Circumventing Crossfire Attacks via Limited-Access Cloud Paths

Cody Doucette*[1], Michel Machado†, Qiaobin Fu‡, John W. Byers§
*Cloudflare, †Digirati, ‡Google, §Boston University,
[1]Work completed while at Boston University

ABSTRACT

Over the past decade, large-scale link flooding attacks (LFAs) have gone from hypothetical threat to alarming reality. Using the massive botnets available in today's Internet ecosystem, adversaries can construct stealthy attacks to deny service to entire geographical regions by flooding links upstream of the intended target. Unfortunately, none of the proposed solutions to the LFA problem have provided a comprehensive and deployable defense. In this short paper, we propose equipping victims with a new topological advantage in the fight against LFAs by using *limited-access* cloud paths as a means to route around flooded links. Supported by a real-world feasibility study using Amazon cloud nodes and the CAIDA Periscope, we show that these cloud paths contain few of the links that attackers would target when constructing an LFA, mitigating harmful effects and driving up the cost to launch a successful attack.

1 INTRODUCTION

Traditional distributed denial of service (DDoS) attacks target the victim directly, either by flooding the CPU, memory, or destination network of the target. When experiencing a sustained spike in resource utilization, victim servers become overwhelmed, and legitimate clients are denied timely access to services. Alternatively, adversaries can shift the locus of the attack upstream by flooding a carefully calculated set of upstream links instead of aiming the attack traffic directly at the target. For an attacker, the main benefit of separating the intended target from the convergence point of the attack traffic is to evade attack detection and mitigation services. If the locus of the attack is far enough away, e.g., in a different network, the victim will not be able to measure the attack directly and may have very few defensive options.

Such *link flooding attacks* (LFAs) have been described in various forms, and the literature has proposed many methods of detecting and defending against them (Section 2). Recently, more attention has been given to LFAs for two reasons. First, sophisticated formulations have been described that can be performed stealthily and at scale [13, 23] (Section 3). Second, LFAs have been seen in the wild (Internet) in at least two documented events [2, 19, 20]. However, all of the proposed defensive methods only partially address, or make simplifying assumptions about, the LFA problem. For example, some of these solutions assume the presence of an alternative Internet architecture, or limit the scope of the attack to only affect links in the same network as the intended target.

To provide a more comprehensive solution, we propose using *limited-access cloud paths* (LAPs) [9–11] to mitigate LFAs (Section 4). During the surveillance phase of an LFA, an adversary conducts reconnaissance of the network topology to compute a set of target links that carries a high proportion of traffic to the target area [23]. LAPs enable victim networks to circumvent those target links by routing along alternative paths whose carrying capacity is unobserved by attackers, so that the links within those paths would likely not be used in the attack construction.

Limited-access cloud paths are so-named because they limit access to the alternative paths in two ways. First, the paths are not apparent to surveillance via methods like traceroute, so the adversary cannot build an accurate map of the network topology to inform their selection of target links. Second, the paths have a high degree of independence from the public Internet paths that may be under attack, allowing traffic to circumvent congestion.

To show the feasibility of this approach, we performed a measurement study to evaluate whether LAPs allow traffic to circumvent target links during LFAs (Section 5). Using the CAIDA Periscope [8], Amazon cloud nodes, and public Internet servers, we computed the target links for, but did not launch, a *Crossfire* attack, which is an especially stealthy and scalable LFA [13]. We compared the set of target links computed from the real world topology with the set of links from LAPs, and found that LAPs greatly reduce the likelihood that traffic crosses target links, mitigating the effects of the LFA. Moreover, we found that even when cloud paths do contain target links, those target links do not necessarily carry high proportions of traffic, thus they may be overlooked as flooding targets by the adversary's selection algorithms.

2 RELATED WORK

We group the prior work on mitigating Crossfire attacks into the following four categories.

Link inspection-based solutions: CoDef [15] proposes collaborative rerouting that requires the presence of a path identifier mechanism and inter-domain collaboration, both of which are not available in the current Internet architecture. SPIFFY [12] proposes mechanisms to force attackers to

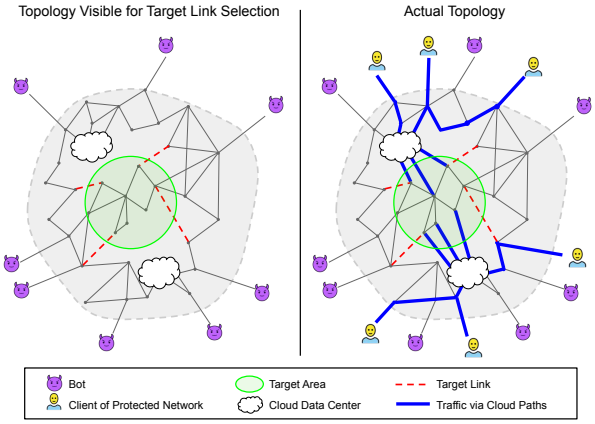

Figure 1: On the left, an overview of the Crossfire attack based on diagram from [13]. On the right, the actual topology of the destination protected with cloud paths.

either increase their attack cost or allow themselves to be differentiated from legitimate traffic. However, this solution assumes that the target links are within the jurisdiction of the deploying network. RADAR [30] is an SDN-based approach to detect and rate-limit link flooding attacks within a single network, however, it has a high false positive rate and brings heavy measurement overhead to the network.

Traffic isolation-based solutions: SIBRA [1] aims to use bandwidth isolation and reservation mechanisms to guarantee a minimum level of service for critical flows. However, it requires novel network architectures such as SCION [29]. Most recently, Nyx [22] proposes to use BGP poisoning to achieve isolation of traffic and route around congested links. However, it is not feasible in practice [24].

Hidden topology-based solutions: There are multiple mechanisms that could be used to obfuscate the topology from the attacker, including redirecting `traceroute` packets to a virtual topology [14], inserting links in `traceroute` replies at critical points [25], and removing potential target links from replies [5]. Similarly, NetHide [18] intercepts and modifies probes in real time. However, NetHide faces deployment hurdles (e.g., requiring programmable routers).

Machine learning-based solutions: Some proposals use machine learning techniques to detect and mitigate Crossfire attacks [3, 21, 27]. However, they typically have low scalability and thus suffer from slow response times.

Overall, these proposals are either partial solutions to Crossfire or are unrealistic in practice. Similar to NetHide [18], LAPs proactively defend against Crossfire attacks by hiding the internal network topology with tunneling (Section 4).

3 CROSSFIRE ATTACK PRIMER

To construct a Crossfire attack, an adversary selects a *target area* as the victim. The target area is a geographic region, and does not necessarily cleanly map to one or more autonomous systems or networks. Example target areas could be a network enclave within an organization, a university network, or the Iberian peninsula.

To deny service to the target area, the adversary seeks to find a set of *target links* that are geographically close to, and that carry a high proportion of traffic to, the target area. If this set of target links is chosen carefully according to these properties, then launching a DDoS attack to saturate these links will cut off a high percentage of traffic to the target area. The saturated target links are ideally upstream of the target area, so that the victim cannot directly detect and mitigate the attack. An overview of the Crossfire attack is shown on the left side of Figure 1.

In order to calculate the set of target links, the adversary finds a set of public servers in the target area as well as a set of *decoy servers* in the vicinity of the target area. We call this set of public and decoy servers D. The adversary then selects a set of bots, B, and sends multiple `traceroute` probes from every $b \in B$ to every $d \in D$. The result of these probes is a set of paths P, each composed of links in the set of all encountered links, L. The adversary then extracts the *persistent* links $L' \subset L$ from these paths: those that always occur in the `traceroute` output across multiple trials for each pair (b, d). The set L' composes a network-layer *link map* (grey links in the left side of Figure 1), which also serves as the set of target link candidates.

To select the target links $T \subset L'$, the adversary analyzes the link map to find the links that would be most useful to flood with an attack. To calculate this, the metric used by the adversary is the *flow density* for each link $l \in L'$, which is defined as the number of paths $p \in P$ that cross l. Links with a high flow density are likely to carry the highest amount of traffic, and are therefore the highest value links for an attacker to flood.

After selecting the set of target links, the adversary gives instructions to each bot, consisting of the set of decoy servers the bot should send to, as well as the rate at which it should send to each server. Bots use low-rate, legitimate-looking traffic that is destined for real, public servers, so that no individual flow looks out of the ordinary. But, in aggregate, the bots are coordinated in such a way that the target links are flooded. To circumvent defensive strategies and maintain attack persistence, the adversary can periodically rotate the set of target links that the bots flood, and/or rotate the set of bots performing the flooding.

The resulting attack is very difficult to detect and mitigate. Since the attack traffic does not necessarily reach the destination server, signature-based detection mechanisms that analyze traffic are not useful. Furthermore, since the attack traffic uses low-rate flows with legitimate-looking traffic to

legitimate destinations, an IDS would be unlikely to flag traffic from Crossfire bots as anomalous anyway. Even if a victim could directly detect the attack, there is no mechanism for networks to protect themselves if the target links are in a different network, and therefore outside of the control of the network operator. One option would be to forge cooperation between the victim network and upstream networks to try to filter traffic, but this approach is not administratively feasible in the current Internet architecture.

Recorded Crossfire attacks. Malicious actors have already launched attacks similar to Crossfire in the Internet. In 2013, the anti-spam company Spamhaus was the victim of a 300 Gbps DDoS attack. The locus of the attack was the set of links in the Internet exchange that Spamhaus traffic crossed on its way to Cloudflare, who provided DDoS protection services to Spamhaus [2, 20]. The attackers were able to find the IP addresses of peers in the Internet exchange points (IXP), and used those IP addresses to saturate some of the IXP links. Two years later, in 2015, ProtonMail was the target of a week-long 50 Gbps attack [19]. The attackers set their sights on the links in the upstream ISPs that connected to the ProtonMail datacenters, and attacking them brought the datacenters offline. These incidents demonstrate that large-scale link attacks are not only feasible, but can be very damaging in practice.

4 LIMITED-ACCESS PATHS TO THE RESCUE

We advocate using limited-access paths (LAPs) to protect a network inside of a Crossfire target area. LAPs neutralize Crossfire attacks by removing the ability of *side traffic* to affect the protected network. Side traffic is any traffic whose destination is not within the protected network, but still affects the target area (e.g., attack traffic to decoy servers). This side effect is possible in the open Internet due to the fact that the core infrastructure is broadly shared between stakeholders, and therefore networks that are topologically close often share the same fate during an LFA. However, if a solution offers a way to receive the traffic to a destination close to the sources of the traffic, and to forward it to the physical destination using links that are not broadly shared in the Internet, it is possible to circumvent Crossfire attacks.

The incoming traffic to a protected destination can be captured close to its source at *entry points*, such as Internet exchanges, points of presence, and cloud providers. Once captured close to its source, the traffic can be forwarded to the physical destination employing LAPs. A LAP is any path built on links that requires a form of subscription for access. This includes, for example, MPLS tunnels offered by transit providers, any form of private links (e.g., dark fiber), and the paths built over the private links available on clouds.

Along the LAP, topology obfuscation can be applied to prevent surveillance by an adversary. For example, tunneling packets between the entry point and the protected network would prevent `traceroute` run from outside of the LAP from seeing the internal characteristics of the path, preventing links in the path from being selected as a target. Other methods for topology obfuscation (see those described in §2) could also be applied.

Cost is the main driver while choosing between the different forms of LAPs. If cost were not a concern, private links such as dark fibers would be the best option, since they are only accessible for traffic to the protected destination, and are therefore completely inaccessible to side traffic. However, for most scenarios, the ideal choice of LAP has both low cost and low accessibility for side traffic. Cloud paths, i.e., the paths that start at cloud resources such as VMs and end at the protected destination, provide both properties. They are low cost because they use shared infrastructure within a cloud provider, and are mostly not accessible to side traffic because cloud paths are largely independent from, and more reliable than, public Internet paths [10, 11]. In fact, as we show in §5, their inaccessibility allows traffic to protected destinations to circumvent target links. We therefore advocate that cloud paths are an economical and effective choice of LAP, and base our measurements on them.

Attackers could gain access to cloud paths by placing bots in clouds, but doing so would require contracting cloud services, which would raise attack cost. Alternatively, attackers could co-opt vulnerable systems, but systems deployed on clouds are typically more secure than user machines (e.g., Shielded VMs in Google Cloud [4]), so counting on vulnerable systems to deploy a Crossfire attack on the cloud significantly raises the bar to deploy these attacks.

The right side of Figure 1 summarizes our solution. Client requests of the protected network are routed to the closest cloud data center, and, from there, are forwarded to the destination using cloud LAPs. Since the figure depicts a Crossfire attack, the bots are not routed to the cloud paths because they target decoy servers instead of the protected destination. Some of the clients of the protected network may be bots, so a protection system to deal with floods, amplifications, and other forms of direct attacks is still necessary.

Finally, to guarantee that attackers cannot bypass the entry points of traffic, each entry point must announce the network prefixes of the destination (e.g., using BGP), thereby establishing an anycast network. The management of this anycast network and all other production demands, such as identifying and filtering direct attacks, can be met, for example using *anonymized* [17], a DDoS protection system. The key to this integrated solution, in contrast to prior work, is that our proposal can be fully deployed by a single network, requiring no coordination between ASes, for example.

In envisioning a deployment of LAPs to protect targets in a production setting, there is still design work ahead of us to address questions that would arise in practice, and which are beyond the scope of this short paper. A first question relates to economical selection of appropriate LAPs: how would a target (or a LAP service provider) best provision LAPs to provide a suitable amount of defense on a budget? Next, there is the question of activation: what would an appropriate trigger mechanism for activating LAPs look like? One would have to be concerned with false positive activation, as the act of "raising shields" incurs unwanted costs for the defender; moreover, a crafty attacker capable of triggering false positives cheaply could use that advantageously during surveillance! And finally, there is the question of whether static defenses are sufficient, or whether something with adaptation and moving targets might be necessary. This latter question is especially important to evaluate; we discuss our future evaluation plans in (§5.3).

5 LAP MEASUREMENT STUDY

To evaluate the effect of LAPs, we emulated the steps that an actual attacker would perform (§3). (We discuss ethical considerations relating to emulating steps a DDoS attacker would take at the end of this manuscript). Full details of our approach are available in [6].

Target area. For a target, we selected Boston, MA, USA.

Botnet. We selected approximately 100 Looking Glass nodes using CAIDA Periscope [8] as a proxy for the Crossfire botnet. We heuristically chose botnet nodes so as to maximize geographic diversity of bots around the world, yet ensuring each had high link persistence to the target area.

Decoy servers. To compose our set of decoy servers, we found approximately 60 public servers in the networks of institutions around the target area. Although [13] recommends finding such decoy servers through port scanning, we took a multi-step approach:

(1) We used Web-based subdomain enumeration tools to find publicly accessible subdomains from institutions around the target area.
(2) For each IP address corresponding to a subdomain, we performed a `traceroute` to that IP address in order to test whether probes are administratively blocked by the network.
(3) If `traceroute` probes succeed, then we looked for other servers in the same network using `nmap` [16] against the `/24` network prefix of the IP address found.
(4) We ran `traceroute` to each of the servers found by `nmap` to verify that they can be probed.

Cloud nodes. We emulated a LAP deployment using Amazon cloud nodes in Ohio (US), Frankfurt (EU-1), Paris (EU-2), Sydney (AU), Mumbai (IN), and São Paulo (BR). We refer to these cloud nodes as the set C. We chose nodes in these regions to reflect a global deployment. An overview of the distribution of bots and cloud nodes is shown in Figure 2. Note that LAPs were not actually deployed in this evaluation; rather we used cloud nodes to determine what the LAPs' likely impacts would be on a hypothetical attack.

Figure 2: Distribution of Looking Glass nodes for bots (blue pins) and cloud nodes (yellow icons).

Choice of parameters. In this study, we chose the well-connected technology hub of Boston as a target area. Our work will be completed in the future with a sensitivity analysis on the parameters, including target areas (e.g., cities, states, countries), botnets (e.g., size and distribution), and cloud providers (e.g., Microsoft Azure, Google Cloud).

5.1 Link Map Construction

With our testbed components selected, we performed a link map construction for a Crossfire attack.

Adversary behavior. In the traditional link map construction, an adversary probes both the decoy servers and the public servers in the target area. However, we assume that the adversary would be aware that the paths to the public servers are hidden, and would therefore avoid wasting resources on probing the public servers. Instead, the adversary would construct the link map solely using decoy servers in unprotected networks, hoping to find target links that might overlap with the hidden paths.

Link map construction. We ran three `traceroute` probes from each Looking Glass node to each decoy server, and used the results to construct a link map with 843 persistent links ($|L'| = 843$). From that set, we picked 20 target links ($|T| = 20$) using the procedure outlined in [13]. Figure 4 shows the top 180 links ranked by flow density, along with the 20 chosen target links highlighted in blue.

Cloud path measurements. To investigate whether cloud paths would enable client traffic to circumvent these target links, we then ran `traceroute` probes from our set C of Amazon cloud nodes to a sample of the target area servers $D' \subset D$ ($|D'| = 48$). We use a random sample to reflect the fact that only a subset of the target area servers are protected by the LAP defense. We then checked whether the $c \in C$ to $d \in D'$ paths contained the target links constructed by the Crossfire attack, the results of which we show next.

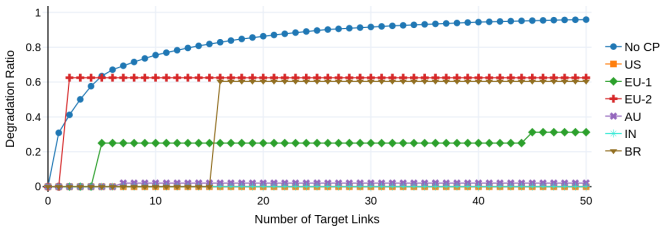

Figure 3: The degradation ratios when paths are forwarded directly to a target area and forwarded through clouds.

5.2 Cloud Path and Target Link Results

In comparing the set of target links with the links in cloud paths, our three main findings are:

(1) Cloud paths contain relatively few target links.
(2) Target links in cloud paths are not necessarily high-value links.
(3) Findings (1) & (2) hold for both a static and a rolling attack, in which target links change dynamically.

We now elaborate on these findings.

5.2.1 Cloud paths contain fewer target links. To evaluate the ability of cloud paths to circumvent target links, we use the *degradation ratio* as a key metric. The degradation ratio is the fraction of paths that contain target links (P_T) to the overall number of paths: $\frac{|P_T|}{|P|}$. Ideally, an attacker would maximize the degradation ratio while minimizing the size of the target link set, since more target links require more attacking bots, therefore raising attack cost.

Figure 3 shows the degradation ratios from our study for different sizes of T, $1 \leq |T| \leq 50$. The **No CP** series represents the degradation ratios for all (b, d) paths, which do not use cloud paths. In general, only a small set T is needed to force most paths through at least one target link: 75% of paths to the target area cross a target link when $|T| = 10$. This result is consistent with [13].

Figure 3 also shows degradation ratios for cloud paths (c, d). We found in almost all cases, the degradation ratios for cloud paths are less than those of non-cloud, (b, d) paths, and in some cases the difference is significant. For example, up to $|T| = 50$, zero paths from the US or the IN clouds crossed any target links, and only one path from AU did so. The EU (Paris) and BR cloud paths overlapped most with target links: over 60% of paths contained at least one target link in both cases, although it took $|T| > 15$ to reach this ratio for BR. Up to $|T| = 50$, only 30% of the EU (Frankfurt) paths contained a target link.

5.2.2 Target links in cloud paths are not necessarily high-value links. For an adversary, it is desirable to minimize the number of target links used to launch an attack. More target links can require more bots to make the attack successful, since Crossfire attacks use low-intensity flows to maintain attack persistence. However, we found that when cloud paths cross target links, they are not necessarily high value links in terms of flow density. To visualize this, refer to Figure 4, which shows all persistent links ranked by flow density. It also highlights the set of target links in blue ($|T| = 20$), and uses arrows to show which target links were crossed by paths from the various cloud nodes.

Paths from EU-2 (Paris), EU-1 (Frankfurt), and AU (Sydney) cross only a single target link in each case, but all cross fairly high-value target links: the second, fifth, and seventh target links by flow density, respectively. All paths from BR (São Paulo) also crossed only a single link, but this link was of lower value: the 16th. In fact, for any attack strength $|T| < 16$, zero of the paths from the BR cloud node would cross a target link. Therefore, Figure 4 shows that (1) even when cloud paths contain target links, they do not contain many of the links in the set, and (2) the paths do not necessarily contain high-value target links, which forces the adversary to attack more links and raises attack cost. We remind the reader that paths from the US (Ohio) and IN (Mumbai) clouds contained no target links, which is an ideal result for a LAP defense.

Note that we have highlighted the case $|T| = 20$ for space considerations, but the results hold for attack strengths up to at least $|T| = 50$. We chose $|T| = 20$ since the degradation ratio of non-cloud paths (b, d) is already high (>85%), and more target links do not significantly change the degradation ratios of (c, d) paths.

5.2.3 Cloud paths can circumvent rotating sets of target links. To avoid mitigation, adversaries can rotate their attack through disjoint sets of target links [13]. During the target link selection phase, each target link set is chosen after removing all links chosen in previous sets. Using this technique, we selected three sets of disjoint links, each of size 10. We then computed the degradation ratios relative to each set of target links. The degradation ratio is 15% for paths (c, d) crossing target link set 1, 6% for target link set 2, and 13% for target link set 3. These degradation ratios are significantly smaller than those for (b, d) paths, which were 75%, 70%, and 63% for the three target link sets, respectively.

	Degradation Ratio		
Cloud Nodes	*Set 1*	*Set 2*	*Set 3*
US	0%	17%	0%
EU-1	25%	19%	8%
EU-2	63%	0%	17%
AU	2%	0%	2%
IN	0%	0%	0%
BR	0%	0%	50%

Table 1: Deg. ratios of rotating target link sets, by cloud node.

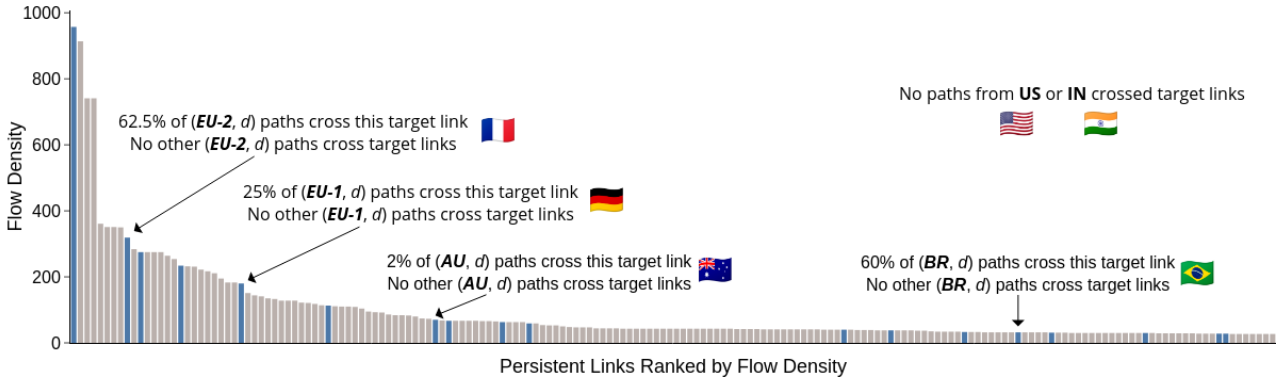

Figure 4: Target links (blue) cut by paths from the various clouds, where $|T| = 20$. Cloud paths do not necessarily cross many target links (at most one for each cloud), and may not cross high-value target links.

We also calculated the degradation ratios by cloud node with respect to each target link set, as shown in Table 1. These results highlight the fact that target link sets selected by the attacker can be "hit or miss" with cloud paths, i.e., some sets may contain a target link that is in a high percentage of paths, but other sets may contain no target links at all. For example, 50% of paths from the BR cloud node cross a target link in set 3, but no paths cross target links in sets 1 or 2.

We found that out of all (c, d) paths, 69.8% do not contain any target links from any of the three sets, 25.3% contain a target link from one set, 4.5% contain target links from two sets, and 0.3% contain a target link from all three sets. Additionally, out of the 30 target links selected between the three sets, only eleven of them ever occurred in a (c, d) path. From these results, we conclude that cloud paths reduce the degradation ratio for clients of the protected network during rolling Crossfire attacks.

5.3 Future Work on Evaluation

So far, we have discussed the strategy of provisioning limited-access cloud paths in advance of an attack. Arguably, this is a step in an ongoing arms race between attackers and defenders, as attackers could conceivably up their game by surveiling likely LAPs. But a further advantage of the cloud is its agility and flexibility in quickly deploying new nodes. Therefore a key question we ask, and expect to affirm in future evaluation work is: is it feasible to deploy new cloud nodes and paths during an attack as a mitigating maneuver?

Such a maneuver would be a *moving target defense* [7, 26, 28], which can range in complexity from simply changing the victim's IP address to shuffling clients to intermediate, cloud-based proxies. In our case, the "moving target" would be the set of cloud paths that carry traffic to the destination network. The adversary's goal is to pick a set of target links that cover as many of these paths as possible. Therefore, if the set of cloud paths could be changed, the adversary would be forced to change their strategy to find a new set of effective links. The effectiveness of the moving target defense will depend on the ability to maximize the independence of these paths from those that are already saturated.

6 CONCLUSION

As DDoS attacks increase in sophistication, they may attack resources far away from and not under the direct control of victims. Even sophisticated defenses, like moving target approaches, are not immune from this attack. One useful tool in the defensive arsenal is to use hardened infrastructure and backup resources that are also distributed. In this paper, we argue that cloud infrastructure in particular is especially helpful to this end, as it can be provisioned in advance, deployed on-demand, and may provide security guarantees beyond that of a victim's operational capabilities. Studying such a defense against the Crossfire attack specifically proves interesting, as it highlights the key issue of surveillance conducted prior to an attack, and counter-measures that a victim can take.

ETHICAL CONSIDERATIONS

The study of DDoS attacks is a setting where networking researchers must tread cautiously. The benefit of publishing novel attacks must clearly be weighed against the societal harm of possibly increasing the potency of genuine attacks. While our work is exclusively on DDoS defense, and thus may seem to be on solid ethical ground, it is worth reminding ourselves that methods providing benefits to victims also better inform would-be attackers. We also thought carefully about conducting network measurements "as if" we were conducting Crossfire surveillance on a target. We did so with a light touch, using only low-rate, low-volume traceroutes from vantage points available for public use. In so doing, we were reminded how innocuous-looking surveillance prior to this type of attack may appear to network operators.

REFERENCES

[1] Cristina Basescu, Raphael M. Reischuk, Pawel Szalachowski, Adrian Perrig, Yao Zhang, Hsu-Chun Hsiao, Ayumu Kubota, and Jumpei Urakawa. 2016. SIBRA: Scalable Internet Bandwidth Reservation Architecture. In *Network and Distributed System Security Symposium (NDSS '16)*.

[2] Peter Bright. 2013. Can a DDoS break the Internet? Sure... just not all of it. https://arstechnica.com/information-technology/2013/04/can-a-ddos-break-the-internet-sure-just-not-all-of-it. (2013).

[3] Yen-Hung Chen, Pi-Tzong Jan, Ching-Neng Lai, ChunWei Huang, Chih-Han Chang, and Yo-Cih Huang. 2020. Detecting linking flooding attacks using deep convolution network. In *Proceedings of the 2020 the 3rd International Conference on Computers in Management and Business*. 70–74.

[4] Google Cloud. 2021. Shielded VMs. https://cloud.google.com/shielded-vm. (2021).

[5] Xuyang Ding, Feng Xiao, and Man Zhou. 2020. Active Link Obfuscation to Thwart Link-flooding Attacks for Internet of Things. (2020). arXiv:cs.NI/1703.09521

[6] Cody Doucette. 2021. *An Architectural Approach for Mitigating Next-Generation Denial of Service Attacks*. Ph.D. Dissertation. Boston University.

[7] Xianjun Geng and Andrew B Whinston. 2000. Defeating distributed denial of service attacks. *IEEE IT Professional* 2, 4 (2000), 36–42.

[8] Vasileios Giotsas, Amogh Dhamdhere, and Kimberly C Claffy. 2016. Periscope: Unifying looking glass querying. In *International Conference on Passive and Active Network Measurement (PAM '16)*. 177–189.

[9] Osama Haq and Fahad R Dogar. 2015. Leveraging the power of cloud for reliable wide area communication. In *Proceedings of the 14th ACM Workshop on Hot Topics in Networks*. 1–7.

[10] Osama Haq, Cody Doucette, John W. Byers, and Fahad R. Dogar. 2020. Judicious QoS using cloud overlays. In *ACM Conference on emerging Networking EXperiments and Technologies (CoNEXT '20)*. 371–385.

[11] Osama Haq, Mamoon Raja, and Fahad R Dogar. 2017. Measuring and improving the reliability of wide-area cloud paths. In *International Conference on World Wide Web (WWW '17)*. 253–262.

[12] Min Suk Kang, Virgil D Gligor, Vyas Sekar, et al. 2016. SPIFFY: Inducing Cost-Detectability Tradeoffs for Persistent Link-Flooding Attacks.. In *Network and Distributed System Security Symposium (NDSS '16)*.

[13] Min Suk Kang, Soo Bum Lee, and Virgil D Gligor. 2013. The crossfire attack. In *IEEE Symposium on Security and Privacy (S&P '13)*. 127–141.

[14] Jinwoo Kim and Seungwon Shin. 2017. Software-Defined HoneyNet: Towards Mitigating Link Flooding Attacks. In *2017 47th Annual IEEE/IFIP International Conference on Dependable Systems and Networks Workshops (DSN-W)*. 99–100. https://doi.org/10.1109/DSN-W.2017.10

[15] Soo Bum Lee, Min Suk Kang, and Virgil D Gligor. 2013. CoDef: Collaborative defense against large-scale link-flooding attacks. In *ACM Conference on emerging Networking EXperiments and Technologies (CoNEXT '13)*. 417–428.

[16] Gordon Fyodor Lyon. 2009. *Nmap network scanning: The official Nmap project guide to network discovery and security scanning*. Insecure.

[17] Michel Machado, Cody Doucette, Qiaobin Fu, and John Byers. 2021. Gatekeeper: the first open-source DDoS protection system. https://github.com/AltraMayor/gatekeeper. (2021).

[18] Roland Meier, Petar Tsankov, Vincent Lenders, Laurent Vanbever, and Martin Vechev. 2018. NetHide: Secure and practical network topology obfuscation. In *USENIX Security Symposium (SEC '18)*. 693–709.

[19] Dan Patternson. 2015. Exclusive: Inside the ProtonMail siege: how two small companies fought off one of Europe's largest DDoS attacks. https://www.techrepublic.com/article/exclusive-inside-the-protonmail-siege-how-two-small-companies-fought-off-one-of-europes-largest-ddos/. (2015).

[20] Matthew Prince. 2013. The DDoS That Knocked Spamhaus Offline (And How We Mitigated It). https://blog.cloudflare.com/the-ddos-that-knocked-spamhaus-offline-and-ho/. (2013).

[21] Raihan Ur Rasool, Usman Ashraf, Khandakar Ahmed, Hua Wang, Wajid Rafique, and Zahid Anwar. 2019. Cyberpulse: A Machine Learning Based Link Flooding Attack Mitigation System for Software Defined Networks. *IEEE Access* 7 (2019), 34885–34899. https://doi.org/10.1109/ACCESS.2019.2904236

[22] Jared M Smith and Max Schuchard. 2018. Routing around congestion: Defeating DDoS attacks and adverse network conditions via reactive BGP routing. In *IEEE Symposium on Security and Privacy (S&P '18)*. 599–617.

[23] Ahren Studer and Adrian Perrig. 2009. The coremelt attack. In *European Symposium on Research in Computer Security (ESORICS '09)*. 37–52.

[24] Muoi Tran, Min Suk Kang, Hsu-Chun Hsiao, Wei-Hsuan Chiang, Shu-Po Tung, and Yu-Su Wang. 2019. On the feasibility of rerouting-based DDos defenses. In *IEEE Symposium on Security and Privacy (S&P '19)*. 1169–1184.

[25] Samuel T. Trassare, Robert Beverly, and David Alderson. 2013. A Technique for Network Topology Deception. In *MILCOM 2013 - 2013 IEEE Military Communications Conference*. 1795–1800. https://doi.org/10.1109/MILCOM.2013.303

[26] Sridhar Venkatesan, Massimiliano Albanese, Kareem Amin, Sushil Jajodia, and Mason Wright. 2016. A moving target defense approach to mitigate DDoS attacks against proxy-based architectures. In *IEEE Conference on Communications and Network Security (CNS '16)*. 198–206.

[27] Jorge Maestre Vidal, Ana Lucila Sandoval Orozco, and Luis Javier García Villalba. 2018. Adaptive artificial immune networks for mitigating DoS flooding attacks. *Swarm and Evolutionary Computation* 38 (2018), 94–108.

[28] Huangxin Wang, Quan Jia, Dan Fleck, Walter Powell, Fei Li, and Angelos Stavrou. 2014. A moving target DDoS defense mechanism. *Computer Communications* 46 (2014), 10–21.

[29] Xin Zhang, Hsu-Chun Hsiao, Geoffrey Hasker, Haowen Chan, Adrian Perrig, and David G Andersen. 2011. SCION: Scalability, control, and isolation on next-generation networks. In *IEEE Symposium on Security and Privacy (S&P '11)*. 212–227.

[30] Jing Zheng, Qi Li, Guofei Gu, Jiahao Cao, David KY Yau, and Jianping Wu. 2018. Realtime DDoS defense using COTS SDN switches via adaptive correlation analysis. *IEEE Transactions on Information Forensics and Security (TIFS '18)* 13, 7 (2018), 1838–1853.