# On Multi-Point, In-Network Filtering of Distributed Denial-of-Service Traffic

Mingwei Zhang*, Lumin Shi*, Devkishen Sisodia*, Jun Li*, Peter Reiher†
* University of Oregon
{mingwei, luminshi, dsisodia, lijun}@cs.uoregon.edu
† University of California, Los Angeles
reiher@cs.ucla.edu

*Abstract*—**Research has shown that distributed denial-of-service (DDoS) attacks on the Internet could often be better handled by enlisting the *in-network* defense of multiple autonomous systems (ASes), rather than relying entirely on the victim's Internet Service Provider at the edge. Less noticed but important is the fact that an in-network defense can also remove DDoS traffic from the Internet early *en route* to the victim, thus decreasing the overall load on the Internet and reducing chances of link congestion. However, it is not well understood to what degree different in-network defense strategies can achieve such benefits. In this paper, we model the existing two main categories of in-network DDoS defense algorithms (PushBack, SourceEnd) and propose a new type of algorithm (StrategicPoints). In particular, we compare their effectiveness in minimizing the amount of DDoS traffic that the victim receives, their impact on reducing the DDoS traffic on the entire Internet, and their resiliency against intelligent adversaries and dynamic attacks. We detail how the comparison results vary according to parameters and provide our insights on the pros and cons of these three categories of in-network DDoS defense solutions.**

## I. Introduction

Distributed denial-of-service (DDoS) attacks are becoming more prevalent and powerful than ever before. The sheer volume of DDoS attack traffic has been reported reaching hundreds of Gbps [1], [2] and even terrifying 1.2 Tbps [3], [4] and 1.3Tbps [5]. Traditional single-point-of-defense mechanisms, usually at the victim network or any mitigation network (e.g., a traffic scrubbing center) at the edge of the Internet, become too costly under such a scale of attacks [6], [7]. As a result, researchers have put forward a number of *in-network DDoS defense* solutions. Instead of defending at the edge, they defend against DDoS attacks before the traffic reaches the victim, often when the DDoS traffic is even further away from the victim's network.

Although many in-network DDoS defense solutions have been proposed, it still remains difficult for a victim to select suitable defense solutions against specific DDoS attacks. The in-network defense solutions (such as DefCOM [8], PushBack [9], and MiddlePolice [10]) vary greatly in resource requirements, training data needed, and expected efficiency. Selecting a sub-optimal defense solution could introduce substantial cost and even result in unsuccessful defense. However, there is no quantitative study on how the solutions compare to each other, nor a general model that describes these solutions in a common language. Further, it is also unknown how these solutions perform under insufficient knowledge of the attacks or against intelligent adversaries who can dynamically revise their attack strategies to escape defense. Without a quantitative comparison, it is hard for a DDoS victim to select the most suitable solution to achieve its defense goal and meet the resource requirements.

In this paper, we introduce a modeling and simulation framework to systematically evaluate in-network DDoS defense algorithms. The framework contains a general model that can describe the attack and defense for various defense algorithms. Using this model, we summarize the existing in-network DDoS defense algorithms into two basic types: PushBack and SourceEnd. A PushBack algorithm employs propagation-based mechanisms to locate suitable defense locations that are close to the victim. A SourceEnd algorithm tries to locate sources of attacks and deploy filtering rules as close to the sources as possible. These two types of algorithms cover most in-network DDoS defense solutions (see Sec.II). We then introduce a new type of in-network algorithms that utilizes the topology of the attack sources and ASes *en route* and locates suitable defending ASes in network at critical locations. We call this type of algorithms StrategicPoints. We study both the existing PushBack and SourceEnd algorithms and our proposed StrategicPoints algorithms in depth on their defense performance, resource cost, and resiliency against intelligent adversaries.

To quantify the performance of the algorithms, we developed a simulation system that can simulate DDoS attack and defense at Internet scale. The simulation system constructs an AS-level Internet topology and simulates the decision-making procedure according to the DDoS defense algorithms. Upon the decision of defense, each selected AS in the simulation will mitigate the corresponding portion of the attack traffic based on its capacity, and the system will collect the overall mitigation status for defense efficiency assessment. We compare the results using two metrics: DDoS traffic leakage to the victim and DDoS traffic pollution on the Internet (see Sec. III). The simulation results provide useful insights into the selection of defense algorithms in response to different attack scenarios. When facing strict resource constraints, PushBack performs slightly better than StrategicPoints and significantly better than SourceEnd in terms of DDoS traffic leakage.

978-3-903176-15-7 © 2019 IFIP

With more available resource, StrategicPoints becomes as effective as PushBack on reducing leakage and significantly outperforms PushBack in reducing pollution caused by DDoS attacks. StrategicPoints strikes a balance between DDoS traffic coverage and pollution reduction, and is more effective in cases when resources are not extremely restrictive.

The paper is organized as follows. In Section II, we review the state-of-the-art related work on in-network DDoS defense algorithms. We first formally define our models in Section III, describing the Internet, DDoS attacks, and DDoS defenses. In Section IV, we propose a classification of in-network DDoS defense algorithms. Section VI examines the simulation results and compares the three defense algorithms against different metrics. Finally, we discuss limitations and open issues in Section VII, and conclude the paper in Section VIII.

## II. Related Work

At a very high level, the existing DDoS defense can be categorized into two styles [26]: single-AS edge defense, or in-network defense. Edge defense mechanisms defend against DDoS attacks within one AS, and usually at the receiving end of the DDoS attack traffic. Single-AS defense solutions, such as work from Sahay et al. [12] (which redirects attack traffic to middleboxes close to the victim), RADAR [11] (which detects and throttles attack traffic at the victim network), SPIFFY [13] (which temporarily increases the effective bandwidth of a congested core link and observes the response to detect and mitigate an attack), and Bohatei [14] (which presents a flexible and elastic DDoS defense system geared towards a single ISP providing customers with DDoS-defense-as-a-service), are easier to deploy and more flexible to implement when compared to in-network defense solutions, especially when network management complexity is reduced by leveraging software-defined networking or SDN. However, many edge defense solutions can incur a very high defense cost due to resource requirement in the term of network connection and network devices [6], [7], and often fail to mitigate attacks when victims' inbound connections are inundated with DDoS traffic.

The in-network DDoS defense solution, on the other hand, can reduce the amount of resources needed at each collaborating AS and relieve ASes from the heavy burden of network and equipment costs. In this work, we focus on modeling the in-network DDoS defense solutions as they do not have the same problems as single-AS solutions. In fact, there are growing interests in both industry and academia in developing and deploying in-network defense mechanisms such as those shown in DOTS [27], AITF [18], DefCOM [8], and StopIt [20].

Almost all in-network defense solutions require placement strategies to decide where on the Internet to deploy traffic filters or defense measure. We categorize existing in-network defense strategies into three categories: *PushBack*, *SourceEnd*, and *other* (shown in Table I).

**PushBack defense**: We define PushBack defense algorithms as those that start defense from a victim AS and expand the defense area to its upstream ASes. Starting from the victim AS, PushBack allows each defending AS to mitigate a portion of the attack traffic, and delegate the rest of the attack traffic to its upstream ASes for further mitigation. The original PushBack style defense propagation is introduced in the *PushBack* paper [9]. Although this work considers router-level defense propagation, the basic idea can be applied to AS-level collaboration. Other distributed defense systems that stem from the classic PushBack paper, such as the work of Yau et al. [16] and ScoreForCore [15], follow the same PushBack algorithm to defend against DDoS attacks.

**SourceEnd defense**: Different from a PushBack algorithm where the defense initiated from the victim side, a SourceEnd algorithm attempts to select the ASes that are the sources of the attack or close to the sources, and only fall back on downstream ASes toward the victim if resources run out. D-WARD system [21], for example, installs rate-limiting rules at border routers in source networks; COSSACK [19] deploys countermeasures at the ASes of attacking sources. Both are early works that employ the SourceEnd strategy. Later work such as AITF [18] introduced the idea of propagating the defense from the attacking sources to the victim, thereby providing more flexibility for defense deployment. Specifically, authors of AITF also observed that the current generation of routers have sufficient filtering resources to mitigate DDoS attacks as long as the attack traffic was blocked close to the attacking sources. Furthermore, AITF was also one of the earliest projects to study hardware rule space during defense. Later work from Huici et al. [23] and StopIt [20] enhance AITF by introducing security measures against DDoS attacks on the defensive infrastructure itself. Unlike PushBack solutions, prior to defending against the attack all SourceEnd solutions need to know the attack topology ( i.e., the attack sources and their AS-level routes toward the victim) for each DDoS victim

**In-network defense systems without clear defense placement strategies:** Besides the aforementioned two categories, there also exist systems that provide DDoS defense frameworks without clear placement strategies. Previous work from Keromytis et al. [24] and Anderson et al. [25] introduce authentication nodes at key locations between the sender and the receiver in order to filter out unwanted traffic. More recent work from Liu et al. [10] (MiddlePolice) utilizes SDN to measure network congestion status, exchanges measurement among collaborating ASes to discover congested links across the Internet, and then places traffic filters at the routers within congested ASes. However, these these works do not clearly state where on the Internet the defense should happen, leaving the decisions to the operators or other algorithms. Without a detailed defense strategy, it is difficult to judge how these systems perform under different DDoS attacks.

## III. Modeling DDoS Attacks and Defenses

In order to evaluate the performance of different types of in-network defense algorithms, we first construct a model to describe DDoS attacks and their defenses. In this section, we

| Work | Single-AS | Multi-AS | | |
|---|---|---|---|---|
| | | PushBack | SourceEnd | Other |
| RADAR[11], Sahay et al.[12], SPIFFY[13], Bohatei[14] | ✓ | | | |
| ScoreForCore[15], Yau et al.[16], Mahajan et al.[9] | | ✓ | | |
| FireCol[17], DefCOM[8], AITF[18], COSSACK[19], StopIt[20], D-WARD[21], Argyraki et al.[22], Huici et al.[23] | | | ✓ | |
| MiddlePolice[10], Keromytis et al.[24], Andersen et al.[25] | | | | ✓ |

TABLE I: DDoS defense solution categorizations

introduce our general model that describes the Internet, DDoS attacks, and in-network DDoS defense.

### A. Modeling the Internet

The Internet is a well-interconnected network consisting of thousands of autonomous systems (ASes). ASes on the Internet can be represented by the set $\mathbb{N} = \{n | n\ is\ an\ AS\ on\ the\ Internet\}$. The size of $\mathbb{N}$ is 63332 at the time of writing [28]. The traffic running on the Internet can be summarized into $flow$s, where each flow $f$ represent a set of packets between the $source$ of the flow (denoted as $f.src$) and the $destination$ of the flow (denoted as $f.dst$) for a transaction. In this paper, we focus on using IP addresses to identify an entity involved in a flow. Each flow is also associated with a volume value, denoted as $|f|$, which can be represented by the number of packets or the number of bytes included in the flow.

### B. Modeling DDoS Attacks

The total set of attack sources is represented by $\mathbb{A} = \{a | a \in \mathbb{N}\ and\ a\ is\ an\ attack\ source\}$, where $a$ represents an attack source in a DDoS attack. Similarly, we define the set of victim end-hosts as $\mathbb{V} = \{v | v \in \mathbb{N}\ and\ v\ is\ a\ victim\ of\ the\ DDoS\ attack\}$. Here, one attacking source represents a machine that is controlled by the attacker and creates unwanted traffic to the victims. Each attacking source can generate multiple flows with varying volume to a victim at any moment during an attack. The set of total attacking flows is represented as

$$\mathbb{F} = \{f | f.src \in \mathbb{A}\ and\ f.dst \in \mathbb{V}\},$$

where $f$ is the attacking flow, generated by an attack source, sent to a victim. While a DDoS attack can potentially strike multiple targets, for simplicity, the rest of the paper focuses only on a single victim. Each flow traverses through a set of ASes on the Internet before reaching the victim. We denote the number of AS links a flow $f$ must travel through to reach its victim as $b_f$.

### C. Modeling In-Network DDoS Defense

Different from single-AS edge defense solutions, in-network defenses employ multiple ASes en route of the DDoS attack traffic to filter unwanted traffic. For each defense solution, we denote the set of all ASes that are able to participate on defense, or *defense pool*, as $\mathbb{D}$ and $\mathbb{D} \subset \mathbb{N}$. However, not all ASes in $\mathbb{D}$ will be used by the defense solution. The set

$$\mathbb{S} = \{n | n \in \mathbb{D}\ and\ n\ is\ selected\ for\ defense\}$$

contains all ASes in the network that are not only able to collaborate on defense, but also selected to filter traffic during the defense. The defense algorithms decides which ASes should be utilized for the defense against specific attacks, which takes the following elements into consideration. Finally, we denote

$$\mathbb{L} = \{f | f \in \mathbb{F}\ and\ f\ is\ filtered\},$$

for each defense solution.

*1) Resource for Defense:* Each defending AS has limited resources in filtering DDoS traffic. Specifically, the main resource limitation is on the number of filtering rules an AS can employ for DDoS defense purposes. We define $Rmax$ as the maximum number of filtering rules that can be installed at an AS. The defense may also face a limitation on the number of total ASes it can use, which we denote as $Dmax$. $Rmax$ reflects the resource limitation at the *intra-AS* level, while $Dmax$ reflects the limitation at the *inter-AS* level. In practice, the defense algorithms should take both the resource limitation $Rmax$ and scale limitation $Dmax$ into consideration when initiating defenses.

*2) Leakage and Pollution:* To quantify the effectiveness of a solution, we look at two main metrics: $leakage$ and $pollution$; $leakage$ represents the total amount of attack traffic that reaches the victim after the defense is in place, and $pollution$ represents the total amount of attack traffic that flows through the Internet before it is filtered. The $leakage$ metric shows the defense's effectiveness from the victim's perspective, while $pollution$ reveals the defense's impact on reducing the overall DDoS traffic on the Internet. DDoS attacks can cause link congestion, therefore a DDoS defense algorithm should not only achieve very low $leakage$ to reduce the attack traffic the victim receives, it also needs to further reduce the traffic on the paths toward the victim ($pollution$) to avoid chance of link congestion.

We define $leakage$ as $leakage = |\mathbb{F}| - \sum_{k \in \mathbb{S}} m_k$, where $m_k$ is the total number of attack flows mitigated by AS $k$ in the defending set $\mathbb{S}$. Note that at which AS the filtering happens does not affect the $leakage$ metric.

We define $pollution$ as $pollution = |\mathbb{F}| - |\mathbb{L}|$. The value of $pollution$ for a defense measures how well a defense strategy does on limiting the amount of traffic running on the Internet. Clearly, $c_f = 0$ when only the victim's AS stops an attack flow $f$. If an attack flow $f$ is stopped at the source AS, the value $c_f$ becomes $c_f = b_f$. Figure 1 shows an example of counting the $pollution$ for one flow. The attack flow $f$, of magnitude $|f| = 1$, travels through the AS path 1-2-3-4-5, and AS3 deploys DDoS defense and filters $f$. In this example,

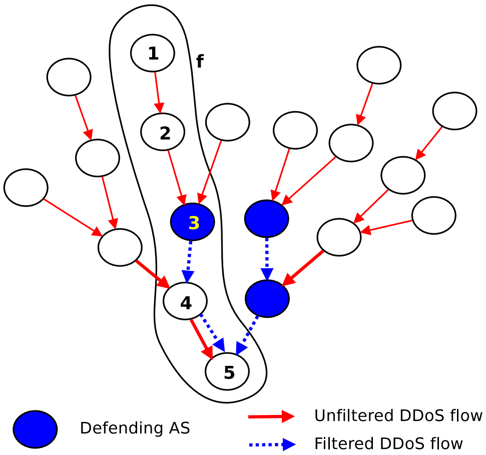


Fig. 1: Example of calculating the $pollution$ for one attack flow $f$ and an defense measures. The pollution flow $f$ remains to have after defense is 2, between $AS1$ and $AS3$.

suppose that AS 3 effectively filters the attack flow, the total length of $f$ is $b_f = 4$, and there are two links after AS3 that will not see $f$, which means $c_f = 2$. Filtered at AS3, $f$ contributes $b_f - c_f = 4 - 2 = 2$ to the overall $pollution$ metric. From this example, we can see that the closer a flow is filtered to the attack source, the less it contributes to the overall $pollution$.

## IV. In-network DDoS Defense Algorithms

In general, the in-network DDoS defense algorithms can be summarized into two major types: **PushBack** algorithm that focuses on placing the defense close to the victim network; and **SourceEnd** algorithm that distribute defense load among networks close to the attack sources. These types of algorithms have been used in various DDoS defense projects (see Sec.II). However, there is no quantitative study on how the solutions compare to each other, nor a general model that describes these solutions in a common language. Therefore, we compare these algorithms and study their strengths and weaknesses. Furthermore, we propose a new algorithm called StrategicPoints algorithm which employs ASes at critical locations of the attacks to achieve the high effectiveness with low cost. In order to compare these algorithms, we generalize each algorithm and study each one individually. In the rest of this section, we will describe the three algorithms using the model introduced in Sec.III.

### A. PushBack Algorithm

The PushBack algorithm propagates the defense workload from the victim AS to upstream ASes. The PushBack algorithm expands the defensive area from the victim AS to its upstream neighbors one AS at a time, and further upstream if necessary. PushBack essentially distributes the DDoS defense load (i.e., the deployment of traffic filtering rules) among the set of collaborating upstream ASes. PushBack can be applied recursively, allowing it to cover a larger set of ASes if necessary.

The PushBack algorithm runs recursively among the collaborating ASes starting from the victim AS. The algorithm begins by selecting the ASes that are nearest to the victim as defending ASes. In each round, each defending AS installs a number of rules to filter a portion of the attack traffic running through it. If more traffic needs to be filtered, the PushBack algorithm will select the next best AS from the *defense pool* to collaborate on defense, where a defense pool consists of the next available upstream ASes of all current defending ASes, sorted by number of flows they can filter. PushBack selects the AS that can filter the most DDoS traffic at each round of defense propagation. The defense propagation ends when there are no available ASes in the *defense pool*, the number of defending ASes exceed the victim's specified parameter $Dmax$, or all traffic has been successfully handled.

### B. SourceEnd Algorithm

In contrast to the PushBack algorithm, the SourceEnd algorithm attempts to select ASes that originate the attack traffic for defense, thereby stopping the attack directly at the sources. The SourceEnd algorithm intuitively performs better in terms of reducing the overall attack traffic on the Internet ($pollution$). However, it requires more participating ASes and traffic filters to be effective at mitigating the attacks.

Ideally, the SourceEnd algorithm should utilize all collaborating ASes that are the closest to the attacking sources. However, facing the maximum available ASes constraint $Dmax$, the SourceEnd algorithm will prioritize collaborating ASes and only select the ASes that can filter the most DDoS traffic. We describe the algorithm from high level as follows. First, SourceEnd locates the initial defense locations as potential defending ASes, i.e., the ASes that are closest to the attackers. Then it sorts all potential ASes by the amount of traffic each AS can filter. It then adds the top AS to the selected ASes list ($S$), and removes it from the potential ASes list. The algorithm will repeat the previous two steps until $|S| \geq Dmax$, or all flows can be filtered, or there are no ASes available.

### C. StrategicPoints Algorithm

The StrategicPoints algorithm, different from PushBack and SourceEnd algorithms, employs ASes based on both traffic and topological information, and tries to deploy defenses at strategic locations inside the network instead of at edges. Although PushBack algorithms can utilize a small number of ASes close to the victim to cover most of the DDoS traffic, it suffers from potential heavy pollution. Similarly, while SourceEnd algorithm can have minimal pollution on the Internet, it requires a large number of participating ASes to cover the source ASes. The StrategicPoints algorithm, selects set of ASes that sits on all attack AS paths, as far into the Internet as possible, to achieve a balance between PushBack and SourceEnd, and target high effective defense with low pollution and low leakage.

The basic idea of StrategicPoints is to find the ASes that are in strategically important locations in terms of forwarding attack traffic to the victim. Here, we believe the most critical

ASes are the participating ASes that 1) observe the most traffic; 2) together consist of a topological cut for all the attack traffic toward the victim; and 3) are closer to the sources if possible. The algorithm first collects the statistics of the attack traffic distribution among the ASes. It begins by adding the victim AS to the set of selected defending ASes, $B$. Then, it builds up $B$ by continuously replacing ASes in this set with their direct upstream ASes until there are no more available ASes in the defense pool or $Dmax$ is surpassed. At each step, we prioritize the ASes by the amount of attack traffic they received, thus pushing the line of defense from the heavily impacted ASes first until all attack traffic is filtered. By doing so, the algorithm maintains a set of selected ASes that together consist of a cut of all attack traffic paths toward the victim, and in the meantime, also moves the defense further toward the attack sources. With sorted selection at each step, the algorithm also balances the defense workload (the amount of traffic an AS needs to filter) among the defending ASes.

To summarize, traditional *PushBack* and *SourceEnd* algorithms work by deploying traffic filtering rules directly at or close to the victim or the attack sources correspondingly. StrategicPoints algorithm, on the other hand, selects critical ASes hops away from the victim that carry the most of the attack traffic and cover all critical paths. It is able to push the defense far to sources for the heavily congested links, and maintaining close-to-victim defending ASes for links that are not congested at the moment. This feature further allows the StrategicPoints algorithm to better handle dynamic attacks with shifting attack sources without overfitting towards one attack pattern at any given moment.

## V. Simulation Setup

To quantitatively investigate different DDoS defense strategies, we built a simulation framework that simulates the Internet, DDoS attacks, and in-network DDoS defense. The simulation follows the model described in Section III and implements the DDoS defense algorithms described in Section IV, with the following details.

### A. Internet Topology

We use the full routing table dump data obtained from RouteViews [29] in August 1st 2018 to build the Internet topology. A full routing table contains the AS-level paths toward all the reachable IP prefixes, which reflects the full Internet topology from the perspective of a route collector. By combining the topologies obtained from all 22 RouteViews collectors, we aim to cover as many AS-level links as possible.

### B. Large-Scale DDoS Attacks

We use two real-world large-scale DDoS attack traces for this study (Table II). One real-world attack trace is a DDoS attack collected by CAIDA in 2007 [30], and the other is a DDoS attack toward an RADB service collected by Merit in 2016 [31]. Both traces involve thousands of attack sources originated from thousands of ASes and can be used to evaluate DDoS defense under real conditions.

| Trace name | # of sources | # of source ASes |
|---|---|---|
| CAIDA-2007 [30] | ∼4,700 | ∼1,400 |
| Merit-2016 [31] | ∼2,300 | ∼1,300 |

TABLE II: DDoS attack traces used in simulation.

We use the route collectors in RouteViews [29] as the victims of DDoS attacks. We also assume that the AS-level paths are symmetric, i.e., the AS path from a victim to an attack source is the same as the AS path from the attack source to the victim, allowing us to build attack flows using the AS paths in the routing tables. Note as this assumption is not always true on the Internet, our simulation thus derive results from a constructed topology that is similar to but not exactly the same as a real Internet topology.

### C. In-network DDoS Defenses

In this simulation we assume all ASes are able to participate in the defense and a DDoS defense algorithm chooses the actual defense ASes. The simulation begins with the scenario of no defense. Then we apply the algorithms introduced in Section IV to decide ASes to be used for defending against the attack. For each simulation run, we vary the $Dmax$ and $Rmax$ restriction, i.e., the maximum number of ASes available for defense, and the maximum number of defense rules available at each AS (for simplicity this is the same across all ASes). At the end of each run, the simulation framework collects the simulation results, including values of the metrics introduced in Sec. III): *leakage* and *pollution*.

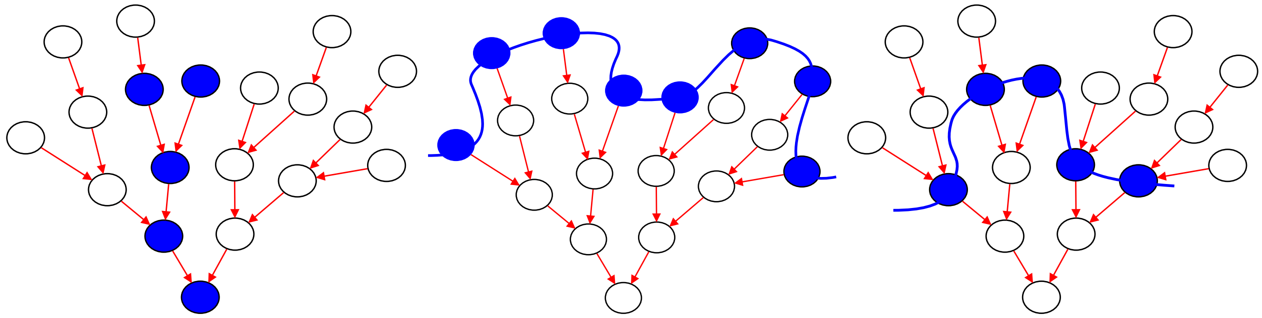
## VI. Evaluation

We compared different DDoS defense algorithms in terms of their resource requirements, time to respond to attacks, and resiliency against the intelligent DDoS attackers. Below we report and analyze the results. Refer to Section V-B for the datasets we use in the evaluation.

### A. $Leakage$

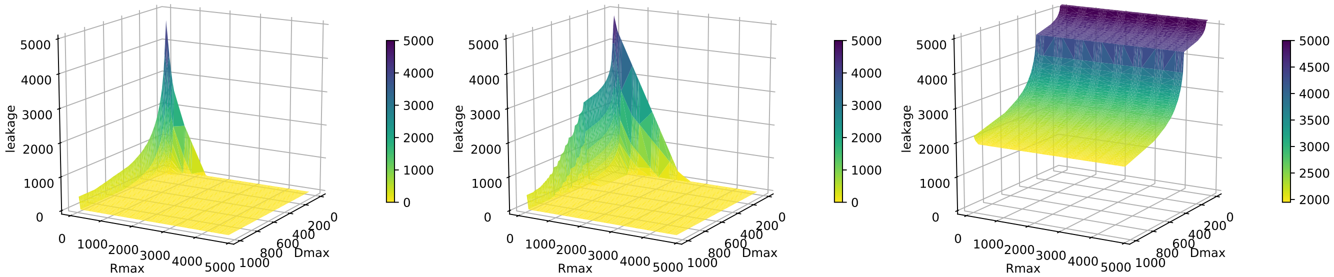
First and foremost, DDoS victims care most about the amount of DDoS traffic still leaking towards them after defense is deployed on the Internet. As we previously defined, we measure $leakage$ under different resource constraints to evaluate each algorithm's effectiveness in filtering out the attack flows toward the victim. Specifically, for each algorithm, we run simulations for various combinations of $Rmax$ (i.e., the maximum number of filtering rules each AS has) and $Dmax$ (i.e., the maximum number of defending ASes the defense can utilize).

Figure 3 shows the $leakage$ simulation results for three algorithms using the CAIDA-2007 dataset, with $Rmax$ ranging from 10 to 5000 (about maximum number of sources of the whole attack) and $Dmax$ ranging from 10 to 1000. It is clear at within the range of the resource limitations in the simulation, both PushBack and StrategicPoints outperform SourceEnd on reducing $leakage$. StrategicPoints performs slightly worse than PushBack when resource limitations are low, and both algorithms perform equally well when resource limitations are high.



(a) PushBack (b) SourceEnd (c) StrategicPoints

Fig. 2: Examples of three in-network DDoS defense algorithms.



(a) PushBack (b) StrategicPoints (c) SourceEnd

Fig. 3: Resource requirements – *leakage*

### B. Pollution

Since both PushBack and StrategicPoints perform similarly in reducing *leakage*, we further examine their performance in reducing the overall *pollution* on the Internet.

Figure 4 shows simulation results for measuring the *pollution* for the three algorithms. It is clear that as $Rmax$ increases, the *pollution* reductions by PushBack become less effective. In fact, given a high enough $Rmax$ PushBack utilizes only the victim AS to defend against the attack, leaving a large portion of *pollution* unhandled. This is undesirable when the attack *pollution* is so high that it could not only cause extra burden on the ISPs to forward traffic, but also trigger traffic congestion on the links close to the victim. On the contrary, StrategicPoints although has similar effectiveness in reducing *leakage*, it also greatly reduces the *pollution* caused by the attack on the Internet, thus further reducing the chances of link congestion.

To show the comparison more clearly, we also fix $Rmax$ while increasing $Dmax$. As shown in Figure 5, both PushBack and StrategicPoints perform well in reducing *leakage*, while StrategicPoints outperforms PushBack and reduces *pollution* significantly.

### C. Effectiveness against dynamic DDoS attack

One other important metric for an effective DDoS defense algorithm is how effective it is at handling DDoS attacks with dynamic attack sources. Specifically, we want to study how the algorithms perform on a real-world attack trace where there is consistently more attack sources joining and leaving the attack.

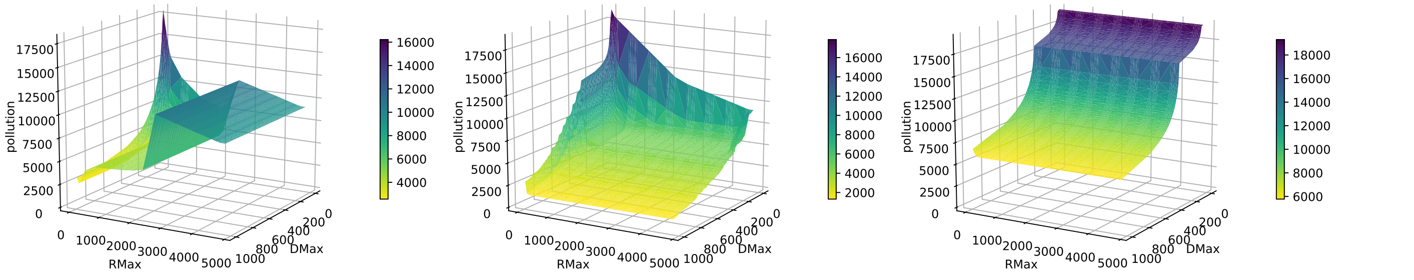
We run three algorithms against the Merit-2016 dataset, using the first 2000 flows (about 15% of the total unique sources) as the training set for locating defending ASes, and then use the same set of ASes for the result of the attack. As the new unique attack sources join the attack, the defending ASes' rules space will be gradually filled, and eventually will not be able to handle any more new attack flows. Results with larger number of flows shows similar results, and thus are omitted.

Figure 6 shows that PushBack is very ineffective in dealing with dynamic attack scenarios because it selects ASes that are just able to handle the training flows, allocating no space for new flows and changes of the attacks. StrategicPoints and SourceEnd both perform much better in handling new flows due to the fact that they both allocate more overall rule space for defense, allowing defenders to have more flexibility in handling dynamic attacks.

### D. Summary

In this section, we evaluated the performance of the three algorithms using real-world traces, and summarize the key points as follows.

On reducing the *leakage* of a defense, both StrategicPoints and PushBack perform similarly well when using a reasonable amount of resources. When $Dmax$ and $Rmax$ are low, PushBack performs slightly better. SourceEnd, on the other



(a) PushBack

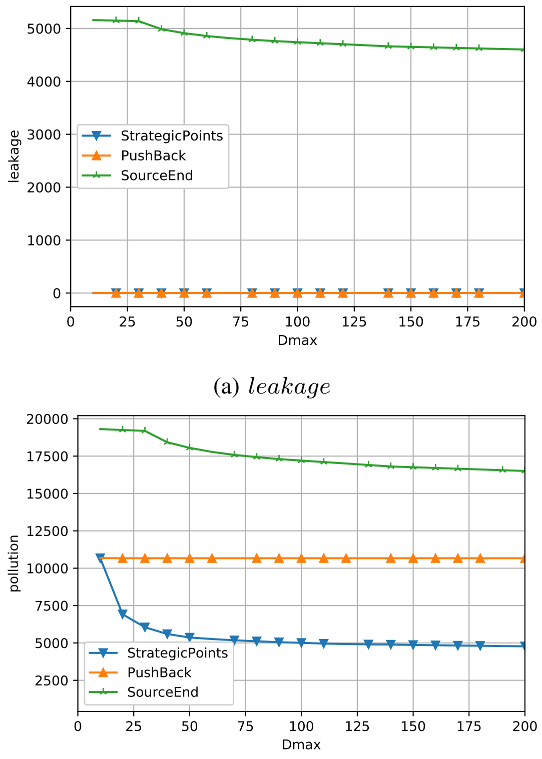
(b) StrategicPoints

(c) SourceEnd

Fig. 4: Resource requirements – *pollution*

| algorithm | leakage | pollution | dynamic attack resiliency | key resource | when to use |
|---|---|---|---|---|---|
| PushBack | low | high | medium | Rmax | very low $Dmax$ or $Rmax$ |
| SourceEnd | high | medium | low | Dmax | $Dmax$ close to total source ASes |
| **StrategicPoints** | **low** | **low** | **high** | **Dmax** | **all other cases** |

TABLE III: Algorithms performance summary and usage suggestion.



(a) *leakage*

(b) *pollution*

Fig. 5: Resource consumption with $Rmax = 3000$

hand, is only viable when $Dmax$ is very large and close to the total amount of attack source ASes.

Reducing *pollution* is also a very important task in that high *pollution* could cause link congestion, which would still directly affect the quality of service for all the traffic toward the victim. On this aspect, StrategicPoints significantly outperforms PushBack due to the algorithm's tendency to deploy rules farther into the Internet thus closer to the sources. When $Dmax$ is very large and close to the total number of source ASes, SourceEnd could also achieve low *pollution*.

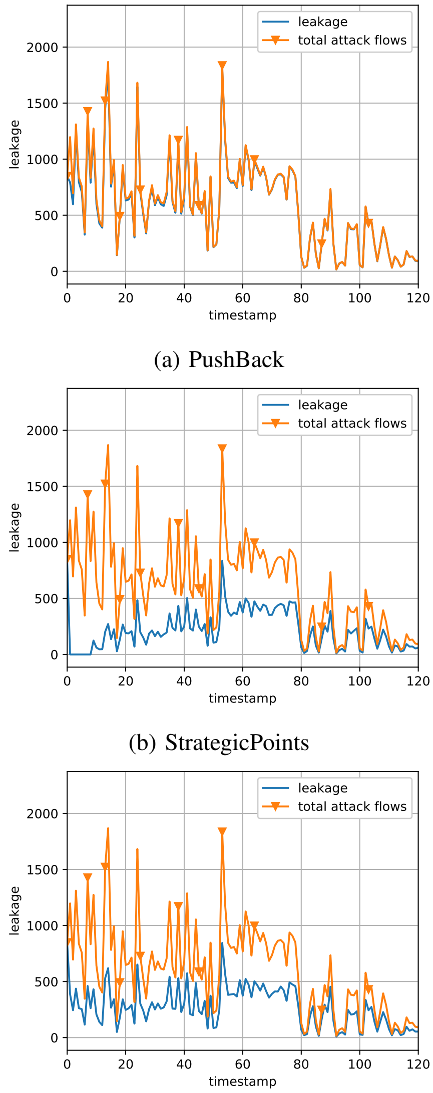
When facing dynamic attacks where attack sources join and leave during the attack, it is important that the defense algorithm is flexible and allows deployment of new filtering rules. Due to its design, PushBack always selects a number of ASes that are "just enough" for the defense, thus leaving little or no extra rule space for new filtering rules to be deployed. On the contrary, StrategicPoints and SourceEnd select defending ASes in a greedy approach, and always fully utilize the available $Dmax$. As a result, ASes selected by either algorithms would have more available rule space to spare for potential future defense rules.

Table III summarizes the key attributes of the three algorithms. Based on these results, we believe that PushBack is only suitable when $Dmax$ an $Rmax$ are very low; in all other cases, StrategicPoints can perform best in terms of reducing *leakage* and *pollution*.

## VII. Open Issues

### A. Flow volume

In evaluating the three algorithms, we currently consider the flows have the same weight, and volume information is not included. In fact, after examining the volume information for each sources in the two attack traces, we observe similar volume for the sources with no significant differences among them. However, our simulation framework is able to incorporate traffic volume information, and we plan to expand our evaluation on that direction in the future.



(a) PushBack

(b) StrategicPoints

(c) SourceEnd

Fig. 6: *Leakage* plot for Merit-2016 traces with strategies using 2000 sources in the first second to locate defending ASes, with both $Dmax = 500$ and $Rmax = 500$

### B. IP spoofing

For any well-designed DDoS defense system, it needs to consider how to protect victims from spoofed traffic. Although we evaluated the three defense algorithms in non-spoof DDoS attack traces, we plan to evaluate them in scenarios with IP spoofing as our future work.

### C. In-network over edge defense solutions

Edge DDoS defense solutions in general are easy to deploy, but costly in operations (i.e., purchasing large bandwidth links or high performance switches). Edge solutions also cannot prevent large-scale volumetric attacks that congest the inbound links of defense networks. For these reasons, we believe in-network DDoS defense solutions are more suitable for defending against large-scale DDoS attacks of the future.

### D. Deployment for evaluation

This study serves as a pilot study for further real-world evaluation. Ideally, our next step is to conduct real-world deployment for evaluation, and collect results from real-world traffic analysis. However, such evaluation presents the following challenges: 1) it is very difficult (if not impossible) to deploy a large-scale study platform to achieve the scale simulated in this paper; 2) it also requires generating a large amount of traffic from multiple vantage points and capturing the traffic passing through ASes on each path, which would result in both high hardware and software requirements at the deployment site; We plan on extending our evaluation to a smaller-scale real-world deployment as our next step.

## VIII. Conclusions

In this paper, we modeled and evaluated different multi-point, in-networking DDoS traffic filtering algorithms. After defining a general model for describing DDoS attacks and defense, we categorized existing in-network DDoS filtering algorithms into two basic types, i.e., PushBack and SourceEnd, that cover the majority of the state-of-the-art research and practice on in-network DDoS filtering. We then introduced StrategicPoints algorithms that outperform PushBack and SourceEnd algorithms in most cases. We designed a simulation framework as a common platform to evaluate major large-scale DDoS attack scenarios and defenses, and evaluated and compared the three types of multi-point, in-network DDoS filtering algorithms in terms of their capability in reducing the DDoS traffic leakage to the victim and the pollution to the whole Internet, as well as their resiliency against dynamic DDoS attacks. With real-world, Internet-scale DDoS attack traces, our evaluation results show that when having a low number of ASes for defense, PushBack performs slightly better than StrategicPoints and significantly better than SourceEnd in terms of reducing DDoS traffic leakage. As the number of available filtering ASes increases, or the number of available rules space increases, StrategicPoints becomes as effective as PushBack on reducing leakage and significantly outperforms PushBack on reducing pollution caused by DDoS attacks. Finally, we summarized the algorithms and suggested what algorithms to adopt for DDoS defense based on the resource availability of the victim.

## IX. Acknowledgment

This project is the result of funding provided by the Science and Technology Directorate of the United States Department of Homeland Security under contract number D15PC00204. The views and conclusions contained herein are those of the authors and should not be interpreted necessarily representing the official policies or endorsements, either expressed or implied, of the Department of Homeland Security or the US Government. We further thank Ann Cox and anonymous reviewers of this paper for their comments.

# References

[1] ArborNetworks, "Worldwide Infrastructure Security Report," 2016. [Online]. Available: https://www.arbornetworks.com/images/documents/WISR2016_EN_Web.pdf

[2] CloudFlare, "400Gbps: Winter of Whopping Weekend DDoS Attacks," 2016. [Online]. Available: https://blog.cloudflare.com/a-winter-of-400gbps-weekend-ddos-attacks/

[3] C. Kolias, G. Kambourakis, A. Stavrou, and J. Voas, "DDoS in the iot: Mirai and other botnets," *Computer*, vol. 50, 2017.

[4] B. Krebs, "Krebsonsecurity hit with record DDoS," *KrebsOnSecurity, Sept*, vol. 21, 2016.

[5] S. Kottler, "February 28th ddos incident report," *GitHub Engineering*. [Online]. Available: https://githubengineering.com/ddos-incident-report/

[6] K. Pagiamtzis and A. Sheikholeslami, "Content-addressable memory (CAM) circuits and architectures: A tutorial and survey," *IEEE Journal of Solid-State Circuits*, vol. 41, 2006.

[7] "CAT 6500 and 7600 Series Routers and Switches TCAM Allocation Adjustment Procedures." [Online]. Available: http://www.cisco.com/c/en/us/support/docs/switches/catalyst-6500-series-switches/117712-problemsolution-cat6500-00.html

[8] G. C. Oikonomou, J. Mirkovic, P. L. Reiher, and M. Robinson, "A Framework for a Collaborative DDoS Defense," in *ACSAC*, vol. 6, 2006.

[9] R. Mahajan, S. M. Bellovin, S. Floyd, J. Ioannidis, V. Paxson, and S. Shenker, "Controlling high bandwidth aggregates in the network," *ACM SIGCOMM Computer Communication Review*, vol. 32, 2002.

[10] Z. Liu, H. Jin, Y.-C. Hu, and M. Bailey, "MiddlePolice: Toward enforcing destination-defined policies in the middle of the Internet," in *ACM SIGSAC Conference on Computer and Communications Security*, 2016.

[11] J. Zheng, Q. Li, G. Gu, J. Cao, D. K. Yau, and J. Wu, "Realtime DDoS Defense Using COTS SDN Switches via Adaptive Correlation Analysis," *IEEE Transactions on Information Forensics and Security*, 2018.

[12] R. Sahay, G. Blanc, Z. Zhang, and H. Debar, "Towards Autonomic DDoS Mitigation using Software Defined Networking," *Workshop on Security of Emerging Networking Technologies*, 2015.

[13] M. S. Kang, V. D. Gligor, and V. Sekar, "SPIFFY: Inducing Cost-Detectability Tradeoffs for Persistent Link-Flooding Attacks," in *NDSS*, 2016.

[14] S. K. Fayaz, Y. Tobioka, V. Sekar, and M. Bailey, "Bohatei: Flexible and elastic DDoS defense," in *24th USENIX Security Symposium*, 2015.

[15] K. Kalkan and F. Alagöz, "A distributed filtering mechanism against DDoS attacks: ScoreForCore," *Computer Networks*, vol. 108, 2016.

[16] D. K. Yau, J. Lui, F. Liang, and Y. Yam, "Defending against distributed denial-of-service attacks with max-min fair server-centric router throttles," *IEEE/ACM Transactions on Networking*, vol. 13, 2005.

[17] J. François, I. Aib, and R. Boutaba, "FireCol: a collaborative protection network for the detection of flooding DDoS attacks," *IEEE/ACM Transactions on Networking*, vol. 20, 2012.

[18] K. Argyraki and D. R. Cheriton, "Active Internet Traffic Filtering: Real-time Response to Denial-of-service Attacks," in *USENIX Annual Technical Conference*, 2005.

[19] C. Papadopoulos, R. Lindell, J. Mehringer, A. Hussain, and R. Govindan, "COSSACK: Coordinated Suppression of Simultaneous Attacks," *DARPA Information Survivability Conference and Exposition*, vol. 2, 2003.

[20] X. Liu, X. Yang, and Y. Lu, "To filter or to authorize: Network-layer DoS defense against multimillion-node botnets," *ACM SIGCOMM Computer Communication Review*, 2008.

[21] J. Mirković, G. Prier, and P. Reiher, "Attacking DDoS at the source," in *10th IEEE International Conference on Network Protocols*, 2002.

[22] K. Argyraki and D. R. Cheriton, "Scalable network-layer defense against internet bandwidth-flooding attacks," *IEEE/ACM Transactions on Networking*, vol. 17, 2009.

[23] F. Huici and M. Handley, "An Edge-to-Edge Filtering Architecture Against DoS," *ACM SIGCOMM Computer Communication Review*, vol. 37, 2007.

[24] A. D. Keromytis, V. Misra, and D. Rubenstein, "SOS: An architecture for mitigating DDoS attacks," *IEEE Journal on Selected Areas in Communications*, vol. 22, 2004.

[25] D. G. Andersen, "Mayday: Distributed Filtering for Internet Services," in *USENIX Symposium on Internet Technologies and Systems*, vol. 4.

[26] S. T. Zargar, J. Joshi, and D. Tipper, "A survey of defense mechanisms against distributed denial of service (DDoS) flooding attacks," *IEEE Communications Surveys and Tutorials*, vol. 15, 2013.

[27] R. Moskowitz and J. Xia, "DOTS over GRE," Internet Engineering Task Force, Internet-Draft draft-moskowitz-dots-gre-00, 2016, work in Progress. [Online]. Available: https://tools.ietf.org/html/draft-moskowitz-dots-gre-00

[28] G. Huston, P. Smith, and T. Bates, "CIDR Report," https://www.cidr-report.org/as2.0/, last accessed on 2019-01-28.

[29] University of Oregon, "Route Views project," 2005. [Online]. Available: http://www.routeviews.org

[30] P. Hick, E. Aben, K. Claffy, and J. Polterock, "The CAIDA DDoS Attack 2007 Dataset," 2007.

[31] I. Merit Network, "A DDoS event against the RADb service," 2016. [Online]. Available: https://www.impactcybertrust.org/dataset_view?idDataset=576