# Evaluating Drill-Down DDoS Destination Detection

Timon Krack*, Samuel Kopmann*†, and Martina Zitterbart*†
Institute of Telematics, Karlsruhe Institute of Technology*
Kastel Security Research Labs†
{timon.krack,samuel.kopmann,martina.zitterbart}@kit.edu

***Abstract*—Volumetric Distributed Denial of Service (DDoS) attacks remain a persistent threat to network infrastructures. The increasing number of devices connected to the Internet, e.g., through the Internet of Things, and growing data rates make volumetric DDoS attacks easier to execute at a large scale, resulting in a high impact on the target. Drill-down DDoS Destination Detection aggregates ingress traffic in two dimensions, the source and destination IP address space, represented as a fixed-state image. A Convolutional Neural Network (CNN) identifies the attack's target subnet using multi-class classification. Iteratively monitoring only the subnet detected by the CNN at a finer granularity leads to increasingly accurate results until the exact destination IP address is determined. The destination drill-down is evaluated on synthetic and authentic traffic, detecting an attack with 2.5% of the background traffic within 275 milliseconds.**

***Index Terms*—DDoS Detection, Machine Learning, Computer Vision, Traffic Monitoring**

## I. Introduction

The ability to efficiently detect the target of *Distributed Denial of Service* (DDoS) attacks helps to mitigate their impact on the network. DDoS mitigation systems can remove attack traffic to reduce the impact on the targeted network and service, however they are resource intensive and introduce packet processing overhead. If the target of an ongoing DDoS attack is identified, the mitigation system only needs to inspect traffic directed towards that destination. *Machine Learning* (ML) enables learning and recognizing attack patterns automatically, avoiding the need for hand-crafted rules. Detecting network attacks with ML requires a transformation of monitored packets to traffic features that serve as input data for the ML algorithm. Especially at high data rates (e.g., during volumetric DDoS attacks), the computational time overhead per packet has to remain below the average packet interarrival time to prevent a backlog. Fine-granular traffic monitoring, e.g., flow-based monitoring [6] [7] using IP addresses, port numbers and the transport protocol can become too slow at high data rates and requires memory for state-keeping dependent on the flow count. Attackers can increase the flow count by sending packets with randomly spoofed IP source addresses and with randomized port numbers, resulting in an arbitrary number of flows that can exhaust the available memory of the monitoring server.

To guarantee low per-packet overhead and memory consumption independent of the traffic volume, the destination drill-down approach [1] aggregates monitored traffic in only two dimensions: The source and destination IP address space.

© 2024 IEEE. Personal use of this material is permitted.
Permission from IEEE must be obtained for all other uses.

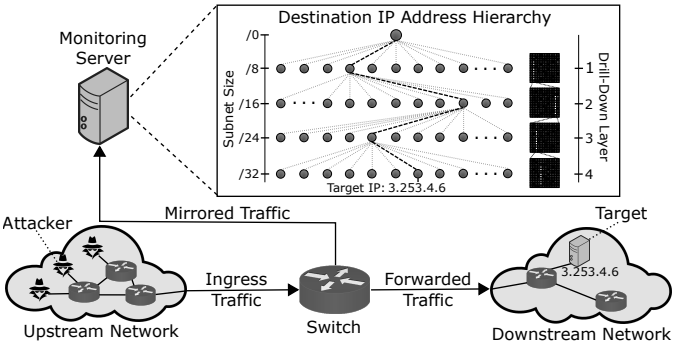



Fig. 1. High-level overview of positioning in the network and effect of destination drill-down detection.

The aggregated traffic is represented as a fixed state size image, which is then classified by a *Convolutional Neural Network* (CNN) [2] [3], determining with high accuracy whether an attack is ongoing or not.

The drill-down approach identifies the destination IP address of DDoS attacks using multi-class classification. If an attack is present, the CNN determines a target subnet containing the detected attack. The identified target subnet is then monitored at a finer granularity. The *drill-down* in the destination IP address hierarchy is illustrated in Fig. 1. Iteratively monitoring only the detected subnets at higher granularity results in increasingly accurate subnets, until the exact destination IP address of the attack is identified. Instead of fine-granular traffic monitoring for all ingress traffic, only a part of the destination address space is considered, depending on the current iteration (drill-down layer). In case the attack vanishes while the drill-down is being performed, a false subnet was chosen or after a successful detection, the granularity is reset to the first layer where the complete IP address space is monitored again.

The drill-down approach monitors traffic with a fixed memory consumption and low per-packet overhead, which makes it attractive for deployment in high-volume traffic environments, like internet exchange points. We focus on the following aspects evaluating drill-down DDoS destination detection:

- Detection performance for different attack intensities and exposure times across all drill-down layers
- Detection performance at individual drill-down layers
- Duration to identify attacks of different intensities for a minimum precision value
- Improvements to further reduce the overall reaction time

## II. Background

This section explains the structure and classification of traffic images as an approach to efficiently aggregate ingress traffic by its source and destination address.

### A. Traffic Image

The left side of Fig. 2 shows the structure of a *traffic image*, which summarizes the observed network traffic as a 2D frequency distribution. The y and x axes represent the monitored source and destination IP address space, which are divided into $\rho$ equal-size subnets. The traffic image consists of a $\rho \times \rho$ grid of counters, which are initialized with 0. The y-axis always covers the full IP source address space. The x-axis contains the IP destination address space starting from the address $\alpha_{\text{IP}}$ and ending after $\rho$ $\gamma_{\text{IP}}$-sized segments.

For each observed packet, its source and destination address is used to find the corresponding counter in the grid, which gets incremented. Assuming 32-bit integers as counters, a traffic image of resolution $\rho = 256$ summarizes traffic using a fixed memory size of $32\rho^2 = 2,1\text{Mbit} = 262\text{KByte}$, independent of the traffic volume.

### B. Multi-class Image Classification

A CNN classifies the obtained traffic images during monitoring. Binary image classification predicts whether there is an attack present in the observed traffic. The results from MIDA [2] show that binary traffic image classification is possible with high accuracy and low latency.

For drill-down destination detection, the goal of the traffic image classification is not only to determine if there is an attack present, but also, in which destination subnet it is located, which enables the drill-down of destination subnets. Fig. 2 illustrates a multi-class image classification. Instead of two possible classes, there are numerous possible classes that can be assigned to each traffic image. Since the traffic image is divided into $\rho$ segments representing the destination subnets, there are also $\rho$ classes representing an active attack in their respective destination subnet. Also, the case where no attack is present has to be considered with an additional class ("no-attack" label). This results in a total of $\rho + 1$ possible classifications for each traffic image. The output class that generates the highest activation value from the CNN is deemed to be the classification of the image. Since only a single class at a time is assigned to each traffic image, the CNN can only detect one attack. It is possible to detect multiple concurrent attacks by classifying the traffic image multiple times. After each iteration, the counters for the detected subnet are reset, to prevent the same attack from being detected again, until the CNN chooses the "no-attack" label.

### C. Related Work

MIDA [2] and HollywooDDoS [3] also perform DDoS detection using traffic images, but are designed for hardware-based monitoring in the TCAM of the switch. Compared to destination drill-down, they operate with low resolution traffic images with $\rho \leq 32$ due to the limited TCAM capacity.

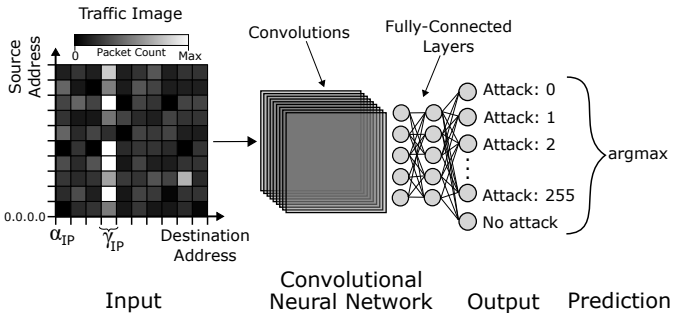



Fig. 2. Schematic view of traffic image processing and label assignment for a traffic image of size $\rho = 256$ (input image depicted smaller)

Drill-down destination detection is implemented in software, as it requires traffic images of higher resolution that may not be possible to create using the switch's limited TCAM capacity. HollywooDDoS does not use a single traffic image like in drill-down, but instead captures a series of consecutive traffic images that serve as input to the machine learning algorithm. When an attack starts, changes in the traffic images are recognized. This enables accurate detection of the presence of an attack within a few milliseconds. However, the exact target destination address is not identified.

The framework discussed in a previous study [4] also detects the exact target destination addresses of DDoS attacks. It uses a two-level approach for coarse- and fine-granular monitoring with sketch data structures. Another approach collects micro-flows over longer time periods [8]. Using clustering algorithms, attacks are then detected with high accuracy and within 2 seconds. Flow segmentation is also used to classify the type of detected DDoS attack [7]. Contrary to destination drill-down, the destination address of the attack is not detected.

However, in all three cases, data structures are used that dynamically vary their size, potentially suffering from scalability issues regarding their memory footprint in the scenario of volumetric DDoS attacks. This is not the case with traffic images, as their memory consumption only depends on the resolution parameter instead of the number of packets.

While drill-down destination detection focuses on volumetric DDoS attacks, low-rate DDoS attacks are detected using micro-flow segmentation [6]. Attack features are then identified by machine learning algorithms. Depending on the exact type, low-rate DDoS attacks may not have enough volume to be visible in traffic images and thus cannot be detected by destination drill-down.

Compared to coarse monitoring strategies with threshold-based detection [5], the drill-down approach does not need to differentiate large legitimate flows from volumetric attack flows. The CNN is trained to identify background and attack patterns independent of the traffic volume. Instead of distributed attacks, an elephant flow has only a single source and does not cause the CNN to detect an attack pattern.

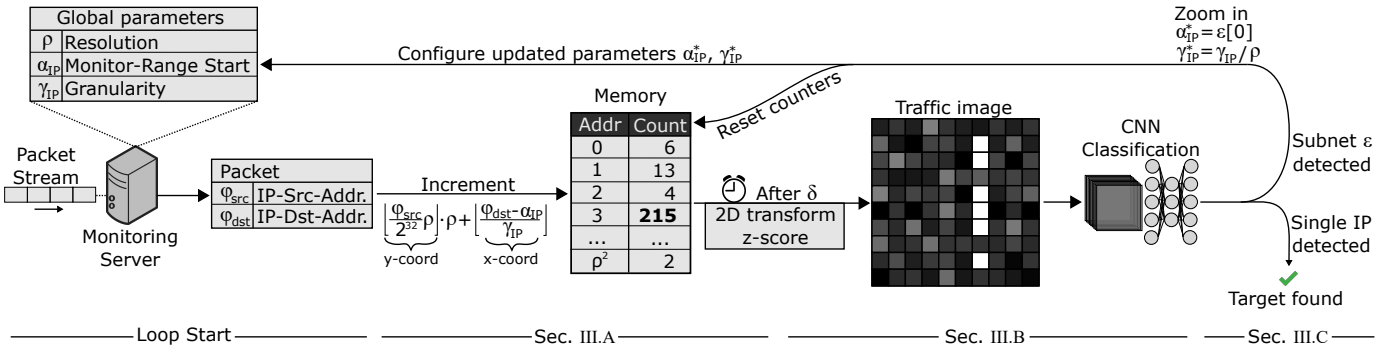



Fig. 3. Drill-down detection loop for monitoring and aggregating traffic, then refining the parameters until detection is successful

## III. Destination Drill-Down in a Nutshell

The drill-down approach continuously executes a detection loop with three consecutive steps, illustrated in Figure 3: ($A$) Parameterized image creation, ($B$) multi-class image classification, and ($C$) parameter adjustment.

### A. Parameterized Image Creation

Drill-down destination detection is running on a server that receives mirrored ingress traffic from a colocated switch. The received traffic is aggregated as a traffic image with a resolution of $\rho = 256$, allowing IPv4 destination addresses to be determined after exactly four drill-down layers. The approach can also be used for IPv6 addresses by using more drill-down layers. The y-axis covers the complete source IP address space, divided into 256 subnets with a prefix length of 8 bits. The x-axis spans the part of the source address space that is currently monitored. The parameter $\alpha_{\text{IP}}$ defines the first destination address that is monitored, and $\gamma_{\text{IP}}$ the subnet granularity (i.e., the number of IP addresses per subnet). Initially, the complete destination address space is monitored, with $\alpha_{\text{IP}} = 0$ and $\gamma_{\text{IP}} = 2^{24}$. As illustrated in Fig. 3, for each packet within the monitored range, the corresponding counter is incremented by calculating the y and x coordinates in the traffic image based on the packet's IP source and destination address. Packets outside the monitored IP address range are ignored.

### B. Multi-Class Image Classification

After a predefined time interval, the exposure time $\delta$, the detection loop continues to the classification step with the goal of detecting attack patterns in the aggregated traffic image. A CNN performs multi-class image classification with each created traffic image to determine ongoing attacks and their destination subnet (i.e., column in the traffic image). The counter values of the traffic image are normalized using z-score normalization with parameters determined from previously monitored traffic (training data). The normalized traffic image is then used as the input for the CNN. Results are one-hot encoded to assign one out of 257 possible classes. The range from $0 - 255$ is used to represent an attack in the respective destination subnet of the traffic image, while the class 256 indicates the absence of an attack. All incoming packets are ignored during the classification, as the parameters of future traffic images depend on the classification result.

### C. Parameter Adjustment

If there is an ongoing attack toward a determined IP destination subnet, the parameters $\alpha_{\text{IP}}$ and $\gamma_{\text{IP}}$ are adjusted for the following image creation to enable a *zoom* on the x-axis of the traffic image (destination addresses) and make the destination subnets more fine-granular. Only the addresses contained in the detected subnet are monitored, but at finer granularity. They are again divided into $\rho$ subnets of equal size. This is done by setting $\alpha_{\text{IP}}$ to the first address of the detected subnet, and dividing $\gamma_{\text{IP}}$ by the image resolution $\rho$. With each zoom, the *layer* of the drill-down increases, initially starting at layer 1. At the 4th layer, each destination subnet represents a single IP address and the drill-down is completed.

When the CNN predicts the 'no attack' label, there are three possible cases. First, in case the prediction was made on layer 1, there is no attack in the observed traffic. The counter values in the traffic image are reset to 0 and the detection loop restarts. Second, it is possible that the attack being tracked has vanished before the final layer of the drill-down is completed. Third, a false positive detection could have caused zooming into a part of the address space without attack traffic. After deciding the wrong subnet for the drill-down, the previously detected attack is no longer visible, which will be classified as benign by the CNN. In both cases, the drill-down cannot continue. Therefore, besides resetting the counter values, the granularity is also reset back to the initial state such that the full address space is monitored in the next cycle.

## IV. Evaluation

The evaluation of the drill-down approach first requires training of the CNN. Training data is required that provides correctly labeled traffic images. With the trained CNN model, the performance evaluation is carried out in a simulated environment to ensure the reproducibility of the results.

### A. CNN Training

The training procedure of the drill-down CNN consists of three main steps. First, the architecture and hyperparameters

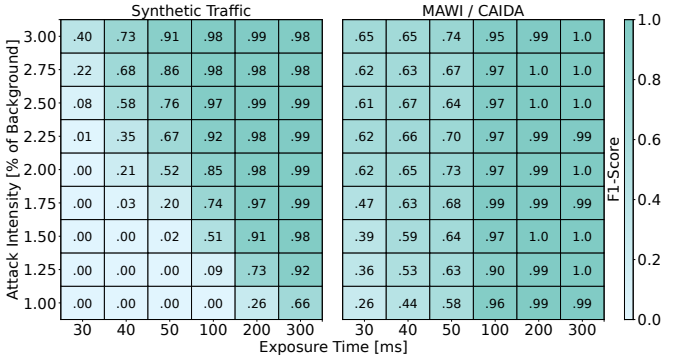



Fig. 4. Achieved F1-Score for different attack intensities and exposure times. 500 samples per cell for 108 cells, 54.000 samples in total.

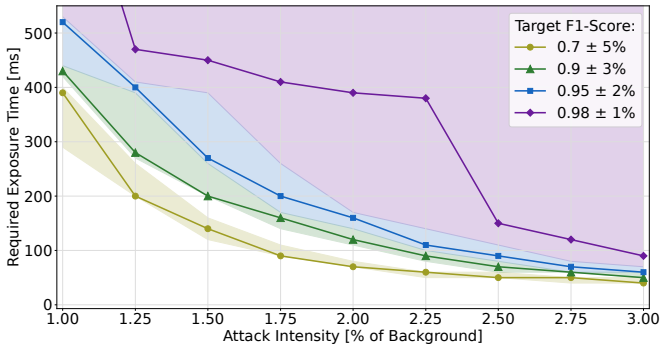



Fig. 5. Minimum required exposure to achieve target F1-Score. Precision 2.000 samples per point for 36 points, 72.000 samples in total.

for the CNN itself and the training data are configured. In this case, $\rho = 256$ and thus, the network has $256 \times 256$ input values and determines the classification of the image using the maximum activation on one of the 257 output classes (including the 'no attack' option).

Training data is generated using randomized traffic for both background and attack to create synthetic traffic images. For each image, 100.000 packets with uniformly random IP source and destination addresses are generated. The *attack traffic intensity* ($I$) then specifies the amount of attack traffic relative to the volume of background traffic. Attack packets also have a random source address, but a common destination address for each generated traffic image. Another set of training data is created using authentic traffic. A packet capture containing backbone traffic from the WIDE MAWI archive [9], recorded on October 1st 2022 and containing 45 million packets, serves as background traffic. The first half of the file is used as training data, while the remainder is used during evaluation. Authentic attack traffic is obtained from the CAIDA dataset [10], which captured an ICMP flood attack. Packets from the CAIDA capture are sampled to match $I$ and are combined with the MAWI traffic.

The models used in this evaluation are trained on 250 batches of traffic images for each of the 4 drill-down layers, with each batch containing one traffic image for each of the 257 possible classifications, thus 257.000 samples in total. For each sample, the attack intensity is varied uniformly randomized between $I = 1\%$ and $I = 10\%$.

### B. Evaluation Setup

Drill-down is executed in a simulated environment to allow for reproducible results and parallelization. The simulator generates traffic images, and then, provided the trained CNN model, observes the model's prediction performance on the generated image. First, the traffic intensity $I$ and exposure time $\delta$ are configured. A random IP address is chosen as the target of the DDoS attack for each generated instance and stored as the correct label for that traffic image. For instances that should just contain background traffic without any attack, $I = 0$ is chosen, and the expected label is set to 256 ('no attack' label). The generated traffic image is submitted to the CNN for predicting its classification. The drill-down then continues as described in section III-C until either the destination detection is completed or the 'no attack' classification is predicted.

After the drill-down has finished, either with an attack detection or a no-attack result, its performance is assessed by a classification in the confusion matrix (*full-cycle performance*). To achieve a true positive (TP), the CNN must predict the correct classification for the respective traffic image from each drill-down layer, such that the last layer reveals the attacked address. If 'no attack' is predicted in any layer, the iteration is considered a false negative (FN) and the drill-down terminates. In the case that $I = 0$, a prediction of the 'no attack' classification yields a true negative (TN). If an incorrect attack classification is predicted, the drill-down continues and zooms into the corresponding image column. If the CNN then predicts 'no attack' before reaching the final drill-down layer, the iteration is counted as TN if there was no attack traffic present in the original image, or as FN is there was an attack, but the CNN chose the wrong column for the drill-down. If a wrong attack prediction is made in the final layer, an incorrect IP address would be decided as the attacked target, resulting in a false positive (FP). The individual decisions on each layer are also saved to be able to analyze wrong predictions on higher layers that are then corrected on lower layers.

### C. Full-Cycle Performance Evaluation

To measure the detection performance of the drill-down destination detection, the *F1-Score* is calculated for a set of iterations with equal parameters. A score of 1.0 indicates flawless decisions, while a score of 0.0 results from worst decisions and thus high FP and FN rates. In Fig. 4, the achieved F1-Score is shown for different traffic image exposure times on the x-axis and attack intensities on the y-axis, with synthetic traffic (left) and MAWI/CAIDA traffic (right).

With the synthetic traffic, exposure times of $\leq 50$ milliseconds are not sufficient to reliably detect attacks, as there is not enough time for collecting meaningful information in the traffic image. High intensity attacks are easier to distinguish from the background traffic and are detected more often. For larger exposure times ($\geq 100$ milliseconds), more information can be collected and attacks at lower intensities are detected with

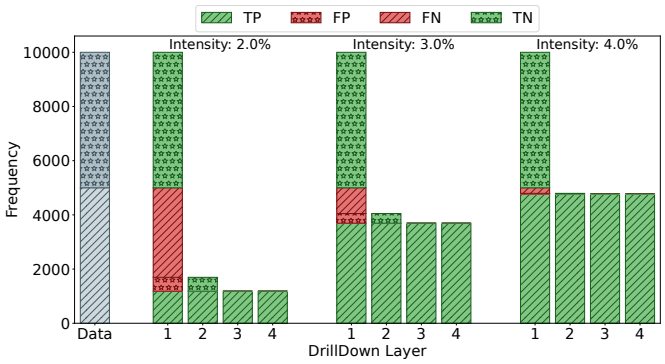



Fig. 6. Decisions of the CNN at each layer of the drill-down for different attack intensities. 10.000 samples per intensity, 30.000 samples in total.

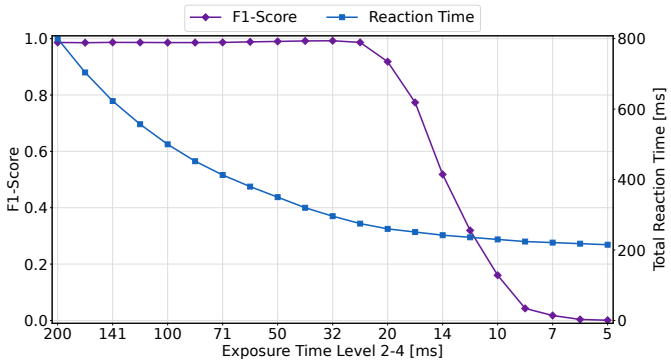



Fig. 7. F1-Score and total reaction time for $\delta_1 = 200$ms and decreasing $\delta_{2+}$. 2.000 samples per point for 21 points, 42.000 samples in total.

higher confidence. However, the time until an attack is detected (*reaction time*) also increases, since each layer takes longer to complete. This results in a trade-off regarding the exposure time: the CNN cannot reliably detect low intensity attacks at low exposure times. Increasing the exposure time beyond 200 milliseconds grants diminishing F1-score improvements.

On the right side of Fig. 4, the same experiments are repeated with MAWI/CAIDA traffic, demonstrating that the drill-down approach works on authentic data. The same trade-off regarding the exposure time applies, eventhough less prevalent in some cases: Contrary to the synthetic scenario, even low intensity attacks are occasionally detected with low reaction times of $\leq 50$ milliseconds. Compared to the synthetic random traffic, there is not simultaneous activity everywhere in the destination IP address space, resulting in less potential wrong choices for the CNN. However, with low exposure times, there are more random errors for high intensity attacks compared to the synthetic scenario. Reliable detections for all considered intensities require exposure times $\geq 200$ milliseconds.

Fig. 5 analyzes the trade-off between exposure time and attack intensity in more detail, using the synthesized scenario. The y-axis shows the minimum required exposure time to reach a target F1-Score for different attack intensities on the x-axis. The minimum required exposure time is determined by increasing the exposure time in steps of 10 milliseconds until the target F1-Score is reached. Each curve shows the required exposure time for a different target F1-Score. The surrounding area of the same color indicates the range of exposure times required for a small deviation, as specified in the legend. There is an exponential decrease in the required exposure time with an increasing attack intensity. Therefore, just a small increase in the exposure time results in a vast improvement of the F1-Score. At an intensity of $I \geq 2.75\%$, the difference between the required exposure time for achieving an F1-Score of 0.7 and 0.95 becomes negligible. Small increases of the exposure time, in the range of milliseconds, can significantly improve the detection performance. However, this observation changes when considering F1-Scores close to 1.0. The required exposure time for a score of 0.98 is unproportionally higher and volatile, as indicated by the large hose created by a deviation of just 1% from the target F1-Score. In this case, perfect F1-Scores were not obtained for any measured exposure time up to 5000 milliseconds.

Since small increases in the exposure time can grant a significant improvement in the F1-Score, minor increases in the reaction time can be worth the improved detection performance. However, random classification errors persist for all configurations, making perfect results close to 1.0 unreachable. The improvement for exposure times greater than $300 - 400$ milliseconds are diminishing. Furthermore, false-negative classifications can be cured by a correct decision in the next drill-down attempt, albeit with the downside of delaying the detection by one cycle. Random classification errors only prevent a detection on the first attempt.

### D. Evaluation of Individual Drill-Down Layers

This section focuses on analyzing the individual decisions made on each layer during a destination drill-down. In this context, false/true positive/negative decisions only relate to a single classification made by the CNN at a specific layer during the destination drill-down. Fig. 6 shows the distribution of false and correct decisions per drill-down layer. The bars are grouped by attack intensity, for values of two, three, and four percent. Each group contains four stacked bars for the four drill-down layers required to obtain the exact attack destination address, as explained in Sec. III-C.

The exposure time is fixed at $\delta = 100$ milliseconds. This is not the optimal exposure time, e.g., using $\delta = 200$ milliseconds reliably detects all three intensities according to Fig. 4. Using a lower exposure time leads to more false classifications, which enables a more detailed analysis of errors during the drill-down. For each of the three scenarios, 10.000 iterations were performed, half of which only consisted of legitimate traffic, while the other half additionally contained an attack against a random IP address. The bar chart then shows the frequency of detection results for each layer. Negative detections always result in a termination of the drill-down, thus both TN and FN iterations are no longer present in subsequent layers. The resulting chart allows for two key insights.

First, false-positive detections only appear at the first layer. This means the drill-down incorrectly continues to the next

layer. However, all FP detections from the first layer are successfully rejected as TNs at the second layer. If the CNN chooses an incorrect column at the first layer that does not contain an attack, subsequently zooming into that column results in a sparse traffic image for the next layer. This is then identified as a TN by the CNN, resetting the drill-down back to the first layer. An FP attack detection would only occur if there was an FP decision at the fourth layer of the drill-down. In the $30.000$ iterations made, this never occurred and all FP decisions of the first layer were corrected at the second layer.

Second, if there is a true-positive detection at the first layer, the destination drill-down is always successful in the other layers, resulting in a correct detection. Since the first layer monitors and aggregates the complete IP address space, it has to deal with the most noise from legitimate background traffic. Lower layers only monitor a fraction of the total traffic with higher granularity, therefore less noise from background traffic is present. If the CNN was able to correctly identify an attack on the first layer despite the high noise, it can also subsequently make correct decisions on the lower layers as the attack is even more distinguishable from the background noise compared to the first layer.

### *E. Reducing Reaction Time*

Detecting the destination of new attacks requires four exposure times and classification durations ($c_d$), i.e., $4(\delta + c_d)$. On average, the CNN classification requires $c_d = 15$ milliseconds on an Nvidia A30 GPU. The delay caused by the exposure time $\delta$ is usually much longer, but it can be optimized. The observations from section IV-D indicate that false decisions only occur in the first drill-down layer. In particular, if the subnet containing the attack is successfully identified at the first layer, zooming results in similar attack features but reduced background traffic since the background traffic outside the detected subnet is no longer monitored. This allows for an easy decision by the CNN. Since the intensity $I$ represents the ratio between background and attack volume, the reduction in background traffic on lower drill-down layers effectively increases the observed $I$ there. Section IV-C demonstrates how the CNN requires less exposure time as $I$ increases. Combining these two observations results in an idea for improving the reaction time. Instead of using the same exposure time $\delta$ for all layers, we define two separate exposure times $\delta_1$ for the first layer and $\delta_{2+}$ for subsequent layers, with $\delta_1 \gg \delta_{2+}$. This allows for enough information to be collected for a correct decision at the first layer, while subsequent layers can be decided quickly since they require less exposure time for correct decisions. This reduces the total reaction time to $\delta_1 + 3\delta_{2+} + 4c_d$ without loosing detection performance, as long as $\delta_{2+}$ leaves sufficient time to capture enough attack traffic such that the CNN can still identify it. Fig. 7 shows the achieved F1-Score for decreasing $\delta_{2+}$ with a constant $\delta_1 = 200$ milliseconds and $I = 2\%$. The F1-Score remains close to 1.0 for decreasing $\delta_{2+}$, indicating that the CNN still makes accurate decisions on lower drill-down layers even with the reduced exposure time, as long as the first layer was decided correctly. This enables a significant reduction in the total reaction time without loosing detection performance. For $\delta_{2+} \leq 20$ milliseconds, the information collected for the lower drill-down layers is no longer sufficient. In this case, the CNN makes wrong decisions on subsequent layers even if the first layer was decided correctly, causing the sharp drop in F1-Score on the right. However, since the first layer still needs to get the full exposure time, the benefit of reducing $\delta_{2+}$ even further is minimal. Choosing $\delta_{2+} = 25$ milliseconds would be optimal in this case, as it reduces the total reaction time from $800$ to $275$ milliseconds ($-65\%$) without affecting the F1-Score of $\approx 1.0$. It is also possible to keep the total reaction time constant, but allocate a larger share of it to the exposure time on the first layer by reducing $\delta_{2+}$ and increasing $\delta_1$. This improves the achieved F1-score by allowing for more accurate decisions in the first layer without increasing the reaction time.

## V. Conclusion

Drill-down destination detection performs sub-second detection of a DDoS attack's target IP address. Two-dimensional traffic images are classified by a CNN to iteratively resolve the attack's target subnet at a finer granularity. The drill-down continues until the exact IP address is identified. Contrary to other approaches, the memory consumption is fixed and not dependent on the number of arriving packets or active flows, while maintaining constant-time processing overhead per packet. The achievable traffic throughput depends on the capabilities of the hardware in the monitoring server, as the implementation is software-based. The drill-down strategy realizes a trade-off between the required exposure time of the traffic image (and therefore, detection speed) and the delivered precision by the CNN. Low intensity attacks can only be detected after long exposure times. For attacks of higher intensity, the required exposure time to make accurate detections decreases exponentially and reaches below 100 milliseconds if the attack traffic makes up $2.5\%$ of the total observed traffic. Analyzing the decisions made by the CNN on each individual layer indicated that the main source of errors is the first layer of the drill-down, as it contains the most noise caused by benign background traffic. This motivates the idea to use decreasing exposure times for lower drill-down layers. The first layer, being most prone to errors, receives the largest exposure time. The exposure time of subsequent layers can be reduced significantly without impacting detection performance. In the evaluated scenario, this strategy improved the total reaction time by $65\%$ while retaining near optimal detection performance.

## VI. Acknowledgements

This work was supported by the bwNET2.0 project, which is funded by the Ministry of Science, Research and the Arts Baden-Württemberg (MWK). This work was supported by funding of the Helmholtz Association (HGF) through the Kastel Security Research Labs (POF structure 46.23.01: Methods for Engineering Secure Systems).

## REFERENCES


[1] S. Kopmann, T. Krack, and M. Zitterbart. “7D. Demonstrating Drill-Down DDoS Destination Detection”. IEEE ICMLCN 2024, Demonstration.

[2] S. Kopmann, H. Heseding and M. Zitterbart. “MIDA: Micro-flow Independent Detection of DDoS Attacks with CNNs”. In: Zirpins, C., et al. Advances in Service-Oriented and Cloud Computing. ESOCC 2022. Communications in Computer and Information Science, vol 1617. Springer, Cham. https://doi.org/10.1007/978-3-031-23298-5_3

[3] S. Kopmann, H. Heseding and M. Zitterbart. “HollywooDDoS: Detecting Volumetric Attacks in Moving Images of Network Traffic”. 2022 IEEE 47th Conference on Local Computer Networks (LCN), pp. 90-97. https://doi.org/10.1109/LCN53696.2022.9843465

[4] L. Haiqin, S. Yan and K. Min. “A Scalable DDoS Detection Framework with Victim Pinpoint Capability”. Journal of Communications, vol 6. https://doi.org/10.4304/jcm.6.9.660-670

[5] M. Silva, J. Marques, L. Gaspary, and L. Granville. “Identifying elephant flows using dynamic thresholds in programmable IXP networks”. Journal of Internet Services and Applications, 2020. https://doi.org/10.1186/s13174-020-00131-6

[6] K. Sudar and P. Deepalakshmi. “Flow-based detection and mitigation of low-rate DDoS attacks in SDN environment using machine learning techniques”. IoT and Analytics for Sensor Networks 2022, pp. 193-205.

[7] M. Sidiq, et al. “Feasibility Evaluation of Compact FLow Features for Real-time DDoS Attacks Classifications”. IEEE COMNETSAT 2022. https://doi.org/10.1109/COMNETSAT56033.2022.9994323

[8] B. Ricks, P. Tague, and B. Thuraisingham. “DDoS-as-a-Smokescreen: Leveraging Netflow Concurrency and Segmentation for Faster Detection”. IEEE 3rd TPS-ISA, 2021. https://doi.org/10.1109/TPSISA52974.2021.00024

[9] K. Cho, K. Mitsuya, and A. Kato. “Traffic Data Repository at the WIDE Project”. Proceedings of the Annual Converence on USENIX, ATC 2000. https://dl.acm.org/doi/10.5555/1267724.1267775

[10] Caida.org “The CAIDA UCSD ’DDoS Attack 2007’ Dataset”. 2007. https://caida.org/catalog/datasets/ddos-20070804_dataset