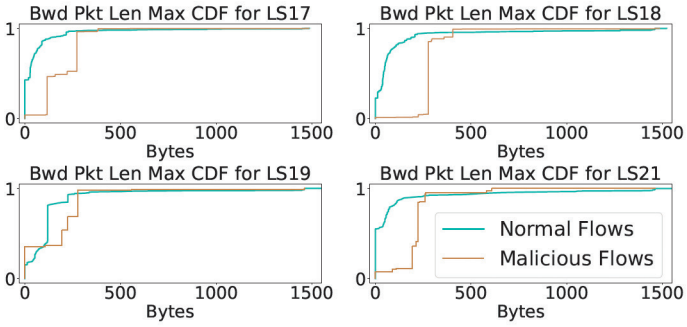
Packet Length-Related Features

In both rankings, all of the top 9 features are related to the packet lengths in a flow. Hence, we analyse one of these features in more depth.

First, we plot the value distributions of Bwd Pkt Len Max representatively for all packet length features as CDFs in Figure 1. Bwd Pkt Len Max is the transport layer payload size of the largest packet in the backward direction. We can see why this feature leads to good performance for most Locked Shields years. The values for malicious flows are clearly higher than the values for normal flows, and many malicious flows have the same feature value. This also explains why the feature does not work as well for 2019, as both CDFs have vertical jumps at 0 and at about 250.

Next, we study some packets to find out why the malicious packets are larger and often have the same size. We stress that the subsequent inferences are primarily based on spot checks and, thus, are not necessarily representative of the entire dataset.

FIGURE 1: DISTRIBUTIONS OF BWD PKT LEN MAX VALUES



We start with the 2018 dataset, as it has the largest vertical jump in the malicious CDF. Almost all communication with a malicious IP is over TLS, implying that the Red Team uses HTTPS connections. While we cannot read the content, we can see exchanges repeating every few seconds. This could be an infected host checking in with a team server. Usually, the team server's answer packet size is 277 bytes, which matches the jump in the CDF. This might be the team server's default answer if there are no new commands. The normal flows with Bwd Pkt Len Max 0 seem to be caused by flows with no backward packets and TCP flows consisting only of zero-length flag packets such as SYNs and ACKs. Therefore, in 2018, there appears to be a lot of beaconing over HTTPS without any new commands.

We also look at the 2019 dataset as its value distributions differ most from the other years. Again, there are many presumed beaconings over HTTPS. The typical answer packets are 223 or 277 bytes in length, which corresponds to two of the jumps in the CDF. It could be that the Red Team is using two Malleable C2 profiles. We can also see more malicious HTTP conversations than in 2018, where the largest packets are

194 bytes. Most malicious flows with zero packet length consist of SYN packets in the forward direction without any answer from the team server. We do not know why the servers were unreachable, but this seems to have been a problem especially in 2019.

6. GENERIC MODELS AND THEIR EVALUATION

In this section, we propose and evaluate two generic model types – a flow-based type and a host-based type – which use a combination of Country A's datasets to select generic features, as explained in Section 5.

Flow-Based Models

Description: The goal of the flow-based models is to detect individual malicious flows. The models are:

- Generic, 10 Feat.: a generic random forest model using RFE to select the best 10 features across all features.
- Generic, 10 t.-i. Feat: a generic random forest model using RFE to select the best 10 time-independent features across all features.
- Generic, 20 Feat.: a generic random forest model using RFE to select the best 20 features across all features.
- Generic, 20 t.-i. Feat: a generic random forest model using RFE to select the best 20 time-independent features across all features.

Evaluation: We evaluate the flow-based models on Country A's datasets and Country B's dataset to assess their transferability. The F1 scores can be found in Table V.

TABLE V: F1 SCORES OF THE GENERIC FLOW-BASED MODELS WITH 10 OR 20 FEATURES (TIME-INDEPENDENT (T.-I.) OR NOT)

Test data	LS17	LS18	LS19	LS21A	LS21B
Generic, 10 Feat.	0.980	0.991	0.426	0.975	0.116
Generic, 10 t.-i. Feat.	0.985	0.992	0.554	0.971	0.162
Generic, 20 Feat.	0.991	0.992	0.621	0.967	0.135
Generic, 20 t.-i. Feat.	0.992	0.993	0.638	0.989	0.185

First, we look at the results of Country A's datasets. The diversity of the training data leads to better and more consistent results than in Section 4.2. While the F1 scores for testing on 2017, 2018, and 2021 data fluctuate by a maximum of 0.06, there are more

significant differences for the 2019 data. The model that achieves the highest overall F1 scores is the generic model using the top 20 time-independent features. The time-independent features also improve the results for the models using only 10 features, suggesting that models can be generalized by ignoring time-dependent features. Using 20 features generally works better than using 10, which is to be expected, as the model has more data points to make a decision. When we inspect the superior results of the models with 20 features in more detail by looking at precision and recall separately (see Tables VI and VII), we can see that recall is above 0.97 for all models and test datasets, even for 2019, indicating that the models detect a considerable percentage of malicious traffic. However, while precision is always 0.93 or higher for all other years, 0.47 is the highest score for 2019. We must remember that the datasets are very imbalanced; precision would be high even if a model classified all flows as normal. This implies that the models classify many normal traffic flows as malicious in the 2019 dataset.

Unfortunately, the scores for Country B's dataset are all below 0.2. When inspecting precision and recall individually (see Tables VI and VII), we can see that neither is high, though the precision scores are similar to those for the 2019 dataset. Again, this means the models classify many normal traffic flows as malicious. At the same time, the deficient recall scores (below 0.2) indicate that the model also fails to classify genuinely malicious traffic. We can partially explain the problem when we consider the value distribution of Bwd Pkt Len Max for Country B's dataset, which shows that there are many malicious flows with a length of 0. In the CDFs of Country A's datasets (Figure 1), barely any of the malicious flows have a length of 0, which probably means they are classified as normal in Country B's data. On top of that, the non-zero malicious flows consist of far greater packets than any flows in Country A's dataset, making it difficult for the model to classify them correctly.

TABLE VI: THE RECALL OF THE GENERIC MODELS USING 20 FEATURES CHOSEN ACCORDING TO THE RFE RANKING, SELECTING ONLY TIME-INDEPENDENT FEATURES (T.-I.) OR SELECTING FROM ALL FEATURES

Test data	LS17	LS18	LS19	LS21A	LS21B
Generic, 20 Feat.	0.985	0.989	0.975	0.988	0.080
Generic, 20 t.-i. Feat.	0.985	0.989	0.978	0.994	0.114

TABLE VII: THE PRECISION OF THE GENERIC MODELS USING 20 FEATURES CHOSEN ACCORDING TO THE RFE RANKING, SELECTING ONLY TIME-INDEPENDENT FEATURES (T.-I.) OR SELECTING FROM ALL FEATURES

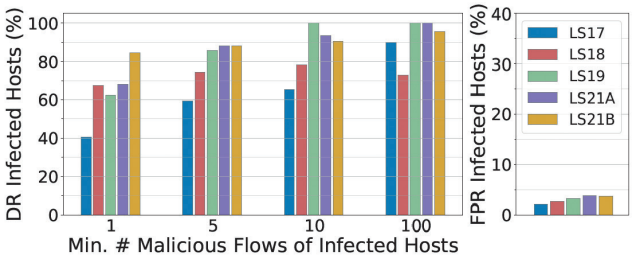
Test data	LS17	LS18	LS19	LS21A	LS21B
Generic, 20 Feat.	0.998	0.994	0.456	0.947	0.420
Generic, 20 t.-i. Feat.	1.000	0.998	0.474	0.985	0.491

Host-Based Models

Description: The purpose of the generic host-based model is to identify infected hosts using the classification of malicious flows. We define infected hosts as IP addresses that are the source IP of at least one labelled malicious flow, and we define detected infected hosts as IP addresses involved in at least $n = \{1, 5, 10, 100\}$ flows predicted as malicious. As the underlying model, we consider the generic model using the top 20 time-independent features, which was the flow-based model with the highest F1 scores in Section 6.1.

Evaluation: We compare the detected infected hosts to the actual infected hosts to calculate the detection rate (DR) and the false positive rate (FPR). We also determine if infected hosts involved in more malicious flows are more accurately detected. The DR and the FPR of the infected hosts for all datasets can be found in Figure 2.

FIGURE 2: DETECTION RATE (DR) AND FALSE POSITIVE RATE (FPR) FOR ALL DATASETS FOR THE GENERIC MODEL WITH 20 TIME-INDEPENDENT FEATURES



Again, we first consider Country A's results. For $n = 1$, the model does not have a very high DR of infected hosts; however, its FPR is below 4% for all years. In absolute numbers, it detected 19 out of 47 infected hosts that were involved in at least one malicious flow in 2017, 33 out of 49 in 2018, 10 out of 16 in 2019, and 15 out of 22 in 2021. The DR improves significantly when we only consider infected

hosts that communicate more often ($n > 1$). It is reasonable to consider the DRs for infected hosts with more than five or ten malicious flows, as such hosts tend to be more precarious for a network. They can siphon out more information, act on more commands, or serve as a pivot for other C2 sessions. There were 26 false alarms for a total of 1,193 non-infected hosts in 2017, 33 for 1,233 normal hosts in 2018, 94 for 2,852 normal hosts in 2019, and 135 for 3,553 normal hosts in 2021.

Next, we look at Country B's results. The performances when testing on Country B's dataset are surprisingly good compared with the F1 scores from Section 6.1. The model detects about 33 of the 39 infected hosts with 119 false alarms out of a total of 3,185 normal hosts. These results are as good as and better than those for Country A's datasets. Interestingly, the detection of infected hosts obtained such good results considering that the F1 scores were quite low for the flow-based models. We suspect that while the Red Team used different methods and commands in Country B's case, the initial connection to the team server had comparable network indicators, leading the model to classify these flows as malicious and hence detecting the infected host despite the network conditions being different.

As a result, we can train supervised models on Country A's datasets that can successfully detect a large portion of the infected hosts in Country A's and Country B's datasets with a relatively low FPR, especially when the hosts communicated with a malicious IP multiple times.

7. CONCLUSION

Developing generic machine-learning models that detect malicious traffic in various network environments is challenging. We analysed the flow classification performance of various random forest models depending on the feature selection, model parameters, and training data. We determined that a mix of training data from different environments leads to models vastly outperforming models trained on only one dataset. These mixed models achieve F1 scores over 0.99 when tested on Locked Shields data from Country A's 2017, 2018, and 2021 datasets and over 0.63 for the 2019 dataset. We identified the time-independent features selected by an RFE ranking over all of Country A's datasets as particularly effective in achieving good classification performances. However, we also saw that achieving high scores in completely unfamiliar environments is an open problem for future research.

Further, we demonstrated that models that sum up the number of malicious flows significantly increase the detection rate in Country A's and Country B's networks. Hosts that communicate with a malicious server more than 100 times have an increased

detection rate of over 90% and FPR below 4%, even for network environments not used for the training.

REFERENCES

[1] S. Wendzel, S. Zander, B. Fechner, and Ch. Herdin, "Pattern-based survey and categorization of network covert channel techniques," *ACM Computer Surveys*, vol. 47(3), pp. 1–26, Article 50, Apr. 2015, doi: 10.1145/2684195.

[2] "HTTPS encryption on the web – Google Transparency Report." Google. 2022. Accessed: Jun. 23, 2022. [Online]. Available: https://transparencyreport.google.com/https/overview

[3] H. Liu and B. Lang, "Machine learning and deep learning methods for intrusion detection systems: A survey," *Applied Sciences*, vol. 9(20), p. 4396, Oct. 2019, doi: 10.3390/APP9204396.

[4] K. Shaukat, S. Luo, V. Varadharajan, I. A. Hameed and M. Xu, "A Survey on Machine Learning Techniques for Cyber Security in the Last Decade," in *IEEE Access*, vol. 8, pp. 222310-222354, 2020, doi: 10.1109/ACCESS.2020.3041951.

[5] "Locked Shields." CCDCOE. 2022. [Online]. Available: https://ccdcoe.org/exercises/locked-shields/

[6] Z. Ahmad, A. S. Khan, Ch. W. Shiang, J. Abdullah, and F. Ahmad, "Network intrusion detection system: A systematic study of machine learning and deep learning approaches," *Transactions on Emerging Telecommunications Technologies*, vol. 32(1), e4150, Jan. 2021, doi: 10.1002/ETT.4150.

[7] A. Khraisat, I. Gondal, P. Vamplew, and J. Kamruzzaman, "Survey of intrusion detection systems: techniques, datasets and challenges," *Cybersecurity*, vol. 2(1), pp. 1–22, Dec. 2019, doi: 10.1186/S42400-019-0038-7.

[8] K. A. P. Da Costa, J. P. Papa, C. O. Lisboa, R. Munoz, V. Hugo, and C. De Albuquerque, "Internet of Things: A survey on machine learning-based intrusion detection approaches," *Computer Networks*, vol. 151, pp. 147–157, 2019, doi: 10.1016/j.comnet.2019.01.023.

[9] A. H. Lashkari, G. D. Gil, J. E. Keenan, K. F. Mbah, and A. A. Ghorbani, "A survey leading to a new evaluation framework for networkbased botnet detection," in *Proceedings of the 2017 the 7th International Conference on Communication and Network Security*, 2017, pp. 59–66, doi: 10.1145/3163058.3163059.

[10] A. Khraisat, I. Gondal, and P. Vamplew, "An anomaly intrusion detection system using C5 decision tree classifier," *Lecture Notes in Computer Science*, LNAI vol. 11154, pp. 149–155, Nov. 2018, doi: 10.1007/978-3-030-04503-6_14.

[11] M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, "A detailed analysis of the KDD CUP 99 data set," *IEEE Symposium on Computational Intelligence for Security and Defense Applications, CISDA 2009*, Dec. 2009, doi: 10.1109/CISDA.2009.5356528.

[12] H. Alqahtani, I. H. Sarker, A. Kalim, S. M. M. Hossain, S. Ikhlaq, and S. Hossain, "Cyber intrusion detection using machine learning classification techniques," *Communications in Computer and Information Science*, vol. 1235 CCIS, pp. 121–131, Mar. 2020, doi: 10.1007/978-981-15-6648-6_10.

[13] "KDD Cup 1999 Data." UCI. 1999. Accessed: Mar. 10, 2021. [Online]. Available: http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html

[14] M. A. Jabbar, R. Aluvalu, and S. S. Reddy, "RFAODE: A novel ensemble intrusion detection system," *Procedia Computer Science*, vol.115, pp. 226–234, Jan. 2017, doi: 10.1016/j.procs.2017.09.129.

[15] Kyoto University. "Traffic Data from Kyoto University's Honeypots." Takakura. 2015. Accessed: Mar. 10, 2021. [Online]. Available: http://www.takakura.com/Kyoto_data/

[16] E. Seker and H. H. Ozbenli, "Concept of cyber defence exercises (CDX): Planning, execution, evaluation," in *2018 International Conference on Cyber Security and Protection of Digital Services (Cyber Security)*, 2018, pp. 1–9, doi: 10.1109/CyberSecPODS.2018.8560673.

[17] M. Smeets, "The role of military cyber exercises: A case study of Locked Shields," *2022 14th International Conference on Cyber Conflict: Keep Moving! (CyCon)*, Tallinn, Estonia, 2022, pp. 9–25, doi: 10.23919/CyCon55549.2022.9811018.

[18] N. Känzig, R. Meier, L. Gambazzi, V. Lenders, and L. Vanbever, "Machine learning-based detection of C&C channels with a focus on the Locked Shields cyber defense exercise," *2019 11th International Conference on Cyber Conflict (CyCon)*, Tallinn, Estonia, 2019, pp. 1–19, doi: 10.23919/CYCON.2019.8756814.

[19] C. Novo and R. Morla, "Flow-based detection and proxy-based evasion of encrypted malware C2 traffic," in *Proceedings of the 13th ACM Workshop on Artificial Intelligence and Security*, Nov. 2020, pp. 83–91, doi: 10.1145/3411508.3421379.

[20] G. Xavier, C. Novo, and R. Morla, Tweaking Metasploit to Evade Encrypted C2 Traffic Detection, 2022, *arXiv:2209.00943*.
[21] "NATO Cooperative Cyber Defence Centre of Excellence." CCDCOE. 2022. [Online]. Available: https://ccdcoe.org/
[22] "Cyber Defence Exercise Locked Shields 2013 After Action Report," CCDCOE, Tallinn, Estonia, 2013. [Online]. Available: https://ccdcoe.org/uploads/2018/10/LockedShields13_ AAR.pdf
[23] OffSec Services. "Kali Linux." Kali.org. 2022. [Online]. Available: https://www.kali.org/
[24] rapid7. "Metasploit Framework." Github.com. 2022. [Online]. Available: https://github.com/rapid7/metasploit-framework
[25] HelpSystems. "Cobalt Strike." Cobaltstrike.com. 2022. [Online]. Available: https://www.cobaltstrike.com/
[26] A. H. Lashkari. "CICFlowMeter." Github.com. 2022. [Online]. Available: https://github.com/ahlashkari/CICFlowMeter
[27] "CICFowMeter Features." Github.com. [Online]. Available: https://github.com/ahlashkari/CICFlowMeter/blob/master/ReadMe.txt
[28] Canadian Institute for Cybersecurity. "Intrusion Detection Evaluation Dataset (CIC-IDS2017)." UNB. 2017. Accessed: Jun. 5, 2021. [Online]. Available: https://www.unb.ca/cic/datasets/ids2017.html
[29] T. E. Duncan, "On the calculation of mutual information," *SIAM Journal on Applied Mathetmatics*, vol. 19(1), pp. 215–220, 1970, doi: 10.1137/0119020.
[30] "sklearn.feature_selection.mutual_info_classif." Scikit-learn.org. [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.feature_ selection.mutual_info_classif.html
[31] Brian C. Ross. 2014. "Mutual Information between Discrete and Continuous Data Sets." *PLOS ONE*, vol. 9(2), e87357, Feb. 2014, doi: 10.1371/JOURNAL.PONE.0087357.
[32] "sklearn.feature_selection.RFE – scikit-learn 1.1.1 documentation." Scikit-learn.org. [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.feature_selection.RFE.html

A. CICFlowMeter Tool Modifications

The CICFlowmeter tool[2] extracts flows from PCAP files and exports each flow as a CSV file entry with the 76 CICFlowMeter features and additional metadata, namely source and destination IPs and MAC addresses, the protocol number, and a flow timestamp. The modifications to this tool for this work are:

- preventing memory overflows when processing big PCAPs by regularly flushing to the CSV file;
- adding a new feature Dst IntExt, which has the value 0 if the destination IP address is inside the internal network and 1 if it is outside;
- adding a time filter for PCAPs, making it possible to only process PCAPs from a directory inside a certain time window;
- filtering out flows with TCP SYN count 0, which are flows created by a suboptimal TCP flow tracking logic.

2 https://github.com/ahlashkari/CICFlowMeter.

B. Top 20 Most Important Features

TABLE VIII: TOP 20 CICFLOWMETER FEATURES FROM [18]

No	Feature
1	Protocol
2	Dst IntExt
3	Flow IAT Max
4	Fwd IAT Tot
5	Subflow Bwd Pkts
6	Subflow Fwd Byts
7	Bwd Header Len
8	Tot Bwd Pkts
9	Fwd Pkt Len Std
10	Fwd Seg Size Min
11	Bwd Pkt Len Std
12	Bwd IAT Mean
13	Active Mean
14	Init Fwd Win Byts
15	FIN Flag Cnt
16	Bwd Pkt Len Min
17	Flow Pkts/s
18	Fwd IAT Max
19	Flow IAT Mean
20	Subflow Fwd Pkts

C. Eliminated Features

TABLE IX: ELIMINATED FEATURES

Constant	High correlation	Low RMI
Bwd PSH Flags	Active Mean	Active Min
Fwd URG Flags	Active Max	Active Std
Bwd URG Flags	Pkt Size Avg	Bwd Blk Rate Avg
URG Flag Cnt	Bwd Byts/b Avg	Idle Mean
	Idle Max	Idle Std
	Fwd Pkts/s	RST Flag Cnt
	Pkt Len Std	Subflow Bwd Byts
	Tot Bwd Pkts	Subflow Fwd Byts

D. Ranking of Time-Independent Features

TABLE X: TIME-INDEPENDENT FEATURES RANKED WITH RFE, SORTED BY AVERAGE RANK

Feature	No	LS17	LS18	LS19	LS21A
Pkt Len Max	1	8	8	2	5
Init Fwd Win Byts	2	1	18	4	1
Fwd Pkt Len Max	3	7	10	9	4
Bwd Pkt Len Std	4	4	17	8	6
Pkt Len Var	5	2	11	17	7
Bwd Pkt Len Max	6	18	14	1	8
Fwd Pkt Len Std	7	3	13	20	10
Pkt Len Mean	8	13	5	15	13
Bwd Header Len	9	9	4	12	23
Init Bwd Win Byts	10	10	19	7	12
TotLen Fwd Pkts	11	12	7	21	9
Bwd Seg Size Avg	12	6	20	11	15

PSH Flag Cnt	13	14	3	25	11
ACK Flag Cnt	14	17	2	19	16
Fwd Header Len	15	22	9	3	21
TotLen Bwd Pkts	16	21	16	6	14
Bwd Pkt Len Mean	17	5	23	13	18
Fwd PSH Flags	18	19	1	22	19
Fwd Seg Size Min	19	20	25	5	17
Fwd Seg Size Avg	20	11	24	14	22
Tot Fwd Pkts	21	24	6	16	26
Fwd Pkt Len Mean	22	16	21	18	24
SYN Flag Cnt	23	25	26	10	20
Down/Up Ratio	24	15	15	26	30
CWR Flag Count	25	29	30	30	2
ECE Flag Cnt	26	27	31	31	3
Fwd Act Data Pkts	27	26	12	28	27
FIN Flag Cnt	28	23	22	27	28
Subflow Fwd Pkts	29	28	28	23	25
Subflow Bwd Pkts	30	32	29	29	29
Pkt Len Min	31	31	33	32	32
Fwd Pkt Len Min	32	34	32	33	31
Bwd Pkt Len Min	33	33	34	34	33