2013 IEEE Symposium on Security and Privacy

The Crossfire Attack

Min Suk Kang
ECE Department and CyLab
Carnegie Mellon University
Email: minsukkang@cmu.edu

Soo Bum Lee
CyLab
Carnegie Mellon University
Email: soobum@cmu.edu

Virgil D. Gligor
ECE Department and CyLab
Carnegie Mellon University
Email: gligor@cmu.edu

***Abstract*—We present the Crossfire attack – a powerful attack that degrades and often cuts off network connections to a variety of selected server targets (e.g., servers of an enterprise, a city, a state, or a small country) by flooding only a few network links. In Crossfire, a small set of bots directs low-intensity flows to a large number of publicly accessible servers. The concentration of these flows on the small set of carefully chosen links floods these links and effectively disconnects selected target servers from the Internet. The sources of the Crossfire attack are undetectable by any targeted servers, since they no longer receive any messages, and by network routers, since they receive only low-intensity, individual flows that are indistinguishable from legitimate flows. The attack persistence can be extended virtually indefinitely by changing the set of bots, publicly accessible servers, and target links while maintaining the same disconnection targets. We demonstrate the attack feasibility using Internet experiments, show its effects on a variety of chosen targets (e.g., servers of universities, US states, East and West Coasts of the US), and explore several countermeasures.**

I. Introduction

Botnet-driven distributed denial-of-service (DDoS) attacks which flood selected Internet servers have been known for some time [1, 2, 3, 4]. In contrast, link-flooding attacks that effectively disconnect chosen Internet servers have been uncommon, possibly because of the complexity of selective server targeting. Instead, most of these attacks cause route instabilities [5] and Internet connectivity disruption [6, 7] rather than selective end-server disconnection (reviewed in Section VII). Nevertheless, when the aim of an attack is to cut off critical infrastructure (e.g., energy distribution, time-critical finance, command and control services) from the Internet, link flooding can be extremely effective; e.g., current peak rates of a single botnet-driven attack can easily exceed 100 Gbps [8], making it possible to flood the vast majority of Internet links.

Link flooding by botnets cannot be easily countered by any of the current Internet defense methods for three reasons. First, bots can use *valid IP addresses*, and thus defenses based on detecting or preventing use of spoofed IP addresses become irrelevant; e.g., defenses based on ingress filtering [9], capability systems [10, 11], or accountable protocol designs [12, 13]. Second, and more insidiously, botnets can flood links *without using unwanted traffic*; e.g., they can send packets to each other in a way that targets groups of routers [7]. Third, a botnet can launch an attack with *low-intensity traffic* flows that cross a targeted link at roughly the same time and flood it; e.g., a botnet controller could compute a large set of IP addresses whose advertised routes cross the same link (i.e., *decoy* IPs), and then direct its bots to send low-intensity traffic towards those addresses. This type of attack, which we call the *Crossfire attack*[1] and describe in this paper, is undetectable by any server located at a decoy IP address, and its effects are invisible to an ISP until (too) late[2]. Furthermore, current traffic engineering techniques are unable to counter these attacks. The latency of offline traffic engineering is impractically high (e.g., hours and days [15, 16]) whereas online traffic engineering techniques cannot offer strong stability guarantees [17], particularly when multiple ISPs need to coordinate their responses to counter an attack, and hence cannot be deployed in the Internet backbone. Worse yet, even if online techniques could be deployed, an adversary attack could change the set of target links in real time thereby circumventing online traffic engineering defenses; viz., discussion in Section IV.

In this paper, we present the Crossfire attack. This attack can effectively cut off the Internet connections of a targeted enterprise (e.g., a university campus, a military base, a set of energy distribution stations); it can also disable up to 53% of the total number of Internet connections of some US states, and up to about 33% of all the connections of the West Coast of the US. The attack has the hallmarks of *Internet terrorism*[3]: it is low cost using legitimate-looking means (e.g., low-intensity, protocol conforming traffic); its locus cannot be anticipated and it cannot be detected until substantial, persistent damage is done; and most importantly, it is *indirect*: the immediate target of the attack (i.e., selected Internet links) is not necessarily the intended victim (i.e., an end-point enterprise, state, region, or small country). The low cost of the attack (viz., Section IV), would also enable

[1]This attack should not be confused with that of Chou *et al.* [14], which also uses the term "crossfire" for a different purpose; i.e., to illustrate *unintentionally* dropped legitimate flows.

[2]Of course, an adversary could easily change the set of bots used in the attack; e.g., typical networks of 1M bots would allow one hundred disjoint, and a very large number of different sets of 10K bots.

[3]Although common agreement on a general definition of terrorism does not exist, the means of attack suggested here are common to most terrorist attacks in real life.

a perpetrator to *blackmail* the victim.

The main contributions of this paper can be summarized as follows:

1) We introduce the Crossfire attack in the Internet and show how it can isolate a target area by flooding carefully chosen links. In particular, we show that it requires relatively small botnets (e.g., ten thousand bots) and is largely independent of the bot distribution. It has no effective countermeasure at either target routers or end-point servers, and as a result, it can degrade and even cut off connections to selected Internet areas ranging from a single organization to several US states, for a long time.

2) We show the feasibility of the Crossfire attack with data obtained from large-scale experiments. In particular, our analysis of Internet traffic to targets shows that very few carefully chosen links are responsible for delivering the vast majority of all traffic to a specific area, a fact which makes this attack fairly easy to launch. Traffic concentration in a small set of links located a few (e.g., three to four) hops away from a targeted area is intuitively attributable to the shortest path routing by the Internet IGP/BGP protocols, and easily discoverable by common tools such as *traceroute*. We show that the attack traffic on these links follows a power-law distribution that depends on the targeted servers and cannot be anticipated by generic Internet-connectivity metrics; e.g., metrics based on router connectivity [18, 19] or betweenness centrality [20].

3) We show that the Crossfire attack is *persistent* in the sense that it cannot be stopped either by individual ISPs or by end-point servers, which are effectively disconnected by flooded links at least three hops away, for a long time. Attack persistence is caused by three independent factors. First, the selected attack routes become stable after the removal of all load balancing dynamics (which is consistent with prior observations [21]). Second, the attack traffic is shaped such that (i) only a data plane of a link is flooded while the control plane remains unaffected, and hence dynamic re-routing can be initiated only after data-plane flood detection, which gives an adversary ample time to select alternate sets of links for the same target area; and (ii) early congestion of links located upstream from a targeted link is avoided by *a priori* estimation of the bandwidth available on the route to that link. Third, the availability of multiple, disjoint sets of target links distributed across multiple ISPs implies that no single ISP can unilaterally detect and handle this attack.

4) We argue that collaborative on-line, rather than offline, traffic engineering techniques would become necessary to reduce the persistence of such attacks. In the absence of such measures, the Crossfire attack must be handled by application protocol layers; e.g., overlays that detect effective host disconnection from the Internet and re-route traffic via different host routes [22, 23]. Botnet market disruption and international prosecution of attack perpetrators may complement technical countermeasures against these attacks.

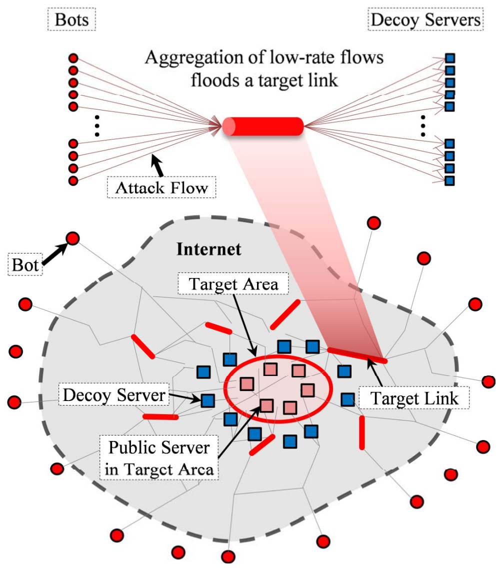

Figure 1: The Elements of the Crossfire Attack

II. The Crossfire Attack

In this section, we present the steps of the Crossfire attack. The adversary's goal is to prevent legitimate traffic from flowing into a specific geographic region of the Internet, and the capability she needs to accomplish that goal is to flood a few network links in and around that region. We begin by defining the two most common terms used in this paper: the *target area* and *target link*. Then, we describe how an adversary designs an attack using the bots she controls. Fig. 1 illustrates the concept of the Crossfire attack.

Target Area: A target area is a geographic region of the Internet against which an adversary launches an attack;[4] viz., the area enclosed by the circle in Fig. 1. A typical target area includes the servers of an organization, a city, a state, a region, and even a country, of the adversary's choice.

Target Link: A target link is an element of a set of network links the adversary needs to flood so that the target area is cut off from the rest of the Internet. These carefully chosen network links are the actual target of the flooding attack whereas the target area is the real, intended target.

To launch a Crossfire attack against a target area, an adversary selects a set of public servers within the target area and a set of *decoy servers* surrounding the target area. These servers can be easily found since they are chosen from publicly accessible servers (viz., Section V-B). The set of public servers is used to construct an attack topology centered at the target area, and the set of decoy servers is used to create attack flows. Then, the adversary constructs a *"link map"*, namely the map of layer-3 links from her bot

[4]The attack may have side effects and affect other non-targeted areas. However, these side effects do not increase attack's detectability. They can be a desired feature whenever the adversary's goal is to cut off most of the traffic at and around a target area, rather than to surgically isolate a small number of specific servers.

addresses to those of the public servers. (The differences between a link map and a typical router-topology map are discussed below.) Once the link map is created, the adversary uses it to select the best target links whose flooding will effectively cut off the target area from the Internet. Next, the adversary coordinates the bot-decoy (server) flows to flood the target links, which would eventually block most of the flows destined to the target area. This can be easily done since target links are shared by flows to the decoy servers and target area. Finally, the adversary selects multiple disjoint sets of target links for the same target area and floods them one set at a time, in succession, to avoid triggering bot-server route changes. The three main steps needed to launch the Crossfire attack consist of the link map construction, attack setup, and bot coordination, as shown in Fig. 2. Note that, to extend the duration of the attack, the last step, namely the bot coordination step, is executed repeatedly by dynamically changing the sets of target links, which we will explain in detail in Section II-D. We describe each of the adversary's steps below.

A. *Link Map Construction*

To flood links leading to a target area, an adversary needs to construct a link map of the Internet surrounding that area.

1) Traceroute from Bots to Servers:

To construct the link map, the adversary instructs her bots to run *traceroute* and find all the router-level routes to the public servers in the target area and the decoy servers. The result of a traceroute is a sequence of IP addresses that are assigned to the interfaces of the routers on the route, where a link is identified by the IP address of the adjacent router's interface. Thus, the sequence of IP addresses represents the sequence of layer-3 links[5] that the attack traffic would travel.

A link map for the Crossfire attack is *different* from a typical router-topology map [18] that attempts to build a router-level connectivity to analyze topological characteristics (e.g., node degree). This attack only needs the list of layer-3 links and their relationships to compute a set of target links on the bot-to-target area routes, while each link's membership to a specific router is *irrelevant*. Note that the link map construction does *not* require IP alias resolution [24]; i.e., determining the set of IP interfaces owned by the same router is unnecessary. As a consequence, an adversary can use the ordinary traceroute for the link map construction regardless of how inaccurate its IP alias resolution may be [25].

A bot runs multiple traceroutes to the same server in order to determine the stability and multiplicity (or diversity) of a route, both of which are used for selecting effective target links (discussed in Section V-D in detail). The traceroute

[5]Although a single layer-3 link consists of several lower layer connections that are invisible to the adversary, the flooding on the layer-3 link is still effective whenever the adversary's maximum bandwidth assumption (e.g., 40 Gbps in our experiments) is correct along the layer-3 link.

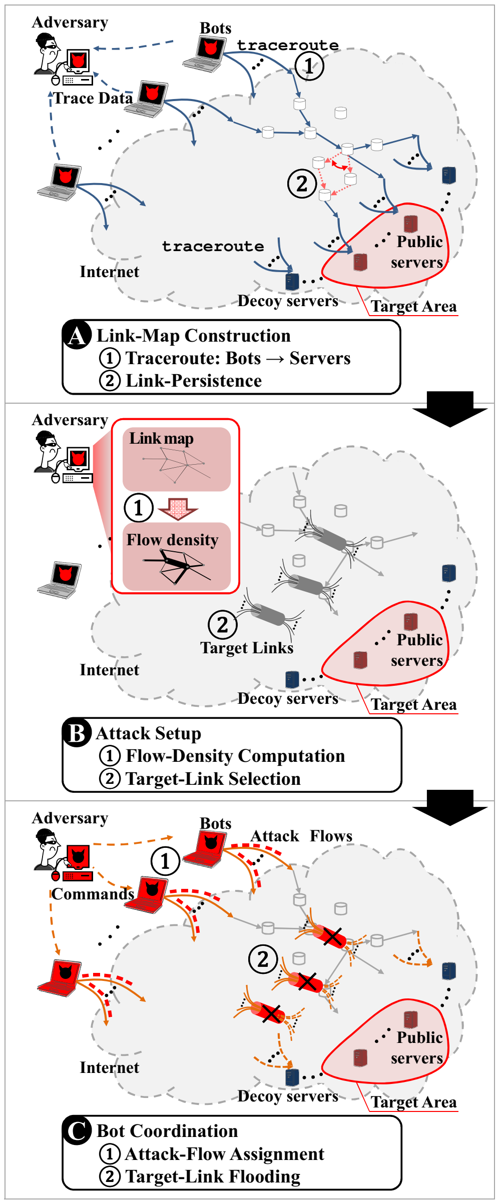

Figure 2: The steps of the Crossfire attack.

results are collected by the adversary and used to construct the link map.

2) Link-Persistence:

The link map obtained in the previous step cannot be directly used to find target links since some of the routes obtained may be unstable. Unstable routes would complicate the attack since the adversary may end up chasing a moving target. Route instability is primarily caused by ISPs' load balancing processes (i.e., forwarding traffic through multiple routes), which are supported by most commercial routers [26]. A consequence of load balancing is that, for the same bot-to-server pair, some links do not always appear on the trace of the route produced by multiple invocations of traceroute (viz., the arrowed links of step A-② in Fig. 2). These links are said to be *transient*, whereas those that always appear on a route are said to be *persistent*. The adversary identifies transient links and *removes* them from the set of potential target links. Our Internet experiment shows that 72% of layer-3 links measured by traceroute are persistent[6].

B. Attack Setup

The adversary uses the obtained link map to discover the set of target links whose flooding cuts off the largest number of routes to the target area. Clearly, the larger the proportion of cut routes out of all possible routes to the target area, the stronger the attack. The attack-setup step consists of the following two sub-steps.

1) Flow-Density Computation:

The adversary analyzes the link map for a target area and computes *'target-specific attack-flow density'*, or simply flow density henceforth, for each network link in the link map. The **flow density** of a persistent link is defined as the number of flows between bots and target-area servers that can be created through that link. Hence, flow density is a target-area-specific metric and can vary widely from one target area to another (viz., Section III-A). It is a very different metric from those used for Internet connectivity, such as the "betweenness centrality" [20] and the degree of routers [18, 19] (viz., Section III-A), and should not be confused with them.

A high flow density for a link indicates that the link delivers both a large number of attack and legitimate (or non-attack) to a specific target area, and thus the link becomes a good attack target. We found that the flow density follows a *power-law* distribution in a link map (viz., Section III-A), and this enables an adversary to easily discover a set of high flow density links that delivers most traffic to a target area.[7] Furthermore, the computed flow density remains largely unchanged for at least several hours due to the well-known, long-term stability of Internet routes [27, 21]. Hence, flow density can be used as a stable and reliable metric by the adversary in selecting target links.

2) Target-Link Selection:

In this step, the adversary finds multiple disjoint sets of target links to be flooded. The adversary selects at least two disjoint sets of target links and uses them one at a time, in succession, to achieve attack persistence (viz., Section II-D). The goal of this step is to maximize the amount of disrupted traffic flowing into the target area by optimal selection of target links using the link map and flow density.

To quantify how much of the traffic to a target area can be cut off by a chosen target-link set, the adversary computes the degradation ratio for that target area. The **degradation ratio** is the fraction of the number of bot-target area routes cut by the attack over the number of all possible bot-target area routes. We say that a route is cut by an attack if the route contains a target link that is flooded by the attack.

To select the target links that maximize the degradation ratio to a target area, the adversary must solve the *generalized maximum coverage problem*, which is a well-known NP-hard problem. Instead of finding an exact solution, the adversary uses an efficient heuristic, namely a greedy algorithm [28], presented in Section IV-D. The execution time of our heuristic is very small, namely less than a minute in all experiments (viz., Section IV-D). This enables the adversary to adapt to dynamic route changes, if necessary. The output of this algorithm shows that flooding a few target links can block a majority of the connections to a target area. For example, flooding ten target links causes a 89% degradation ratio for a small target area; flooding fifteen target links can block 33% of connections flowing to the West Coast of the US (viz., Section V-D).

C. Bot Coordination

Once target links are selected at step B-② (Fig. 2), the adversary coordinates individual bots to flood the target links. To create flooding flows for a given set of target links, the adversary assigns to each bot (1) the list of decoy servers and (2) the send-rates for packets destined to individual decoy servers. The send-rates are assigned in such a way that individual attack flows have low intensity (or low bandwidth) while their aggregate bandwidth is high enough to flood all target links. This step consists of two sub-steps.

1) Attack-Flow Assignment:

The goal of the attack-flow assignment is to make the aggregate traffic rate at each target link slightly higher than the link bandwidth so that all the legitimate flows are severely degraded in those links. Two assignment constraints must be satisfied. The first is that the adversary must keep each per-flow rate low enough so that none of the network protection mechanisms in routers or intrusion detection systems (IDS) at or near a server can identify the flow as malicious. The second is that the aggregate attack traffic necessary to flood all the targeted links is relatively evenly assigned to multiple bots and decoy servers. The first constraint ensures *indistinguishability* of attack flows whereas the second addresses *undetectability* both at servers in the target area and at decoy servers; viz., Section VI for details. The adversary first sets the maximum target bandwidth for each target link and exhausts it with attack flows. Then, she assigns individual flows for each target link.

[6]The link map obtained may not include *backup* links since these links typically do not show up in traceroutes. The existence of such links is largely immaterial to the effectiveness of Crossfire. If attack traffic spills over onto backup links and its intensity dampens appreciably, the adversary could easily switch to a new set of target links for the same server area, as shown in Section II-D.

[7]The power-law of flow density should *not* be confused with *connectivity properties* derived from traceroute, such as those for the degree of router level topology [19].

The rate of an attack flow at a target link is lower-bounded by the flow density. The average per-flow rate for the target link should be higher than the target bandwidth divided by the maximum number of available attack flows on the link, which is proportional to its flow density. Moreover, the assignment of the per-flow rate must take into account the maximum flow rate a decoy server can handle without triggering traffic alarms. For example, if a decoy server is a public web server, one web click per second on average (a HTTP GET packet per second $\simeq$ 4 Kbps) would not be classified as abnormal traffic at the server. Therefore, the adversary can easily assign a large enough number of attack flows with low per-flow rates. The adversary also has to assign per-bot and per-decoy server rates that are evenly distributed. For enhanced undetectability of attack traffic at the bots and the decoy servers, the adversary must account for all previously assigned traffic rates at the bots and decoy servers whenever assigning new attack flows. The adversary conservatively sets the target bandwidth to 40 Gbps, which is the most widely used link bandwidth currently deployed (OC-768) for high bandwidth backbones.

Despite an adversary's careful attack flow assignment, non-target links located upstream of the target links could still become congested, which we call *early congestion*, if they have limited bandwidth and/or the bot density in a certain area is too high. The adversary can avoid potential early congestion using *a priori* link bandwidth estimation, which we discuss in detail in Section IV-C.

2) Target-Link Flooding:

The adversary directs her bots to start generating the attack flows. Each bot is responsible for multiple attack flows, each of which is assigned a distinct decoy server with the corresponding required send-rate. Bots slowly increase the send-rates of their attack flows up to their assigned send-rates, which makes the attack flows indistinguishable from the traffic patterns of typical "flash crowds" [29]. Bots can adjust the intensity of their flow traffic dynamically, based on the state of each target link; i.e., if the actual bandwidth of a target link is less than the assigned attack bandwidth (set in Section II-C1), the bots stop increasing the rates of attack flows as soon as the target link is flooded.

D. Rolling attacks

The adversary can dynamically change the set of target links (among the multiple sets found previously) and extend the duration of the Crossfire attack virtually indefinitely. Continuous link flooding of the same set of target links would lead to bot-server route changes since it would inevitably activate the router's failure detection mechanism. Hence, changing the set of target links assures attack persistence and enables the attack to remain a pure data-plane attack. The adversary can also dynamically change the set of bots to further enhance the undetectability of the Crossfire attack. These dynamic attack execution techniques are called *rolling attacks* in Section IV-B where they are described in more detail.

III. Technical Underpinnings

In this section, we discuss the two characteristics of the current Internet which enable the Crossfire attack, namely (1) the power law of flow-density distribution, which is target-area specific, and (2) the independence of the geographical distribution of bots from target links and attack targets, which gives an adversary has a wide choice of bots in different locations on the globe.

A. Characteristics of Flow-Density Distribution

Before analyzing the distribution of flow density, we must distinguish between the attack-specific flow density and connectivity-specific metrics, such as the *betweenness centrality* [20] and the *degree of routers* [18], which characterize an Internet topology. Recall that the flow density of a link represents the number of source-to-destination (i.e., bot-to-server in the target area in the Crossfire attack) pairs whose traffic crosses the link *persistently*. In contrast, betweenness centrality, which is the number of shortest routes among all vertices that pass through an edge in a graph, does not reflect actual traffic flows and their dynamics. Similarly, the connectivity degree of a router, which represents the router's layer-3 direct connections to neighbor routers, namely the topological connectivity of the router, does not capture any dynamics of traffic flows. Thus, neither of these metrics could be used to evaluate the feasibility of the Crossfire attack.

Our analysis on the flow-density distribution is two-fold; first, we show that it is easy to find target links that have extremely high flow density for a selected target area; and second, we show that flow density of a link is not a constant but varies depending on a selected target area (i.e., flow density is a target-area specific metric).

1) Universal power-law property of flow-density distribution: A power-law distribution exhibits a heavy-tail characteristic, which indicates that extreme events are far more likely to occur than they would in a Gaussian distribution. More formally, a quantity x obeys a power-law if it follows a probability distribution

$$p(x) \propto x^{-\alpha} \text{ for } x > x_0, \tag{1}$$

where α is a constant parameter of the distribution known as the scaling parameter [30]. The power-law property appears in the tail of the distribution (i.e., $x > x_0$)[8]. If a power-law

[8]Some past research relied on simple data-fitting methods to conclude that their datasets follow a power-law distribution [18, 31]; i.e., if a histogram of empirical datasets is well fitted to a straight line on log-log scale, a power-law behavior would be ascribed to the datasets. However, recent studies [32, 25] show that these data-fitting methods are insufficient to conclude the power-law compliance of empirical data. According to Clauset *et al.* [30], the majority of purported power-law datasets fail to pass the rigorous statistical hypothesis test on their power-law distribution.

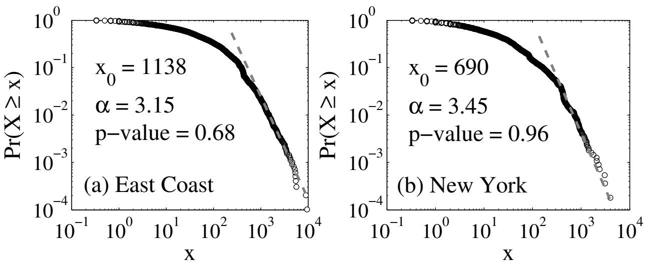

Figure 3: Flow-density distributions for various target areas: (a) East Coast and (b) New York. The complementary cumulative distribution functions (CCDFs) (i.e., Pr($X \geq x$)) of flow density (x) for both areas are plotted on log-log scale.

distribution holds for flow density, that would imply that an adversary could easily find links whose flow density is many orders of magnitude higher than average. These links would become good targets for attack for a particular target area.

We use the rigorous statistical test proposed by Clauset *et al.* [30][9] to show that a *power-law holds for flow-density distributions*. We first estimate the parameters (i.e., x_0 and α) of power-law distribution on our flow-density datasets and test the power-law hypothesis with the estimated parameters. Fig. 3 shows the flow-density distributions of two different target areas: (a) East Coast and (b) New York. The complementary cumulative distribution function (CCDF) (i.e., Pr($X \geq x$), where x is flow density) of the flow-density datasets is plotted on a log-log scale. As the graphs show, both distributions are well fitted to the diagonal lines at the tail. More precisely, we apply the power-law hypothesis test proposed by Clauset *et al.* [30] to the measured flow-density dataset and obtain the p-value, which indicates the degree of plausibility of a hypothesis, for each test. The p-values for the two target areas (i.e., 0.68 and 0.96) are much higher than the significance level, which is often set to 0.05. Hence, the plausibility of the null hypothesis (i.e., the flow-density distribution follows a power law) is accepted [33].

2) Target-area dependency of flow density: Unlike connectivity-related metrics, which are dependent only on *physical* network connectivity but independent of attack targets, flow density is an attack-specific metric; i.e., a target link that has high flow density for a target area may have a very different density for other areas.

Table I illustrates the top 20 links ordered by flows densities for three target areas of different sizes: the East Coast, Massachusetts, and Univ2. Naturally, one would expect that the links' flow densities would follow the obvious link-map inclusion relations, namely the link map of Univ2 $\subset$ link map of Massachusetts $\subset$ link map of the East Coast. However, Table I shows that the top 20 links for these related target areas are very different: not only these links do not follow the link-map inclusion, but also whenever some are

[9]The statistical tools, proposed by Clauset *et al.* [30], are available at http://tuvalu.santafe.edu/~aaronc/powerlaws/.

Target area	Indices of top 20 flow-density links
East Coast	01, 02, 03, 04, 05, 06, 07, 08, 09, 10, 11, 12, **13**, 14, 15, 16, 17, 18, **19**, 20
Massachusetts	**19**, 21, **13**, 22, **23**, 24, 25, 26, 27, 28, 29, **30**, 31, 32, 33, 34, 35, 36, 37, 38
Univ2	39, 40, **30**, 41, 42, **23**, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56

Table I: Top 20 flow-density links for three different target areas: the East Coast of the US, Massachusetts, and Univ2. Each link IP address is mapped to a link index. Bold indices denote the links shared by different areas.

shared between areas they have different density ranks. For example, link **19** has the highest flow-density rank when the state of Massachusetts is targeted, and yet it only ranks next to the last for the East Coast. Furthermore, link **19** does not even appear in the top 20 link densities of Univ2. This clearly shows that flow density is a target-area specific metric, which reveals a link's usefulness in an attack that targets a specific area.

B. Geographical Distribution of Bots

Although the selected target links are highly dependent on the target area of the attack, they are nearly *independent* of the choice of bot distributions; i.e., even if an adversary uses different sets of bots that have different geographic distributions to flood a target area, the effectiveness of the Crossfire attack would remain nearly unchanged. To show this, we performed the following experiment. First, we partitioned the set of bots into several subsets based on bots' geolocation (viz., subsets denoted by S_j, $j = 1, ..., 8$ in Table II). Then, we selected different subsets to form six different bot distributions (viz., distributions denoted by $Distr_i$, $i = 1, ..., 6$ in Table II), and simulated a separate Crossfire attack for each distribution against three different target areas; i.e., East Coast, Pennsylvania, and Univ1. Finally, we analyzed how the different distributions affect the degradation ratios.

The geographical distributions of 620 PlanetLab nodes and 452 LG servers are as follows: 42% were located in Europe, 39% in North America, 13% in Asia, and 6% in the rest of the world (viz., Figure 5). Since the distributions of PlanetLab nodes and LG servers in North America and Europe cover wider areas than those in the rest of the world, we (1) assigned three disjoint subsets to each; i.e., S_1, S_2, and S_3 to North America and S_4, S_5, and S_6 to Europe; and (2) constructed the bot distributions such that $Distr_1$, $Distr_2$, and $Distr_3$ cover a similar number of bots in North America and Asia, and $Distr_4$, $Distr_5$, and $Distr_6$ a similar number of bots in Europe and Asia.

Fig. 4 shows the degradation ratios for the six different bot distributions shown in Table II and three different-size target areas chosen; i.e., East Coast, Pennsylvania, and Univ1. For each target area, we defined a baseline degradation ratio (denoted by "Baseline" in Fig. 4) as the degradation ratio

	North America			Europe			Asia	Others
	S_1	S_2	S_3	S_4	S_5	S_6	S_7	S_8
Baseline	✓	✓	✓	✓	✓	✓	✓	✓
$Distr_1$	✓			✓	✓	✓	✓	✓
$Distr_2$		✓		✓	✓	✓	✓	✓
$Distr_3$			✓	✓	✓	✓	✓	✓
$Distr_4$	✓	✓	✓	✓			✓	✓
$Disrt_5$	✓	✓	✓		✓		✓	✓
$Distr_6$	✓	✓	✓			✓	✓	✓

Table II: Different geographic distributions of bots ($Distr_i$) created using different subsets of PlanetLab nodes and LG servers (S_j).

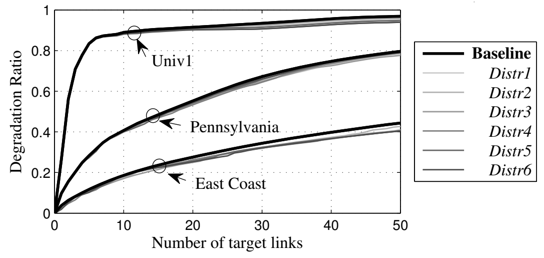

Figure 4: Degradation ratios for different geographic distributions of PlanetLab nodes and LG servers.

given by an attack launched by *all* bots available. The six degradation ratios are computed using the same total number of routes as that used in the baseline ratio. Thus, if the degradation ratio of a certain distribution is close to the baseline, that distribution of bots is as damaging to the target area as the baseline (i.e., as all available bots). As shown in Fig. 4, the choice of the six different distributions does not diminish the effectiveness of the attack in a measurable way. That is, the effectiveness of an attack is nearly independent of the geographical distribution of bots. This is particularly noticeable in the case of the small and medium target area where the degradation ratios are almost indistinguishable from the baseline.

IV. Attack Persistence and Cost

A. *Data-Plane-Only Attack: Indefinite Duration*

In this subsection, we discuss how the Crossfire attack maintains its effectiveness, namely a high connection degradation ratio for selected target areas caused by link flooding (data plane only), by avoiding any route change (by the control plane) in the Internet. Clearly, the goal of the adversary is to avoid control plane reaction since that would cause routes to change dynamically in response to any unexpected network-state variations (e.g., due to link failures or high traffic load akin to link flooding).

The Crossfire attack takes advantage of the fact that the current Internet's dynamic response to link flooding is too slow for an adaptive adversary. That is, if the adversary periodically changes the set of predetermined target links in less than 3 minutes, she can maintain a very high connection degradation ratio without inducing any Internet route changes. Thus, the attack duration can be extended virtually indefinitely. The technique of changing the set of target links, namely the rolling attack, is discussed in detail in Section IV-B. The following two subsections illustrate how slowly the current Internet would react to the Crossfire attack.

1) Link failure detection: Link-failure detection refers to a function of a routing protocol that enables a router to assess the physical connectivity of its network link to its neighbor router [34]. A router which misses several consecutive control packets (e.g., *hello* packets for OSPF or *keepalive* messages for BGP) in a specific time interval (default 40 seconds for OSPF or default 180 seconds for BGP) will conclude that the link failed and broadcast the link failure to other routers. The consequence of the link failure is two-fold. First, if an intra-AS link fails, the failure notification is sent to all the routers within the same AS, which leads to internal topology changes. In contrast, if a link between two neighbor ASs (i.e., an inter-AS link) fails, the failure, in the worst case, could propagate to all the BGP speaking routers in the Internet and cause a global topology change. These topology changes would redirect the attack traffic to alternate routes and invalidate the flow densities computed for the on-going Crossfire attack.

To measure Internet reaction to link failures, Shaikh *et al.* [34] inject traffic that consumes 100% of the capacity of a link and measure the time for the router to detect the link failure. This experiment shows that it takes 217 seconds for a IGP router (that runs OSPF or IS-IS) and 1,076 seconds for a BGP router to diagnose congestion as a failure[10]. Note that failure detection takes much longer than its default waiting time interval for the control packets, namely 40 seconds for OSPF and 180 seconds for BGP. This is because some control packets that are queued at the congested interface at a router can successfully reach a neighbor router even in severe link congestion. Clearly, the congestion diagnostic times are too long to enable rapid reaction to the Crossfire attack where the adversary can change the set of target links for an area in much less than 3 minutes; viz., the rolling attacks of the next subsection.

2) Traffic engineering: Most commonly, ISPs use *offline* traffic engineering techniques, whereby network parameters are periodically re-optimized based on the estimated traffic matrix among the ingress/egress points of their networks [16]. The network parameters can be the link weights of IGP protocols (e.g., OSPF or IS-IS) in pure IP networks [37] or bandwidths of LSP (label switched path) tunnels in MPLS networks [38, 39]. Offline traffic engineering produces new

[10]We assume that the OSPF and BGP protocols do not use shorter intervals for fast failure detection [35], but use default timers (HelloInterval & RouteDeadInterval for OSPF and KeepaliveTimer & HoldTimer for BGP). Since most optical fiber connections (e.g., SONET or SDH) provide failure reports in less than 50 ms, additional system configuration for faster link failure detection at layer-3 is obviously unnecessary [36].

routes on a time scale ranging from tens of minutes to hours and days [15], though more commonly in days and weeks [38, 39, 16]. Even though it is not frequently used by ISPs due to its potential instability problem, *online* traffic engineering occurs on a smaller time scale, namely from minutes to hours [16, 17]. Given that the adversary can repeatedly relaunch the Crossfire attack for new routes, neither current offline nor online traffic engineering can offer effective countermeasures.

B. Proactive Attack Techniques: the Rolling Attack

A Crossfire attack is said to be *rolling* if the adversary changes the attack parameters (e.g., bots, decoy servers, and target links) dynamically while maintaining the same target area. A rolling attack can be employed by an adversary to further increase indistinguishability of attack traffic from legitimate traffic and undetectability of all target links by target area. Based on the types of attack parameters that can be dynamically changed, rolling attacks can be categorized into two types: one that changes bots and decoy servers while maintaining the same target links, and the other that changes target links while maintaining the same target area.

The main advantage of the first type of attack is that it further increases the indistinguishability of the Crossfire flows from legitimate flows while maintaining the same attack effects. Since the source and the destination IP addresses seen at the selected target links change over time, the ISPs cannot easily identify the source and the destination IP addresses that contribute to the attack. A potential disadvantage is that this attack requires more bots and decoy servers than the minimum necessary to flood the target links. However, the current cost of bots suggests that this disadvantage is insignificant (viz., discussion of bot costs below).

The second type of rolling attack uses multiple sets of *disjoint* target links for the same target area. To find the multiple disjoint sets, the adversary executes the target-link selection algorithm (viz., Section II-B2) successively; i.e., the n-th best set of the target links is selected after removing the previously selected links. The use of multiple disjoint sets of the target links enhances attack undetectability by ISPs since ISPs could not anticipate the adversary's choice of targets with certainty. More importantly, this type of rolling attack enables Crossfire to remain a pure data plane attack, as discussed in the previous subsection. A potential disadvantage is that this type of rolling attack may degrade the effectiveness of the Crossfire attack since the degradation ratio caused by attacking a non-best target set can be lower than that of attacking the best set. However, the degradation ratios of different sets of target links shown in Table III indicate that this degradation is minimal. In order to maximize attack effects while being undetected, the adversary can alternate the target sets; she would use the best set for the most of attacks and switch to the non-best sets only for a short time interval. For example, if the adversary repeatedly schedules 3 minutes for the attack on the best set and next 30 seconds for the second-best set, she can maintain the attack towards a target area indefinitely while limiting the reduction of the degradation ratio less than 4%.

Target area	Target link set		
	Best set	2^{nd} best set	3^{rd} best set
Univ1	89%	77%	63%
Pennsylvania	42%	30%	24%
East Coast	21%	16%	14%

Table III: Degradation ratios for different disjoint target link sets. Each set has 10 target links.

C. Avoidance of Early Congestion

Crossfire avoids early congestion, namely the event whereby a non-target link, or more, located upstream of the target links becomes congested. We argue that early congestion does not affect attack *feasibility*, but instead is a matter of attack *provisioning*, which is a very distinct and easily handled issue by an adversary.

Bots can easily detect early congestion by regularly performing traceroutes to the target area since if it happens, they would not receive most of replies (i.e., ICMP time exceeded messages) from the congested router and the subsequent routers on the route. When early congestion is reported by the bots, the adversary can re-assign some attack flows to over-provisioned bots, to avoid the early congestion. In other words, the adversary *adaptively* assigns attack flows to geographically distributed bots, so that a sufficient number of attack packets reach the target links and flood them. Note that additional attack routes to target links can always be found *before* the attack and used only if necessary.

In addition to the dynamic assignment of attack flows, the adversary can instruct bots to estimate the available bandwidth towards the target links using *a priori* bandwidth estimation tools (e.g., Pathneck [40]) and predict early congestion before assigning attack flows. In this way, the adversary can provision the bots so that early congestion would not happen.

D. Execution Time of Target Selection Algorithm

The greedy algorithm of selecting a set of T target links runs as follows:
Let $\mathcal{R}$, $\mathcal{L}$ and $\mathcal{T}$ be the set of all bot-to-target area routes, the set of candidate links for the target area, and the set of target links, respectively. Let l_i be a link on a route.

(1) Add all distinct links ($l_i's$) of $\mathcal{R}$ to $\mathcal{L}$.
(2) Take out the highest flow density link, l_i^{max}, from $\mathcal{L}$ and add it to $\mathcal{T}$.
(3) Recompute the flow density for all l_i's in $\mathcal{L}$.
(4) Repeat (2) and (3) until T target links are selected, i.e., until $|\mathcal{T}| = T$.

The above algorithm finds the T best target links that disconnect the target area in terms of the degradation ratio,

Target area	$T = 10$	$T = 20$	$T = 30$	$T = 40$	$T = 50$
Univ1	0.94	1.87	2.79	3.72	4.65
Pennsylvania	3.10	5.46	7.38	8.99	10.38
East Coast	13.44	24.93	35.13	43.96	52.05

Table IV: Execution time (in seconds) to select T target links for different target sizes.

in T iterations of steps (2) - (3). Step (3) re-evaluates flow densities after removing all routes of $\mathcal{R}$ that include l_i^{max} and as a consequence, the step ensures that the adversary selects the target link that maximally disconnects the target area at each iteration. Table IV shows the execution times taken by our experiments. As expected, the execution time is proportional to the number of target links (T) for all target areas, and grows significantly for a large target area (e.g., 52 seconds in selecting 50 target links for the East Coast of the US), since more unique links can be found in large target areas. However, the number of unique links is bounded by a *limited* number of routes. This number is limited because bot-decoy pairs in the same source and destination subnets produce a *single* unique route. Hence, the execution time of the algorithm is short enough (e.g., at most a couple minutes) for an adversary to adapt to all potential route changes even for a large target area, in practice.

E. The Cost of the Crossfire Attack

To launch a Crossfire attack, an adversary needs bots. To get them, she can either infect user machines and install her own bots or buy the bots from *Pay-Per Install* (PPI) botnet markets [41]. For cost estimation, we assume that the adversary buys the bots from the markets. Our cost estimates are based on a recent analysis of PPI botnet markets [41]. A possible option would be to rent cloud services for bot operation from many, say one hundred, providers around the world. Given the low computation and communication requirements of Crossfire bots and the high-bandwidth connectivity of data centers to the Internet, the bots' behavior during an attack would not trigger providers' alarms.

PPI botnet markets have region-specific pricing plans. Generally, bots in the US or the UK are most expensive and cost \$100-\$180 per thousand bots. Bots in continental Europe cost \$20-\$60 whereas bots in the rest of the world cost less than \$10 per thousand bots. The mix of bots used in our experiments (presented in Section III-B) has 49% of bots in the US or UK, 37% in continental Europe, and 14% in the rest of the world. If we assume the size of a bot cluster (β) is 500, the total cost of the Crossfire attack is roughly \$46K. Our experiments also show (viz., Section V-D) that the minimum number of required bots that can flood 10 target links can be as low as 107,200 bots, and hence the attack cost can be as low as \$9K. This implies that a single organization or even an individual can launch a massive Crossfire attack. If the attack is state- or corporate-sponsored, many more bots can be purchased and a much larger number of links can be targeted. In this case, the Crossfire attack could easily disconnect almost 100% of the Internet connections to a large target area.

V. Experiment Setup and Results

In this section, we demonstrate the feasibility of the Crossfire attack and its effects on various target areas using real Internet data. In particular, we show how one sets up the bots, decoy servers, and target area for a Crossfire attack.

A. Bots

Instead of using real bots to perform our experiments, which would raise ethical [42, 43] and/or legal concerns [44], we use *PlanetLab nodes* [45] and *Looking Glass (LG) servers* as attack sources. PlanetLab is a global research testbed that supports more than one thousand nodes at 549 sites. An LG server is a publicly available router that provides a Web-interface for running a set of commands, including *traceroute* [46]. They have been used as vantage points for discovering Internet topology [47, 48, 49].

The PlanetLab and LG server networks provide a faithful approximation of a globally distributed bot network. As seen in Fig. 5, the 620 PlanetLab nodes and 452 LG servers are located 309 cities in 56 countries. In Section III-B, we will show that different bot distributions created using PlanetLab nodes and LG servers, result in practically the same attack effectiveness. Hence, the Crossfire attack using real bots (e.g., leased from botnet markets) would experience similar attack effects as in our experiments. A single PlanetLab node or LG server represents several hundred bots, given (1) the high degree of clustering observed in real-bot distributions [50, 51], and (2) the fact that bot-originated traffic from the same AS domain would converge at a router and then follow the same route, due to the BGP's single best route selection policy. Hence, the routes we trace from the PlanetLab nodes or LG servers to the public servers in the target area, allows us to build the actual Internet link map of the target area. We call the group of bots represented by the same PlanetLab node or LG server a *bot cluster*, and experiment with cluster sizes of 100, 200, and 500 bots.

B. Decoy servers

Decoy servers, which are the destinations for attack traffic, can be any public server whose physical location is nearby a target area. Among various possible ways an adversary could select decoy servers, one way is to find servers of public institutions (e.g., universities and colleges) physically located surrounding the target area. For example, the servers of a university or college are typically located on their

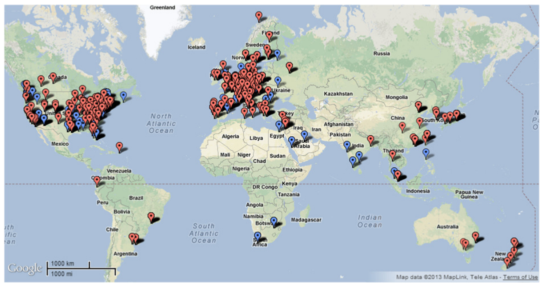

Figure 5: A map of geographic locations of the 620 PlanetLab nodes (red pins) and 452 LG servers (blue pins) used in our experiments.

Target area	Number of public servers in target area	Number of decoy servers
Univ1	1,000	350,000
Univ2	1,000	350,000
New York	86,000	265,000
Pennsylvania	82,000	269,000
Massachusetts	54,000	297,000
Virginia	34,000	317,000
East Coast (US)	351,000	351,000
West Coast (US)	201,000	201,000

Table V: The extrapolated numbers of public servers in target areas and decoy servers used for attacking each target area in our experiments

campus[11].

We found 552 institutions (i.e., universities and colleges) on both the East Coast (10 states) and West Coast (7 states) of the US, which can provide large numbers of decoy servers. The list of institutions in a specific US state is easily found on the Web[12]. An adversary can find a minimum of 1,000 public servers within an institution. For example, we found 2,737 and 7,411 public web servers within Univ1 in Pennsylvania and Univ2 in Massachusetts, respectively, via port-scanning. Had we used real bots, port scanning duties would be distributed to each bot and would be performed over a period of time, to avoid triggering IDSs or firewall alarms at those institutions. Similarly, an adversary could use 351,000 public servers located in 351 institutions on the East Coast of the US, and 201,000 public servers in 201 institutions on the West Coast.

C. Target area

A target area is the geographic location where an adversary wants to block Internet traffic. To establish that the Crossfire attack works for various target-area sizes, we used three different configurations: small, medium, and large. For the small area size, we set a single organization as the target area. Specifically, we set Univ1 and Univ2 as examples of small-sized target areas. As examples of medium-sized areas, we picked four US states, namely New York, Pennsylvania, Massachusetts, and Virginia. Finally, we picked ten states on the East Coast and seven on the West Coast as two examples for large target areas. Note that the large target areas' sizes could conceivably represent a medium-size country. For a small or medium target area, we chose decoy servers outside the target area for the undetectability of attack flows. However, for a large target area, we chose decoy servers inside the target area since the wide array of decoy servers within the area would not diminish the Crossfire's undetectability.

Table V illustrates the extrapolated numbers of public servers in the target areas and decoy servers used for attacking those areas. Note that the extrapolation is based on that an adversary can find 1,000 public servers within an institution.

D. Results

We performed Internet-scale experiments to verify the feasibility and the impact of the Crossfire attack based on the steps described in Section II. For each attack target area illustrated in Table V, we construct a link map (Step 1, viz., Section II-A) and select the target links (Step 2, viz., Section II-B), using the PlanetLab nodes and LG servers, and public servers in the target area. Bot-coordination (Step 3, viz., Section II-C) is performed via simulations, for obvious ethical and legal reasons. However, the simulations use the real link map and data obtained from the first two attack steps illustrated in Fig. 2. In this section, we summarize the results of our experiments.

Link map. We gather traceroute data from all the PlanetLab nodes and LG servers (i.e., sources) to all the institutions in the target areas (i.e., destinations) and construct the link maps centered on the target areas of the East and West Coasts of the US. For each source-destination pair, we run a traceroute six times to diagnose link persistence. Since multiple traceroute packets (i.e., ICMP packets) to the same destination are independently load-balanced at a load-balancing router [26], running six traceroutes is enough to determine whether a link on the route is persistent or transient. We classify a link as persistent if the link appears in all six traceroute results. The false positive probability, namely the probability that we falsely determine a transient link as persistent, is at most 0.016 ($\simeq 2^{-6}$). This is the case because the highest false positive probability is reported when a router, which has *two* load-balancing links to the next hop router, happens to select the *same* link in forwarding six traceroute packets originated from the same source. If

[11]The adversary might use a public search engine, such as SHODAN (http://www.shodanhq.com), to gather a large number of publicly accessible IPs at a geographical location. However, use of SHODAN would require cross-validation of the IP addresses in a geolocation due to possible search inaccuracies. Cross-validation would be a fairly simple matter of comparing results of multiple IP geolocation services for a certain target area

[12]http://www.4icu.org/

Target area	Percentage of persistent links
Univ1	79.99 %
Univ2	70.37 %
New York	69.70 %
Pennsylvania	75.68 %
Massachusetts	74.11 %
Virginia	70.32 %
East Coast (US)	71.78 %
West Coast (US)	72.37 %

Table VI: Percentage of persistent links per target area

the router has more load-balancing links, the false positive probability becomes lower.

We summarize the percentages of persistent links found by traceroutes in Table VI. Regardless of the size of a target area, the majority of the discovered links are persistent and hence can be used for the Crossfire attack. This result shows that even though traffic load-balancing through multiple links is widely implemented by ISPs in the current Internet, a large portion of Internet links are persistent. This enables the adversary to easily find (persistent) target links. In the following subsection, we discuss how the adversary finds the target links whose congestion would effectively disconnect a target area.

Link Coverage. Although one could not demonstrate that all links leading to a target area can be found by traceroute, one could show that *all critical links* can be found for a target area. To show this we selected different uniformly-distributed subsets of the 1,072 bots used (i.e., PlanetLab nodes and LG servers); e.g., subsets of 10%, 20%,..., 90% of all bots. We computed their degradation ratios for three target areas and plotted those against the *baseline degradation ratio* produced by all 1,072 bots. Figure 6 shows that, for each target area, beyond a certain bot-subset size, the differences in deviations from the baseline degradation ratios taper off, indicating that additional critical links which would increase degradation ratios can no longer be found; i.e., that size is approximately 10% of all bots for Univ1, 20% for Pennsylvania, and 50% for the East Coast. In similar experiments, if we vary server-subset sizes beyond a certain target-area related *threshold*, additional critical links that would increase the degradation ratios could not be found any longer. These two experiments suggest that the critical links we find adequately cover the flows toward a target area.

Flow density. To compute flow densities of all persistent links of the link map, we count the number bot-to-target area routes on those links. As expected, the distribution of flow densities is highly non-uniform, namely it follows a power-law distribution; i.e., a few links have unusually high flow densities while most of the other links have much lower flow densities (viz., Section III-A). The power-law distribution of flow densities makes the Crossfire attack very effective indeed. That is, flooding only a few high flow-density links would effectively disconnect a large number bot-target area

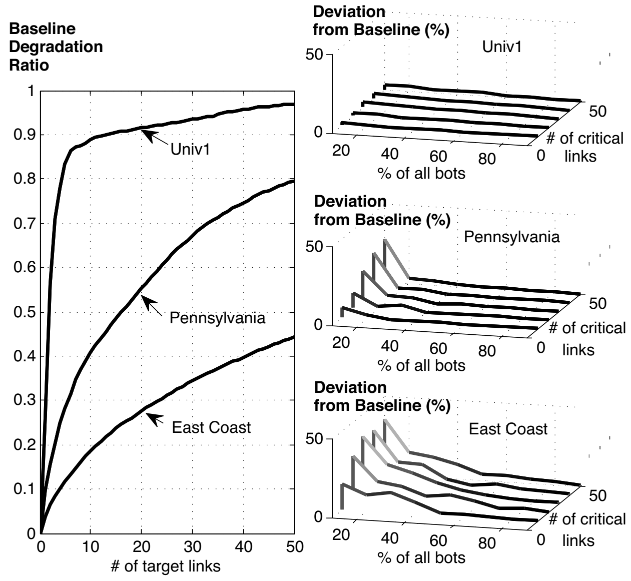

Figure 6: Deviations from baseline degradation ratios for different bot subsets.

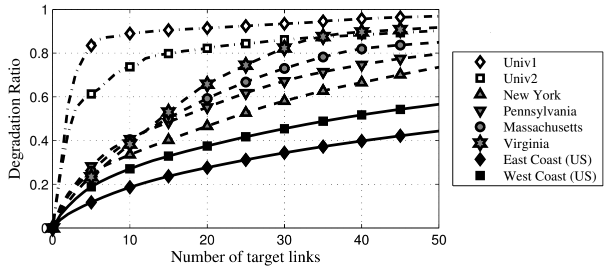

Figure 7: Degradation ratios for various target areas for different numbers of target links.

routes.

After computing the flow densities of all persistent links, we select a set of target links using the greedy algorithm specified in Section IV-D. Recall that we do not select links that are located close to a target area (more precisely, links whose distance from the target area is less than or equal to three hops) to avoid attack detection by any servers in the target area. For example, the average hop distance from the selected target links to Univ1 and Univ2 are 3.67 and 4.33, respectively[13]. Note that even though we eliminate links that are less than three hops away from the target area, we can effectively find target links with sufficiently large flow densities as discussed in the following subsection.

Degradation ratio. Fig. 7 shows the degradation ratios for various target areas with different numbers of target links. As shown in this figure, the increase in the degradation

[13]For medium and large areas, the hop distance can be measured relative to the peripheral servers.

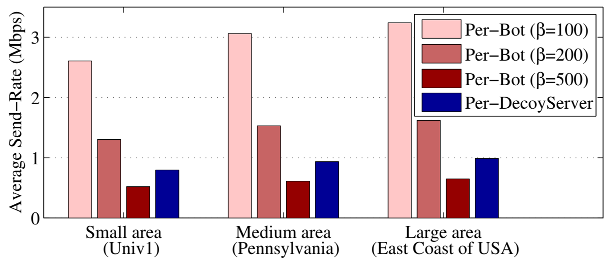

Figure 8: Per-bot, per-decoy server average send-rates for different bot cluster sizes (β).

ratio achieved by flooding additional target links diminishes as we flood more links; e.g., flooding the first five target links for attacking Univ1 results in an 83% degradation ratio whereas flooding five additional target links increases the degradation ratio by only 6%. This trend clearly shows that the power-law distribution of the flow density enables the adversary to achieve a high degradation ratio by flooding only a few target links. In general, the smaller the target area, the higher degradation ratio, because smaller target areas have relatively few links that deliver most of the traffic to them. For example, when flooding 15 target links, the degradation ratio of a large area (i.e., West Coast of US) is as high as 32.85%, that of a medium area (i.e., Virginia) is as high as 53.05%, and that of a small area (i.e., Univ1) is as high as 90.52%. This result may be misinterpreted and conclude that the Crossfire attack would damage only small target areas. In reality, when the attack effects are measured in terms of the *total number* of effectively disconnected end-users (or hosts) in a target area, the attack appears to be far more lethal to a large target area than a small one. For example, a Crossfire attack against West Coast using 15 target links effectively disconnects only 32.85% of traffic, yet the number of affected servers is huge.

Attack bots and flows. To flood the selected target links, we assign attack flows to bots by providing the list of decoy server IPs and corresponding flow rates. In our experiments, we set a 4 Kbps per-flow rate, which can be achieved by sending one HTTP GET message per second, for the indistinguishability of the Crossfire attack. While maintaining the low per-flow rate, we assign the attack flows evenly to the multiple bots and decoy servers. We conservatively assume that the bandwidth of target links is 40 Gbps, which ensures the presence of at least 10^7 (i.e., $40\text{ Gbps}/4\text{ Kbps}$) attack flows through each target link.

Fig. 8 shows the per-bot and per-decoy server average send-rates for three target areas of different sizes when flooding ten selected target links. Notice that for the large bot cluster size (β), we achieve lower per-bot send-rate since the attack flows can be more evenly distributed. An important observation is that for any target area, the per-bot average send-rate can be much lower than 1 Mbps when the bot cluster size (β) equals 500 (i.e., 536,000 bots in total). This shows that the adversary can aggregate a sufficiently large number (i.e., 10^7) of low-rate (i.e., 4 Kbps) attack flows at each selected target link and thus successfully exceed the bandwidth (i.e., 40 Gbps) of the target link while maintaining low per-bot and per-decoy server average send-rates. If the adversary uses more bots and decoy servers in practice, these average rates would become even lower.

VI. Attack Characteristics

The Crossfire attack has four distinct characteristics which distinguish it from ordinary DDoS attacks, namely *undetectability*, attack-flow *indistinguishability*, *flexibility* in the choice of targets, and *persistence* in terms of attack duration.

Undetectability at the Target Area. The Crossfire attack uses all *legitimate* flows to flood target links. Each bot creates *ordinary* connections (e.g., HTTP) with a set of decoy servers following the adversary's (i.e., the bot-master's) assignments, and hence individual connections do not trigger an attack alarm at the servers. Since a target area is not directly attacked and the decoy servers near the target area do not see any suspicious traffic, the servers in the target area would be unable to detect the attack. Even decoy servers would be unable to detect the attack since the well-coordinated flows to the decoy servers would cause only a few Mbps bandwidth increase to each server. Furthermore, the adversary can easily select target links among the links in the target set that are several hops (i.e., at least 3 hops in our experiments) away from the target area since links with high flow density are usually located in the core backbone networks. This makes it difficult even for the target links to identify an attack.

Indistinguishability of Flows in Routers. In the Crossfire attack, a large number of low-rate attack flows pass through a target link. Hence, a router connected to the target link cannot distinguish the attack flows from legitimate ones. In other words, since all the attack flows carry different source IP addresses and destination IP addresses, the high bandwidth aggregation mechanisms (e.g., Pushback [52], PSP [14]) become ineffective even if they are employed at all routers along the attack routes. Inspecting the payload of each packet would not help either because the attack flows carry the same payload as that of legitimate flows. Moreover, flooding target links with different sets of bots (e.g., the rolling attack, viz., Section IV-B) would further enhance this inherent indistinguishability of attack flows in routers.

Persistence. The Crossfire attack is able to disconnect a target area persistently by controlling the bot traffic so as not to trigger any control plane changes (e.g., route changes). This is achieved by using stable routes in rolling attacks, which change an active set of target links dynamically (viz., Section IV-B). In essence, a rolling attack makes the Crossfire attack a *pure data plane attack*, thereby leaving the control plane of the Internet unchanged. This extends the attack duration virtually indefinitely. The details of the attack persistence are presented in Section IV-A.

Flexibility. The Crossfire attack can be launched against *any* target area (regardless of its size) since an adversary can usually find a large number of public servers inside that target area and decoy servers near it; e.g., the adversary can select any of the many publicly accessible servers without needing permission from that server. This offers a great deal of flexibility in the adversary's choice of a target area, which is one of the most important characteristics that distinguish the Crossfire attack from other link-flooding attacks (viz., Related Work in Section VII). Our adversary's choice is enhanced by its low-rate flows used by the bots since the resulting attack flows would not trigger individual alarms in any potential target area.

VII. Related Work

A. Control Plane DDoS Attacks

DDoS attacks against a network link, even if launched with low-rate traffic, can disrupt a routing protocol (e.g. BGP) and ultimately trigger instability in the Internet. This class of attacks, which we call Control Plane DDoS attack, first proposed by Zhang *et al.* [53], exploits the fact that the control plane and data plane use the same physical medium. This fate-sharing allows an unprivileged adversary to convince a BGP speaking router that its BGP session has failed. Schuchard *et al.* in [5] extended this attack to multiple BGP sessions, which were selected based on the betweenness centrality measures of the network topology. They showed that their CXPST attack can generate enough BGP updates to cripple the Internet's control plane.

In contrast, the Crossfire attack is pure data plane attack, which maintains the effects of the attack persistently by suppressing any control plane reaction.

B. Attacks against Links

The recent Coremelt attack [7] demonstrates how a set of bots can send packets to each other and flood a set of AS backbone routers. The key characteristic of Coremelt is that it creates only wanted traffic and thus it eludes all defense mechanisms that filter unwanted traffic. Furthermore, this traffic is not subject to the congestion-control mechanisms of TCP and can thus exceed typical TCP traffic bounds. This unique advantage cannot be exploited in Crossfire, since the ends of its attack flows are not bots. Thus, Crossfire uses protocol messages that are unencumbered by congestion control; e.g., HTTP GET requests. In contrast with Coremelt, Crossfire creates very low intensity traffic (e.g., 4 Kbps flows) to decoy servers, which can be *any* public IP addresses. Furthermore, it can flood any of the *selected* target links regardless of the distribution of bots, and its server-disconnection effects at a target area are easily predictable. Crossfire is more *persistent* than Coremelt, since

Design Goal	Crossfire	Coremelt
Flexibility of targeting server areas	High	N/G
Bot-distribution independence	Y	N
Persistence · Data vs. control plane distinction · Robustness against route changes	Higher	Lower
Distribution of target links across multiple ISPs	Y	N/G
Indistinguishability at routers	Y	Y*
Undetectability at target area servers	Y	N/G
Reliance on wanted flows only	N	Y

(* only if bot-to-bot flow intensity does not exceed router bounds.
N/G = "Not a design Goal")

Table VII: Crossfire vs. Coremelt [7] Differences

it does not trigger control-plane reaction (e.g., BGP route changes [5]) and it can easily evade route-change countermeasures produced by online traffic engineering. Finally, unlike Coremelt, which targets the backbone routers of an AS, Crossfire aims to select routers and links that are distributed across ASs of different ISPs, such that no single ISP could counter the attack. In short, the Crossfire attack is different from Coremelt as it shares neither all the goals nor the attack techniques of Coremelt. Table VII summarizes the key differences between Crossfire and Coremelt.

C. Large-Scale Connectivity Attacks

The technical underpinnings of the Crossfire attack are also related to research on the robustness of Internet connectivity to attacks that disable routers or links [54, 55, 56]. Albert *et al.* [54] illustrate that if an adversary disables 4% of the highly connected routers, the entire Internet would break up into small isolated pieces. However, later work by Magoni [55] and Wang *et al.* [56] concludes that all such attacks would be *infeasible* because of the huge number of routers or links that need to be disconnected.

The main distinction between the Crossfire attack and this line of work is that our notion of (dis)connectivity captures the practical realities of the Internet; we say that a node A is (effectively) disconnected from a node B whenever the persistent route from A to B is severely congested (viz., Section II-A2).

The Crossfire attack also has a clearly different goal from the routing attack proposed by Bellovin and Gansner that cuts multiple network links to attract a certain traffic to compromised routers for eavesdropping purposes [6].It is also different from the DoS source-detection technique proposed by Burch and Cheswick [57] whereby a victim server attempts to flood various routers and measure decreases in attack traffic – a telltale sign identifying attack sources on the router's path.

D. Brute-Force DDoS Attacks

The goals of the Crossfire attack are fundamentally different from those of conventional brute-force DDoS attacks [1, 2, 29, 58] in at least three respects. First, it has a flexible choice of targets in a much more scalable range than those of DDoS attacks (e.g., from servers of a single enterprise, to those of a state or country). Second, its attack sources (i.e., bot hosts) are undetectable by any targeted servers, since they do not receive attack messages, and by network routers, since they receive only low-intensity, individual flows that are indistinguishable from legitimate flows. Third, its persistence against the same set of targets can be extended virtually indefinitely by changing attack parameters. The Crossfire advantage of the flexible choice of targets in a geographic area is shared by the geo-targeted DDoS attacks in cellular networks proposed by Traynor *et al.* [59]. However, these attacks are less relevant for the Internet.

VIII. Conclusion

The proliferation of bot networks seems unavoidable, for at least as many reasons as that of malware, the primary reason being successful large-scale social engineering scams against unsuspecting users world-wide. End-server bots flooding the Internet router fabric to effectively disconnect other end-server systems is the *penultimate insult* to the end-to-end argument in network design, the ultimate being, of course, the loss of end-to-end trust caused by malware in end-servers [60].

The question of whether it is possible to counter an attack such as Crossfire arises naturally given the current Internet architecture and ISP operations. Preliminary analysis suggests that combinations of multiple countermeasures against such attacks may, in fact, become necessary. As argued in Section IV, no single ISP can counter such an attack whenever the flooded links reside in different ISP domains, regardless of the quality of traffic engineering techniques employed by individual ISPs. Whether ISP coordination becomes practical despite of competitive concerns, remains to be seen. Another possibility would be to support application layer overlays that would route around flooded links by selecting different server routes in response to link-flooding alerts. Yet another possibility would be to deter massive attacks by both preemptive and retaliatory disruption of bot markets with certainty. This would require analysis of bot markets along the lines of described by Caballero *et al.* [41]. Finally, international agreements regarding prosecution of telecommunication-infrastructure attacks may also become necessary [61].

Acknowledgements

We are grateful to Hsu-Chun Hsiao, Yongdae Kim, Adrian Perrig, Vyas Sekar, and the symposium reviewers for their insightful comments and suggestions. This research was supported in part by CyLab at Carnegie Mellon under contract W911NF-09-1-0273 from the US Army Research Office, and by the National Science Foundation (NSF) under grants CNS1040801. The views and conclusions contained in this document are solely those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of any sponsoring institution, the U.S. government or any other entity.

References

[1] V. D. Gligor, “Guaranteeing access in spite of distributed service-flooding attacks,” in *Security Protocols Workshop*, 2003, pp. 80–96.

[2] J. Mirkovic and P. Reiher, “A taxonomy of DDoS attack and DDoS defense mechanisms,” *SIGCOMM Comput. Commun. Rev.*, vol. 34, no. 2, pp. 39–53, Apr. 2004.

[3] F. C. Freiling, T. Holz, and G. Wicherski, “Botnet tracking: exploring a root-cause methodology to prevent distributed denial-of-service attacks,” in *Proceedings of ESORICS’05*. Berlin, Heidelberg: Springer-Verlag, 2005, pp. 319–335.

[4] D. Dagon, G. Gu, C. Lee, and W. Lee, “A taxonomy of botnet structures,” in *Computer Security Applications Conference. ACSAC 2007. Twenty-Third Annual*, dec. 2007, pp. 325 –339.

[5] M. Schuchard, A. Mohaisen, D. Foo Kune, N. Hopper, Y. Kim, and E. Y. Vasserman, “Losing control of the internet: using the data plane to attack the control plane,” in *Proceedings of NDSS 2011*. ACM, 2010, pp. 726–728.

[6] S. M. Bellovin and E. R. Gansner, “Using link cuts to attack internet routing,” *Tech. Rep., ATT Research, 2004, Work in Progress 2003 USENIX*.

[7] A. Studer and A. Perrig, “The Coremelt attack,” in *Proceedings of ESORICS’09*. Berlin, Heidelberg: Springer-Verlag, 2009, pp. 37–52.

[8] J. Nazario, “DDoS attack trends through 2009-2011,” *NANOG 54*, Feb. 2012.

[9] P. Ferguson, “Network Ingress Filtering: Defeating Denial of Service Attacks which employ IP Source Address Spoofing,” *RFC 2827*, 2000.

[10] A. Yaar, A. Perrig, and D. Song, “SIFF: A Stateless Internet Flow Filter to Mitigate DDoS Flooding Attacks,” in *Proceedings of the IEEE Security and Privacy Symposium*, 2004.

[11] Xiaowei Yang and David Wetherall and Thomas Anderson, “A DoS-limiting network architecture,” in *SIGCOMM ’05*, 2005.

[12] R. Moskowitz and P. Nikander, “Host Identity Protocol (HIP) Architecture,” *RFC 4423*, 2006.

[13] D. G. Andersen, H. Balakrishnan, N. Feamster, T. Koponen, D. Moon, and S. Shenker, “Accountable Internet Protocol (AIP),” in *ACM SIGCOMM*, 2008.

[14] J. C.-Y. Chou, B. Lin, S. Sen, and O. Spatscheck, “Proactive Surge Protection: a defense mechanism for bandwidth-based attacks,” *IEEE/ACM Transactions on Networking (TON)*, vol. 17, no. 6, pp. 1711–1723, 2009.

[15] A. Feldmann, A. Greenberg, C. Lund, N. Reingold, J. Rexford, and F. True, “Deriving traffic demands for operational IP networks: methodology and experience,” in *ACM SIGCOMM Computer Communication Review*, vol. 30, no. 4. ACM, 2000, pp. 257–270.

[16] N. Wang, K. Ho, G. Pavlou, and M. Howarth, “An overview of routing optimization for Internet traffic engineering,” *Communications Surveys Tutorials, IEEE*, vol. 10, no. 1, pp. 36 –56, quarter 2008.

[17] K. Levanti, “Routing management in network operations,” Ph.D. dissertation, Carnegie Mellon University, 2012.

[18] M. Faloutsos, P. Faloutsos, and C. Faloutsos, “On power-law relationships of the internet topology,” in *Proceedings of SIGCOMM ’99*. ACM, 1999, pp. 251–262.

[19] A. Lakhina, J. W. Byers, M. Crovella, and P. Xie, “Sampling biases in IP topology measurements,” in *Proceedings of INFOCOM*, vol. 1. IEEE, 2003, pp. 332–341.

[20] M. E. Newman, “A measure of betweenness centrality based on random walks,” *Social networks*, vol. 27, no. 1, pp. 39–54, 2005.

[21] I. Cunha, R. Teixeira, and C. Diot, "Measuring and characterizing end-to-end route dynamics in the presence of load balancing," in *Proceedings of PAM'11*. Berlin, Heidelberg: Springer-Verlag, 2011, pp. 235–244.

[22] Y. Amir and C. Danilov, "Reliable communication in overlay networks," *IEEE/IFIP International Conference on Dependable Systems and Networks (DSN 2012)*, vol. 0, p. 511, 2003.

[23] A. D. Keromytis, V. Misra, and D. Rubenstein, "SOS: secure overlay services," in *Proceedings of SIGCOMM '02*. New York, NY, USA: ACM, 2002, pp. 61–72.

[24] J. Sherry, E. Katz-Bassett, M. Pimenova, H. V. Madhyastha, T. Anderson, and A. Krishnamurthy, "Resolving IP aliases with prespecified timestamps," in *Proceedings of IMC '10*. New York, NY, USA: ACM, 2010, pp. 172–178.

[25] W. Willinger, D. Alderson, and J. C. Doyle, "Mathematics and the Internet: A source of enormous confusion and great potential," *American Mathematical Society*, 2009.

[26] B. Augustin, T. Friedman, and R. Teixeira, "Measuring load-balanced paths in the internet," in *Proceedings of IMC '07*. New York, NY, USA: ACM, 2007, pp. 149–160.

[27] V. Paxson, "End-to-end routing behavior in the internet," in *Proceedings on SIGCOMM '96*. New York, NY, USA: ACM, 1996, pp. 25–38.

[28] R. Cohen and L. Katzir, "The generalized maximum coverage problem," *Information Processing Letters*, vol. 108, no. 1, pp. 15 – 22, 2008.

[29] J. Jung, B. Krishnamurthy, and M. Rabinovich, "Flash crowds and denial of service attacks: characterization and implications for CDNs and web sites," in *Proceedings of WWW '02*. New York, NY, USA: ACM, 2002, pp. 293–304.

[30] A. Clauset, C. R. Shalizi, and M. E. J. Newman, "Power-Law Distributions in Empirical Data," *SIAM Review*, vol. 51, no. 4, pp. 661–703, 2009.

[31] O. Narayan and I. Saniee, "Scaling of load in communications networks," *Phys. Rev. E*, vol. 82, p. 036102, Sep 2010.

[32] M. P. H. Stumpf and M. A. Porter, "Critical truths about power laws," *Science*, vol. 335, no. 6069, pp. 665–666, 2012.

[33] S. Ross, *Introduction to Probability and Statistics for Engineers and Scientists*, ser. Wiley series in probability and mathematical statistics. Academic Press/Elsevier, 2009.

[34] A. Shaikh, A. Varma, L. Kalampoukas, and R. Dube, "Routing stability in congested networks: Experimentation and analysis," in *Proc. of ACM SIGCOMM*, 2000, pp. 163–174.

[35] P. Francois, C. Filsfils, J. Evans, and O. Bonaventure, "Achieving sub-second IGP convergence in large IP networks," *SIGCOMM CCR*, vol. 35, no. 3, pp. 35–44, Jul. 2005.

[36] G. Iannaccone, C.-N. Chuah, S. Bhattacharyya, and C. Diot, "Feasibility of ip restoration in a tier 1 backbone," *Network, IEEE*, vol. 18, no. 2, pp. 13 – 19, mar-apr 2004.

[37] B. Fortz, J. Rexford, and M. Thorup, "Traffic engineering with traditional IP routing protocols," *Communications Magazine, IEEE*, vol. 40, no. 10, pp. 118 – 124, oct 2002.

[38] B. Davie and A. Farrel, *MPLS: Next Steps*, ser. Morgan Kaufmann Series in Networking. Elsevier/Morgan Kaufmann Publishers, 2008.

[39] T. Nadeau, *MPLS Network Management: MIBs, Tools, and Techniques*, ser. Morgan Kaufmann Series in Networking. Elsevier Science, 2002.

[40] N. Hu, L. E. Li, Z. M. Mao, P. Steenkiste, and J. Wang, "Locating internet bottlenecks: algorithms, measurements, and implications," in *Proceedings of SIGCOMM '04*. New York, NY, USA: ACM, 2004, pp. 41–54.

[41] J. Caballero, C. Grier, C. Kreibich, and V. Paxson, "Measuring Pay-per-Install: The Commoditization of Malware Distribution," in *Proceedings of the 20th USENIX Security Symposium*, Aug. 2011.

[42] *IEEE Policies*. IEEE, February 2012, ch. 7.8 IEEE Code of Ethics.

[43] "ACM Code of ethics and professional conduct," *Commun. ACM*, vol. 35, no. 5, pp. 94–99, May 1992.

[44] FBI National Press Office, "Over one million potential victims of botnet cyber crime," http://www.fbi.gov/news/pressrel/press-releases/over-1-million-potential-victims-of-botnet-cyber-crime, June 13, 2007.

[45] PlanetLab., "http://www.planet-lab.org/."

[46] Traceroute.org, "Public route server and looking glass site list," http://www.traceroute.org/.

[47] L. Subramanian, S. Agarwal, J. Rexford, and R. Katz, "Characterizing the internet hierarchy from multiple vantage points," in *Proceedings of INFOCOM 2002*, vol. 2, 2002, pp. 618 – 627 vol.2.

[48] F. Wang and L. Gao, "On inferring and characterizing Internet routing policies," in *Proceedings of IMC '03*. New York, NY, USA: ACM, 2003, pp. 15–26.

[49] B. Augustin, B. Krishnamurthy, and W. Willinger, "IXPs: mapped?" in *Proceedings of IMC '09*. New York, NY, USA: ACM, 2009, pp. 336–349.

[50] D. Dagon, C. Zou, and W. Lee, "Modeling botnet propagation using time zones," in *In Proceedings of the 13 th Network and Distributed System Security Symposium NDSS*, 2006.

[51] S. Staniford, V. Paxson, and N. Weaver, "How to own the Internet in your spare time," in *Proceedings of the 11th USENIX Security Symposium*. Berkeley, CA, USA: USENIX Association, 2002, pp. 149–167.

[52] R. Mahajan, S. M. Bellovin, S. Floyd, J. Ioannidis, V. Paxson, and S. Shenker, "Controlling high bandwidth aggregates in the network," *SIGCOMM Comput. Commun. Rev.*, vol. 32, no. 3, pp. 62–73, Jul. 2002.

[53] Y. Zhang, Z. M. Mao, and J. Wang, "Low-rate TCP-targeted DoS attack disrupts internet routing," in *Proc. 14th Annual Network & Distributed System Security Symposium*, 2007.

[54] R. Albert, H. Jeong, and A.-L. Barabasi, "Error and attack tolerance of complex networks," *NATURE*, vol. 406, p. 378, 2000.

[55] D. Magoni, "Tearing down the Internet," *Selected Areas in Communications, IEEE Journal on*, vol. 21, no. 6, pp. 949 – 960, aug. 2003.

[56] Y. Wang, S. Xiao, G. Xiao, X. Fu, and T. H. Cheng, "Robustness of complex communication networks under link attacks," in *Proceedings of ICAIT '08*. New York, NY, USA: ACM, 2008, pp. 61:1–61:7.

[57] H. Burch and B. Cheswick, "Tracing anonymous packets to their approximate source," in *Proceedings of the 2000 Usenix LISA Conference*, 2000, pp. 319–327.

[58] D. Moore, G. Voelker, and S. Savage, "Inferring Internet Denial-of-Service Activity," in *Proceedings of the 10th Usenix Security Symposium*, 2001, pp. 9–22.

[59] P. Traynor, W. Enck, P. Mcdaniel, and T. La Porta, "Exploiting open functionality in SMS-capable cellular networks," *Journal of Computer Security*, vol. 16, no. 6, pp. 713–742, 2008.

[60] D. D. Clark and M. S. Blumenthal, "The end-to-end argument and application design: The role of trust," in *Federal Communications Law Journal*, vol. 63, 2011, pp. 357–390.

[61] International Telecommunication Union, "http://www.itu.int."