2ND WORKSHOP
"MACHINE LEARNING & NETWORKING" (MaLeNe)
PROCEEDINGS

SEPTEMBER 4, 2023

CO-LOCATED WITH
THE 5TH INTERNATIONAL CONFERENCE ON NETWORKED SYSTEMS (NETSYS 2023)
POTSDAM, GERMANY

Reducing Memory Footprints in Purity Estimations of Volumetric DDoS Traffic Aggregates

Hauke Heseding†, Timon Krack*, Martina Zitterbart†
†*Institute of Telematics,* †*KASTEL Security Research Labs*
Karlsruhe Institute of Technology (KIT), Karlsruhe, Germany
hauke.heseding@kit.edu, timon.krack@student.kit.edu, zitterbart@kit.edu

***Abstract*—Distinguishing between attack and legitimate traffic in volumetric DDoS scenarios is challenging. Hierarchical Heavy Hitter algorithms can efficiently monitor high-volume traffic aggregates, but provide no insight into traffic composition. Monitoring complementary traffic features enables classification of traffic aggregates with machine learning, but increases the memory footprint of Hierarchical Heavy Hitter algorithms. Since the performance of these algorithms depends on the efficiency of memory usage, we evaluate feature importance to find a compact feature set for accurate distinction of legitimate and attack traffic.**

***Index Terms*—Distributed denial of service, hierarchical heavy hitters, machine learning, feature importance**

I. Introduction

Hierarchical Heavy Hitter (HHH) [1] algorithms can process high-volume traffic in volumetric DDoS attack scenarios at line speed (e.g., [2], [3]). These algorithms monitor traffic volume distribution, but not traffic composition. To make HHH algorithms useful for attack traffic removal, additional information is required to distinguish between attack and legitimate traffic. By monitoring complementary features (besides traffic volume), machine learning (ML) can be used to estimate the purity of attack traffic in a traffic aggregate and to blacklist subnets that primarily send attack traffic. However, using complementary features increases the memory footprint of HHH algorithms, which impedes monitoring efficiency.

To reduce the number of required features, we evaluate the impact of individual features on attack traffic purity estimations. For this, we utilize permutation feature importance [4] to measure the increase in the absolute estimation error after shuffling the values of a single feature randomly. Our results based on authentic MAWI traffic and synthesized attack patterns indicate that only few features are required to achieve high accuracy in attack traffic purity estimations.

II. Attack Traffic Purity Estimation

HHH algorithms perform traffic volume aggregation to detect high-volume IP prefixes, i.e., prefixes whose traffic comprises a certain fraction ϕ of the total traffic (not including the volume of longer high-volume prefixes). Estimating the attack traffic purity of such high-volume prefixes enables blacklisting of highly malicious IP source subnets to protect network infrastructures from attack traffic.

*Research performed while student at KIT.

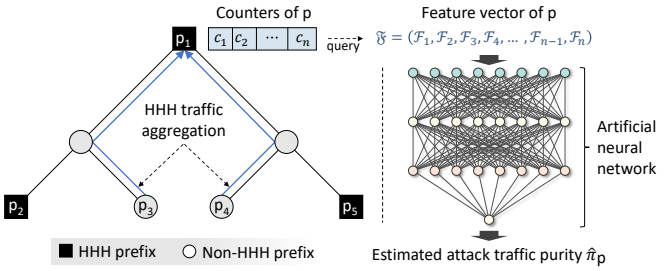

Fig. 1: ML-based regression to estimate attack traffic purity of aggregates.

Specifically, we denote the *attack traffic purity* of a prefix p as the ratio $\pi_\mathsf{p} = A_\mathsf{p}/V_\mathsf{p}$ of attack traffic volume A_p to total traffic volume V_p. To obtain an estimate $\widehat{\pi}_\mathsf{p}$ of π_p, ML-based regression can be applied to complementary features as depicted in Fig. 1. To obtain these features, we modify an HHH algorithm to count how often packets in the source address range of a high-volume prefix exhibit certain characteristics. For example, the counter c_1 of prefix p_1 can count the number of times the TCP protocol occurred in its address range. By continually monitoring different traffic characteristics, the HHH algorithm provides insight into aggregate composition.

The counters can be queried at regular intervals to obtain a feature vector $\mathfrak{F}$ with features $\mathcal{F}_1, \mathcal{F}_2, \ldots$. Given a set of feature vectors and target values of π_p, an artificial neural network can be trained to estimate attack traffic purity. For this, we use a straightforward model architecture (implemented in TensorFlow). The model applies z-score normalization (fitted on training data) in the first layer followed by an alternating sequence of eight fully-connected layers (128 neurons and ReLU activation) and eight dropout layers that reduce the risk of over-fitting. A final layer outputs the attack traffic purity estimation $\widehat{\pi}_\mathsf{p}$ using a single neuron with a linear activation function. The model is trained on dynamic traffic patterns described in Sec. IV that model multiple volumetric DDoS attack vectors overlaid on authentic, legitimate traffic.

III. Feature Importance

To assess the impact of individual features on model performance, we use permutation feature importance on the features and vectors in Tab. I. We then shorten the vectors to reduce memory footprints and retrain the models for comparison.

TABLE I: Features and Feature Vectors

Feature	Meaning
$\mathcal{F}_{\text{PKT}}$	Aggregate packet count
$\mathcal{F}_{\text{VOL}}$	Aggregate traffic volume in bytes
$\mathcal{F}_{\text{TCP}}$, $\mathcal{F}_{\text{UDP}}$	#occurrences of TCP and UDP protocols
$\mathcal{F}_{\text{PORTS}\in[X,Y]}$	#occurrences of source ports in the range $[X, Y]$
$\mathcal{F}_{\text{SIZE}\in[X,Y]}$	#occurrences of frame sizes in the range $[X, Y]$
Feature vectors	**Included features**
$\mathfrak{F}^{\text{PROTO}}$	$\{\mathcal{F}_{\text{TCP}}, \mathcal{F}_{\text{UDP}}\}$
$\mathfrak{F}^{\text{PROTO+SIZE}}$	$\mathfrak{F}^{\text{PROTO}} \cup \{\mathcal{F}_{\text{SIZE}\in[0,199]}, \mathcal{F}_{\text{SIZE}\in[200,1999]}\}$
$\mathfrak{F}^{\text{PROTO+PORTS}}$	$\mathfrak{F}^{\text{PROTO}} \cup \{\mathcal{F}_{\text{PORTS}\in[0,2^{10}-1]}, \mathcal{F}_{\text{PORTS}\in[2^{10},2^{16}-1]}\}$
$\mathfrak{F}^{\text{ALL}}$	$\mathfrak{F}^{\text{PROTO}} \cup \{\mathcal{F}_{\text{PKT}}, \mathcal{F}_{\text{VOL}}\}$ $\cup \{\mathcal{F}_{\text{PORTS}\in[i\cdot 2^{10},(i+1)\cdot 2^{10}-1]} \mid i \in [0, 63]\}$ $\cup \{\mathcal{F}_{\text{SIZE}\in[i\cdot 200,(i+1)\cdot 200-1]} \mid i \in [0, 9]\}$

Initially, the vector $\mathfrak{F}^{\text{all}}$ uses all features and serves as a baseline. To find its important features, we perform two-steps:

Feature importance assessment. First, for a given feature vector, we calculate the cumulative distribution function (CDF) of the absolute errors $|\pi_{\mathsf{p}} - \widehat{\pi}_{\mathsf{p}}|$ of a trained model. We then calculate the area under the curve (AUC) of the CDF. A higher AUC implies better estimations. This can be used to assess permutation feature importance. For this, we shuffle each feature in a test dataset individually and perform attack traffic purity estimations with a trained model. A reduction in the AUC indicates the importance of a feature to the model.

Feature reduction. Second, we eliminate or combine features of $\mathfrak{F}^{\text{ALL}}$ that yield low AUC reductions (e.g., by combining short port ranges). This yields the vectors $\mathfrak{F}^{\text{PROTO+SIZE}}$, $\mathfrak{F}^{\text{PROTO+PORTS}}$, and $\mathfrak{F}^{\text{PROTO}}$ from Tab. I that focus on protocol, frame size, and/or port information. These vectors are significantly shorter to reduce the number of counters required by the HHH algorithm. By measuring the AUC after re-training on shorter vectors, we identify relevant complementary features.

IV. Evaluation

We train the model on a synthesized dataset with authentic, legitimate MAWI traffic [5] and synthesized attack patterns with randomized time-dynamic behavior:

- UDP-based DNS, NTP and OpenVPN amplification attacks with frame size distributions reported in [6].
- TCP-based flood attacks with random frame sizes and ports in the range $60 - 1492$ and $49152 - 65535$ (resp.).

The DNS and NTP attacks use high-volume sources, while OpenVPN and TCP attacks use wide-spread, low-volume sources to generate different aggregates with varying traffic volume. Randomizing sources and attack traffic characteristics renders the estimation of π_{p} challenging.

The feature importance of the most relevant features for each feature vector is shown in Fig. 2. Protocol information (particularly TCP) has the highest impact on model performance. Based in the feature importance of $\mathfrak{F}^{\text{ALL}}$ we determine shorter feature vectors. Notably, the short frame size and port ranges of $\mathfrak{F}^{\text{ALL}}$ have low individual impact. Therefore, we combine them into larger ranges to shorten vector length.

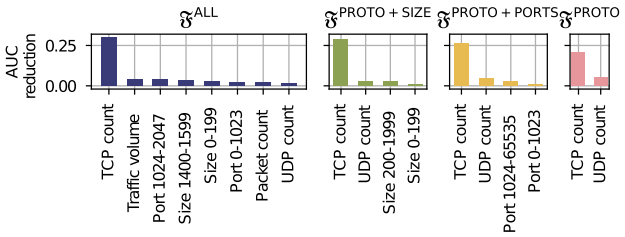

Fig. 2: Feature importance ranked by AUC reduction for all feature vectors.

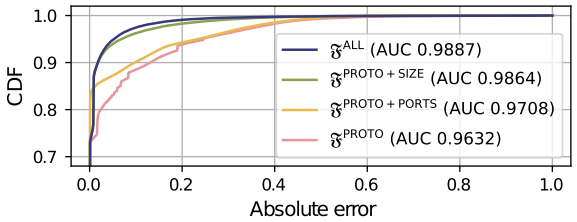

Fig. 3: Cumulative distribution function of the absolute errors in estimations of the attack traffic purity π_{p} when using different feature vectors.

Since there is no clear preference for port over frame size information, we use the complementary vectors $\mathfrak{F}^{\text{PROTO+PORTS}}$ and $\mathfrak{F}^{\text{PROTO+SIZE}}$ for comparison and additional size reduction.

Fig. 3 summarizes the CDF of absolute errors for different feature vectors. The vector $\mathfrak{F}^{\text{ALL}}$ achieves the highest estimation performance with an AUC of 0.9887. In comparison, protocol information alone results in a significant AUC reduction (0.9632 using $\mathfrak{F}^{\text{PROTO}}$). Including complementary port information in $\mathfrak{F}^{\text{PROTO+PORTS}}$ increases the AUC slightly to 0.9708. However, including frame size information instead (in $\mathfrak{F}^{\text{PROTO+SIZE}}$) retains an estimation performance close to the full feature vector $\mathfrak{F}^{\text{ALL}}$ (AUC = 0.9864). This provides a significant feature size reduction from 76 features ($\mathfrak{F}^{\text{ALL}}$) down to 4 features ($\mathfrak{F}^{\text{PROTO+SIZE}}$) with low impact on estimation performance. Through this, the memory footprint of an HHH algorithm used for blacklisting can be reduced by $94.7\,\%$.

Acknowledgment

This work was supported by funding of the Helmholtz Association (HGF) through the KASTEL Security Research Labs (KASTEL) (POF structure 46.23.01: Methods for Engineering Secure Systems).

References

[1] G. Cormode, F. Korn, S. Muthukrishnan, and D. Srivastava, "Finding hierarchical heavy hitters in data streams," in *Proceedings of the 29th International Conference on Very Large Data Bases*, 2003.

[2] D. A. Popescu, G. Antichi, and A. W. Moore, "Enabling fast hierarchical heavy hitter detection using programmable data planes," in *Proceedings of the Symposium on SDN Research*, 2017.

[3] R. Ben Basat, X. Chen, G. Einziger, and O. Rottenstreich, "Designing heavy-hitter detection algorithms for programmable switches," *IEEE/ACM Transactions on Networking*, 2020.

[4] L. Breiman, "Random forests," *Machine learning*, 2001.

[5] R. Fontugne, P. Borgnat, P. Abry, and K. Fukuda, "MAWILab: Combining Diverse Anomaly Detectors for Automated Anomaly Labeling and Performance Benchmarking," in *ACM CoNEXT '10*, 2010.

[6] D. Kopp, C. Dietzel, and O. Hohlfeld, "Ddos never dies? An IXP perspective on ddos amplification attacks," in *Passive and Active Measurement*, 2021.