8-2020

Leveraging Conventional Internet Routing Protocol Behavior to Defeat DDoS and Adverse Networking Conditions

Jared M. Smith
jms@vols.utk.edu

To the Graduate Council:

I am submitting herewith a dissertation written by Jared M. Smith entitled "Leveraging Conventional Internet Routing Protocol Behavior to Defeat DDoS and Adverse Networking Conditions." I have examined the final electronic copy of this dissertation for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Doctor of Philosophy, with a major in Computer Science.

Maxfield J. Schuchard, Major Professor

We have read this dissertation and recommend its acceptance:

Scott Ruoti, Audris Mockus, Bryan Lyles, Jeff Nichols

Accepted for the Council:

Dixie L. Thompson

Vice Provost and Dean of the Graduate School

(Original signatures are on file with official student records.)

Leveraging Conventional Internet Routing Protocol Behavior to Defeat DDoS and Adverse Networking Conditions

A Dissertation Presented for the

Doctor of Philosophy

Degree

The University of Tennessee, Knoxville

Jared Michael Smith

August 2020

To Kaleigh

This PhD is yours too

Acknowledgments

The long road that I began several years ago to earn this degree was full of anxiety, uncertainty, and lots and lots of stress. However, the road to this moment was filled more with endless laughs with my colleagues, massive bar trivia winnings, fruitful collaborations, and a sense of hope and patience that things would work themselves out. If there is anything I am coming out of this process with other than more knowledge than anyone in the world on this most obscure of topics, its that we **need** others to pull us through when we cannot do it by ourselves. Personally, I also needed God in the darkest times, and without realizing, during the good times.

To that end, I had a lot of help to get here. I want to first thank my wife Kaleigh for walking alongside me during this journey. Without her, none of this would have been possible (I would have surely already ended up in a ditch on the side of the road after having an anxiety attack). I haven't given her enough credit for the intellectual contributions (and endless editing) she has contributed to my ideas, papers, and rants along the way. She is the *ultimate* partner, and this might as well be her PhD too.

Next, I was mentored by an advisor that taught me much more than just how to be an academic and independent researcher. Max Schuchard showed me how to push through the

thickest and most crushing of times by allowing me to retain hope throughout the process. He taught me that life is not all about work, more than anyone else has ever made me realize that. He taught me many valuable skills that I have already been employing and will continue to employ when mentoring my own students. He taught me to focus, and that is especially hard to teach to someone that gets as distracted by shiny things and new opportunities as I do. Finally, Max was a friend, and I do not think I would have made it through this PhD if he was not my friend.

I especially want to thank my parents, Jeff and Danyelle, and my brother, Noah. My parents raised me to be curious about the world and would answer *endless* questions I had growing up. I will always remember the trips my dad took me on to the Kennedy Space Center as a child, which inspired my passion for science. My mom, on the other hand, has always helped me focus on my health and sanity, and without her, and all the years of her raising me, I would not be where I am. She taught me an insanely lot about balance, which was absolutely necessary for making it through this PhD. My parents and brother have always shown interest in my work and stressed that I needed to push through while realizing my limits, but to still stay grounded to the other important parts of life. My brother Noah has always been around if I have needed him, and won't let me talk about my school work for much longer than a few minutes, which has helped distract me from the tougher times.

My PhD committee, Dr. Bryan Lyles, Dr. Audris Mockus, Dr. Scott Ruoti, and Dr. Jeff Nichols, have all contributed greatly to my development as a person and academic since I started this endeavor and even before as an undergraduate. I am grateful especially for Bryan's many hours of discussions he has taken out of his day to have with me about my PhD while I worked at ORNL. Jeff deserves extra thanks for coming onto my committee last

minute to handle what I hope was the last time UT handed me a very annoying administrative obstacle. I would also like to thank Dr. Justin Beaver, my former group leader at Oak Ridge National Lab, who always gave me the slack (and accompanying motivation) to focus on my PhD while still being a staff member. After Max, Justin has been the most impactful to my professional direction in my career of all the people I have worked with. Dr. Bobby Bridges taught me a lot about math and getting outside, and for that I am extremely thankful. Of my ORNL mentors, the last one I would like to thank is Dr. Jason Carter, since he was one of the first people at ORNL to tell me to focus on my PhD as much as possible and to enjoy the experience while it lasted.

I want to thank my friends outside of school that got me through this degree by distracting me with long days on the lake, heartwarming small groups, and scorching football game tailgates. Deery Street crew, you all played a much larger role in me getting this far than you probably think. I am extremely thankful for each of you, especially the guys. Joseph, Garza, Kyle - thank you so much for sticking with me all these years, I know I am a lot to deal with. Sam, Nick, Rob, Toye, Emma, Summer, Stephanie, Erika - you all are hilarious and refreshing, thank you.

Finally, and I leave them for last because they will always be in the back of my mind, I want to thank my friends from VolSec. Not only are they some of my best friends, but they are the smartest, most dedicated collaborators I have ever had, and we had a blast over the last few years. Tyler, Joseph, Savannah, Kyle, Parker, and Jordan: I cannot thank you each enough for all you have done for me and my sanity. You all are literally the best.

Abstract

The Internet supports the livelihoods, businesses, governments, and critical infrastructure that keep our modern society moving. The Internet, at its most basic level, uses purpose-built computers to deliver messages between parts of the Internet, called routers. To build paths which these messages will travel, routers carry out a routing protocol called the Border Gateway Protocol, or BGP. Unfortunately, the Internet's success at using BGP and other protocols to connect the unconnected has made it an exceptionally valuable target for adversaries, from nation-states to novice computer users. Increasingly devastating Distributed Denial of Service attacks, or DDoS, continue to bring down core Internet services and websites. Yet, to date, the only viable solutions for these attacks are excessively expensive, require an Internet redesign, or require cooperation among routers. This dissertation focuses on examining the following thesis statement. *Rather than seek to redefine the way the Internet works to combat advanced DDoS attacks, we can leverage conventional Internet routing behavior via BGP to mitigate modern distributed denial of service attacks.*

The research in this work breaks down into a single arc with three independent, but connected thrusts. These thrusts demonstrate that the aforementioned thesis is *possible, practical, and useful.* The first thrust demonstrates that this thesis is *possible* by building

and evaluating Nyx, a system able to protect Internet networks from DDoS using BGP, without an Internet redesign and without cooperation from other networks. We show that Nyx is effective in simulation for protecting Internet networks and end users from the impact of advanced forms of DDoS. The second thrust examines the real-world *practicality* of Nyx, as well as other systems which rely on real-world BGP behavior. Through a comprehensive set of active Internet measurements, this second thrust confirms that Nyx works effectively in practice beyond simulation as well as revealing novel insights about the effectiveness of other Internet security defensive and offensive systems. We then follow these live measurements by evaluating Nyx under the real-world routing constraints discovered in practice. The third thrust explores the *usefulness* of Nyx for mitigating DDoS against critical U.S. energy infrastructure. After first exposing the latent vulnerability of U.S. electric utilities to DDoS, we explore how Nyx can protect these utilities. This final thrust finds that the current set of exposed U.S. power facilities are widely vulnerable to DDoS that could induce blackouts, and that Nyx can be leveraged to reduce the impact of these targeted DDoS attacks.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

The Internet is a cornerstone of modern society. At a high level, the Internet is composed of many networks, run by human operators. These networks are called *autonomous systems*, or ASes. Core to the Internet's functionality is the way in which traffic on the Internet gets from one destination to another across these ASes. Special purpose computers, called routers, transit traffic made up of *packets* along paths between these ASes by following a *routing protocol*. The routing protocol that defines how traffic travels along these paths is known as the Border Gateway Protocol, or BGP. BGP, along with the other protocols underpinning the modern Internet, allows every phone, computer, tablet, supercomputer, smart device, and every other Internet-connected device to reach out and interact with other devices and online services in the world. Without this inherent ability to connect devices and exchange data, modern society would not exist in the way it does today.

However, the Internet is under attack. When these attacks succeed, the livelihoods, businesses, governments, and infrastructure that rely the Internet also come under attack. Increasingly devastating attacks against the Internet threaten to undermine the Internet's success at using BGP and other protocols to connect the unconnected. Of all the adversarial campaigns waged against the Internet and the organizations that rely on it, distributed denial of service, or DDoS, tops the list of the most volatile attacks. In recent years, DDoS attacks have been responsible for large swaths of the Internet blacking out, while other attacks have completely overwhelmed key Internet services, websites, and even entire countries [3, 6, 4, 88].

These DDoS attacks have grown in prevalence, reaching up to terabits per second (Tbps) in volume [68]. For perspective, the largest ever DDoS attack known to date was capable of 1.3 Tbps, sending malicious messages intended to overwhelm the victim at a rate of 126.9 million per second [30]. This is *25,000 times* the rate which the average internet-connected

household can send packets to the Internet. Punishing DDoS attacks threaten to unwind the progress made so far in connecting our society. Critically, as adversaries devise new DDoS attack types and build larger networks of bots to execute them, nations and companies continue to build both implicit and explicit reliance on the Internet into their systems and products. This reliance, coupled with the growing presence of DDoS-capable adversaries, has put the Internet in a dangerous place.

Yet, to date, the only viable solutions to these DDoS attacks are either excessively expensive, require an Internet redesign, or require cooperation among ASes. What solution then can ASes rely upon to protect themselves from DDoS when they neither have the resources nor ability to purchase protection or cooperate with others? Even if an AS can purchase DDoS protection, advanced DDoS attacks known as Link Flooding Attacks (LFAs) can overwhelm core Internet paths *without* the ability for a defender to mitigate them with existing DDoS solutions. How then can the Internet adapt to defeat these attacks, including advanced LFAs, without sacrificing the deployability of such defenses? Can networks on the Internet rely on an easily deployable solution to mitigate advanced DDoS, while also not sacrificing the quality of service needed to support the demands of modern society's reliance on the Internet?

1.1 Thesis Statement

The central thesis of this dissertation is as follows:

Rather than seek to redefine the way the Internet works to combat advanced DDoS attacks, we can leverage conventional Internet routing behavior via BGP to mitigate modern

distributed denial of service attacks.

My research supporting this thesis breaks down into a single arc with three independent, but connected thrusts. These thrusts demonstrate that the aforementioned thesis is *possible, practical, and useful.* The first thrust demonstrates that this thesis is *possible* by building Nyx, a system which can employ implicit BGP behavior to route around congestion caused by both simple and advanced DDoS attacks. The second thrust explores the *practicality* of the core techniques used in Nyx and other Internet security offensive and defensive systems with live Internet measurements. We then re-evaluate Nyx with the practical routing constraints discovered from this measurement study. The third and final thrust explores how U.S. critical energy infrastructure relies on the Internet for power generation, and how our Nyx can successfully mitigate DDoS against Internet-connected utilities.

1.2 Outline

This dissertation will explore each of these three thrusts. Prior to exploration of these three thrusts, we cover relevant background on BGP, DDoS, and LFAs in Chapter 2. The outline of the three thrusts in this dissertation and their major contributions are as follows:

Routing Around Congestion with Nyx (Chapter 3)

Chapter 3 presents Nyx, the first system to mitigate Link Flooding Attacks and traditional DDoS against Internet links without requiring Internet redesign or cooperation among ASes. This thrust presents an evaluation of Nyx protecting a deploying AS against DDoS from multiple modern botnets, as well as an evaluation of protecting inbound traffic from multiple

remote *critical* networks. Nyx, at its core, builds on a conventional Internet routing behavior known as BGP poisoning, and expands its usage to re-route benign traffic *around congestion* caused by DDoS. The major contribution of this thrust is the Nyx DDoS defense system, which is easily deployable and tested with a wide range of Internet-scale simulations using a purpose-built BGP simulator we built called *Chaos*.

Assessing the Practicality of Nyx and Similar Systems on the Live Internet (Chapter 4 and 5)

Chapter 4 introduces a systematic study of BGP poisoning's practicality on the *actual Internet*. By conducting real BGP advertisements thousands of times from multiple vantage points, we discover where and how well BGP poisoning functions on the live Internet. The major contributions of this thrust lie in our findings that inform the practicality of Nyx, as well as our findings for how well other security systems, including censorship circumvention systems, work under the presence of BGP poisoning. Chapter 5's concludes this second thrust by contributing a detailed analysis of Nyx under the practical routing constraints discovered from our measurement study. This evaluation of Nyx with added practical constraints helps inform the parameters real-world operators should use when deploying Nyx.

Applying Nyx to Defeat DDoS against Internet-reliant Critical Infrastructure (Chapter 6)

Chapter 6 covers the third and final thrust with three key contributions. First, we build a state-of-the-art model for tying U.S. electric utilities to the links on the Internet they rely

on for controlling power generation. Second, we explore how DDoS against these utilities is possible and can be devastating to SCADA communications between the utility's generation sites and the utility's AS. We then examine how successful DDoS from LFAs in the earlier chapters can impact the ability for the power grids maintained by these utilities to respond to sudden increases or decreases in power demand when the DDoS interrupts normal SCADA communications used for responding to these demand increases/decreases. Third, we present a rigorous evaluation of how Nyx can help mitigate the DDoS waged against these utilities by providing the utilities with a deployable and useful tool to route critical SCADA traffic around congestion imposed by adversaries.

Chapter 2

Background

2.1 Internet Routing with BGP

We begin with a discussion of the Internet and how traffic travels on the Internet via its main routing protocol, BGP.

The Internet is composed of many autonomous systems, or ASes, which are sets of routers and IP addresses under singular administrative control [56]. Each AS has one or more IP prefixes allocated to it, containing large amounts of IP addresses (e.g. an /8 or /16 subnet), or they can contain relatively few IPs (e.g. a /23 or /24). Today, a /24 is the most specific, or smallest, prefix recommended to be allowed by the most current best practices documents [39]. The Border Gateway Protocol [114] (BGP) is the de facto routing protocol of the Internet. BGP allows the exchange of information, called advertisements, between ASes about routes to blocks of IP addresses (e.g. prefixes), allowing each AS to have knowledge of how to forward packets toward their destinations. BGP advertisements are confined to the control-plane of the Internet, while protocols such as TCP and UDP are confined to the data-plane.

To carry out the routing decision process, BGP harnesses a path-vector routing algorithm *with policies* to build and propagate AS paths, or routes, via BGP advertisements. Individual routers can define their own policies for which routes are considered "best" and then use the preferred routes to forward packets. In practice, these routes are often not the shortest, but rely on the specific policies defined in router configurations. These can include the cheapest route, the most favorable for congestion directly upstream, or any number of preferences a network operator sets for which upstream AS should be used. Outbound AS-level BGP paths are controlled by using the local routing policy to force a particular installed route

as the first choice. BGP also includes a "loop detection" mechanism, where a BGP router receiving a new advertisement will first scan the entire path, and if it is already on the path, will drop (ignore) the advertisement and refuse to propagate the path to its neighbors.

To stabilize the control plane, mechanisms such as route-flap dampening [151, 112] (RFD) and Minimum Route Advertisement Interval (MRAI) timers [114] limit the number of advertisements a single AS can propagate to amounts capable of being handled by connected ASes. These mechanisms can slow the process of BGP convergence, or the time taken for the Internet to settle on a set of stable routes to destinations based on BGP updates. However, as router processing power has increased, RFD becomes less widely used and is now disabled by default in Cisco routers [102]. Additionally, RIPE recommends setting RFD with a high BGP update suppression threshold [102]. MRAI timers also vary widely in configuration, with a default value of 30 seconds.

When speaking of sets of ASes, often operators group ASes into relative sizes. A widely-adopted AS classification scheme presented in [107] divides ASes into 4 "tiers": Tier 1, Large ISP, Small ISP, and Stub ASes. Tier-1 ASes can transit traffic to all ASes without compensation. From a graph-theoretic perspective, these ASes form a clique. In graph theory, a clique is a subset of vertices in an undirected graph such that every two distinct vertices in the clique are adjacent. For this reason, the Tier-ASes essentially form the "core" of the Internet itself. Large ISPs are those with over 50 customers but without being in the Tier-1 clique. For comparison, a large ISP may have 500 customers while one of the largest Tier-1 ASes has nearly 6,000 (e.g. Cogent). Small ISPs have between 5 and 50 customers. Example of such ASes include large universities at the small end and ISPs that serve many customers, but still buy significant transit from larger ISPs. Finally, Stub ASes, edge ASes,

or fringe ASes, are those with less than 5 customers. We consider all classifications of ASes other than Stub ASes to be in or near the "core" of the Internet, as the vast majority of ASes (over 60,000 out of nearly 70,000) are stub ASes.

2.1.1 RPKI and BGPSec

The Resource Public Key Infrastructure (RPKI) [80] is a feature in partial deployment across the Internet [50, 29] designed to tie ASes to the prefixes they are allowed to originate via Route Origin Authorizations (ROAs). ROAs are designed to prevent prefix hijacking [103]. BGPsec [73] extends BGP with cryptographic hardening of AS PATHs in advertisements. If fully deployed, BGPsec would prevent certain behaviors possible in BGP, such as the ability to lie about what AS is on the path when advertising routes. However, while BGPsec has been specified in RFC 8205, no commercial implementations exist [137], and BGPsec is largely ineffective unless widely implemented [53].

2.2 Distributed Denial of Service

In this section we discuss distributed denial of service (DDoS) attacks in more detail.

2.2.1 DDoS and Botnets

Typical volumetric DDoS attacks provide a high level of impact with a low degree of technical complexity to launch. The ease of executing DDoS has resulted in an increased number of occurrences in recent years. Typically, DDoS attacks have originated from infected hosts on the Internet, as is the case with the Conficker botnet [126]; however, new botnets are often

originating primarily in IoT-based devices, such as Mirai [143], or SCADA sources such as Blackenergy [99]. We analyzed the distributtion of these botnets, and we found that these botnets are concentrated in a small number of ASes.

This can be seen by the distribution in Figure 2.1. The plot uses a Cumulative Distribution Function (CDF) on the y-axis, relative to the number of bots per AS along the x-axis. This plot displays the number of bots per botnet for each of the total 60,000+ ASes from Februrary 2018. These CDF plots will appear throughout this work. In this example, because the distribution spikes near zero bots per AS and only curves further on the y-axis at nearly 0.995 on the y-axis, this means that at the point where a single botnet hits 0.995 on the y-axis, that in fewer than 0.005% of ASes are there more than the number of bots the curve hits on the x-axis. In particular, the Mirai botnet has over *200,000* of its nearly 2.9 million bots in a single AS, which leads to less bots being scattered throughout other diverse ASes.

The bandwidth adversaries can harness to conduct DDoS attacks has been steadily increasing annually. Researchers have observed a more than 140% increase in attacks of greater than 100Gbps [67] from 2015 to 2016, with Mirai generating over *1 Tbps* of malicious traffic on multiple occasions. Historically, traditional DDoS attacks originating primarily in hosts see adversaries sending bot traffic directly at the victim network, forcing traffic at the edge of the victim's network to be dropped, thus significantly degrading quality of service. Throughout this work, we will discuss how Nyx defends against traditional DDoS, which is illustrated in Figure 2.2a.

2.2.2 Link Flooding Attacks

Even with the prevalence of traditional volumetric DDoS, a new DDoS attack strategy has emerged from academia and has been actively executed *in the wild*. These new attacks target core transit links which serve the victim host's *entire network*. These attacks are called *transit-link DDoS* or *Link Flooding Attacks* and are shown in detail in Figure 2.2b. In practice, transit-link DDoS have been seen in recent attacks on the major DNS provider Dyn [3], the prominent security journalist Bryan Krebs with KrebsOnSecurity [4], and the country of Liberia [6]. With transit-link DDoS, the adversary directs bot traffic upstream of the network that is the actual victim, which causes traffic directed to the target to be dropped far ahead of reaching its final destination. Bots directed by transit-link DDoS adversaries address their traffic to networks other than the victim, which ensures that the victim *cannot filter the traffic or blackhole it in any way.*

Examples of these attacks in literature include the Coremelt attack [140] and the Crossfire attack [65]. The Coremelt attack is a transit-link DDoS attack that takes any number of N bots participating in the attack and sets up N^2 connections between them, inflicting significant damage to the transit core of the Internet. At the time of Coremelt's introduction, no other transit-link DDoS attacks existed, but since then, others have emerged, such as the Crossfire attack. Crossfire, in a method similar to Coremelt, directs traffic to "wanted" locations expecting the attack traffic, such that attack traffic can never be dropped or filtered by targeted ASes along the chosen attack paths. By doing so, Crossfire can bring down connections to selected critical servers in the transit-core simply by congesting their available capacity.

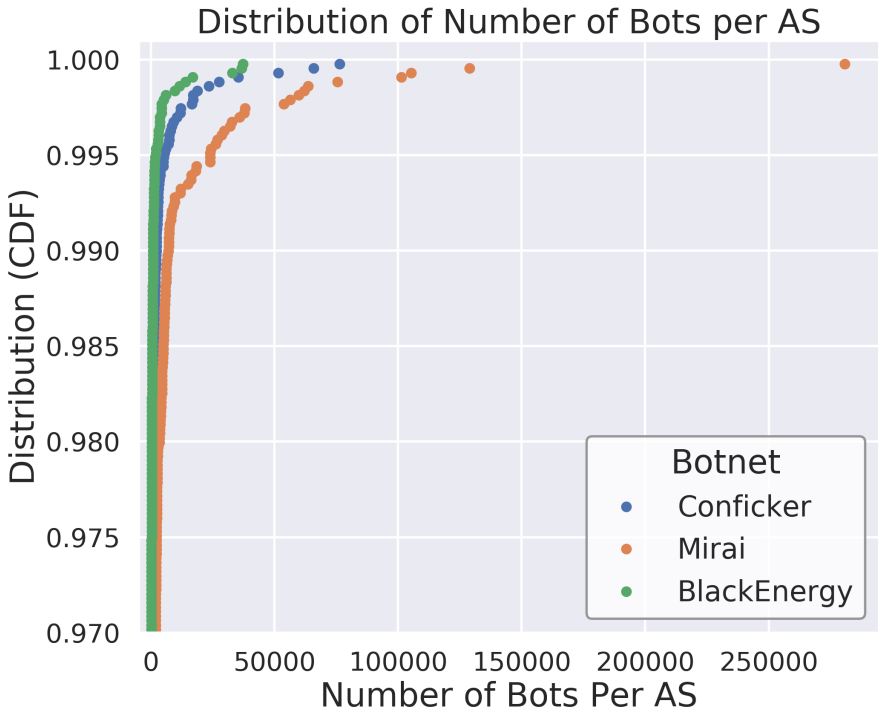
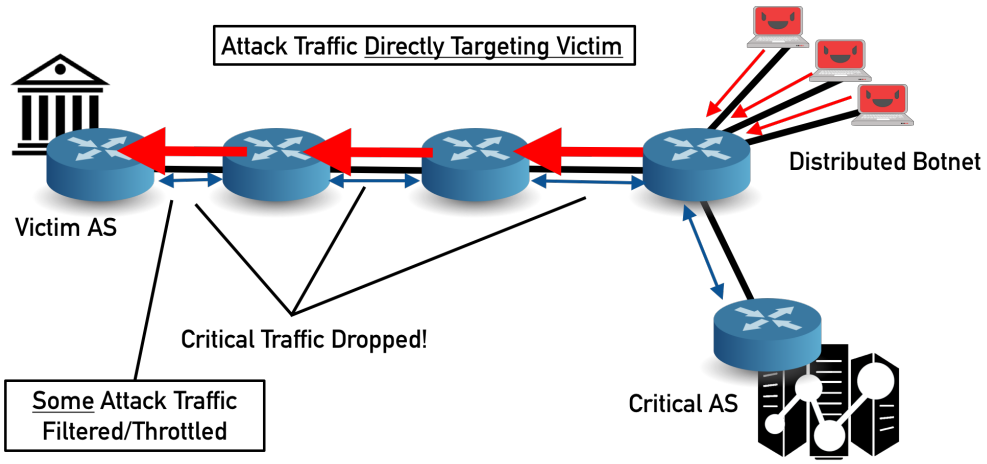
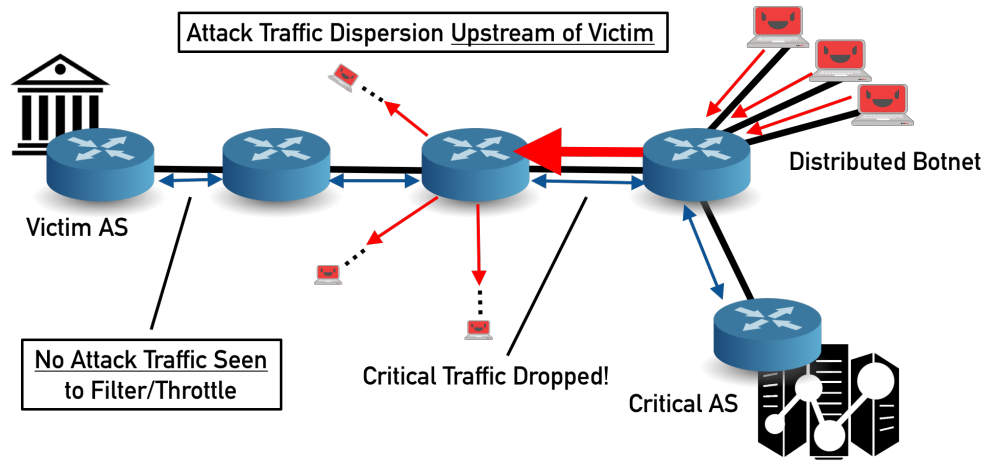


Figure 2.1: Distribution of Bots per Botnet over all ASes on the Internet



(a) Traditional DDoS: The victim AS is directly targeted



(b) Transit-Link DDoS (Link Flooding Attack): Transit AS upstream of the victim is targeted without sending traffic to the victim, thus nullifying the effects of filtering and throttling techniques employed by the victim

Figure 2.2: Traditional and Transit-Link DDoS

Chapter 3

Nyx: Defeating DDoS and Adverse Network Conditions by Routing Around Congestion

3.1 Introduction

Our first set of work focuses on mitigating advanced distributed denial of service (DDoS) attacks using BGP. Due to their high level of impact, combined with low degree of technical complexity, DDoS attacks continue to represent one of the largest unsolved persistent threats on the Internet. Recent successful DDoS attacks by the Mirai botnet against root DNS providers [143] and core transit links [6] highlight both the lack of an effective deployed solution to DDoS attacks and the impact such attacks have on critical network infrastructure. To make matters worse, increased botnet bandwidth has allowed adversaries to launch attacks against shared transit links located *outside* of the intended victim, rather than directly against the victim's end hosts, an attack methodology proposed in academic research by Kang [65] and Studer [140], which we call *transit-link DDoS*.

While DDoS represents one of the oldest and most well known security problems facing the Internet, research has yet to propose a solution that both provides effective mitigation against transit-link DDoS attacks **and** has a realistic deployment scenario. For example, filtering and prioritization techniques [129, 21, 158, 95, 81, 157] require costly per-stream calculations, presenting scalability concerns with modern DDoS attacks. Load balancing and Content Distribution Network (CDN) backed solutions [46] become a test of who possesses more bandwidth, the defender or the adversary, a tenuous proposition in an era of multi-Tbps attack flows, something the Mirai botnet and it's variants have repeatedly achieved. More importantly, *none of these defenses are capable to mitigate DDoS attacks launched against the Internet's transit infrastructure.* Attacks such as Kang's Crossfire are outside of the threat model considered by current DDoS defenses, which focused on protecting the last-mile links

and provide no protection for transit links. Systems such as SCION and SIBRA [163, 19], which integrate DDoS defense into the transit fabric of the Internet, present promise but require a complete redesign of the Internet, raising doubt about their deployability in the foreseeable future.

In this first thrust exploring the main thesis, rather than considering DDoS mitigation as a filtering or prioritization problem, we approach DDoS mitigation as a **routing problem**. We develop a system called **Nyx**, which allows the defending or deploying network, specifically a multi-homed AS, to isolate critical traffic from attack traffic at a path level, preventing the critical traffic from competing against malicious traffic for limited resources. An AS deploying our system, which we term the *Deployer AS*, when negatively impacted by a DDoS attack will adjust the routes of outgoing *and* incoming traffic from a single remote *Critical AS*, known a priori, around links degraded by the DDoS attack. The inbound critical traffic will be routed to non-attacked paths with sufficient capacity using *currently deployed routing protocols*, specifically BGP. This approach to DDoS mitigation has several advantages over existing approaches. Instead of filtering, Nyx instead operates at the route selection level, avoiding costly per-stream decisions. Our system functions independently of the location of the link actually being attacked, even if that link is outside of the deployer's directly connected links, allowing our system to mitigate the impact of DDoS attacks against transit providers the deployer depends on that would normally be outside of the deployer's control. Since our system prevents malicious traffic from DDoS attacks and benign traffic from Critical ASes from being co-located, our capacity to successfully mitigate a DDoS attack in *not dependent* on the volume of malicious traffic, thus allowing our system to succeed against today's large-scale DDoS attacks, often reaching sustained traffic levels of

1 Tbps or more, which no known filtering mechanisms can handle. **Lastly, our system functions using existing routing protocols and protects traffic to and from the deployer without outside assistance**, allowing for a realistic deployment scenario of our system, unlike prior work presented to combat the problem of transit-link DDoS.

In order to realize Nyx, we address three key challenges. First, we address how the AS deploying Nyx, which we also call the *reactor or deployer AS*, can successfully maneuver both outgoing and *incoming* traffic off of attacked links. While altering outgoing paths is trivial, BGP provides the destination no direct way to control incoming paths. We overcome this limitation through the use of currently existing traffic engineering mechanisms, while controlling path propagation via strategicaly lying about the networks on a path in an effort to trigger loop-detection. We utilize an extension of a known traffic engineering technique, BGP poisoning, which we call *Fradulent Route Reverse Poisoning*. BGP poisoning works in the presence of deployed RPKI and is discussed later in Section 3.2. Our solution causes more preferable paths, with respect to packet forwarding, to propagate around, but never actually reaching, the links under a DDoS attack, which we can detect via observing loss of quality service on the links connected to the deployer when the deployer does not actually observe attack traffic directly.

Second, our deployer must limit the number of non-critical networks, which also change their best path as a result of adjusting paths used by critical networks or ASes; a property we term *disturbance*. Disturbance can result in two undesired scenarios. First, disturbance can result in malicious traffic also flowing along the alternate path, resulting in the alternate path itself suffering a DDoS attack. Second, even if the disturbed networks are not sources of attack traffic, too much traffic from ASes other than the chosen critical ASes might congest

the alternative path, as it is likely not provisioned to handle a large amount of traffic beyond normal loads. In order to mitigate disturbance, we expand our path propagation control techniques, preventing propagation of the path to all networks outside of the critical network and the networks appearing along the alternative path.

Lastly, our system needs to ensure that the resulting alternative path has sufficient spare capacity to handle traffic from the critical network, along with traffic from any disturbed networks. If our system detects that the path is struggling to handle the added load, it will attempt to search for a different alternative path. It accomplishes this by withdrawing the alternative path and attempting to re-propagate it, avoiding propagating the route to both links under DDoS attack and the bottleneck links in the previous alternative path. Nyx **does not require** knowledge of either the malicious traffic sources (i.e. the ASes containing malicious bots) or the actual capacity of upstream links to find alternate paths not under attack.

We demonstrate the ability of Nyx to accomplish all three of these tasks using Internet scale simulations in which our system attempts to mitigate a variety of DDoS attack scenarios. We find that it is possible to move critical traffic off attacked links and onto functional paths in 78% of cases where the primary link connecting the deploying AS to the Internet is attacked, which we call *Traditional DDoS* and greater than 98% of all other cases where the attacked link is upstream of the deployer, which we call **Transit-Link DDoS** as illustrated by Kang and Studher with Crossfire and Coremelt [65, 140]. We see that implementing techniques to limit changes in the best path to the deployer of non-critical networks reduces unintended path changes to less preferred networks to 10 networks on average, as opposed to between 1000 and 5000 networks prior to employing reduction

strategies. In addition we find that our system results in little to no added costs with respect to path length, and does not result in best paths switching to less economically advantageous routes. Lastly, we demonstrate that our alternative paths **provide some degree of relief from DDoS attacks in 98% of cases**, and we find that we can move the critical traffic impacted by DDoS attacks onto **completely uncongested paths with in at least 70% of the time**. We also show how Nyx can defend *several* critical ASes, finding that Nyx can still successfully re-route up to 200 critical ASes for a single deployer 80% of the time, and onto a path that is totally uncongested over 70% of the time.

Contributions

We present the following contributions throughout first thrust:

- We present the Nyx DDoS/transit-link DDoS defense system in Section 3.2 and present evaluation results in Section 3.3. Nyx is the first-of-its-kind to leverage the Internet's routing plane to circumvent link flooding attacks.

- We show Nyx works in a multi-critical AS scenario in Section 3.2.7 and present evaluation results in Section 3.4. We find that a single critical AS impacted by transit-link DDoS can be routed onto an uncongested path in 75% of cases. If we consider 500 critical ASes, the deployer can re-route over 30% of critical ASes to completely uncongested paths and to less congested paths over 80% of the time.

- We present our results for multiple botnets, including host-based, IoT-based, and SCADA-based in Sections 3.3-3.4.

- Finally, we open source our modern BGP simulator with bandwidth and botnet models for reproducibility at https://github.com/volsec/chaos.

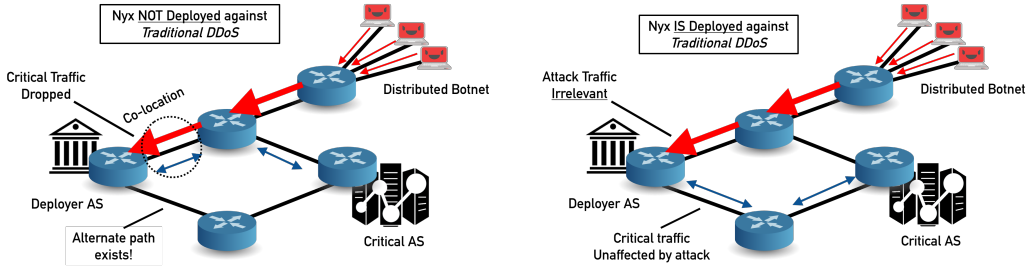
The rest of this chapter is laid out as follows. Section 3.2 will present details of our system design, including design constraints, our approach to DDoS mitigation, and the mechanisms by which we realize our mitigation strategy. Section 3.3 will cover details of our simulation methodology and the results of simulations testing the viability of our system. Section 3.4 covers Nyx protecting multiple critical ASes. Lastly, in Section 3.5 we will compare our system to other DDoS mitigation systems.

3.2 System Design

3.2.1 Routing Around DDoS

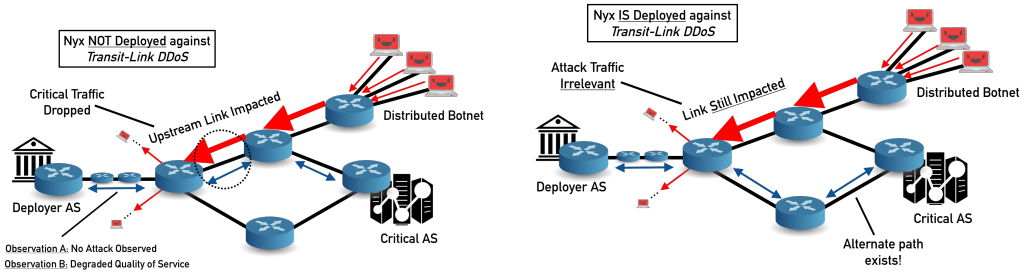
To combat the unmitigated threat posed by transit-link DDoS, we have designed **Nyx**, a system that mitigates DDoS attacks by routing traffic between a Nyx deployer and a chosen critical AS known ahead of time, around links degraded by a DDoS attack or other adverse network conditions. By operating on a per-route basis, rather than a costly per-stream basis, Nyx utilizes BGP at the deployer to route around DDoS without adversely affecting the routing information of other ASes *and* by moving traffic inbound for the critical AS onto new paths with sufficient capacity to handle the added load. At a high level, Nyx makes attack traffic from botnets *irrelevant*, achieving the property of **botnet-level independence**. The ability of Nyx to route around DDoS and make attack flows irrelevant is illustrated in Figure 3.1 for Traditional DDoS and Figure 3.2 for Transit-Link DDoS.

Recall that **traffic filtering and prioritization are ineffective** against modern DDoS with multi-Tbps traffic flows. Furthermore, the transit-link DDoS attacks proposed in literature, Crossfire and Coremelt [65, 140], as well as real-world attacks seen against Liberia [6], do not send their attack traffic directly to the targeted AS, thus eliminating the possibility of applying any filtering or prioritization technique to incoming traffic since critical traffic is dropped upstream and typically outside the control of the victim AS. Nyx approaches the problem differently, by focusing on the problem of *route selection*, utilizing normal BGP and traffic manipulation techniques to **route around DDoS**. By continually selecting alternate paths with the ability to handle traffic otherwise due to be dropped on congested links, Nyx does not rely on existing filtering or prioritization techniques.



(a) Nyx NOT Deployed **(b)** Nyx IS Deployed

Figure 3.1: Nyx Deployment Against **Traditional DDoS**



(a) Nyx NOT Deployed **(b)** Nyx IS Deployed

Figure 3.2: Nyx Deployment Against **Transit-Link DDoS**

3.2.2 Realistic Deployment

Unlike prior systems which mitigate transit-link DDoS via bandwidth contracts [19], Nyx requires no outside cooperation from other ASes, including the critical AS. Furthermore, Nyx does *not* have any knowledge of where attackers originate. Nyx only assumes it knows the AS relationships via open-source data from CAIDA [150]. In Tables 3.1 and 3.2 we show the information required and not required by Nyx. In detail, Nyx does *not* need information about the bandwidth or capacity of links on the Internet. The simulator which we use to validate Nyx utilizes a bandwidth model for the capacity of links in the topology, but this information is not known to the deployer AS or Nyx. Our system also does not have knowledge about the location of bots, which ASes have bots, and where in the Internet bots live; instead, Nyx continues to use packet flow performance as an indicator that the current path between the critical AS and the deployer AS is congested. When Nyx discovers the current path is congested, we use our strategies to route around DDoS and attempt to find, via an evolutionary algorithm, an alternate path with sufficient capacity, as we will discuss later in Section 3.2.6. Finally, Nyx does not need to know what traffic is malicious or benign, since our system knows the critical AS a priori and treats all traffic from that AS as "benign". By forcing traffic from the critical AS onto a path outside of the sphere of influence of a DDoS event or other adverse network conditions, malicious traffic is completely irrelevant due to Nyx's ability to route *around* links impacted by malicious botnet traffic, which gives Nyx the property of botnet-size independence when mitigating DDoS.

Beyond the information Nyx does and does not know, we make the following assumptions about the deployment of Nyx in practice:

- Nyx should only require the defending AS to deploy Nyx. This means we do not rely on a full deployment of our system across the Internet to work. This means that our critical AS will **not** provide our defender any assistance, nor will any other ASes on the Internet, which is a key feature of Nyx that distinguishes our system from any prior work proposing to mitigate transit-link DDoS.

- Nyx should not negatively impact other ASes. Nyx should not alter any paths outside of routes to and from the defender.

- Nyx should not significantly impact other ASes' normal activities. In order to utilize our techniques, the AS operator solely needs to be able to control the BGP advertisements on the routers that are BGP speakers for the deployer AS.

- Nyx should function without any changes to BGP, since the technique we have devised to manipulate inbound traffic from known critical ASes can be performed only via adjustment of routing policies at the deployer.

3.2.3 Adversarial Model

In accordance with how traditional DDoS *and* transit-link DDoS are typically controlled, our adversary does not control the underlying network structure and is *not routing-aware*, thus unable to make routing decisions. Instead, our threat model considers adversaries which control massive distributed botnets or a subset of hosts with the ability to generate massive attack flows. With this restriction, the adversary can control the selection of bots for a particular attack, how much traffic the bots distributed across the Internet will send, and where in the topology each bot should send its traffic. In our current adversarial model, we

did not consider a global adversary in the design of Nyx. As mentioned earlier and shown in Table 3.2, Nyx does not know where the bot ASes live, how much traffic they are sending for a given attack, or the quantity of malicious bots in a given attack.

In the rest of this section, we will explore how Nyx achieves its three core goals within the design restrictions we have placed to ensure deployability and resistance to adversaries.

In Section 3.2.4 we will examine how Nyx adjusts incoming traffic to alternative paths, which is a functionality not controllable directly within BGP. When Nyx successfully migrates critical traffic off of links suffering DDoS we call this **routing success**, and will discuss our evaluation of routing success later in Section 3.3.3. Next, in Section 3.2.5 we look at how Nyx reduces **disturbance**, where ASes outside the the critical AS and those along the alternative path switch to the alternative path. Finally, in Section 3.2.6, we establish how Nyx attempts to maximize the number of instances of where the new link has sufficient capacity to handle the critical traffic.

3.2.4 Migrating Critical Traffic with BGP Poisoning

Recall from earlier in Chapter 2 that outbound traffic from an AS is trivial to adjust via local preferences at the external BGP router; however, manipulating the paths inbound traffic takes to an AS would typically only be possible via coordination between the ASes on either end, as existing systems such as SCION and SIBRA do to route around DDoS [163, 19]. Nyx, however assumes *no coordination* between the deployer AS and any other AS, specifically the critical AS. The deployer cannot directly adjust the local preferences of the critical AS to traverse links which avoid DDoS attacks and other adverse network conditions. We address

Table 3.1: Information Needed by Nyx

Information Needed	How Nyx Uses Information	Information Source
Critical AS	Traffic from Critical AS moved around degraded or attacked links	Chosen by Deployer AS
Paths between Deployer AS and Critical AS	Alternate, non-degraded paths between Critical AS and Deployer AS chosen based on any known paths	Deployer BGP speaker's Routing Information Base (RIB)
Packet flow performance	Used to detect service degradation due to DDoS event or adverse network conditions over alternate paths	OpenFlow
ASes bordering alternate paths between Deployer AS and Critical AS	BGP loop detection is used during BGP poisoning to reduce disturbance by appending ASes bordering alternate paths	Deployer BGP speaker's Routing Information Base (RIB) and Inferred AS Relationships Data from CAIDA [150]

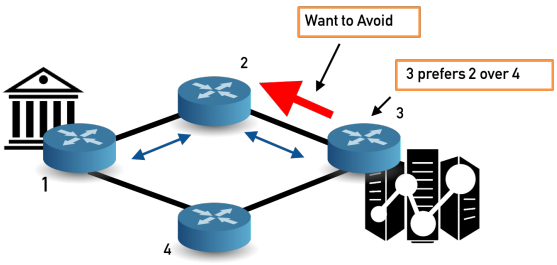
Table 3.2: Information **NOT** Needed by Nyx

Information Not Needed	How Nyx Works Without
Bandwidth/Capacity of links in the Internet	Packet flow performance used as a proxy for congestion
Location of malicious bots and botnets in the Internet	Nyx continually discovers alternate paths until a path with sufficient capacity is found, without ever knowing the attack sources
Malicious and benign traffic	Nyx considers traffic from critical AS, known ahead of time as, "benign", without needing to know malicious traffic

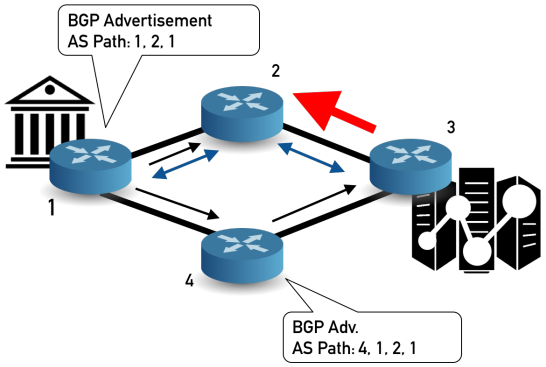
this issue by giving the deployer AS the ability to *restrict* the AS-level paths the critical AS can take to the deployer to only paths which *do not traverse the congested or attacked links* within the topology, such as those affected by traditional or transit-link DDoS attacks. We do this without restricting the critical ASes connectivity to any other ASes, and without causing the critical AS to see any less BGP advertisements from ASes other than the deployer. At a high level, Nyx strives to *route around DDoS* as illustrated in Figures 3.1 and 3.2, where we show how Nyx makes attack events and congested links irrelevant, as critical traffic headed to the deployer is forced onto uncongested, alternate paths.

To give the deployer this ability, we have developed a strategy used by Nyx called **Fraudulent Route Reverse Poisoning** (FRRP), or more simply, *BGP Poisoning*. Nyx employs BGP poisoning to ensure that any BGP advertisements which propagate to the critical AS, originated by the deployer AS, are guaranteed to *not* traverse links that are congested or under attack from DDoS or adverse conditions such as broken links or surges in bandwidth usage creating congestion. BGP poisoning takes away the choice of the critical AS to route outbound traffic headed to the critical over the attacked links by ensuring advertisements which originate at the deployer do not reveal the paths with attacked links.

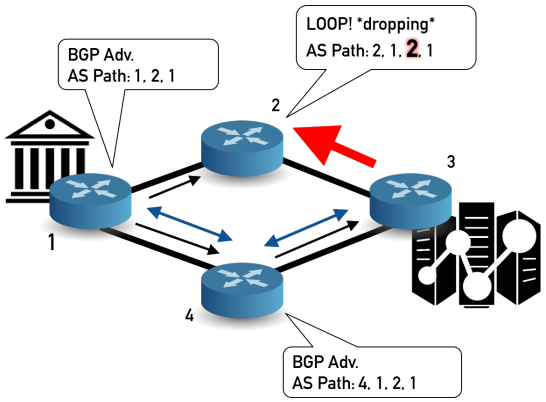
In detail, BGP poisoning is illustrated in Figure 3.3 and works as follows: the normal traffic from the critical AS 3 to deployer AS 1 usually flows over AS 2 from 3, since the critical AS prefers using AS 2 over AS 4 (shown by Part 3.3a). However, attack traffic has congested the link from 3 to 2. In order to avoid this link and route the critical traffic over AS 4, the deployer lies about the path by appending AS 2 to its BGP advertisements. The deployer also appends it's own AS number to the end of the path, which as we will discuss shortly, allows BGP poisoning to function under deployed RPKI. When AS 4 receives this



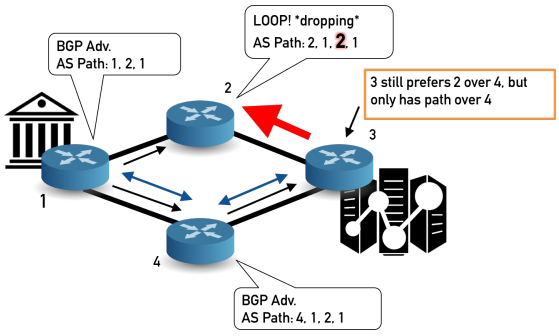
(a) Critical links are congested



(b) Lying about paths and appending ASes to avoid



(c) Loop detection



(d) Critical AS now traverses alternate path

Figure 3.3: BGP Poisoning

path, it advertises it to AS 3 (as shown in Part 3.3b). When AS 2 sees that itself is in the path advertised from the deployer, BGP's built-in loop detection causes AS 2 to not forward its route to AS 1 (shown by Part 3.3c). Thus, the critical AS 3 will no longer see the path to 1 over 2, and it will use it's only other available path, which is over AS 4 (shown in Part 3.3d). Nyx utilizes BGP poisoning at an *Internet-scale* to migrate incoming traffic from a chosen critical AS onto alternate paths in situations where many alternate paths exist.

By using BGP poisoning, we achieve over 98% success for the ability to move traffic off of links under DDoS. Figure 3.1 shows Nyx both deployed and not deployed against a traditional DDoS attack, and Figure 3.2 shows Nyx both deployed and not deployed against Transit-Link DDoS. In both cases, Nyx utilizes BGP Poisoning to achieve reactive route selection and subvert attacked links, rather than relying on filtering or prioritization of traffic from the critical AS.

BGP Poisoning and Network Connectivity

In order to maintain network connectivity, the deployer still advertises it's normal paths, but the poisoned paths will be hole-punched prefixes as discussed earlier in Section 2.1. The deployer will advertise normal aggregates to maintain connectivity to ASes other than the critical, and will utilize de-aggregate advertisements for BGP Poisoning via hole punching. BGP Poisoning coupled with hole-punching ensures that the deployer running Nyx can successfully manipulate traffic inbound from the critical AS without losing any connectivity to other ASes.

As discussed in this section, BGP Poisoning gives Nyx the ability to route around DDoS attacks and adverse network conditions. Whether the alternate paths can handle the added

Table 3.3: References to Loop Detection Support in BGP Implementations

BGP Implementation	Type	Loop Detection Support Reference
Cisco	Commercial	https://www.cisco.com/c/en/us/support/docs/ip/border-gateway-protocol-bgp/13753-25.html
Juniper	Commercial	https://www.juniper.net/documentation/en_US/junos/topics/reference/configuration-statement/loops-edit-protocols-bgp-family.html
Arista	Commercial	https://www.arista.com/en/um-eos/eos-section-33-4-bgp-commands#ww1116932
Quagga/Zebra	Open-Source	https://github.com/Quagga/quagga/blob/88d6516676cbcefb6ecdc1828cf59ba3a6e5fe7b/bgpd/bgp_aspath.c#L1245
BIRD	Open-Source	https://gitlab.labs.nic.cz/labs/bird/-/blob/master/proto/bgp/attrs.c#L1365
FRRouting	Open-Source	https://github.com/FRRouting/frr/blob/5c83709171f4e61e2639a9f93d2d194c0f3ad76d/bgpd/bgp_route.c#L1727
OpenBGPD	Open-Source	https://cvsweb.openbsd.org/src/usr.sbin/bgpd/rde.c?rev=1.502 (search for "loop")

load is discussed later in Section 3.2.6. Before exploring this issue, we first examine the ability of Nyx to reduce the side-effects of utilizing BGP Poisoning in the next section.

Router Support for BGP Poisoning (via Loop Detection)

Given that the BGP RFC [114] specifies loop detection, router frameworks, both open-source and commercial, should implement this feature. While we cannot peer into commercial router source code and firmware from Cisco, Juniper, or other vendors, we can examine the major open-source router software projects. For commercial vendors, we examine the documentation for Cisco, Juniper, and Arista, three of the largest networking hardware vendors. Each commercial vendor surveyed by default drops paths with the router's own AS in the path. The main open-source router projects include Quagga/Zebra, BIRD, FRRouting, and OpenBGPD (part of OpenBSD). The software for each of these projects includes specific source code for abiding by BGP's loop detection feature. The findings are documented in Table 3.3.

Effects of BGP Hold Timers, Route Flap Dampening, and BGP Convergence

BGP hold timers, minimum advertisement intervals, and Route Flap Dampening (RFD) are mechanisms built into BGP designed to protect Internet's routing infrastructure from being overwhelmed. The former is a timing mechanism to keep BGP routers connected to

each other when KEEPALIVE messages are exchanged and several successive KEEPALIVE messages fail to be received. Advertisement intervals are timers that set a minimum delay between updates for a neighboring BGP session. Finally, RFD can detect when a neighboring BGP session is repeatedly sending BGP updates too quickly, and timeout that neighbor [1].

BGP hold timers are defined in Cisco to be 30 seconds, and the Minimum Route Advertisement Interval (MRAI) is set to 30 seconds for external peers and 0 seconds for internal BGP peers. This amount of time would not prevent Nyx from leveraging BGP poisoning to re-route remote ASes. While RFD was initially proposed to help limit flapping updates, the RFD mechanism has been disabled by many network operators according to recent RFCs [112].

All of these safety mechanisms impact BGP's convergence time, which is roughly the time it takes once a BGP advertisement is sent for the rest of the Internet to update their routing tables, if necessary, based on that update. Though we discuss the effects of BGP convergence later in Chapter 4, we show in Figure 3.4a and Figure 3.4b graphs of poisoned vs. normal BGP updates received by BGP route collectors in May 2020 using our measurement infrastructure presented in the next chapter (Chapter 4). Clearly, BGP updates advertised take little time (less than 30 seconds) to spread throughout the Internet, and poisoned paths are no different.

While RFD would not be a problem for Nyx, and hold timers are sufficiently low that they should not affect the system, an AS leveraging Nyx can limit their advertisements to account for these safety controls. By only advertising poisoned paths per deployer at low rates until

[1] https://networkgeekstuff.com/networking/cisco-bgp-timers-re-explained/ and https://www.noction.com/blog/bgp-timers

an alternate route is found via BGP poisoning, the Nyx deployer can avoid triggering RFD and other timing mechanisms.

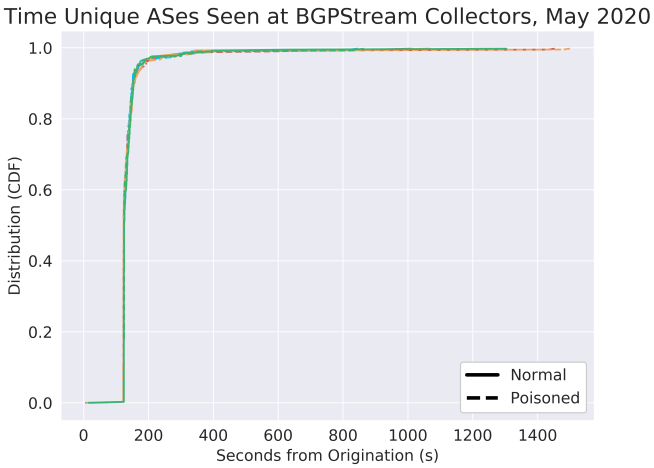
What if BGP poisoning disappeared or stopped working tomorrow?

While BGP poisoning is a behavior *built-in* to the core protocol RFC and needed for BGP itself to work, it is worth considering how Nyx would operate without BGP poisoning. Specifically, Nyx could rely on a recently standardized and increasingly used technique to steer inbound traffic similar to poisoning. BGP communities [27] allow AS operators to tell other ASes how to route traffic bound for their AS. Communities are affixed to BGP advertisements via a special BGP path attribute, and propagate along with the path when it transits the control-plane of the Internet. Specific BGP communities exist which would allow AS operators to encode the same types of behavior enabled via BGP poisoning, including the no-export community [23].

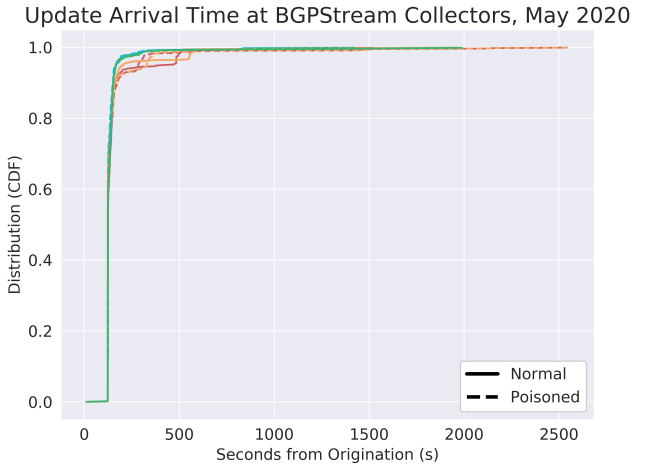
Along with BGP poisoning, new systems that re-architect the Internet could be deployed that would provide similar defensive properties like Nyx. SCION/SIBRA [163, 19], discussed later in Section 3.5, can be deployed on top of existing routers and allow ASes to coordinate to reserve bandwidth for traffic to mitigate DDoS. However, it is worth noting this solution is not easily deployable at a wide-scale and would only allow ASes operating SCION-compatible router software to mitigate DDoS.

3.2.5 Reducing Disturbance

By utilizing BGP Poisoning, we may unintentionally alter the preferred paths to the deployer of ASes other than the critical AS. In the worst case, we alter the path utilized by ASes



(a) Distribution of the times that unique AS advertisements are seen at BGP collectors



(b) Distribution of update arrival time at BGP collectors

Figure 3.4: BGP update time statistics for poisoned vs. normal BGP updates sent in May 2020

containing large numbers of bots, potentially causing DDoS traffic to now flow over the alternate path. We term this effect **disturbance**. To address disturbance, we have implemented two techniques that modify the process of BGP Poisoning:

- Selective Advertisement: We first advertise the poisoned path, observing what the most preferable alternative path from critical AS to the deployer is. We then withdraw the poisoned path and re-advertise it only to the AS directly connected to the deployer AS on the preferred alternative path.

- Path Lining: Using the preferred alternative path, we utilize BGP poisoning to blacklist every AS adjacent to the path and their customer cone, but not the ASes along the path. When the blacklisted ASes see the deployer-originated advertisement, they drop the new path due to loop detection in the same way that BGP poisoning was used to avoid the attacked links due to DDoS. By halting the propagation of our alternate path, disturbance is reduced. Keep in mind, path lining requires **no** outside cooperation or coordination from ASes outside of the deployer, since the deployer simply includes the ASes it wishes to blacklist in its fraudulent advertisements.

In our evaluation, to be discussed in Section 3.3, selective advertisement alone actually increases the disturbance caused by BGP poisoning, a byproduct of how the path propagates through the topology. Path lining *does*, however prevent disturbance, since we are able to add ASes which we do not want our poisoned routes to propagate beyond to our list of ASes to blacklist via BGP loop detection. When employing path lining, we see on average less than 10 ASes disturbed as a result of the deployer's actions, which will be discussed further in Section 3.3.4.

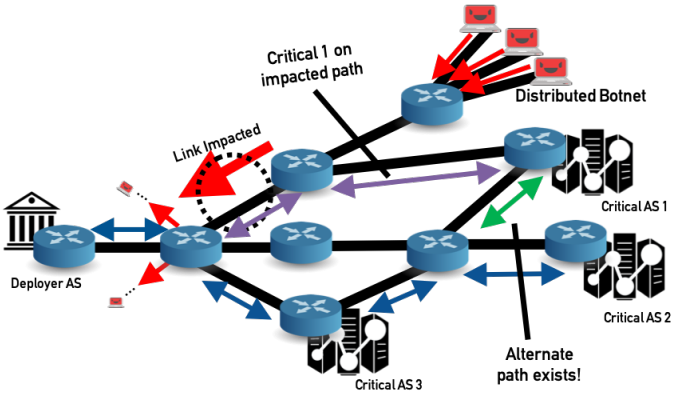
3.2.6 Finding Performant Paths

Even when our system finds paths around ASes we want to avoid, the new paths may not be optimal with respect to available bandwidth along the new path's links. When we move traffic from one path to another path, we force the alternate path to carry its original traffic in addition to traffic from the critical AS and any disturbed ASes. If the new links cannot support the amount of added bandwidth we are placing on them, we will still experience congestion, and can even end up in a worse situation than not using Nyx at all.

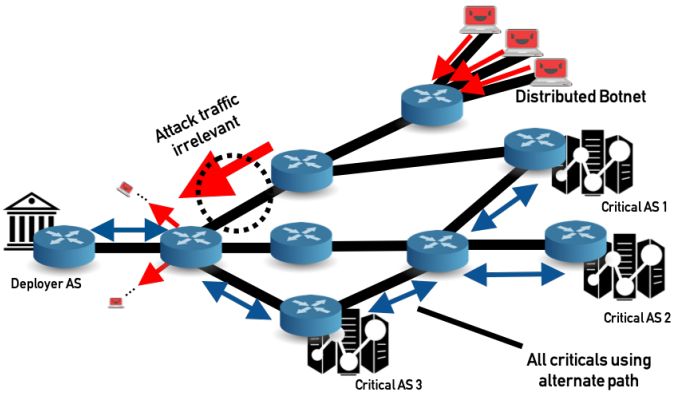
To counter the problem of moving traffic onto new links without enough bandwidth capacity, we have developed a *searching* method to find the most performant paths when alternate paths exist, which when deployed, will repeatedly use BGP poisoning and path lining to migrate critical traffic to an alternative path, and then evaluate if congestion is being experienced. This searching is an *evolutionary algorithm*, where the fitness function is packet flow performance over each alternate path. When searching, if the alternative path is experiencing congestion, Nyx withdraws the alternative route and repeats the poisoning and path lining process, but additionally treats the hops along the former alternative path as if they are experiencing DDoS as well, thus blacklisting them and causing the critical AS to not route traffic to the deployer AS over the insufficient alternate paths. In other words, Nyx repeats the alternative path generating process, avoiding ASes experiencing DDoS and those who have failed to provide a performant alternative path.

3.2.7 Extending to Multiple Critical Autonomous Systems (ASes)

The simplest form of Nyx protects a single critical AS's traffic, but Nyx can be extended to protect *multiple critical ASes*. Described in Figure 3.5a, a deployer may have multiple ASes that it needs to receive traffic from during an impending DDoS or transit-link DDoS. In this case, single-critical Nyx can only re-route *one* of the critical ASes onto the alternate path. Nyx can also route multiple critical ASes onto unimpacted paths, or the same alternate path, shown by Figure 3.5b. We re-route the critical ASes by poisoning the link targeted by DDoS, and then continually shifting each critical AS until an alternative path is found. These alternative paths do not cross the target link and should be less affected by the attack. We use the practical routing constraints discussed earlier on the live Internet.



(a) Multi-Critical Nyx NOT Deployed



(b) Multi-Critical Nyx IS Deployed

Figure 3.5: Multi-Critical Nyx deployment against **transit-link DDoS** or Link Flooding Attacks

3.3 Evaluation

3.3.1 Simulation Methodology

To evaluate the effectiveness of our system, we built our own BGP simulator, which has been used in prior work by Schuchard *et al.* [2] [122]. The simulator is essentially a collection of software routers who speak BGP configured in a realistic topology. The topology used in the simulation is from CAIDA's AS relationships dataset taken from December, 2016 [150]. The BGP policies used by the simulated routes match the current best practices used by operators. Additionally, we have used three bandwidth models, covered in Section 3.3.1, which are used to calculate link capacities, and two botnet models, which are used to calculate attack volumes available to each attacking AS throughout our simulation. We will show later in Section 3.3.5 that our system is resilient to changes in these models.

Using our simulator, we can examine both the effectiveness and cost of a deployer using Nyx, which we refer to as the *deployer AS*, to migrate critical traffic off of links suffering from DDoS or adverse network conditions. Our experiment repeatedly picks two random ASes from the Internet's default-free zone, a region of the Internet where the ASes are not stub ASes, and fixes one of the ASes as the deployer AS and the other as an AS generating critical traffic (i.e. the critical AS mentioned earlier). We then simulate the deployer attempting to respond with Nyx to a DDoS attack that is impacting links on the current best known path between the deployer and critical AS.

When simulating a DDoS attack, we measure the used capacity of links on the best path between the deployer and critical AS in two cases by simulating traffic flow through the

[2]Source code at https://volsec.eecs.utk.edu/nyx

Internet: (1) we measure the used link capacity after the attacks effects have congested a link but before we have used Nyx to migrate traffic off links, (2) we measure after we use Nyx to migrate traffic onto an ideally more performant path. We call the used link capacity for a given link the **subscription factor** of that link, and we calculate these values using a combination of our bandwidth model, which we will discuss in the next section, Section 3.3.1, two constant fixed values varied between simulations that we call the "Bandwidth Tolerance" and "Congestion Factor", and the number of IPs in bot ASes that are attacking the deployer AS per run. The number of IPs per bot AS is determined via the earlier mentioned botnet models we use, which we will discuss further in Section 3.3.1.

The aforementioned bandwidth tolerance and congestion factors inform us of whether our system can hold up under varying attack strengths. The **bandwidth tolerance** for the link between any given AS pair is a constant value between 1 and 2 that describes how much additional capacity the link has based on a normal capacity of 1.0. For example, if the bandwidth tolerance is 1.5, then the AS can handle 50% more traffic than it's normal capacity of 1.0, where any number higher than 1.0 means that link is congested and may drop traffic flowing over it.

The **congestion factor** is a value that is specific to a simulation instance. The congestion factor informs the simulator to send an amount of traffic to the current link we are simulating an attack on that would put the traffic flowing over that link at such a congestion factor. In our simulation, a value of 1.0 for the amount of traffic on a link is the max capacity, and anything over 1.0 means it is congested. We vary our congestion factors between 2.0 and 5.0 over runs, in order to simulate a moderate amount of congestion and a significant amount of congestion.

In order to calculate the pre- and post-traffic migration subscription factors, we need to calculate normal traffic levels flowing over every link in the topology. Using our bandwidth model, we get a predicted level of traffic that should flow over each link, which is then multiplied by the bandwidth tolerance to give us the normal traffic values for that link. Using our botnet model to determine the magnitude of bots per AS, which we will discuss in in Section 3.3.1, we then direct bot traffic at the links between the critical and deployer AS in the case of transit-link DDoS, and at the deployer AS itself for traditional DDoS, by allocating traffic first to the ASes with the most bots. With the combination of the normal traffic over the deployer-critical links and the bot traffic from the DDoS attack impacting them, we calculate our pre-subscription factor as a value above 0.0, where less than 1.0 means the link is uncongested, and above 1.0 means the link is congested. After we utilize Nyx to move traffic off the impacted links and ideally onto more performant paths, we flow traffic again in our simulator and calculate the post-subscription factor, which we use to determine our **performance success** metric.

We use our congestion factor as a proxy for packet loss, and we use path length as a proxy for latency. Modeling latency on the Internet itself is extremely difficult for massively distributed systems; therefore, we adopt the common notion of using path length as a proxy metric for latency, since we can measure path length easily within our simulator since the simulator knows the current Internet topology. Nyx must also know the topology to find alternate paths via our evolutionary algorithm for capacity alleviation, where an individual AS can gather this data from known open source datasets updated frequently via organizations such as CAIDA [150] or gather this on its own via targeted traceroutes. Table 3.4 shows a summary of the information visible to our Internet simulator, and

illustrates that Nyx and the deployer AS know very little in practice, which is shown in Tables 3.1 and 3.2 earlier in Section 3.2.

Bandwidth Model

We recognize that establishing a complete and irrefutable bandwidth model for the modern Internet is an unsolved problem without wielding large-scale collaboration from nearly all existing ASes; therefore, we have developed what we believe to be an accurate and generalized model that effectively allows us to assign bandwidth capacities to links on the Internet. In addition to this model, we have tested our system with two simpler models, one based on the degree of ASes and one on the total IPs associated with ASes, and we show that it works effectively with simpler models later in Section 3.3.5.

To achieve this, we need an Internet scale model of where traffic originates from, where its destination is, and how much of it there is. We base our model on existing work, specifically that of Gill *et al.* [51], supported by the measurements of Labovitz *et al.* [72], the World Bank [154], PeeringDB [111], and Sandvine [118]. We call this model the *Inferred Model*, as we have used known and reputable Internet-wide data to assign approximate traffic constraints to links in the Internet known solely to the *simulator*, and *not* by the Nyx deployer.

In order to establish the relative values of traffic leaving and entering ASes three data sets were combined. Sandvine provides the amount of bandwidth consumption from an "average" user in various regions [118]. This information was combined with the World Bank's estimation of the number of Internet users in each country to get relative inbound and outbound bandwidth on a per nation state basis [154]. In order to assign that

Table 3.4: Information needed by the simulator

Information Used by Simulator	Use of Information	Revealed to Nyx	Information Source
AS Relationships	Simulator needs to model the interaction of all known ASes, and Nyx needs to know ASes bordering the chosen alternate paths during path lining	YES	CAIDA AS Relationships [150], Route Views Project [117]
Inferred (with machine learning) Bandwidth Model	Simulator uses this model as the primary means to calculate congestion factors for links in the topology during simulation. Contains mapping of AS to a "traffic factor" for how much traffic that AS sends	NO	CAIDA AS Relationships [150], PeeringDB [111], IANA [IANA], World Bank [154], Sandvine [118], Labovitz *et al.* [72], Gill *et al.* [51]
AS Degree Bandwidth Model	Used as secondary bandwidth model for validation, contains a mapping between every AS to its degree (number of connected ASes)	NO	CAIDA AS Relationships [150]
AS Total IP Count Bandwidth Model	Used as secondary bandwidth model for validation, contains a mapping of every AS to the number of total IPs known to live inside that AS based on traceroutes from RIPE Atlas	NO	Route Views Project [117]
Mirai Botnet Model	Botnet model used for attack traffic origination based on the Mirai botnet between August 2016 and June 2017, contains ASes with the number of Mirai infections within them	NO	Netlab360 [101]
Conficker Botnet Model (Conficker)	Host-based botnet (Conficker) model used for attack traffic origination based on the Conficker botnet as measured between 2012 and 2013 contains ASes with the number of Conficker infections within them	NO	Thomas *et al.* [147]
Malicious Traffic	Traffic from bot ASes is sent from the originating bot ASes to other ASes such that their traffic flows over the simulator's currently attacked link (upstream of the Deployer AS), or in the traditional DDoS scenario, targets the Deployer AS directly	NO	Botnet Models

bandwidth to ASes, we first assigned each AS to the nation state it primarily resides in using IANA's assigned AS numbers [IANA]. We then consulted PeeringDB, which is a system that allows ASes to advertise their willingness to peer with other ASes [111]. ASes which elect to participate in PeeringDB have the ability to optionally disclose the average amount of inbound and outbound bandwidth from their AS that peers should expect. Of the roughly 55,000 ASes which exist in our topology, where our topology is built based on CAIDA's AS relationship dataset [150], just over 8,000 report bandwidth estimates exist. In order to establish relative bandwidth values between all ASes, a Decision Tree classifier was trained based on AS features including AS degree, the AS customer cone size, the AS's primary country of operation, and the size of IP space advertised by the AS using Scikit-Learn, a popular machine-learning framework [110]. The resulting classifier had a correlation coefficient of 0.89, indicating that the PeeringDB data combined with additional AS information models bandwidth estimates with high accuracy.

Again, we recognize that our inferred bandwidth model is not perfect, but currently no literature has established a model for bandwidth sufficient for approximating traffic levels across the entire Internet.

Botnet Model

For simulating attacks on links in our topology, we have *three* botnet datasets. The first dataset comprises 2.9 million unique Mirai [143] hosts, observed between August 2016 and June 2017, which we call our **Mirai Botnet** model. This model was gathered by a Chinese CDN with a passive scanner setup to detect connections from IPs on known Mirai ports [101]. Given that the Mirai botnet caused massive failures of systems on the Internet in transit-link

attacks, as seen in the DynDNS attack and in Liberia being knocked offline [3, 6], we use this botnet so that we can simulate Nyx standing up against a DDoS attack generated by the same model in which nearly all modern DDoS defenses have recently failed to protect against. For Mirai, the distribution of bots among ASes reveals that the majority of bots are clustered in a relatively small number of ASes, as seen in Figure 2.1 from the background, with less than 50 bots in over 97% of ASes with at least one bot. Note, that in Figure 2.1, the y-axis is trimmed to only show ASes with bot quantities of over 97% of the total botnet size.

The second is a dataset of 23 botnet families collectively known as Conficker, which were observed launching DDoS attacks between late August 2012 and March 2013 with a total of 2.2 million unique hosts [147]. This data was gathered by enumerating all of the botnets Command and Control domains in advance based on exposed code from the botnet variant, and then track the hosts contacting the domains. The distribution of the host-based botnet is nearly identical to Mirai as shown by Figure 2.1 in the Appendix, with again less than 50 bots in over 97% of ASes.

The third botnet dataset is a *fully distributed botnet* where every AS in the topology, except for the current deployer and critical AS, is a bot AS with the ability to send malicious traffic. We use this final botnet model to prove that Nyx is able to mitigate transit-link and traditional DDoS even when facing a fully distributed adversary.

Finally, in the next section, when we explore deployers protecting multiple critical ASes, we add the BlackEnergy botnet [99], a distributed SCADA-focused botnet.

In our simulator, when we iterate over each link of the original path between the deployer and critical ASes, we direct the traffic of the bot ASes to the deployer side of the current

link if and only if the bot ASes best paths to every other AS in the Internet travels over the current link, thus modeling the Crossfire attack by Kang *et al.* [65] popularizing transit-link DDoS by setting up an N^N amount of connections between bots, where N is the number of bots in the current botnet model that can flow over the currently attacked link. This ensures that out of all the bot ASes we have in each bot dataset, we only use the subset of bots that can direct traffic either flowing over the attacked transit-link, in the case of transit-link DDoS, or the deployer AS, in the case of traditional DDoS. In our evaluation, we show graphs from both datasets, illustrating Nyx's resiliency to choice of botnet model.

3.3.2 Attack Scenarios

In our simulation, we aim to mitigate attacks and protect the deployer AS in two distinct DDoS attack scenarios:

- **Transit-Link DDoS**: In this scenario, the bots in our dataset target links upstream of the deployer AS as described above. This is the primary scenario and most closely represents transit-link DDoS, where attacking ASes do not directly address attack traffic to the victim AS, and instead try to block up links outside of their direct sphere of influence. Success in moving traffic to less congested links in this scenario means that our system can effectively mitigate transit-link DDoS. Recall, the deploying AS *cannot filter or prioritize traffic when under transit-link DDoS*, as shown earlier in Figure 2.2b.

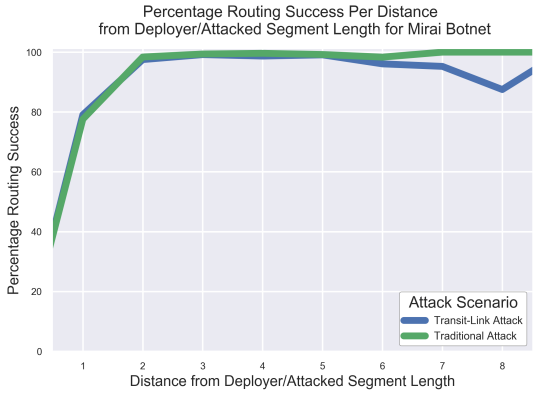
- **Traditional DDoS**: In this scenario the bots in our dataset directly target the deployer AS using the bots that actually have paths to the deployer. This scenario is the more

difficult of the two scenarios, and success here is a major improvement to current DDoS defenses. Here, instead of measuring our routing success in terms of distance to the deployer, we are actually measuring the routing success against **attacked segments** starting with the deployer AS. In this scenario, bots not only address their traffic to the deployer AS directly, but to every segment of hops between the deployer and critical AS; therefore, it is worth noting that fewer bot ASes have paths to long segments of ASes within the default-free zone of the Internet, as opposed to a given transit core AS.

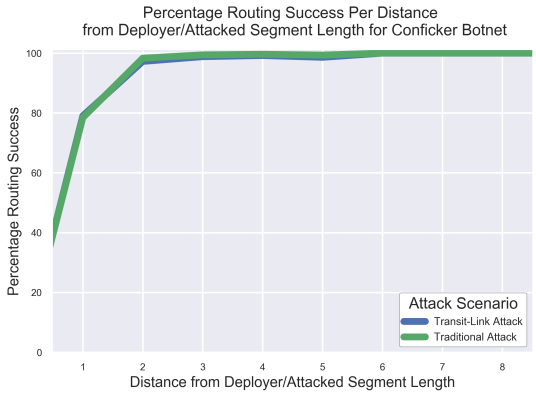
As we will show in the rest of this section, we are largely insensitive to the scenario chosen, which illustrates that we are able to defend against the two major forms of DDoS attacks seen today. Now, we will explore our ability to migrate incoming traffic off their original paths, then show how we can migrate incoming traffic without disturbing significant numbers of neighboring ASes along the deployer-critical AS path, and finally reveal that we can migrate traffic off impacted links onto links that are less congested or totally uncongested relative to the original best path.

3.3.3 Can Nyx Migrate Traffic Onto Links Not Impacted by DDoS Attacks?

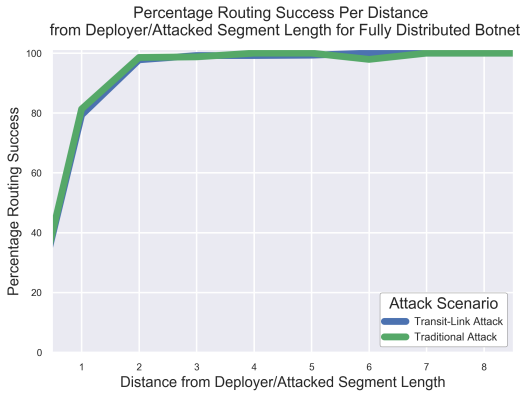
Nyx is able to find valid paths and *move incoming traffic around impacted links* with a great degree of success, which is the first step in mitigating transit-link and traditional DDoS of the volumes where current systems fail. We use our simulator to measure this result for both types of DDoS scenarios, and we label this result as **routing success**.



(a) Percentage routing success for both attack scenarios for the Mirai botnet.



(b) Percentage routing success for both attack scenarios for the Conficker botnet.



(c) Percentage routing success for both attack scenarios for the Fully Distributed botnet.

Figure 3.6: Routing Success for the Mirai, Conficker, and Fully Distributed Botnet models.

As shown in Figure 3.6a, our system achieved nearly 100% routing success over all simulation runs when using FRRP with selective advertisement and path lining to influence the incoming traffic from ASes between 2 and 8 hops out from the deployer. This means that when transit-links upstream of the deployer AS are being targeted, the deployer AS can successfully cause incoming traffic from a chosen critical AS to move around the impacted links.

Not only can we do so with very high success when transit-links are attacked, but when we are under a traditional DDoS attack, our success in routing incoming traffic was above 78% at the extremely low end, and nearly 100% when migrating traffic off links 2 hops or greater away from the deployer itself. This means that as an attacking botnet targets the two links closest to the deployer AS on the path from the deployer to critical AS, the deployer can migrate traffic from that critical AS around the two impacted links with nearly 100% success.

In Figure 3.6b, we show that we can migrate incoming traffic off of nearly any arbitrary link in the Conficker model, and not only when under attack from the Mirai botnet. In this case, our success is also above 98% for hops between 2 and 8 out from the deployer, both when upstream transit-links are under attack and when the deployer is directly under attack.

Finally, we see that Nyx can migrate incoming traffic from the critical AS nearly 100% of the time when under attack from a *globally distributed botnet*, as shown by Figure 3.6c.

Modeling Latency

Recall that we use the widely accepted notion of path length increase as a proxy for increased latency while modeling our system. In practice, measuring the latency of chosen, alternate

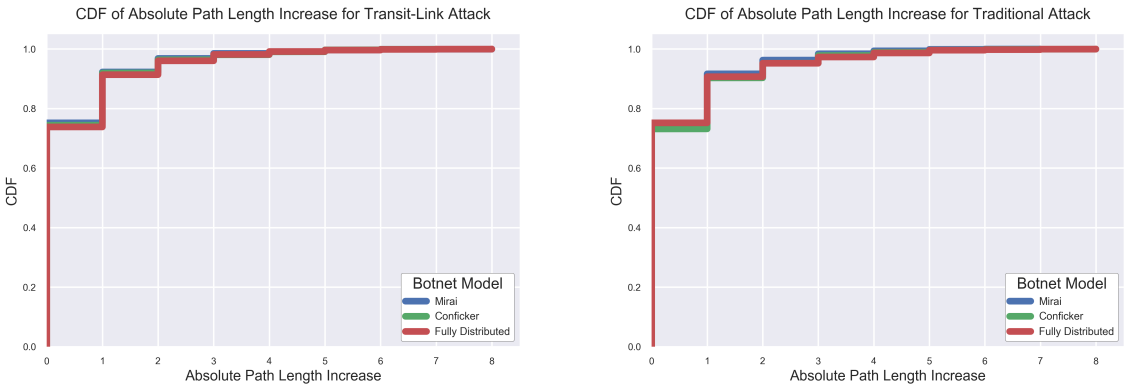
paths on the Internet depends heavily on the layer 1 technologies uses, such as the physical cables between ASes, as well as geographical distance between ASes, and the quality of the hardware running the BGP daemons.

We see path length increases of greater than 5 hops in only 2% of runs, and for 94% of runs, we see *no path length increase*, which is shown in Figure 3.7 regardless of the botnet model used. This is significant as well, because now we additionally do not cause incoming traffic to take longer paths to the deployer AS when traveling around impacted links due to the influence of Nyx. Figure 3.7 also shows the path length increase when using the Conficker and Fully Distributed botnet models, which is roughly equivalent and illustrates that Nyx is resilient to changes in the botnet model with respect to path length increases, even for a *globally distributed botnet*.

Routing success can be achieved independent of whether the new path is actually more congested than the original path, and routing success where the network congestion is alleviated on the new path is discussed later in Section 3.3.5. In the next section, we discuss how we address the second challenge described earlier in Section 3.2, disturbance mitigation.

3.3.4 Can Nyx Migrate Traffic Without Disturbing Other ASes?

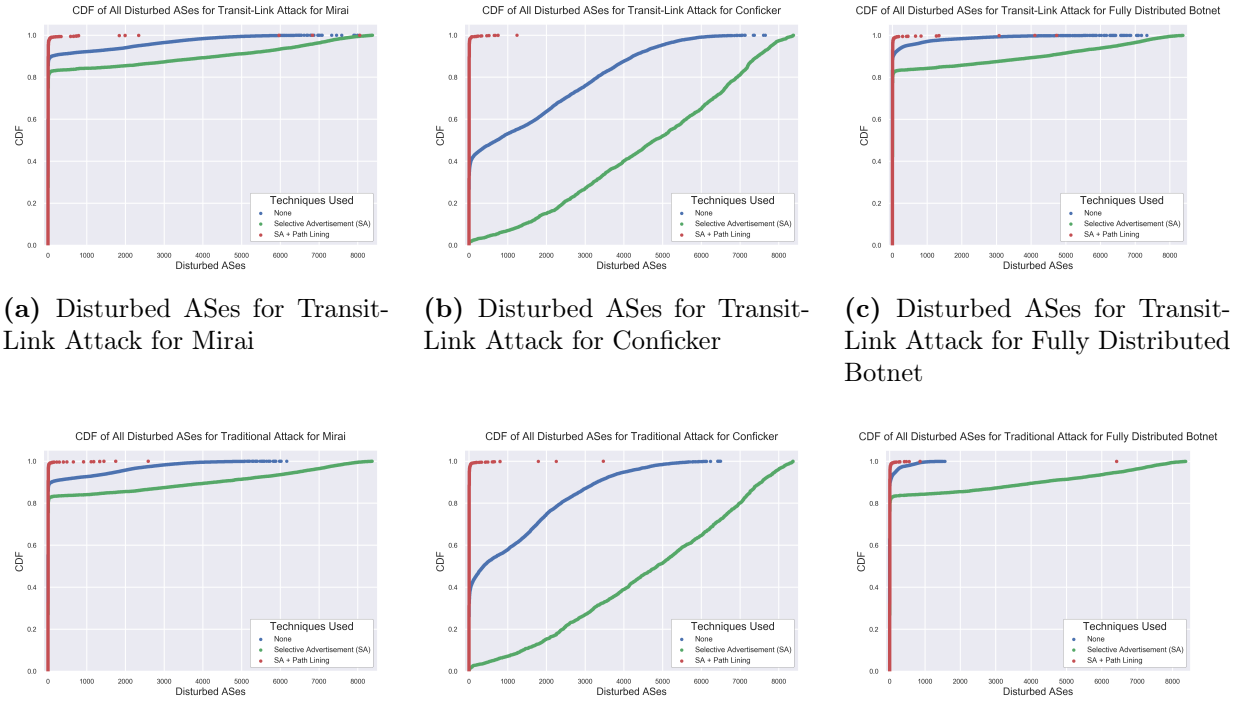
Despite being able to migrate incoming traffic onto new paths outside of the influence of a major DDoS attack, we discovered that the FRRP technique used by Nyx disturbed significant numbers of ASes. To overcome the problem of disturbance, we introduced two strategies in Section 3.2.5: selective advertisement and path lining. When utilizing those



(a) Path Length Increase for Transit-Link Attack

(b) Path Length Increase for Traditional Attack

Figure 3.7: Path length increase for both attack scenarios for Mirai, Conficker, and Fully Distributed botnets.



(a) Disturbed ASes for Transit-Link Attack for Mirai

(b) Disturbed ASes for Transit-Link Attack for Conficker

(c) Disturbed ASes for Transit-Link Attack for Fully Distributed Botnet

(d) Disturbed ASes for Traditional Attack for Mirai

(e) Disturbed ASes for Traditional Attack for Conficker

(f) Disturbed ASes for Traditional Attack for Fully Distributed Botnet

Figure 3.8: Disturbed ASes with and without disturbance mitigation for all botnet models for Traditional and Transit-Link DDoS

strategies in unison, we significantly lessened the disturbance to the ASes close to the deployer AS when Nyx was employed to migrate incoming traffic.

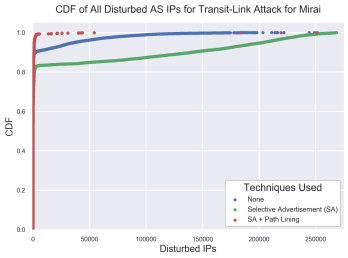
As shown in Figure 3.8, before employing any strategies to mitigate disturbance, we disturbed between 1,000 and 6,000 ASes nearly 90% of the time, which in the modern Internet is roughly 10% of all existing ASes [3]. This is true when under either attack scenario: transit-link DDoS or traditional DDoS. When we implemented selective advertisement alone, we did not see the disturbance drop, which indicated we needed to try another strategy. Then, we implemented path lining as described in Section 3.3.4, and brought the number of disturbed ASes to less than 10 disturbed ASes on average. Using path lining and selective advertisement, we effectively mitigated the disturbance of ASes in the default-free zone, thus reducing the deployment costs of Nyx when both upstream transit-links are attacked and when the deployer AS is targeted directly. Furthermore, for each of those ASes, Nyx also disturbed the IPs residing within them, as they may see new routes taken for traffic being sent. This is shown in Figure 3.9, with a similar resulted for disturbed ASes.

In summary, by employing our disturbance mitigation techniques, we are able bring the *costs of disturbance to virtually zero* in nearly 90% of cases. In the next section, we will discuss local preference disturbance.

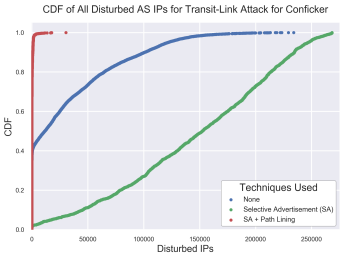
Are There Any Local Preference Changes?

The costs of link usage from one provider to another provider in the actual Internet are closely guarded secrets; therefore, we use the act of switching onto a peer- or provider-learned path as a proxy for added monetary cost. In our simulations, the deployer AS using Nyx *never*

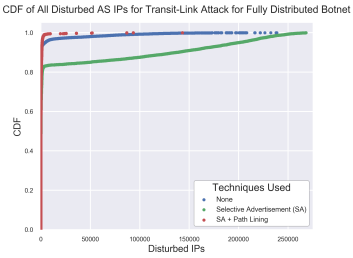
[3] As of October 2017, the number of ASes according to CAIDA's Internet topology was roughly 58,000.



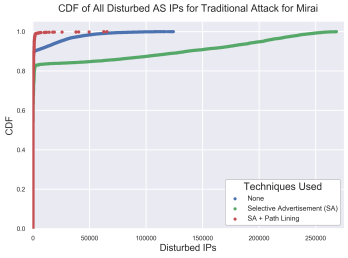
(a) Disturbed IPs for Transit-Link Attack for Mirai



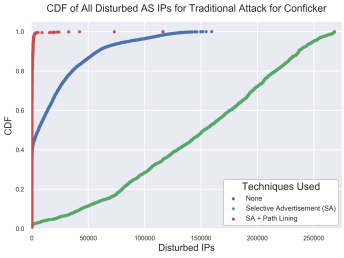
(b) Disturbed IPs for Transit-Link Attack for Conficker



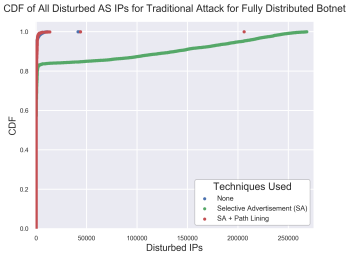
(c) Disturbed IPs for Transit-Link Attack for Fully Distributed Botnet



(d) Disturbed IPs for Traditional Attack for Mirai



(e) Disturbed IPs for Traditional Attack for Conficker



(f) Disturbed IPs for Traditional Attack for Fully Distributed Botnet

Figure 3.9: Disturbed IPs with and without disturbance mitigation for all botnet models for Traditional and Transit-Link DDoS

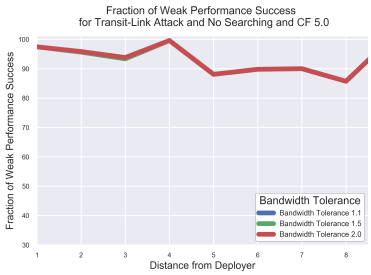
switches from a customer learned path to peer- or provider-learned path, or a peer-learned path to a provider-learned path.

3.3.5 Do the Alternate Paths Have Enough Capacity?

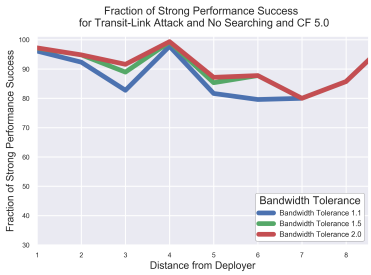
Now that we have shown that Nyx can successfully migrate incoming traffic and do so with little to no disturbance, we now show that Nyx can migrate traffic onto more performant and uncongested paths in nearly all cases for transit-link DDoS and a majority of cases in traditional DDoS. In order to measure performant paths, we use several bandwidth tolerances and congestion factors, as discussed in Section 3.3.1. We additionally show that our system has the ability to search for performant paths as described in Section 3.2.6, which greatly enhances the success of migrating onto uncongested paths after a DDoS attack.

Our system achieves performance success in two distinct ways: 1) when the post-subscription factor is less than the original subscription factor, and 2) when the post-subscription factor is less than 1.0 (indicating that we have completely alleviated congestion from either an original subscription factor of 2.0 or 5.0 to less than 1.0). The bandwidth tolerances are between the range of 1.0 and 2.0, which model links where the capacity might actually have more room than our bandwidth model dictates, and this tolerance is applied across all of our simulations.

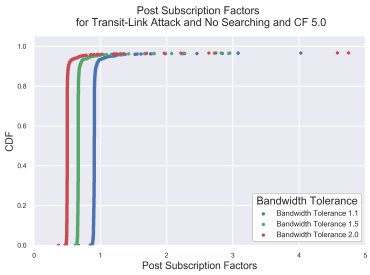
As shown in Figure 3.10, when the deployer AS is under transit-link DDoS, we are able to find paths more performant than the pre-attack path on average in over 90% of cases without searching for the hardest setting of bandwidth tolerance and congestion factor, but with searching as shown in Figure 3.10d, our performance success is essentially 100% for all



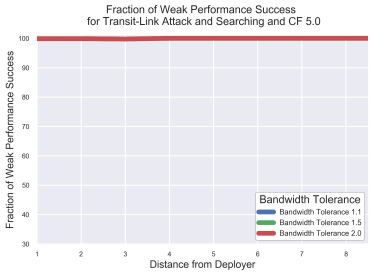
(a) Weak Performance Success with No Searching for the Mirai Botnet and Normal Bandwidth Model



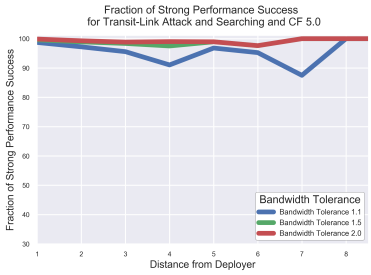
(b) Strong Performance Success with No Searching for the Mirai Botnet and Normal Bandwidth Model



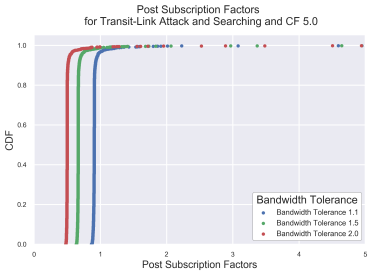
(c) CDF of Post-Subscription Factor with No Searching for the Mirai Botnet and Normal Bandwidth Model



(d) Weak Performance Success with Searching for the Mirai Botnet and Normal Bandwidth Model



(e) Strong Performance Success with Searching for the Mirai Botnet and Normal Bandwidth Model



(f) CDF of Post-Subscription Factor with Searching for the Mirai Botnet and Normal Bandwidth Model

Figure 3.10: Performance success metrics for the transit-link attack scenario with and without searching.

distances from the deployer AS. This means that no matter how far out a transit-link is being attacked, we can alleviate *some amount of congestion* in nearly 100% of cases. But what about alleviating *all* of the congestion?. We show this in Figure 3.10b without searching, where we are still able to find performant paths that are completely uncongested as compared to an original congestion of *5 times more than the capacity* in over 89% of cases on average. When we employ searching in Figure 3.10e, we bring that average up to over 95% on average for the hardest setting of bandwidth tolerance at 1.1.

These results indicate that when transit-links are under a DDoS attack upstream of the deployer AS, we can guarantee *no* traffic loss for a particular critical AS in over 95% of cases. To state it in another way, we give the deployer the ability to operate under normal conditions for traffic being delivered from some critical network despite the transit-links between the deployer and critical network sustaining attack traffic loads of arbitrary amounts.

For transit-link DDoS with and without searching, we also show the post-attack subscription factors in Figure 3.10c and 3.10f, which indicates that for over 95% of cases we can migrate traffic onto uncongested paths out of the way of any DDoS attacks on upstream transit-links.

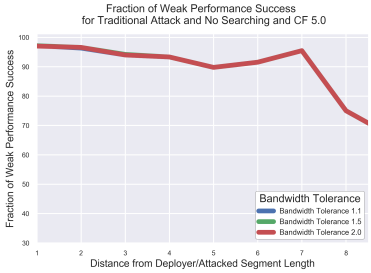
Not only can we protect the deployer AS when it is under transit-link DDoS, but we show we can protect the deployer AS when it is targeted directly in a traditional DDoS scenario for the *hardest settings* of bandwidth tolerance and congestion factor. As we show in Figure 3.11, we are able to migrate traffic onto links that are more performant than the original paths in on average 93% of cases, and for strong performance success we can migrate traffic onto paths that are on average completely uncongested in 75% of cases. When employing searching, though we see a higher weak performance success in Figure 3.11d, with

an average of nearly 98% success in alleviating some amount of congestion, we do not see searching helping nearly as much for strong performance success, as shown in Figure 3.11e.

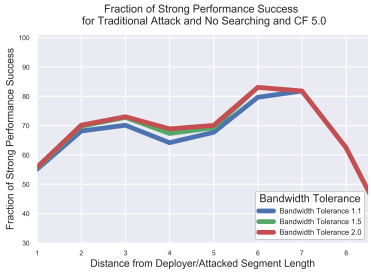
Despite this, we demonstrate that using Nyx, even when under a traditional DDoS attack, with little cost of deployment to an AS and no outside cooperation, we can migrate incoming traffic onto links that are not impacted by the DDoS at the ASes *doorstep*, targeted directly at the deployer, on average 75% of the time, as shown in the CDF of the post-subscription factors in Figure 3.11f. Why is traditional DDoS the harder case to protect against? The answer lies in how Nyx utilizes FRRP. When we advertise out our hole-punched paths from the deployer AS while under traditional DDoS attacks, we can end up dragging along large amounts of bot traffic that is being addressed directly to the deployer AS, whereas the bot traffic in transit-link DDoS is never addressed to the deployer AS, and the bot traffic will not be dragged towards the deployer AS. Regardless of this side effect, we have still demonstrated that our system can protect a significant amount of traffic from a chosen critical AS known ahead of time, which often cannot be done in any capacity with traditional DDoS defense methods that we will discuss in Section 3.5.

We show in Figure 3.12, that when our system utilizes searching, the depth to which we search is small except in greater distances from the deployer. This means that the deployer does not have to force the BGP speakers implementing Nyx to waste precious time finding more performant paths around impacted links, and that it can be done in the case of transit-link DDoS in nearly 0 iterations on average and 14 iterations on average in the worst case for distances in excess of 8 hops from the deployer.

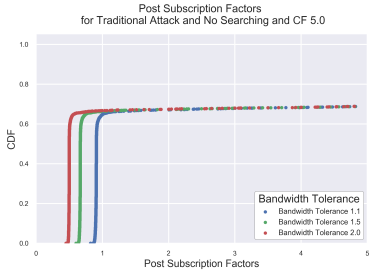
We have discussed our results for performance success in the case of bandwidth tolerances and congestion factors, but how do we show that these values are not chosen simply to



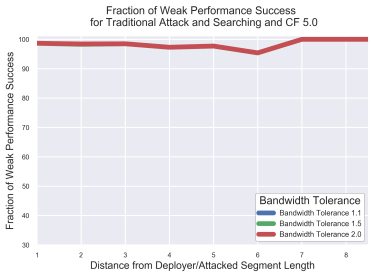
(a) Weak Performance Success with No Searching for the Mirai Botnet and Normal Bandwidth Model



(b) Strong Performance Success with No Searching for the Mirai Botnet and Normal Bandwidth Model

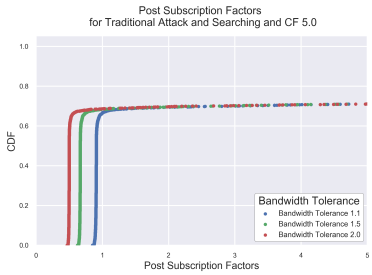


(c) CDF of Post-Subscription Factor with No Searching for the Mirai Botnet and Normal Bandwidth Model



(d) Weak Performance Success with Searching for the Mirai Botnet and Normal Bandwidth Model

(e) Strong Performance Success with Searching for the Mirai Botnet and Normal Bandwidth Model



(f) CDF of Post-Subscription Factor with Searching for the Mirai Botnet and Normal Bandwidth Model

Figure 3.11: Performance success metrics for the traditional attack scenario with and without searching.

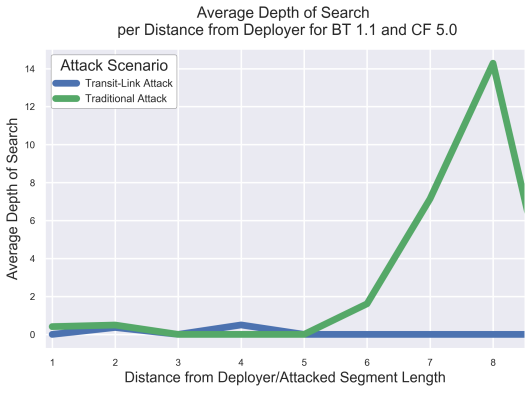


Figure 3.12: Average depth of search for the hardest setting of bandwidth tolerance and congestion factor.

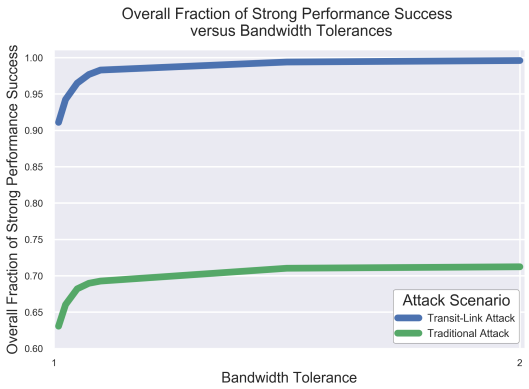


Figure 3.13: Strong performance success over varying bandwidth tolerances. Notice that once the bandwidth tolerance is greater than 1.1, the overall strong performance success stabilizes.

guarantee the success of our system? We show in Figure 3.13, that for the Mirai botnet model, once our bandwidth tolerance is at 1.1 or higher, the gains received by increasing the tolerance stabilize and do not increase further. This indicates that regardless of how much room you give the link capacities around a DDoS attack, the strong performance success does not increase; therefore, our chosen values in the simulation are not in place to guarantee we have greater success.

For congestion factors, we see only slightly higher performance success for smaller congestion factors, such as our other tested factor of 2.0, but not by significant amounts. As shown in Figure 3.14, the smaller congestion factor of 2.0 has little effect on the strong performance success, where we must migrate traffic onto links that are uncongested. This is the case when either transit-links are attacked or when the deployer is attacked directly. Given these results, our simulation's choice of congestion factor indicates that you can continue to congest links on the normal path between the deployer and critical AS and

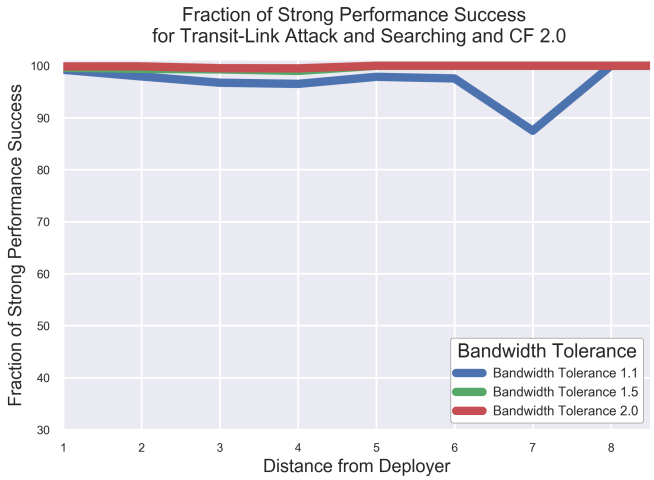
still be able to successfully migrate traffic around the impacted links, while still alleviating congestion.

Is the Performance Success of Nyx Insensitive to the Botnet Model?

In Section 3.3.1, we described three botnet models: Mirai, Conficker, and a fully distributed botnet. In the previous section, we showed that Nyx significantly mitigates the effects of Traditional DDoS and nearly defeats any congestion due to Transit-Link DDoS when the adversary controls a botnet with the size and topology of Mirai. However, Nyx performs as well with other models, including Conficker, which has a distribution and cardinality similar to Mirai (see Figure 2.1), and a fully distributed botnet. For Conficker, the results are similar in success to Mirai and are shown in Figure 3.15. For the fully distributed botnet, Nyx achieves strong performance success in 99% of cases on average for Transit-Link DDoS for the hardest settings of bandwidth tolerance and congestion factor, and 78% strong performance success on average for Traditional DDoS as shown in Figure 3.16. This means that a globally distributed adversary, such that essentially *every AS in the modern Internet* possesses bots that can send attack traffic upon command, can be subverted by routing around the DDoS events with Nyx, deployed at a single AS and without outside cooperation from other ASes.

Is Nyx Insensitive to the Choice of Bandwidth Model?

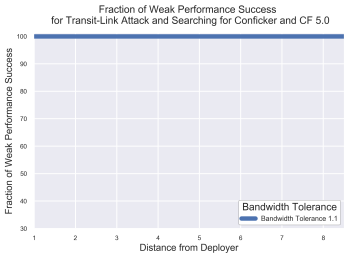
In Section 3.3.1, we described the main bandwidth model used in our evaluation. This model is fairly complex, but approximates the typical traffic levels on existing ASes through the application of several well-known datasets of AS-level and geographic data, which is needed



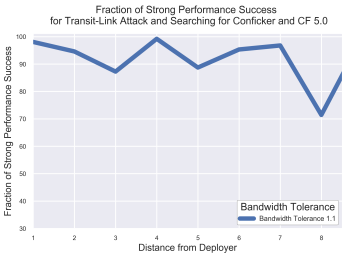
(a) Strong Performance Success for Transit-Link Attack for CF of 2.0

(b) Strong Performance Success for Traditional Attack for CF of 2.0

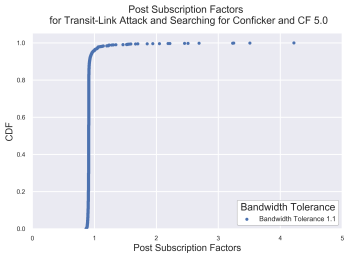
Figure 3.14: Strong performance success with searching for both attack scenarios for congestion factor of 2.0.



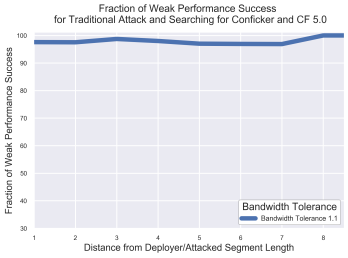
(a) Weak Performance Success for Transit-Link Attack with Searching for the Conficker Botnet



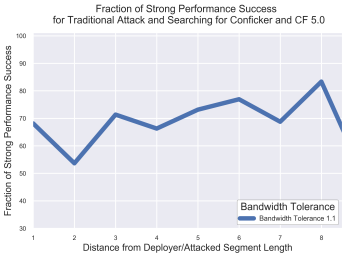
(b) Strong Performance Success for Transit-Link Attack with Searching for the Conficker Botnet



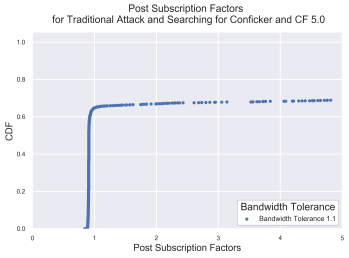
(c) CDF of Post-Subscription Factor for Transit-Link Attack with Searching for the Conficker Botnet



(d) Weak Performance Success for Traditional Attack with Searching for the Conficker Botnet

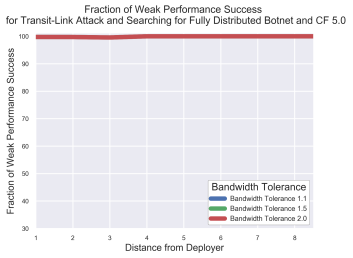


(e) Strong Performance Success for Traditional Attack with Searching for the Conficker Botnet

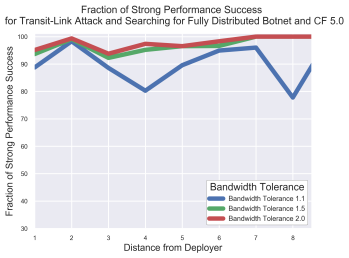


(f) CDF of Post-Subscription Factor for Traditional Attack with Searching for the Conficker Botnet

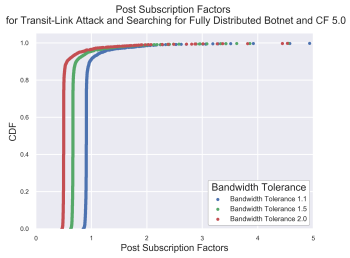
Figure 3.15: Performance success metrics for both Traditional and Transit-Link attack scenario, normal bandwidth model, with searching for the **Conficker** botnet.



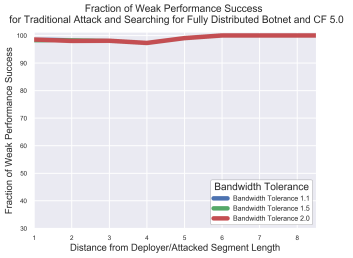
(a) Weak Performance Success for Transit-Link Attack with Searching for the Fully Distributed Botnet



(b) Strong Performance Success for Transit-Link Attack with Searching for the Fully Distributed Botnet

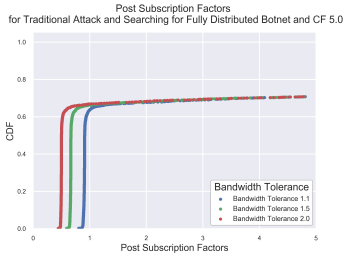


(c) CDF of Post-Subscription Factor for Transit-Link Attack with Searching for the Fully Distributed Botnet



(d) Weak Performance Success for Traditional Attack with Searching for the Fully Distributed Botnet

(e) Strong Performance Success for Traditional Attack with Searching for the Fully Distributed Botnet



(f) CDF of Post-Subscription Factor for Traditional Attack with Searching for the Fully Distributed Botnet

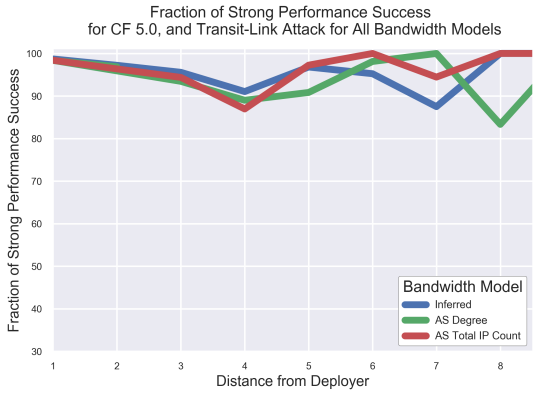
Figure 3.16: Performance success metrics for both Traditional and Transit-Link attack scenario, normal bandwidth model, with searching for the **Fully Distributed** botnet.

because the real link capacities between major ASes are a in many cases a closely guarded secret. To evaluate our system's ability to still work with simpler models, we have built two other bandwidth models that influence our system's performance success (i.e. the ability to reduce congestion on the new routes chosen after successfully migrating traffic):

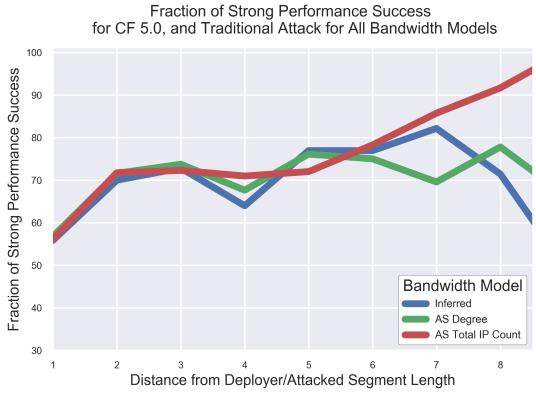
1. AS Degree Model: This model chooses the degree of each AS as the traffic factor value, or the approximation of the traffic sent by a given AS with arbitrary traffic units, described in Section 3.3.1. The AS Degree for any given AS is defined as the number of ASes where the given AS has a direct connection, as inferred from CAIDA's AS Relationship dataset [150].

2. AS IP Count Model: This model chooses the number of IPs associated with a given AS as it's traffic factor value. The number of IPs associated with a given AS is determined by the RIPE NCC RouteViews dataset [117].

In Figure 3.17, we show that our system still achieves nearly identical strong performance success for all tested bandwidth models, with our most complex and standard inferred model performing the worst overall. For the transit-link DDoS scenario, our models all averaged around 95% strong performance success, and for the traditional DDoS scenario, our models averaged around 70% to 75% success. Therefore, by modeling the link capacities on the Internet as a function of the AS Degree and AS Total IP count, we have achieved similar results as when we model the link capacities with our more complex, inferred bandwidth model that attempts to model the true link capacities in the Internet.



(a) Strong Performance Success for Transit-Link Attack Scenario for All Bandwidth Models



(b) Strong Performance Success for Traditional Attack Scenario for All Bandwidth Models

Figure 3.17: Strong performance success for both attack scenarios over all bandwidth models.

3.4 Multi-Critical Nyx

Next, we show how Nyx can isolate the inbound traffic of multiple critical ASes from DDoS, rather than only isolating a single critical AS's traffic. We now evaluate the routing success, weak performance success, and strong performance success of this multi-critical scenario. Our results and analysis focuses on the scenario described in evaluation methodology from Section 3.2.7, where a botnet uses transit-link DDoS to target a major provider of a deploying AS defending between 3 and 500 critical ASes.

3.4.1 Can Nyx Migrate Traffic from Multiple Criticals onto Non-Impacted Links?

First, we examine whether a deploying AS can successfully re-route between 3 and 500 critical ASes around congestion and onto an alternative path. This metric, routing success, mirrors the single-critical scenario from earlier, except at the scale of several critical to deployer paths. We consider an attack scenario where the Mirai botnet targets a provider two hops upstream of the deployer, and employ a poison limit of 245. This poison limit is found in real-world measurements in the next set of work in Chapter 4. This poison limit forces Nyx to use path lining with a limit, which directly affects the performance success we discuss in the next section. We find that Nyx is able to find valid paths around congested links over 95% of the time for 1 to 50, 50-100, and 100-150 critical ASes shown by Figure 3.18. We see that from 150 to 200 ASes and 200-300 ASes, the deployer can re-route all 150 to 300 ASes around congested paths and onto any alternative path no less than 85% of the time. At 500 critical ASes, Nyx can achieve routing success in over 80% of cases across

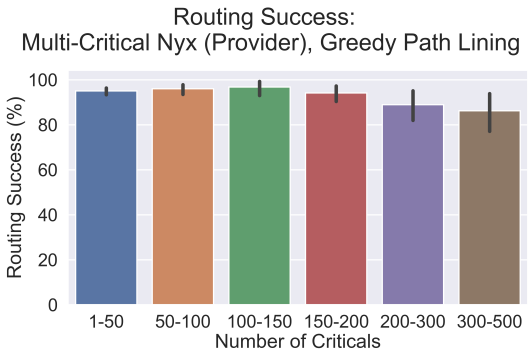


Figure 3.18: Multi-Critical Routing Success

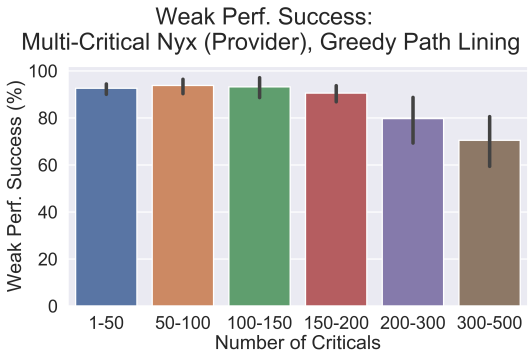


Figure 3.19: Multi-Critical Weak Performance Success

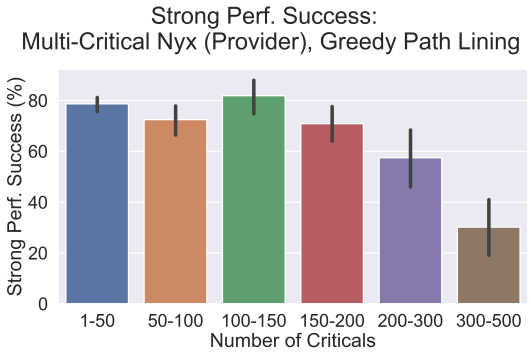


Figure 3.20: Multi-Critical Strong Performance Success

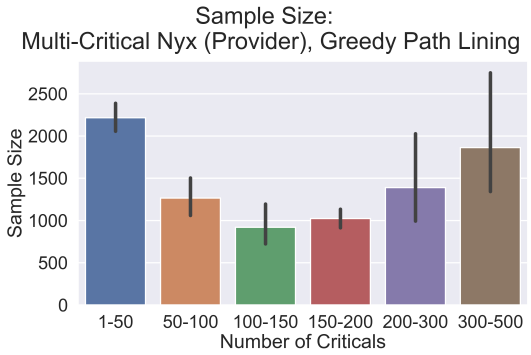


Figure 3.21: Sample Size of Multi-Critical Experiments from Figures 3.18-3.20

1,000 simulation runs shown by the sample sizes at each bin in Figure 3.21. These results demonstrate the practicality of Nyx for allowing a deployer of any size to achieve quick wins against a transit-link DDoS targeting its major traffic providers.

Despite the limit on path lining in place, Nyx can still route several critical ASes onto alternative paths. Nyx only poisons the single targeted provider that the critical ASes use on their return path to the deployer, thus the constraint of 245 poisoned ASes is not reached with the poisons tied to re-routing. To limit disturbance, Nyx can poison the top n sets of neighbors along each critical to deployer path using the greedy path lining algorithm. While the limited path lining algorithm in the multi-critical case will not limit disturbance to the levels of unrestricted path lining, or even the disturbance for a single critical AS, the diversity of re-routed paths chosen by the critical ASes leads to the dispersion of the disturbed traffic. Thus, while a single critical AS has only alternative path at any given time, and disturbance will affect that path and the original path, multiple critical ASes spread disturbance to the deployer across many return paths.

3.4.2 Do Alternate Paths from Multiple Criticals Have Enough Capacity?

Second, we show that Nyx can not only route multiple critical ASes around congested paths, but also onto *uncongested paths*. Again, like routing success, the performance success varies by the attack scenario. Figure 3.19 reveals Nyx can achieve weak performance success of over 80% on average for up to 300 critical ASes. In other words, when Nyx is deployed to protect 300 critical ASes, 80% of the time it can both find alternative paths around the

target of the transit-link DDoS *and* find paths that have less congestion than the original 5x over-subscribed path. At 1 to 50 critical ASes, the success for moving onto less congested paths rises above 90%. For certain deployment scenarios, such as military or end-user networks, the number of important critical ASes may be lower due to tactical procedures or CDNs, respectively. At the extreme end, ASes such as Amazon Web Services, Cloudflare, or other CDNs may have large amounts of critical ASes. However, these ASes also possess the bandwidth to approach the problem of DDoS with the traditional filtering-based approach or by using their large distributed networks to re-route inside their sphere of influence.

While re-routing onto less congested paths is useful, Nyx can also re-route up 200 critical ASes onto totally uncongested paths over 70% of the time. Shown in Figure 3.21, this result mirrors the single-critical scenario success. With only small changes to the re-routing algorithm, Nyx thus achieves with 2 to 200 ASes what it can also achieve with only 1 critical AS. At the high end of criticals, Nyx performs worse. At 30% success at moving 500 critical ASes onto totally uncongested paths, the trend towards failure increases as the deployer needs to re-route further amounts of critical ASes. This finding could be due to the fact that at 500 critical ASes, the amount of available substantially different return paths needed to avoid multiple re-routed criticals from congesting their new alternative path drops significantly. Again, here we highlight that the deployer AS is going from no amount of filtering being able to defeat transit-link DDoS to effectively and practically re-routing up to 200 critical ASes around the congestion over 70% of the time for over 5,000 evaluation runs.

3.5 Related Work and Other DDoS Defense Systems

Traditional and current defense systems attempt to mitigate the negative effects of DDoS attacks through a variety of means; however, no systems we found defend against DDoS via route-altering techniques such as ours. In this section, we will discuss several classes of DDoS defense systems in recent literature, then we will discuss why these systems fail to protect against recent transit-link DDoS attacks and massive traditional DDoS attacks leveraging the Mirai botnet [3, 4, 6], then we will discuss why our system does not suffer from the same flaws as these existing systems.

Traditional DDoS defense systems that attempt to alleviate DDoS attacks via packet filtering [93] using techniques such as packet marking [129, 21, 158, 95, 81, 157] and push-back techniques [161, 78, 83, 64, 17] filter traffic at ingress and egress points on the network, but are incapable of withstanding DDoS attacks of the size of the Mirai botnet used to test Nyx. Additionally, transit-link DDoS typically does not send attack traffic directly to the reactor AS, and instead to upstream links; therefore, filtering on attack traffic would not be feasible since the victim would not see the traffic. Furthermore, our system can handle massive inbound flows sent from distributed botnets because not only do we not physically have to handle malicious traffic in the case of transit-link attacks, but we can arbitrarily manipulate the paths that attack traffic takes, and by doing so spread the incoming attack traffic across links upstream of the deployer AS such that no traffic from the critical AS is dropped along the way.

Other techniques that filter traffic targeted at specific services [36, 28, 152] are ineffective against DDoS attacks that attack different services or even the underlying routing

infrastructure. Because all internetwork traffic must be sent over paths determined by BGP speakers, as simulated in this work, our system is able to reactively alter advertised paths such that no matter the type of traffic being sent by the adversary, the victim AS will move traffic from a chosen critical AS onto paths not impacted by the malicious traffic.

Strategies using game-theoretic approaches model the defender's best case strategy to maximize cost for an attacker [135, 20], but these approaches are ineffective when massive DDoS attacks can be launched with the click of a button at little cost to the attacker. Zhou *et al.*'s work to protect the Internet's backbone and highly connected ASes [165] fails to defend against transit-link DDoS, since the proposed system only handles traffic once it reaches the deployed system within the victim AS. Other recent works take this same deployment approach, where an attempt to detect and model botnet traffic is done at the victim AS using statistical methods [164, 125], which cannot be done in the case of transit-link DDoS.

Chapter 4

Measuring and Analyzing the Ability to Re-Route in Practice

4.1 Introduction

In our second set of work, we set out to measure how well systems like Nyx function on the live Internet. In doing this, we measured many other properties of BGP poisoning on the live Internet in a comprehensive Internet measurement study.

Responsible for ensuring packets find their home once they begin their journey, the Internet routing infrastructure serves a key role in ensuring the reachability, availability, and reliability of online services. Given the importance of its fundamental role, the security of the routing infrastructure as a set of protocols and routing process has underpinned much of the past two decades of distributed systems security research. However, the converse is becoming increasingly true. Routing and path decisions are now important for the security properties of systems built *on top* of the Internet.

In particular, research has begun to harness the de facto routing protocol on the Internet, the Border Gateway Protocol (BGP)[114] and the functionality it provides to implement new offensive, defensive, and analytical systems. Growing in use, a method known as BGP poisoning has been leveraged by censorship circumvention, DDoS defense, and topology discovery. Rooted in the BGP RFC, BGP poisoning can be used by routing-capable entities. These entities are known on the Internet as Autonomous Systems, or ASes [56]. Fundamentally, BGP poisoning is now being used to maneuver an ASes' return traffic around specific AS-to-AS links, new regions of the Internet topology previously not visible to certain ASes, and other regions of interest. Critically, BGP poisoning and the re-routing it provides is being employed for *security purposes*.

For example, Nyx, a DDoS defense system leverages the ability to manipulate inbound traffic paths with BGP for a security purpose: to route around attacked links on the Internet [130]. Nyx relies on altering paths on the Internet to circumvent links affected by DDoS, and is evaluated on the entire Internet via simulation. Prior to Nyx, Katz-Basset *et al.* demonstrated the use of BGP poisoning in practice for single link-failures, as opposed to DDoS-inflicted failures, with LIFEGUARD [70]. In the domain of censorship circumvention, decoy routing (DR) has become a standard means to avoid censoring eyes [156, 155, 66, 59, 24]; though, Schuchard *et al.* presented the *Routing Around Decoys* (RAD) attack and follow-on E-Embargos [122, 123], which utilized the routing infrastructure to circumvent decoy routers by re-routing both outbound *and* return paths to completely avoid decoy routers. In general, these security systems rely on the real-world feasibility of BGP poisoning to carry out defensive security goals.

Yet other systems rely on the **opposite** assumption to provide security guarantees or to attack poisoning-enabled defenses, that BGP poisoning is *infeasible* in the real-world. In response to Schuchard *et al.*'s RAD attack on decoy routing systems, Houmansadr *et al.* presented the *Waterfall of Liberty* system, which demonstrated that RAD could be prevented with downstream decoys [60], relying on altering return paths via BGP poisoning to be infeasible, and Goldberg *et al.* added additional security guarantees to Waterfall [25]. In the world of DDoS, Tran & Kang *et al.* presented an adaptive Crossfire/Link-Flooding Attack (LFA) [149] that challenged Nyx, which we term *Feasible Nyx*, yet their approach is only measured in simulation, supported by passive observations gathered by a single major network. Additionally, Tran's claims about how ASes filter poisoned paths are supported by

passive, not active, measurements. These particular security systems rely on the *infeasibility* of BGP poisoning to carry out both offensive and defensive security goals.

We recognize that the lack of evidence exposing BGP poisoning's real-world feasibility has given rise to diverging claims in the research community. Worse, even the network operator community is being affected, as multiple NANOG mailing list threads have led to long, heated debates regarding the feasibility, positives, and negatives of BGP poisoning as a re-routing mechanism [106, 105, 104]. While some published research claims re-routing based on poisoning is feasible, like Nyx, RAD, and LIFEGUARD, other research, such as Waterfall and Adaptive LFA attacks, **assumes the opposite** claim is true when presenting their experiments and analysis. While this problem is unaddressed, network operators can not reasonably depend on BGP poisoning for defensive purposes, or refute the feasibility of and leave out BGP poisoning when designing threat models, pushing critical networks to lose out on the benefits of any of the systems described above. While deployment of poisoning is still possible without a ground truth of Internet behavior, no prior study outlines with real-world active measurements 1) how feasible re-routing with BGP poisoning is in practice, 2) how networks and ASes treat poisoned AS paths propagated by a Nyx defender, RAD adversary, or network operator, and 3) clarifies the security implications of 1) and 2). This paper serves as that study.

Understanding and Analyzing Re-Routing Feasibility

In this second thrust, we present the first self-contained collection of novel, real-world, Internet-scale measurements that validate or refute assumptions made in simulation or passively in recent security literature, such as RAD, Waterfall, Nyx, and Adaptive CrossFire/Link-Flooding Attacks. We provide insight into utilizing BGP poisoning as a

topology and congestion discovery tool. We re-evaluate the Nyx DDoS defense platform, examine the graph-theoretic aspects of return paths available to ASes, and build predictive models for ASes wishing to understand their vulnerability to poisoning. We examine not only the filtering of BGP advertisements using poisoning, but also what ASes and what policies are deployed by such ASes that filter. To understand whether routing operator groups walk the walk when it comes to poison filtering, we measure their behavior against ASes not in a popular security-first ISP consortium.

These findings were uncovered via a 6-month long series of active measurements, beginning in January 2018 until July 2018. These measurements employed an array of control-plane and data-plane Internet infrastructure. This infrastructure included a collection of ASes from our own organization and the broader Internet via PEERING [121], a real-world BGP testbed, nearly all responsive and one-per-AS 5k traceroute sources from RIPE Atlas [RIPE NCC], and live streams of BGP announcements from RouteViews [117] and the RIPE Routing Information Service [116]. In practice, we found that the Internet's treatment of BGP poisoning lies on a *spectrum* of behavior when evaluated across 1,400+ experiments, conducted with permission and guidance from 5 geographically and topologically diverse ASes on the Internet.

Specifically, we find that for 1,460 instances of BGP poisoning, over 77% of the distinct instances could be successfully maneuvered onto new, previously unreachable AS-links at some point in the original path. An average of 8 new links were discovered per path, for a total of 3 completely new paths on average. In 20% of cases, more than 5 completely new paths were discovered, with a maximum of 19 unique paths in one particular case and 23 total new links in another case. Beyond enumerating new paths, we found that BGP poisoning

can be used to route around 80% of ASes with less than 2,500 customers, considered small to medium-sized transit ASes. By further refining these measurements, we uncovered additional cases of poison filtering for highly connected ASes such as L3 and Cogent. When poisoning downstream ASes in systems such as Nyx, RAD, and others, connectivity can be lost to the poisoned ASes if a less specific IP block is not advertised to cover for the poisoned AS. In our work, we conducted an assessment of the ability of a poisoning AS without a less specific covering prefix to maintain reachability to the poisoned ASes, finding that 30% of all ASes on the Internet can reach a /25 prefix advertised by an AS with only a /24, which is critical to effective BGP poisoning in an IPv4-dominated Internet. Next, we investigated default routing on the Internet and found that for 36% of ASes with only 2 providers (that is, multi-homed in the simplest case), even when the primary provider is poisoned, the AS will continue to route through it. This finding hints that placing Waterfall resistors nearest the last-mile yields benefits even in the presence of routing-capable censors. We set out to uncover the raw amount of poisoning that a routing-capable AS can carry out, which has direct implications for the effectiveness of systems such as Nyx in practice. We find that ASes can propagate paths up to 251 in length that are accepted by 99% of the Internet via customer cone inference. Critically, we also find that the Nyx system performs roughly 30% worse in practice than in simulation, and that routing working groups do "walk the walk", and do not only "talk the talk". Perhaps intuitively, the larger the AS or ISP, the more filtering of poisons occurs, and the smaller, the less filtering. Throughout the rest of this paper, we dive deeper into these and additional analysis of BGP poisoning's feasibility on the Internet. We summarize our results, key takeaways, impacts on existing security systems, and security ramifications in Table 4.1.

Table 4.1: Experiment summaries and their takeaways, impacted security systems, and ramifications for Internet routing security in general.

Experiment Conducted	High-Level Description	Key Takeaways	Existing Security Systems Impacted	Security Ramifications
Section 4.5: Steering Return Paths	Explores which ASes can effectively conduct poisoning, as well as the properties of alternative paths	77% of cases with successful path steering, Avg. 3 new unique traversed paths, minimal poisons needed, $< 1\%$ latency increase for alternate paths	Waterfall of Liberty, RAD, Nyx, Feasible Nyx, LIFEGUARD	Real-world evidence supports the claims of poisoning-enabled systems, with caveats for specific topological cases
Section 4.5.2: Re-Evaluation of Nyx	Re-evaluates the Nyx DDoS defense system with active measurements directly compared to simulated results	Nyx performs 30% less effective in practice, the inferred topology of the Internet used in simulation often does not match the topology and policies in practice	Nyx, Feasible Nyx, LIFEGUARD	Success of a system in simulation and/or passive measurement *does not guarantee* success (or the same findings) in the real-world
Section 4.6.1: Filtering of Poisoned Advertisements	Investigates the ASes that filter BGP poisoned advertisements as well as their relative size and other metadata	80% of ASes with less than 2,500 customers can be poisoned to 99% of the Internet	Waterfall of Liberty, RAD, Nyx, Feasible Nyx, LIFEGUARD	For specific parts of the Internet topology, poisoning does not work very effectively, allowing systems that would otherwise be hampered by poisoning to thrive in specific regions of the Internet
Section 4.6.2, 4.6.3 Filtering of Long Poisoned Paths	Establishes an upper bound for the maximum path length able to be advertised on the Internet with BGP poisoning	Max path length of up to 255 ASes propagated to 99% of the Internet	RAD, Nyx, Feasible Nyx, LIFEGUARD	Security systems which use poisoning have a fixed budget of poisons in reality, specifically 245 when factoring in the length of a normal AS path
Section 4.7.1: Declining Presence of Default Routes	Discovers the prevalence and distribution of default routes on the Internet	For 1,460 samples, 55% of fringe or no-customer ASes had default routes, while $< 10\%$ of transit ASes had default routes	Waterfall of Liberty, RAD, Nyx, Feasible Nyx, LIFEGUARD	Default routes do impact poisoning-enabled systems negatively, but can be avoided in specific topological cases, especially when the system is not deployed at the edge of the Internet
Section 4.7.2: Growth of /25 Reachability	Uncovers how many ASes must lose reachability to poisoned ASes when leveraging BGP poisoning	56% of observed ASes will propagate /25 prefixes and 31% of ASes respond to traceroutes for a /25	RAD, Nyx, Feasible Nyx, LIFEGUARD	Reachability of /25 prefixes limits some systems using poisoning, but for most cases, poisoning-enabled systems claims hold up in an Internet that has changed greatly since the earliest measurements of /25 reachability

Contributions The rest of this chapter explores the following contributions:

- We conducted the largest measurement study on BGP poisoning to date, comprising 1,460 successful/1,888 total poisoning cases. We publish our dataset, source code, and data analysis from the final results of this paper. [1] See Section 4.5.
- We reproduce recent security papers done in-simulation or passively, but now with active BGP poisoning on the live Internet. See Sections 4.5.2 and 4.6.3.
- We constructed statistical models that serve as a first-step towards utilizing BGP poisoning as an AS operator *without* requiring active tests or convincing senior IT administrators. See Section 4.5.3.
- We assess the extent and impact of poisoned path filtering from several perspectives. For this analysis, see Section 4.6.
- We reassessed the Internet's behavior with respect to default routes and /25 reachability one decade after the first exploration. See Sections 4.7.1 and 4.7.2 for these findings.
- We discuss insights and recommendations for the use (or threat model inclusion) of poisoning in security and measurement work going forward. We cover these insights within Sections 4.5, 4.6,and 4.7, with summaries in subsections at the end of these sections.
- We conclude this thrust with a discussion in Section 4.8 about the reproducibility and limitations of our experiments.

[1] https://github.com/VolSec/active-bgp-measurement

4.2 Background

We discussed BGP poisoning as a primitive in Chapter 3, and evaluated it via simulation. Here, we lay out BGP poisoning's impact on real-world deployment of Nyx and other security systems.

4.2.1 Impact of BGP Poisoning on Internet Security

There are certain security systems that *directly use BGP poisoning* to achieve their stated goals. In addition, other security systems rely on certain AS path properties to provide security guarantees. If an adversary could choose routes used by these security systems via BGP poisoning, then the claims of these systems would be undermined.

In the realm of censorship, BGP poisoning has been used by Schuchard *et al.* [122] with *Routing Around Decoys* (RAD) to attack censorship circumvention systems, specifically those predicated on Decoy Routing (DR). Decoy routing is a recent technique in censorship circumvention where circumvention is implemented with help from volunteer Internet autonomous systems, called decoys. These decoys appear to route traffic to a decoy destination, but instead form a covert tunnel to the actual destination to evade the censor. In the RAD paper, only outbound BGP paths were altered to allow censors to route around decoys, but inbound paths could also be altered to avoid decoy routers. In response to this approach to routing around decoys, work by Houmansadr *et al.* [60, 98] presented defenses against RAD, including the *Waterfall of Liberty*. Waterfall places decoy routers on return paths under the assumption that RAD adversaries can not control these paths. Our study exposes the relative invalidity of this assumption.

Following from Waterfall, additional work was done by Goldberg *et al.* and others [25, 91] built on top of the return path decoy placement; thus, literature continues to emerge while operating under assumptions not entirely true in practice. Arguing that RAD placement was infeasible financially, Houmansadr *et al.* [97] showed the costs of RAD in practice, while Gosain *et al.* [54] places decoy routers to intercept the most traffic. Both approaches could be circumvented when BGP poisoning works successfully at certain topological positions.

In particular, we use BGP poisoning to provide DDoS resistance with Nyx [130] and Katz-Bassett *et al.* uses poisoning for link failure avoidance with LIFEGUARD [70]. Nyx uses poisoning to alter the return paths of remote ASes to a poisoning AS, in an attempt to route the remote ASes' traffic around Link Flooding DDoS Attacks. LIFEGUARD uses poisoning to route around localized link failures between cloud hosts in AWS. Despite their success in simulation and limited sample sizes in practice, these systems assumptions need expansion and further validation at a wider scale to be used effectively for network defense. Tran *et al.*'s [149] feasibility study of Nyx raises issues with poisoning needed to steer traffic, but fails to evaluate their assumptions via real-world active measurements. Instead, they rely on passive measurement and simulation. Our findings demonstrate how the real-world limitations of BGP poisoning affect these systems, specifically when BGP path steering via poisoning is used in a defensive context.

Since BGP poisoning is a non-standard technique, it is not widely known how feasible it is across the entire Internet, and further it is not known how its real-world feasibility differs from its de facto feasibility when simulated. This lack of understanding directly impacts existing security systems. As a result, we need to understand the real-world feasibility of BGP poisoning to shed light on the validity of these security systems' claims.

4.2.2 Key Terminology in this Measurement Study

We use the following terms in the rest of this paper:

Steered AS: The steered AS is a remote AS whose traffic is steered by the *poisoning AS* onto new paths revealed via poisoning.

Steered Path: Steered AS traffic is moved onto a new *steered path* by the poisoning AS' advertisements.

Poisoning AS: The poisoning AS exerts control over the *steered AS* for security, measurement, performance, or other purposes.

Poisoned AS: Poisoned ASes are those being *prepended* to advertisements by the *poisoning AS* to steer paths.

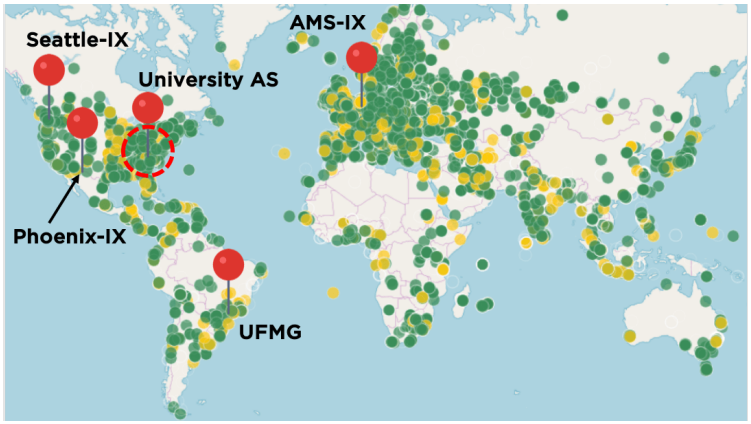


Figure 4.1: Distribution of RIPE Atlas traceroute probes at time of experiments with overlaid BGP routers

4.3 Experiment Infrastructure

The software-driven infrastructure used in our experiments to uncover the feasibility of BGP poisoning coordinates a vast amount of Internet infrastructure. We leverage thousands of network probes across 10% of the ASes on the Internet and 92% of the countries around the world, 5 geographically diverse BGP router locations —including two within Internet Exchange Points (IXPs) —and 37 BGP update collectors spread throughout the Internet. Our sample size of experiment vantage and measurement points represents the best available publicly at the time of the experiments. The major components of our measurement infrastructure are shown in Figure 4.2. For a detailed discussion of our ethical considerations, please see the next section after we first cover our experiment infrastructure. We employ both existing and new network infrastructure in the *control-plane* and *data-plane*:

Control-Plane Infrastructure: We use BGP routers to advertise paths with poisoned announcements. The routers originate in a cooperating university AS, the University of Tennessee, Knoxville (AS 3450), and 4 routers from PEERING [121] advertised as AS

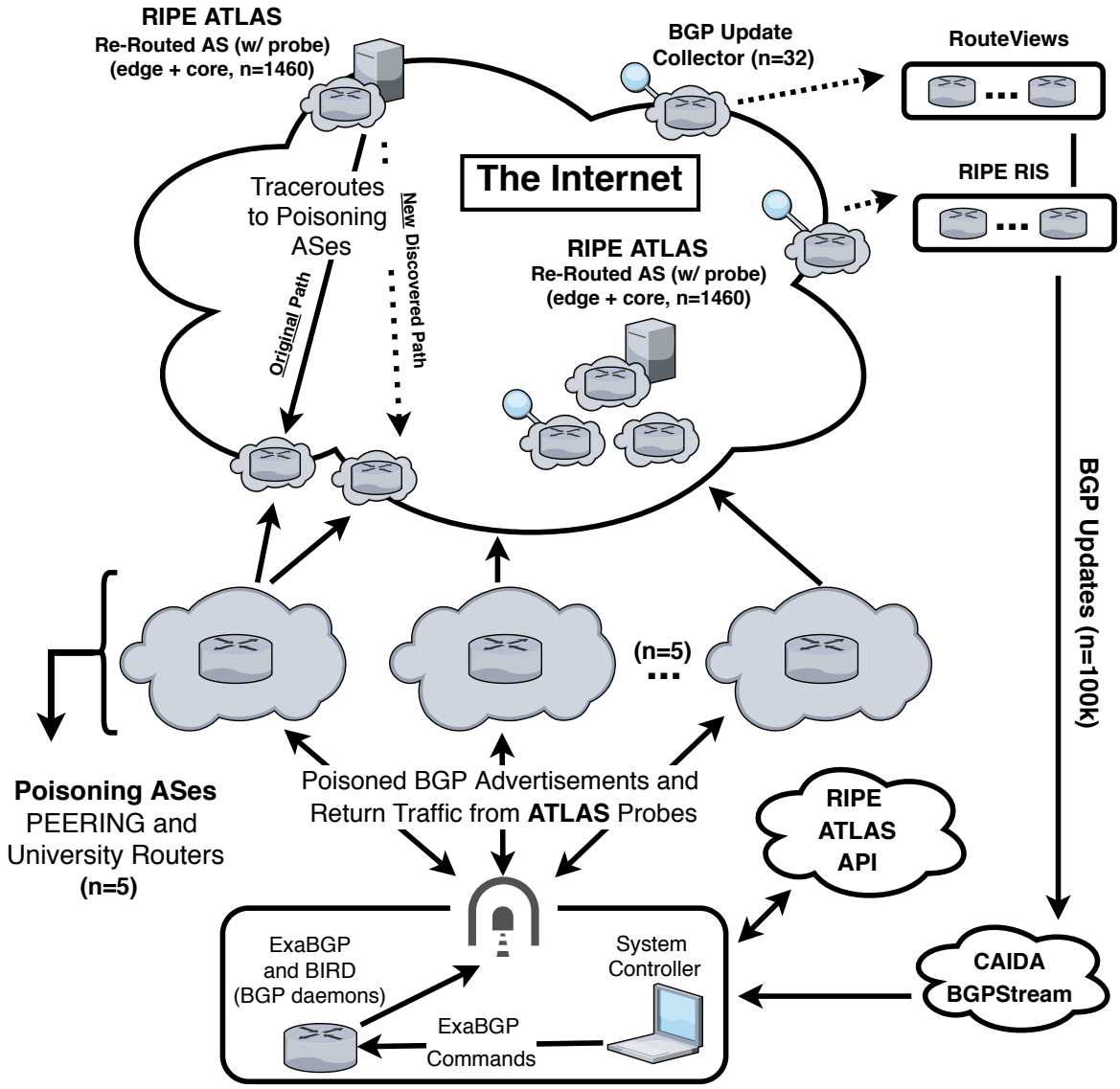


Figure 4.2: Measurement infrastructure from our experiments; incorporating CAIDA's BGPStream, RIPE Atlas, PEERING, a university AS, RouteViews, and RIPE RIS

47065. Figure 4.1 shows the routers distributed both geographically and topologically across 3 countries: USA, Brazil, and the Netherlands. While this geographic diversity does not necessarily correspond with topological (i.e. AS-level) diversity, we used all available BGP routers from PEERING to generalize our results.

Advertisements were sent to 26 upstream transit ASes plus 300 peers. This includes two IXPs within PEERING. We employ 8 unused, unique /24 prefixes from PEERING and two /24 prefixes from the university AS. Active experiments pause 2 minutes between measurements after each BGP advertisement, and in some cases 10 minutes or more for different measurements depending on infrastructure constraints. These wait times help prevent route-flap dampening [151, 112] and ensure expiration of MRAI timers [114].

Poisoned advertisements are in the following format, which provides neighbor validation for the first AS and ROV for the last. This setup mirrors existing usage of AS-path prepending for traffic engineering use cases.

$$\{\ AS_{origin},\ AS_{AV_1},\ AS_{AV_2}, \ldots,\ AS_{AV_N}, \underbrace{AV_{origin}}_{\text{For ROV}}\} \tag{4.1}$$

Finally, we monitor our BGP advertisements propagation from all 37 BGP update collectors available from CAIDA's BGPStream [109]. These collectors live physically within RouteViews [117] and RIPE NCC's network [116].

Data-Plane Infrastructure: We utilize RIPE Atlas [RIPE NCC] to measure data-plane reactions to poison announcements. In total, we were able to conduct traceroutes across 10% of the ASes on the Internet and 92% of the countries around the world. We leverage RIPE Atlas's mapping of IPs to ASNs for discovering the AS-level path. For the path steering

measurements using BGP poisoning, we only use 1 probe per AS, since we care about measuring new AS-level return paths, not router-level paths. We attempted to use every AS within the Atlas infrastructure as long as the probe was responsive and stable. We tried *all probes available*, but only 10% of ASes could be covered with responsive Atlas probes. Specifically, we found many of RIPE Atlas probes were either misconfigured, overloaded, or simply not working. This led us to ignore them and label them as "non-responsive". While a system such as PlanetLab may also have been useful, PlanetLab has significantly smaller AS coverage compared to Atlas.

Timing Considerations: To ensure that our advertisements on the control-plane have propagated successfully by the time of traceroutes, our experiments wait at least 2 minutes between measurements after each BGP advertisement, and in some cases 10 minutes if conducted via PEERING infrastructure. These wait times help prevent route-flap dampening [151, 112] and ensure expiration of minimum route advertisement timers [114]. We highlight additional data on BGP convergence times in the related work under Section 4.9.

4.4 Ethical Considerations

Our study conducts *active* measurements of routing behavior on the live Internet. As a result, we took several steps to ensure that our experiments did not result in the disruption of Internet traffic and were ethically sound. In particular, we ensure our experiments conform with the Menlo Report [35] and the policies of our infrastructure providers and external network operators. To that end, we first engaged with the operator community and leveraged their expertise throughout our experiments. Second, we designed experiments to have minimal impact on routers and the normal network traffic they carry. In this section we will touch on these steps.

Working with Operators: To ensure care was taken throughout all experiments, we worked extensively with the network operator responsible for campus-wide connectivity, quality-of-service, and routing at the university AS used in our experiments. This individual assisted in designing our experiments such that the concerns of external network operators on the Internet would not be affected adversely by our study, while also not biasing our results. In addition to the university operator, we worked extensively with PEERING operators from USC and Columbia throughout our study's design and execution. PEERING operators have a large amount of collective experience running active measurements on the Internet, which we leveraged to build non-disruptive experiments.

Significant care was taken to *notify* various groups of our activities. In accordance with the PEERING ethics policy, we announced to the RIPE and NANOG mailing lists prior to experiments the details of the study, allowing operators the ability to opt out. Over the course

of our experiments we monitored our own emails and the mailing lists. In total, 4 emails were received. Of the e-mails received, *no parties asked to opt-out*. For each email received, we responded promptly, explained our study, and incorporated any feedback provided.

Minimizing Experimental Impact: BGP path selection is conducted on a per-prefix basis. Meaning that advertisements for a particular prefix will only impact the routing of data bound for hosts in that prefix. The prefixes used for our experimental BGP advertisements were allocated either by PEERING or the university for the *express purpose of conducting these tests*. Outside of a single host that received traceroutes, no other hosts resided inside these IP prefixes. No traffic other than traceroutes executed as part of our experiments were re-routed. This includes traffic for other IP prefixes owned by the poisoning AS and any traffic to or from the poisoned ASes.

Another potential concern is the amount of added, and potentially unexpected, bandwidth load we place on links we steer routes onto. Since the only traffic that was re-routed as a result of the experiments was traceroute traffic bound for our own host, this added traffic load was exceptionally low. The bandwidth consumed by our measurements *did not exceed 1 Kbps at peak*.

Besides minimizing non-experimental traffic, we minimized the impact our BGP advertisements had on the routers themselves. Our BGP advertisements were spaced in intervals also ranging from *tens of minutes up to hours*. Resulting in a negligible increase to router workloads given that on average BGP routers currently receive 16 updates per second during normal operation [14]. All updates were withdrawn at the conclusion of each experiment,

preventing unnecessary updates occupying space in routing tables. Furthermore, all BGP updates conformed to the BGP RFC and were not malformed in any way.

The largest concern to operators were our experiments measuring the propagation of long paths on the Internet, described earlier in Section 4.6.2. Several historical incidents, most notably the SuproNet incident [sup], have demonstrated that exceptionally long AS level paths can potentially cause instability in BGP speakers. Underscoring this point were emails from operators on the NANOG mailing list [2] that appeared several months *before* our experiments complaining about instability in Quagga routers as a result of the advertisement of AS paths in excess of 1,000 ASes. As a result, all experiments involving long paths conformed to the filtering policies of our next hop providers. In the case of PEERING experiments, administrators limited our advertisements to 15 hops, and for the university, our upstream providers (two large transit providers located primarily in the United States) limited us to advertisements of 255 total ASes. These limits were enforced with filters both in the experimental infrastructure and at the upstream provider. In addition, such experiments were conducted with 40 minute intervals between announcements in an effort to allow operators to contact us in the case of any instability resulting from our experiments.

[2] https://lists.gt.net/nanog/users/195871?search_string=bill%20herrin;#195871

4.5 Feasibility of Steering Return Paths

Our first set of experiments explore the degree to which a poisoning AS can, in practice rather than simulation, change the best path an arbitrary remote AS uses to reach the poisoning AS. We call this rerouting behavior *return path steering*. Many security-related reasons for an AS to utilize return path steering focus on finding paths which avoid *specific ASes*. As a consequence, we are interested in more than simply *if* an AS can steer returns paths. We are interested in the diversity of paths available to a poisoning AS, the graph-theoretic characteristics of new usable paths, and to understand where such return paths play into security systems. In this section we both quantify the number of potential return paths we can steer a remote AS onto and dive deeply into the properties of these alternative paths. This analysis includes quantifying the diversity of transit ASes along those alternative paths, computing weighted and unweighted minimum cuts of the topology based on AS properties, and exploring latency differences between alternative paths. We also attempt to reproduce past security research and build statistical models that represent how successfully an AS can conduct return path steering.

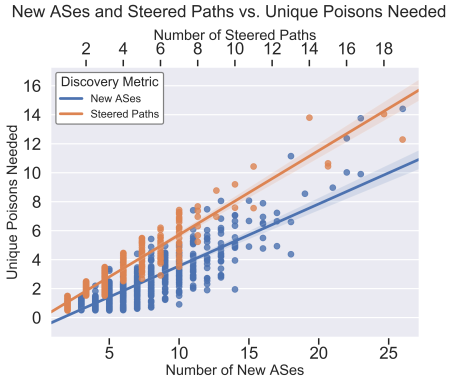
4.5.1 Experimental Design and Data Collection

We explore the properties of alternative return paths by enumerating the paths a poisoning AS can move a remote AS onto via return path steering. Our set of poisoning ASes consisted of all ASes hosting a PEERING router plus the university AS. When conducting poisoning, the poisoning AS will only steer *one remote/steered AS* at a time, where the remote AS is at least two AS-level hops away from the poisoning AS. This is critical to what we want to

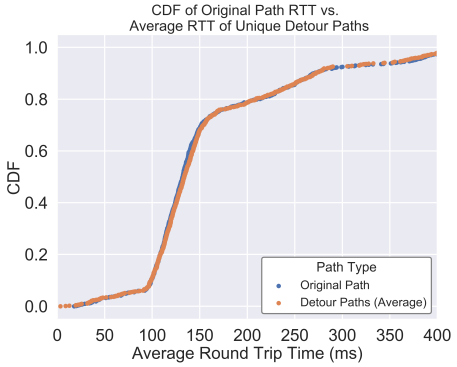
measure for security purposes. In this component of our study, we do not intend to measure new policies or congestion directly, as this has been done in prior work by Anwar *et al.* [13] which used multiple poisoning ASes from PEERING to steer the same set of remote ASes. However, Anwar *et al.*'s algorithm is fundamentally similar to ours. We use all available and responsive RIPE Atlas probes in unique ASes as steered AS targets. We collect BGP updates during the process in order to ensure our routes propagate and no disruption occurs. In total, we conducted our return path enumeration experiment for 1,888 individual remote ASes, or slightly more than 3% of the IPv4 ASes that participate in BGP [14]. We present the algorithm for the experiment below. The recursive function *SteerPaths* builds a *poison mapping*. This data structure maps the poisoned ASes required to reach a steered path. For all 1,888 instances, we capture the ASes and IPs of the original and all new paths, latencies at every hop, geographic IP locations, the set of poisoned ASes need to steer onto each successive path, and other relevant path metadata. This dataset will be made public upon acceptance. Our poisoning algorithm is measured to be successful when we see RIPE Atlas switch the path it is using to the poisoning AS. We infer that this sudden switch in path shortly after we confirm our poisoned BGP updates have propagated is due to the poisoning itself.

4.5.2 Steering Return Paths

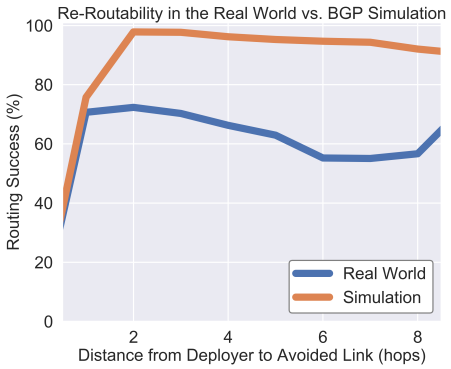
We successfully steered 1,460 out of 1,888 remote ASes, or 77%, onto at least one alternative return path. The unsuccessful cases arose due to default routes (discussed in Section 4.7.2) or poison filtering (discussed in Section 4.6). For each case of successful poisoning we analyzed



(a) Poisons Required to Discover Unique ASes and Entire Unique Return Paths with a Regression Line Fit



(b) RTT Comparison, Original vs New Return Paths



(c) Active Measurement vs Simulation for an Identical Set of Poisoning-Steered AS Pairs

Figure 4.3: Return path steering metrics. Figure 4.3a shows the number of poisons required to reach steered paths. Figure 4.3b shows the difference in measured RTT between original paths and steered paths. Figure 4.3c re-evaluates the Nyx defense

Algorithm: Recursive path steering algorithm

```
recursive function SteerPaths (src, dest, nextPoison, currentPoisons, mapping)
  Input: poisoning AS src, steered AS dest, next poisoned AS nextPoison, current poisons
         currentPoisons, poisonMapping mapping
currentPath = src.pathTo(dest)
poisonDepth = currentPath.indexOf(nextPoison)
previousHop = currentPath[poisonDepth − 1]
newPoisons = currentPoisons + nextPoison

dest.poison(newPoisons)
currentPath = src.pathTo(dest)
mapping.put(newPoisons, currentPath)
if currentPath == ∅ then
    disconnected = true;
end
newPrevHop = currentPath[poisonDepth − 1]
if !disconnected && newPrevHop == previousHop then
    SteerPaths(src, dest, currentPath[poisonDepth],
    newPoisons, mapping);
end
dest.poison(currentPoisons)
poisonIndex = currentPath.indexOf(nextPoison)
if currentPath[poisonDepth + 1] == dest then
    SteerPaths(src, dest, currentPath[poisonDepth + 1], currentPoisons, mapping);
end
```

several metrics: the number of unique alternate steered paths discovered and ASes traversed, the number of poisoned ASes needed to reach those paths, centrality measures of the graph formed by the steered paths, minimum cuts of this graph, and latency differences between the original path and the alternate return paths. Summary statistics of several of these measurements are shown in Table 4.2.

As shown by Figure 4.3a, for three quarters of (steered, poisoning) AS pairs, between 2 and 3 unique paths were reached on average using return path steering. However, for some pairs, we find nearly 20 unique paths. Clearly, some ASes are better positioned to execute return path steering. We dive deeper into which ASes can more easily execute return path steering using an array of statistical and machine learning models later in Section 4.5.3. The number of poisons required to reach these paths scales linearly with both the number

Table 4.2: Summary of return path steering metrics

Metric	Result
Cases of Unsuccessful Return Path Steering	428
Cases of Successful Return Path Steering	1,460
Overall Unique New ASes	1369
Average Unique Steered Paths Per Atlas AS	2.25
Average Unique New ASes Per Atlas AS	6.45
Max Unique Steered Paths	19
Max Unique New ASes	26
Avg. Poisons Needed vs. Avg. New ASes	2.03/6.45
Unique New ASes vs. Unique Poisons Needed	1369/468

of discovered alternate paths and the number of unique new ASes on those paths. This is relevant for many systems relying on return path steering, as each poison increases the advertised path length by one. We will demonstrate in Section 4.6 that path length is a major factor in AS operators' decision to filter or propagate received advertisements.

A comparison of round trip times between original and steered paths as measured by traceroutes is shown in Figure 4.3b. The original and steered round trip time (RTT) values are nearly indistinguishable. We find that on average we only observed a 2.03% increase in latency on alternative return paths, a positive indication that the alternative return paths have similar performance characteristics. Interestingly, we also found that the new paths tested out of the university AS performed 2.4% *better* than the original paths, while the steered paths out of PEERING ASes performed 4% *worse* than the original paths. We believe this is attributable to the proximity of the university AS to the Internet's core, versus the relative distance from the PEERING ASes to the core. These latency measurements provide supporting evidence that the alternative paths are fit to carry traffic from an approximate performance perspective, though the best indicator of path performance would come from knowing the bandwidth of the links traversed. Unfortunately, such data is highly sensitive

and considered an industry trade secret for an ISP. Our approximation via the link round trip times is our best estimate to link viability, with more real-world applicability than the PeeringDB estimated bandwidth model approach used in simulation by work from Nyx, as well as Tran and Kang [130, 149, 123].

Re-evaluation of Nyx

Next, we attempt to reproduce the performance of the Nyx Routing Around Congestion (RAC) system [130] with active measurements. Using the open source simulator from Nyx [131], we find 98% routing success (ability to steer an AS around a congested link) for the same 1,460 samples we measured in our previous experiment. We perform an exact comparison between simulated results and those measured actively. We find that in practice these ASes perform approximately 30% *worse*. We show these findings in Figure 4.3c using the same metric from the Nyx paper for the simulation and in practice comparison.

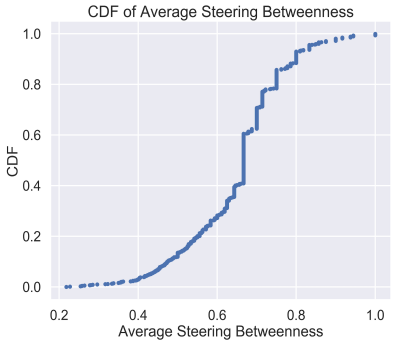
This apples-to-apples comparison illustrates that in most cases return path steering functions in practice, but the extent of that functionality is not necessarily as substantial as simulations based purely on AS-relationship models [150] imply. In Nyx simulation, CAIDA AS-Relationship models were used to show that ASes have tens to hundreds of available paths based on paths observed via advertisements. This is *significantly larger* than what we found in practice (2-3 unique paths). This is due to the CAIDA AS-relationships dataset only attempting to show *connectivity*, not policy. In other words, an AS-to-AS link observed in BGP advertisements does not indicate real-world willingness to send traffic over such links. While CAIDA's data represents the best possible model for simulation, it is clear that simulations relying on the routing infrastructure should be validated by active measurement.

While Anwar *et al.* [13] find connectivity that CAIDA does not, we find that when considering only a single poisoning AS steering a single remote AS, the poisoning AS can not achieve the full spectrum of return paths shown by AS connectivity alone. We also find that a poisoning/steered AS pair in simulation often had a longer original path length than was measured in the real world. The simulator found no original paths where the length was 3 AS hops. For the same sample set actively measured, we found paths with an original path lengths of 3 hops in 165/1,460 successful steering cases.

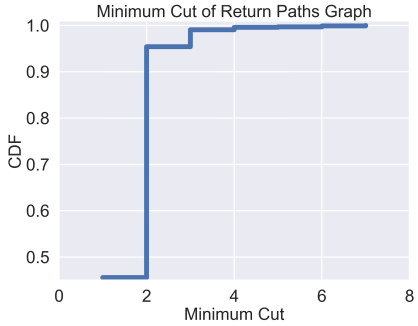
Clearly, assumptions from the simulation of Nyx did not match what was discovered in reality. First, as stated before, inferred policies of Internet routes *do not* match what is found in practice. Thus, both simulation of the inferred topology, and passive measurement of all paths seen, cannot directly justify individual policies that will actually be taken by a poisoning AS, which may contribute to the 30% difference in success. Second, Nyx did not factor in the effects of ASes that filter poisons on the success of routing around congestion. While the simulation allowed paths to propagate without filtering, this is not true in practice for certain cases, as we discuss in the next Section. Third, Nyx did not limit the amount of new paths that could be used to re-route during simulation, while in practice this is restricted to 2.5 available alternate paths. Therefore, the success from simulation will appear higher due to the lack of this restriction.

Graph Theoretic Analysis of Return Path Diversity

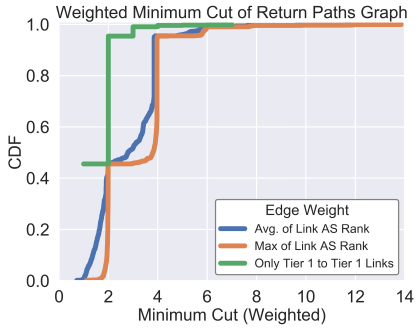
Here we analyze characteristics of the directed aclycic graph formed by combining the original and alternate return paths from the steered AS to the poisoning AS across all steering experiments (1,888 instances). We call this the *return path graph*.



(a) Average Normalized Steering Betweenness



(b) Unweighted Min Cut



(c) Weighted Min Cuts

Figure 4.4: Centrality measures of the importance of individual ASes in the directed acyclic graph formed by the original path and steered paths. Figure 4.4a shows the average vertex betweenness for ASes in each of the graphs, normalized by the number of distinct paths between steered and poisoning AS. Figure 4.4b and 4.4c show the unweighted and weighted min cuts of these graphs

One first concern is AS-level path diversity of the return path graph; how different are potential return paths are in terms of the AS-level hops they contain? This is relevant because security systems built on return path steering may seek to avoid specific links (e.g., to route around congestion). In this case, the availability of alternate return paths alone is not sufficient. The poisoning AS requires a return path *that does not contain* the congested link. Here we quantify the diversity of return paths by calculating the average betweenness for AS nodes on the return path graph. For each AS in the graph, we count how many paths the AS appears on, and divide by the number of total return paths (original and all discovered alternates). This yields a normalized betweenness for each AS between 0 (exclusive) and 1. The average betweenness for ASes on the return path graph, which we call *steering betweenness*, is designed to explore the diversity of ASes along the original and alternate return paths. A steering betweenness approaching 1 implies that the set of possible return paths differ in AS hops very little, while a number close to 0 implies that there are few ASes found in multiple return paths.

Figure 4.4a shows steering betweenness for each poisoning/steered AS pair in our experiments. We see that on average, a transited AS from the return path graph has a betweenness centrality of roughly 0.667. This indicates that some ASes appear on the majority of return paths. However, these paths are *not* essentially identical.

Next, We also compute the unweighted and weighted minimum cut of the return path graph. Here we seek to explore the prevalence of bottlenecks, or links that can not be steered around, in the set of return paths. This metric is especially meaningful for systems like Nyx that use BGP poisoning to maintain connectivity between a selected AS (the steered AS in the context of this experiment) and a Nyx deployer (the poisoning AS) in the presence of

a DDoS attack, since a low minimum cut reflects an unavoidable bottleneck for DDoS to target. Figure 4.4b demonstrates that in just under half of cases a single bottleneck exists, and for more than 90% of steered/poisoning AS pairs, a bottleneck of at most two links exists in this graph. In general, a bottleneck indicates that re-routing is limited due to there being an unavoidable link to cross. This finding weakens the use of re-routing for that particular topological case where the bottleneck is only a single link.

To explore where in the topology these bottlenecks occur, we constructed different methods for weighting the graph, seen in Figure 4.4c. First, we assign infinite weight to all Tier-1 to Tier-1 links to effectively remove them from consideration in the minimum cut, as the real-world capacity of links between large providers is, intuitively, much greater than links between other ASes. Consequently, we expect they are more difficult to degrade. Tier-1 ASes are those ASes who have no providers, and can therefore transit traffic to any other AS without incurring monetary costs [107]. Interestingly, this did not change the minimum cut for any graph, meaning that the bottlenecks did *not occur as a result of single unavoidable Tier-1 provider*.

To account for the difference in link bandwidth that likely exists between links serving larger ASes compared to smaller ones, we also assigned weights based on CAIDA's AS rank [7]. This rank orders ASes by their customer cone size. An AS's customer cone is the set of ASes that are reachable by customer links from the AS [79]. While CAIDA's AS rank is in descending order (rank 1 having largest customer cone) we invert the order for weighting purposes so that higher link weights indicate larger endpoint AS customer cones. To capture link capacity as a function of AS endpoint customer cone size, we use both the average and maximum rank (of link AS endpoints) as edge weights. The results demonstrate

that within the set of graphs with the same unweighted minimum cut there exists widely different difficulties for attackers attempting to disconnect an AS. In fact, a large plurality of steered/poisoning AS pairs require a cut equivalent to one link between ASes with an average AS rank double that of the average AS rank (or two links between ASes of average rank). A majority require a cut at least twice as large, implying that *bottlenecks reside on edges touching large ASes.*

Return Path Diversity and Security Impact

For Nyx, our findings agree that return path steering can reach alternative paths. While our betweenness results show the same ASes appear often on multiple steered paths, our reproduction of Nyx shows that in more than 60% of cases there exists at least one steered path that avoids an arbitrary AS from the original return path. Therefore, Nyx may help the poisoning AS when it is an impacted bystander or when the adversary is targeting the Internet as a whole. Our min. cut measurements reveal that bottlenecks occur in these steered paths, but it is unlikely they are on the weakest links. This means that an adversary strategically targeting the poisoning AS could target the min. cuts, but must work harder to disconnect a Nyx AS over others. In a similar manner, operators can leverage our insights to gain insight into the types of available paths to use after a set of real-world link failures.

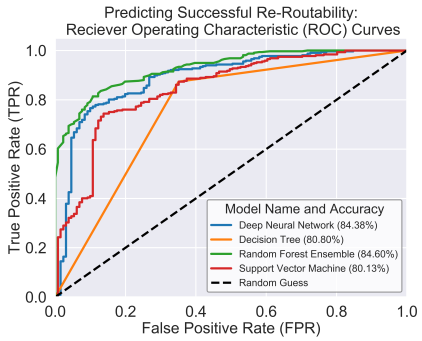
For censorship tools such as Waterfall [98], the success shown for return path steering presents issues. These systems now must consider attacks similar to Schuchard *et al.*'s RAD attack [122]. However, our centrality results reveal a significant betweenness, demonstrating that while alternative return paths exist, on average these paths transit a particular set of ASes that can not be steered around. The min. cut results further buttress this result,

and indicate strategic locations where censorship circumventors could place decoy routers to prevent a routing-capable adversary from routing around them with poisoning. Such an adversary would then not be able to use BGP poisoning to thwart the circumventor simply due to the filtering of the poisoning which inhibits their re-routing. Some work already approaches finding more diverse paths [97, 24, 54], but these systems also do not consider adversaries which can steer traffic around decoys on the return path. We suggest future studies examine poisoning from routers in censoring nations (e.g. China or Iran).

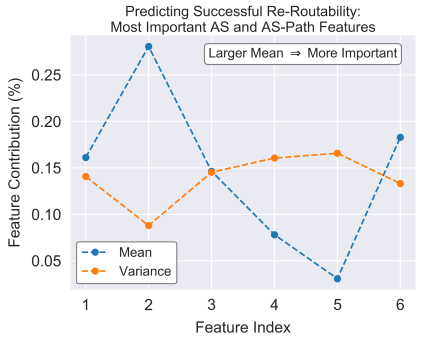
4.5.3 Predicting Successful Steering

To understand which ASes can execute return path steering most successfully, we constructed a set of statistical models. These models 1) predict which ASes can successfully steer traffic with poisoning and 2) determine the most important predictors for success of return path steering. Using the entire 1,460 sample dataset, we extract the following features from the real world active measurement data: distance on original path from poisoning AS to steered AS, poisoning AS's next-hop AS Rank, the steered next-hop AS rank, original path average edge betweenness, steered AS Rank, and original path average latency (over all hops). We selected these features based on properties that can be easily determined using standard traceroutes and by referencing open datasets such as CAIDA's AS Rank [7].

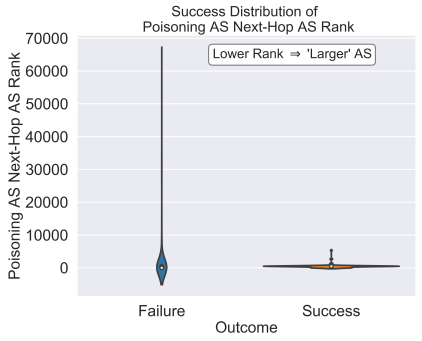
We first split the data into a 70/30 train-test-split. Then we scale the data by removing the mean and scaling to unit variance. In total, we employ 4 models: 4-layer fully-connected neural network, decision tree classifier, random forest ensemble classifier, and support vector machine. After fitting the data, we test the models with a 10-fold cross-validation. Then,



(a) ROC curves for different models predicting ASes that can execute return path steering



(b) Features analyzed with Principal Component Analysis, where Index 2 is the Poisoning AS's Next-Hop AS Rank



(c) Distribution of the Poisoning AS Next-Hop AS Rank vs the outcome of the return path steering

Figure 4.5: Predicting successful return path steering with both public and experimentally-derived path-based features: 1) Distance on original path from poisoning AS to steered AS, 2) poisoning AS's next-hop AS Rank, 3) steered next-hop AS rank, 4) original path average edge betweenness, 5) steered AS Rank, and 6) original path average latency (over all hops)

we plot Receiver Operating Characteristic curves in Figure 4.5a, which show the success of a given AS at return path steering. Specifically, the curves show the true positive rate vs. false positive rate distribution across models.

Overall, the models perform strongly. At 80.80% accuracy, the decision tree classifier both trains and tests new samples the quickest at < 1 second *and* is the most explainable. Explainability of machine learning models is critical here, since operators must inform their network administrators *why or why not* their network is fit for employing return path steering. Using only the feature vectors and their distribution, we now examine the features that express the most variance.

Figure 4.5b shows a Principal Component Analysis (PCA) algorithm used to rank all features by their mean and variance. The features with higher means indicate more important properties of the poisoning AS, steered AS, and pre-steering AS path. We find the most important predictor is the *next-hop AS rank of the poisoning AS*. As the number of available links to steer onto increases, the poisoning AS finds more unique paths. We can see this by examining Figure 4.5b Feature Index 2. The successful cases evolve from influential ASes as the poisoning ASes' next-hop provider or peer. By drilling down further into the distribution, we see in Figure 4.5c unsuccessful cases clustered around much smaller ASes. Note that path lengths average around 4 hops on the current Internet. In cases where a poisoning AS can not steer through the available paths at its next-hop, other diverse AS choices should exist at the later hops. Perhaps counter-intuitively, the least important predictor is the AS rank of the steered AS. This indicates that the relative influence, or size, of the steered AS *does not* affect a poisoning ASes' ability to steer them.

4.5.4 Security Ramifications and Takeaways

Our findings on the feasibility of steering return paths impact all security systems mentioned in Section 4.2.1, including Nyx, LIFEGUARD, RAD, Waterfall of Liberty, and Feasible Nyx [130, 70, 122, 98, 149]. Notably, the claims made by systems that leverage BGP poisoning are more in line, but not an exact match, with the behavior of the live Internet. Nyx can successfully execute its re-routing defense using poisoning, though with 30% less success than simulations show. In particular, poisoning-enabled victim ASes can defeat link-flooding adversaries that target the victim's provider links by executing Nyx to re-route onto alternate, uncongested ASes. Besides demonstrating that BGP poisoning does function in practice for many cases, these experiments also help underscore the need for real-world experiments when validating system design.

From the perspective of censorship, the feasibility of BGP poisoning allows censors to leverage RAD [122] to thwart the efforts of those wishing to avoid censorship with decoys or advanced defenses like Waterfall [98]. However, as we saw with Nyx, we do see that BGP poisoning functions less often than simulations would lead us to believe in specific cases. In the next section, we explore policies such as AS-level filtering which hamper the effectiveness of BGP poisoning, yet open the door for systems such as Waterfall to function effectively.

4.6 Extent and Impact of Filtering

In this section we present experiments that uncover ASes throughout the Internet which refuse to propagate BGP paths with poisoned ASes prepended. We term this effect *poison filtering*. We present evidence for how often ASes conduct poison filtering, a behavior that impacts the success of BGP poisoning. As we will discuss, ASes, especially larger ASes, may implement strict policies in their BGP routers which disregard "anomalous" paths. If the policies label poisoned paths as "anomalous", then the poisoning AS may see their route dropped at that filtering AS. This directly impedes a poisoning AS when they target a filtering AS in their poisoned advertisements.

We explore how many ASes propagate poisoned routes, how long of poisoned paths can be propagated, and additionally conduct a rigorous graph-theoretical analysis of the specific ASes by size inferred to be filtering long poisoned paths. We also attempt to reproduce recent work by Tran & Kang *et al.* [149] who used a dataset gathered through passive measurement (as opposed to active BGP measurements). In this analysis, we yet again demonstrate that simulation or passive measurement is *not enough* to empirically determine the behavior of the Internet.

4.6.1 Filtering of Poisoned Advertisements

Systems which depend on return path steering need the ability to avoid ASes of a variety of sizes. Since a poison is essentially a lie about the AS level path, it is natural to ask if ASes disregard lies about large ASes. This type of poison filtering would prevent systems using

return path steering from avoiding key ASes in the topology. In order to explore this, we measured the ability to propagate poisoned routes containing various sizes of poisoned ASes.

Experimental Design

First, we randomly sampled 5% of ASes seen in BGP updates from January 2018 by their degree of connectivity. In cases (like Cogent) where an AS has a unique degree of connectivity, we sample just that AS. However, when many ASes share a degree (e.g., 3), we sample 5% of those ASes uniform at random. With these ASes, we proceeded to advertise poisoned paths with one sampled AS prepended as the poison per advertisement. This announcement would appear as AS_P, AS_F, AS_P, where AS_P is our poisoning AS or measurement point, AS_F is the AS being tested for poison filtering. Prior work has found that the relative connectivity of an AS often determines its reaction to anomalous Internet events [107, 26] due to larger ASes necessarily enforcing certain policies based on the customers it serves. For each iteration, we initially announce the normal, non-poisoned prefix to all providers and peers connected to the university's AS. After waiting 40 minutes for BGPStream to continuously pull from update collectors in batches, we then fetch all observed updates from the prior 40 minutes, though our updates actually propagated within 30 seconds when observed from the actual collector based on update timestamps. We then measured how many unique ASes were observed advertising the original announcement. [3]

With a baseline taken for the non-poisoned announcement per prefix, we repeated this process, but now poisoning ASes by degree from high-low in the path example shown earlier. Again, we wait 40 minutes before collecting all updates and additionally *implicitly*

[3]BGP convergence happens nearly instantly with poisoned routes, see LIFEGUARD [70].

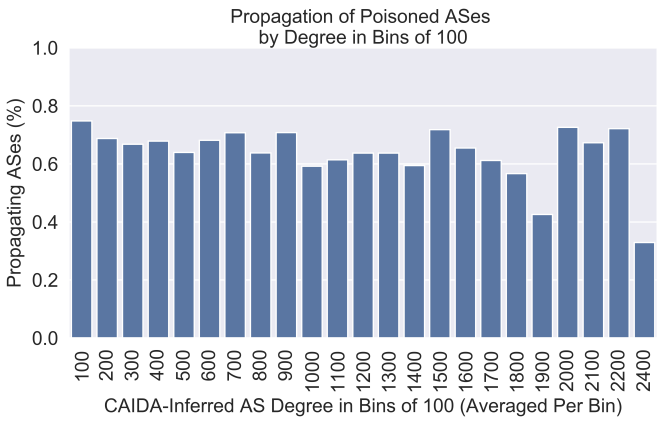


Figure 4.6: Filtering of AS paths increases as the poisoned AS increases in degree, an approximation for its influence on the Internet

withdraw each poisoned path after each iteration. We then compute the normalized percent of ASes propagating the poisoned paths. This measures the fraction of ASes advertising specific poisoned paths versus those who advertised the non-poisoned baseline path from the equivalent poisoning AS and poisoned prefix. If an AS propagated the non-poisoned path from our AS, and they also propagated a poisoned path, then the normalized percent is higher. In other words, this metric illustrates the percentage of ASes from our random sample that **do not** employ poison filtering.

Results and Discussion

The results of this measurement are shown in Figure 4.6. We have aggregated the normalized propagation percentages by AS degree into averages in bins of AS degrees from 0 to 99, 100 - 199, ..., 2300 - 2399. We observe that for AS degrees of less than 2,500, the ASes accepting and propagating the poisoned ASes is roughly the same, with between 70% to 80% of ASes continuing to propagate poisons. We did not show the most connected ASes

Table 4.3: Top 10 ASes by Degree and their normalized percent of ASes propagating paths with these ASes poisoned

Rank by Degree	ASN and Name	Degree	Number of Customers	Registered Country by ASN	Normalized Propagation Percentage
1	6939 - Hurricane Electric	7064	1202	United States	11.9%
2	174 - Cogent	5352	5272	United States	11.6%
3	3356 - Level 3	4980	4898	United States	11.6%
4	24482 - SG.GS	3382	24	Singapore	96.1%
5	3549 - Level 3 GBLX	2538	2446	Unites States	11.6%
6	7018 - AT&T	2373	2330	United States	0.05%
7	58511 - Anycast	2351	13	Australia	60.1%
8	49605 - IVO	2193	11	Italy	66.7%
9	8492 - OBIT Ltd.	2153	46	Russia	71.4%
10	8220 - COLT Tech. Grp.	2143	716	United Kingdom	78.2%

in Figure 4.6 due to their outlier status; instead, the top 10 ASes by degree are shown with their propagation data and other relevant AS metadata in Table 4.3. Notably, the largest degree AS is Hurricane Electric, a nearly Tier-1 AS. at 7,064 degree. Hurricane Electric has roughly 20% propagation compared to ASes with under 2,500 customers at roughly 70% propagation. In fact, the extent to which ASes refuse to propagate high degree poisons is confined to a very small sample of high-degree ASes. Only 4 have a propagation percentage of less than 30%, with AS degrees of 2,538, 4,980, 5,352, and 7,064.

First, systems such as Nyx [130] and RAD [122] assume all ASes *do not* conduct poison filtering. We present evidence that significant parts of the Internet do not allow poisoned routes to propagate, especially for the small amount of ASes with degrees greater than 1,000. This finding exemplifies the reason why systems such as Nyx do not find the nearly limitless available paths in practice as what is shown via CAIDA data. To that end, future systems employing BGP poisoning for defensive or offensive purposes should not assume all available paths can be steered onto.

For decoy-routing systems, decoy routers should be placed on AS paths that because of filtering, the adversary can not easily steer said path around the decoy. In scenarios where decoy placement leverages these strategies, the censors may face a losing scenario.

Also shown in Table 4.3, the number of customers an AS has seems to indicate the extent of poison filtering. For example, despite AS 24482 (an ISP in Singapore) having the 4th highest level of AS connectivity, it only provides direct customer transit to 24 ASes. Accordingly, this Singaporean AS has a much higher propagation percentage relative to ASes with similar degree but more customers. In the case of AS 24482, the non-transit ASes pumping up the AS degree may be peers. Clearly, while paths with larger ASes seen in poisoned paths may be filtered more often, it is not always the case based on AS 24482. With over 3000 ASes reported as connected by CAIDA [150], the amount of propagation was still 96% of a normal non-poisoned path.

4.6.2 Filtering of Long Poisoned Paths

Our next experiment investigates the maximum amount of poisoned ASes a poisoning AS can spread throughout the Internet via successively longer path lengths. In existing security systems, Nyx [130] advertises long poisoned paths to avoid dragging along non-critical traffic when steering remote ASes around congestion. RAD [122] and censorship tools using BGP poisoning must rely on many poisons to steer traffic around decoy routers. AS relationship and policy inference methods could use our path steering algorithm from Section 4.5 coupled with longer poisoned paths to explore broader AS-to-AS business relationships [13]. Congestion discovery systems could also benefit from greater topological visibility.

To that end, we have conducted what we believe is the *most exhaustive measurement of maximum path length on the Internet.* This experiment provides valuable information on whether common models of routing hold in practice. Though the BGP specification [114] does

not place an upper bound on path length, the BGP best practices RFC [39] recommends that excessively long paths should be filtered. Furthermore, statistics from the APNIC routing registry [14] show most maximum path lengths observed well-under what should be possible. Many Cisco forum posts also hint at operators that assume all paths are filtered over 50 in length. Fortunately, we were able to conduct our experiment from the university AS with permission over two large ISP transit links, without the path length restrictions of PEERING. The university AS's providers have the explicit policy of filtering BGP advertisements longer than 255 hops. Therefore, even though paths may extend beyond this in some router's policies, we can only observe the propagation of path lengths up to 255.

Experimental Design

Similar to the poison-filtering approach in Section 4.6, we first announce a normal baseline path with no poisons. After collecting the baseline number of ASes advertising the normal path and withdrawing the baseline advertisement, we then iteratively poison paths of increasing lengths in intervals of 40 minutes, from 1 poisoned AS prepended to the path to 135 poisons by one at a time. Once we reached 135 poisons, we shifted to poisoning in successive iterations of 5, going from 135 to 500. After every iteration of path length increase, we implicitly withdrew the prior advertisement. During propagation throughout the Internet, we collect all BGP updates from collectors managed by BGPStream [109], which we again use to measure the normalized percentage of ASes propagating the poisoned paths. In practice, the path would look similar to the path in Equation 4.2, where AS_I, AS_J, and AS_K are normal ASes forwarding the prefix; AS_{Orig} is the poisoning AS; and AS_{P_1} through AS_{P_n} are the prepended poisons.

$$AS_I,\ AS_J,\ AS_K,\ AS_{Orig},\ AS_{P_1},\ AS_{P_2},\ AS_{P_3},\ ...,\ AS_{P_n},\ AS_{Orig} \tag{4.2}$$

We conducted this experiment with two sets of ASes to prepend: 1) randomly sampled, in-use ASes from the CAIDA topology to most closely mirror a poisoned path needed for return path steering, and 2) using the university AS as a self-prepend. We ensured part of the in-use AS sample included both ASes on the edge of the topology (those with no customers), as well as transit ASes small and large (those with more than 5 customers) according to prior classifications of AS types by UCLA [107].

Results and Discussion

Displayed in Figure 4.7 for both the randomly sampled ASes from CAIDA and for the self-prepended university AS, we present a rigorously evaluated upper bound on the max path length of the Internet of **251**. This path propagated to over 99% of the Internet when including customer cones of AS's forwarding the path. This included highly connected ASes such as Level 3 and Cogent. Figure 4.7 matches an operator's intuition that as paths grow longer, they are less accepted throughout the Internet, though still roughly 75% of BGP collectors observed the longest path lengths detected.

With this information, systems such as Nyx [130] now have an upper bound for the amount of poisoned ASes usable for path lining, which was estimated with passive, not active, measurements in Tran & Kang *et al.*'s re-routing feasibility study [149]. Since Nyx did not limit the poisons, our reproduction of Nyx earlier incorporates this poison limit, finding less success overall when steering return paths. When re-routing around localized

failures, as Katz-Bassett *et al.* [70] did between Amazon AWS instances in LIFEGUARD, this maximum length limits the amount of path steering in practice that can be achieved. There are implications for RAD [122] and other decoy routing adversaries as well: the more poisons possible, the harder Waterfall [98, 25, 97] must work to place decoys.

4.6.3 Which ASes Filter Long Paths

Filtering Inference Algorithm

Here we investigate which ASes are filtering paths based on data collected in the prior experiment. We develop a new inference algorithm to discover which ASes filter long poisoned paths based on a comparison of paths received by route collectors at each advertisement of successively greater length. First, we build a directed acyclic graph D of all paths p observed on paths from the university AS to collectors. The nodes of D are ASes appearing on paths; edges represent links between them. Next, for each advertisement i of successively greater path length, we build a set of ASes A_i composed of all ASes appearing on our advertised paths that reached route collectors. Finally, we remove all $a \in A_i$ from a copy of D, creating D_i. For each weakly connected component remaining in D_i, we learn that 1) at minimum, the roots of each component filtered the advertisement, and 2) at maximum, all AS nodes $a \in D_i$ filtered it. Using this method, we iteratively build maximum and minimum inferred filtering AS sets for every path length in our experiment.

Results and Discussion

Our results are grouped using the aforementioned, widely-adopted AS classification scheme presented in [107]. ASes are divided into Tier-1 (can transit traffic to all ASes without compensation and form a clique), Large ISPs with over 50 customers, Small ISPs with between 5 and 50 customers, and Stub ASes, those with less than 5 customers. Figure 4.8a displays our results for Tier-1s and Large ISPs; Figure 4.8b gives the same information for Small ISPs and Stub ASes. Naturally, the ephemeral structure of the Internet topology introduces noise into our results. Additionally, it is more difficult to draw conclusions about Tier-1 and Large ISP filtering behavior using our method, as the minimum and maximum inferences diverge significantly. This is likely due to advertisements being filtered before reaching these ASes as they propagate outward from the university AS. So, these ASes are rarely the root of the weakly connected components used to infer minimum filtering, and we conjecture that the true filtering rate for these classes is closer to the maximum inference.

Overall, the results indicate that Tier-1's and Large ISPs filter long paths more aggressively than Small ISPs and Stub ASes, and that AS filtering policies are highly fragmented. In a feasibility study on Nyx/RAC by Tran *et al.* [149], the authors utilize a distribution of observed path lengths from passive measurement to hypothesize about AS filtering rates. In short, they suggest that some filtering occurs on paths of length 30 - 75, no increase in filtering occurs between 75 and 255, and paths of length 255 or greater are almost universally filtered. We were limited by university AS provider policy from experimenting with paths over length 255, but their findings align well with our own for Small ISPs and Stub ASes. For the larger ASes, our experiments indicate that the rate of filtering does

in fact increase after a path length of 75. Additionally, our results capture the intuition that larger, more influential ASes should filter often. We find that of all tiers of ASes, the Tier 1 ASes filter most, while larger ISPs filter less but close behind Tier 1 ASes shown in Figure 4.8a. Finally, small ISPs and stub ASes filter very little as shown by Figure 4.8b.

4.6.4 Case Study: Filtering by an ISP-driven Working Group

MANRS [84], Mutually Agreed Norms for Routing Security, is a global Internet routing security initiative that develops and publishes best practices for network operators. Path filtering is one area of concern for MANRS, and they publish standards for following RPKI and other BGP security mechanisms that member ASes are expected to implement. The 120+ MANRS ASes represent a distinct set of ASes that intuitively should be most likely to filter BGP advertisements similar to poisoned updates. They include Cogent, Charter Communications, CenturyLink, and Google.

In Figure 4.8c, we display the results of the same filtering inference algorithm used in the previous section, with results divided by MANRS and non-MANRS ASes. We observe a significant deviation in the inferred filtering range between MANRS and non-MANRS ASes, suggesting that MANRS operators may implement tighter filtering policies. This key result indicates that an ISP's participation in an Internet consortium such as MANRS may actually correspond with stricter implementations by the operators responsible for day-to-day network activity and filtering policy, rather than aligning with MANRS only at an organizational level.

4.6.5 Security Ramifications and Takeaways

Our findings on poison filtering impact all security systems mentioned in Section 4.2.1, including Nyx, LIFEGUARD, RAD, Waterfall of Liberty, and Feasible Nyx [130, 70, 122, 98, 149]. While the results of Section 4.5 show that BGP poisoning *broadly* functions, these experiments reveal that AS-level filtering in portions of the Internet results in *specific topological locations* where poisoning *does not* function. As a result, systems like Waterfall, which depend on BGP poisoning not to function should seek out those locations for deployment. By seeking out topological regions of the Internet where poisoning is not effective, some of which were described earlier, censorship circumvention systems can avoid RAD adversaries wishing to route around decoys.

From the perspective of DDoS, Nyx cannot easily route around DDoS in the filtered regions of the Internet. This filtering contributes to the weakened effectiveness of Nyx in practice that we saw in the prior section. The filtering of long paths also affects Nyx by limiting its ability to poison the neighbors of the alternate paths, which prevents dragging along unintended traffic, thus hampering the relief of attack congestion. Notably, these experiments also put a hard real-world limit on Internet path length. At 255 ASes, Internet paths fail to propagate; however, up to 255, paths propagate to 99% of ASes, unlike what was found with passive measurement in prior studies [149]. ASes seeking to propagate long paths should limit the number of poisons used to 245, which accounts for up to 10 ASes for the actual path. Given that the average path length is between 3 and 4 ASes [14], this amount of room is suitable for real-world deployment of poisoning.

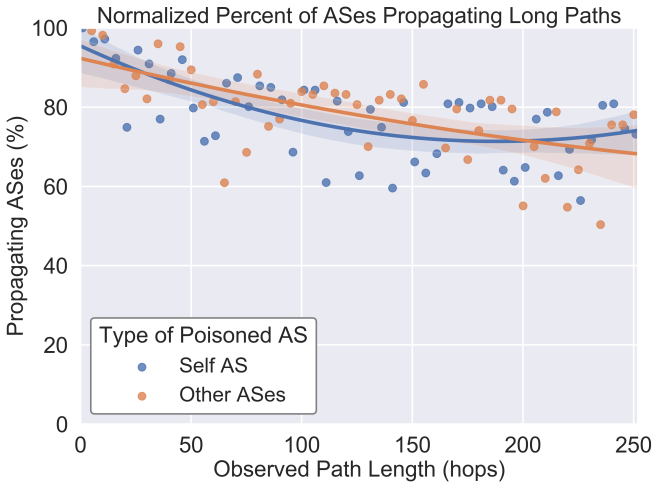
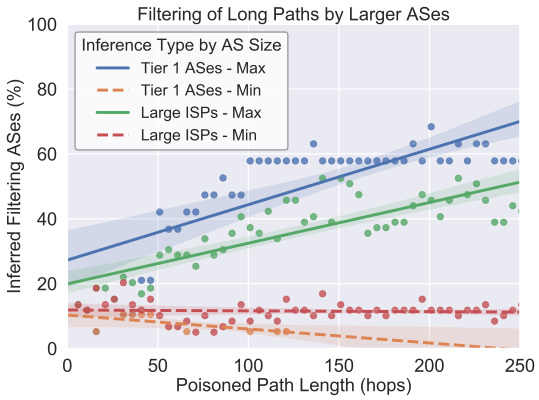
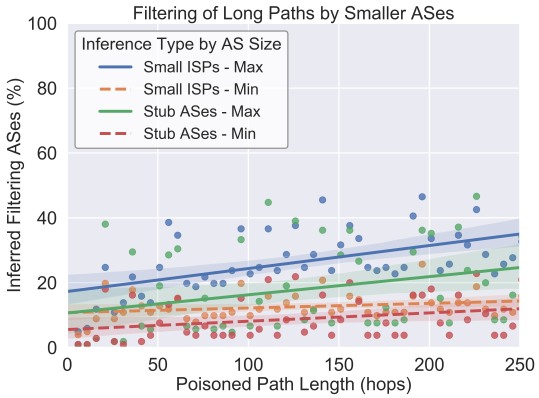


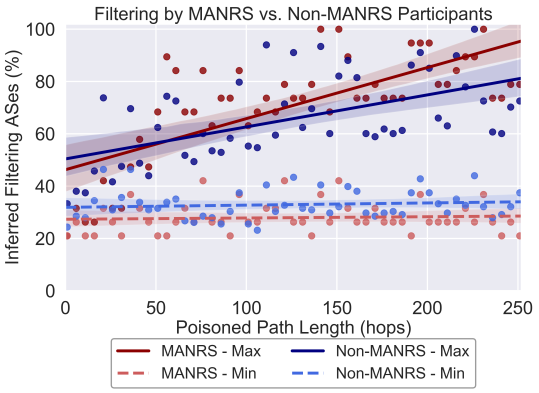
Figure 4.7: With paths up to 250 in length, we found over 80% of ASes treated 250-length paths the same as normal paths (Regression Fit of Order 2)



(a) Tier-1 and Large ISPs



(b) Small ISPs and Stub ASes



(c) MANRS vs non-MANRS ASes

Figure 4.8: Minimum and maximum inferred filtering for ASes classified by tier and MANRS membership, each with a regression fit

4.7 Reassessing Reachability

As part of our study, we setup our infrastructure to attempt to reproduce Internet measurements from nearly a decade ago by Bush *et al.* in Internet Optometry at IMC 2009 [26]. These measurements from 2009 have a distinct impact on the feasibility of BGP poisoning for security. This section presents our findings for an Internet in 2018 with over 60,000 ASes and estimated 3.8 billion unique users, compared to nearly 25,000 ASes and 1.7 billion users in 2009 [138, 14]. Once again, we found cases in this evaluation where common assumptions from the operator community *did not coincide* with actively measured Internet behavior.

4.7.1 Declining Presence of Default Routes

Experimental Design

Default routes exist when an AS has two or more providers and refuses to choose a second provider when the first provider is "removed" from the topology via BGP poisoning. A poisoning AS can theoretically remove the first provider from the steered AS's topology by causing the first provider to drop (and not propagate) its route to the poisoning AS. However, when measured in 2009 [26], this did not always occur when poisoning. Default routes were evaluated as part of the earlier return path steering experiments.

Results and Discussion

We found 330/1,460 successful poisoning cases exhibited default routes at the steering AS's next hop. Thus, steering traffic onto a second, third, or other provider bordering the remote

Table 4.4: Default Route Findings

Measurement	Number of Instances
Fraction of Total Samples with Only 1 Provider (not multi-homed)	28.7% (419 / 1,460 total samples)
Fraction of Total Multi-Homed Samples with Default Routes	48.6% (506 / 1,041 multi-homed samples)
Fraction of Transit ASes with Default Routes	26.8% (196 / 731 total Transit ASes)
Fraction of Stub/Edge/Fringe ASes with Default Routes	36.7% (310 / 845 total Fringe ASes)

Table 4.5: /25 Reachability Findings

Prefix Length	Measurement	Findings	Timespan of Measurement
/25	BGP Observability	Seen at 21/37 (56.7%) collectors	96 hours of collection
/25	Traceroute Reachability	31% reached /25 prefix on average	7 hours; 5,000 distinct traceroutes every 1 hour
/24	BGP Observability	Seen at 34/37 (91.8%) collectors	96 hours of collection

AS was impossible for these ASes. As shown by Table 4.4, we examined the properties of these ASes by their transit or fringe status. Default routes existed for 26.8% of the transit ASes we sampled, those being ASes with 5 or more customers. We also found that 36.7% of the fringe ASes, or those with less than 5 customers, had default routes. In 2009, 77% of stub ASes had default routes (out of 24,224 ASes measured from the poisoner with the ping utility). In 2018, we found 36.7% of stubs had default routes (out of 845 ASes measured from the steered AS with RIPE Atlas probes using traceroute).

Based on the prevalence of default routes, decoy routing systems [98, 97, 25] could optimize placement of decoy systems on the immediate next-hop of remote ASes which a RAD [122] adversary wishes to steer. This approach may yield stronger security guarantees against poisoning-equipped adversaries when the middle of the path exhibits strong compatibility with poisoning. For systems such as Nyx [130], a steered AS which has a default route may not be able to provide QoS to its customers when under a direct DDoS due to the inability to steer return paths around its next-hop AS that will always be used for return traffic.

4.7.2 Growth of /25 Reachability

When executing return path steering, an originating AS utilizing poisoning on a set of their own prefixes can cause the poisoned ASes and their traffic to *lose* connectivity to the poisoning AS's prefixes, because without a less specific covering prefix, the poisoned AS will have no path to the poisoning AS. Fortunately, a poisoning AS can still maintain reachability given a sufficient allocation of prefixes. However, not all ASes benefit from an ample supply

of controlled prefixes. To maintain reachability while poisoning, an AS must have both a more specific prefix and a less specific prefix. Current best practice documents recommend ASes filter advertisements with a prefix to be advertised longer than /24 [39]. Recall that the /24 prefix is what is advertised via BGP, leading to other ASes discovering a path to the hosts in that /24 prefix. Therefore, any AS which only has /24 prefixes available will lose reachability to poisoned ASes *unless* it can advertise a /25 as the poisoned prefix and use the /24 as the covering prefix. We searched the BGP RIB for an AS's shortest advertised prefix length and found that 48% of them only have /24 prefixes advertised, except for ASes that may have had /25's already in the default free zone. For an in-depth examination of BGP prefix delegation and who gets the privilege of many prefixes, see recent work by Krenc *et al.* [71]. With this necessary primitive for poisoning in mind, we set out to examine the amount of reachability a /24-only poisoning AS could retain to the poisoned ASes.

Experimental Design

For the control-plane measurement, we started by announcing a unique /24 across all 8 prefixes. Over the following 96 hours, we collected aggregated BGP updates from BGPStream [109]. This provides a baseline number of ASes propagating the path. We then withdrew each /24 prefix. Next, we announced a /25 prefix from the same set of locations across PEERING and the university AS, collecting the number of updates in the same manner again over 96 hours. With these two sets of ASes, we compute the normalized percent of ASes propagating the /24 versus the /25.

Next we measure data-plane reachability. We announced a /25 from our 8 prefixes. Then we scheduled 5,000 traceroutes, randomly sampled from all Atlas probes, directed to

the advertising /25 prefixes every hour for 7 hours. We recorded the number of traceroutes that reached the /25, noting this as the approximate reachability of the /25. We opted not to do the traceroutes for the /24 both due to PEERING being used for other experiments and because we expect a /24 to be reachable except in the case of faulty Atlas probes.

Results and Discussion

Our results from these experiments are shown in Table 4.5. We found that the current data-plane reachability of a /25 is roughly 30%, while the number of ASes propagating a /25 BGP announcement is over 50%, or *50x higher than 2009 results* in Bush *et al.* [26]. Notably, our analysis shows that the 48% of ASes without sufficient *recommended* prefixes to steer traffic actually *can maintain reachability*. We note that our results may not be able to be directly compared with Bush's work due to the use of traceroutes here over ping, but the comparison at least serves as a measuring stick for the growing Internet.

Our findings also have crucial implications for both existing and future security systems. Censors that do not have a less specific prefix than a /24 will be unreachable by ASes affected by path steering for a non-negligible amount of steered ASes. This may dampen a RAD [122] adversary's success in terms of economic means by lost traffic, while strengthening the case for circumvention tools [98, 25, 97]. Though loss of reachability may not be an issue for censorship entities in general, smaller censoring nations may sustain significant economic costs. Smaller nations may have few egress BGP paths to the broader Internet. Any additional sacrificed reachability may have substantial impacts on the businesses and users behind the border. However, the cases where BGP routers propagate a /25 may be enough in some attack scenarios for a RAD adversary, though this is topology-dependent

given a poisoning AS. We recommend that future iterations of the decoy routing attack and defense schemes factor in our findings by evaluating their success for classes of ASes with only a /24 prefix. As a defensive mechanism, operators of Nyx or LIFEGUARD [130] must be willing to sustain losses of reachability. Defensive measures must account for these new AS reachability metrics for ASes with few prefixes.

4.7.3 Security Ramifications and Takeaways

Our findings on reachability and default routes impact all security systems mentioned in Section 4.2.1, including Nyx, LIFEGUARD, RAD, Waterfall of Liberty, and Feasible Nyx [130, 70, 122, 98, 149]. Like filtering, default routes and losses of reachability limit what can be claimed by poisoning-enabled systems such as Nyx, RAD, and LIFEGUARD. From the perspective of DDoS, Nyx functions even with default routes, since attacks are often more than several hops away from the target in the case of Link Flooding Attacks (LFAs) [130]. Furthermore, our finding that 48% of ASes with only a /24 can in fact use a /25 is significant for Nyx and LIFEGUARD deployers with few IP prefixes. Thus, smaller ISPs can still execute BGP poisoning feasibly and maintain their connectivity via a less and more specific prefix.

However, our findings again limit these systems in specific cases. For example, decoy routers benefit from RAD [122] being unable to steer return paths due to default routes near the poisoning AS, or AS deploying RAD. Poisoning also impacts the reachability of censoring ASes' prefixes when a less specific prefix is not available. This presents a difficult decision to censors unwilling to shut off partial Internet access to customer traffic. While

censors are inhibited by poisoning in some cases, a censor willing to lose reachability to some parts of the Internet *and* aware of decoys not impacted by default routes can still benefit from BGP poisoning. Finally, our discovery that the Internet has significantly changed from 2009 to 2018 with respect to reachability should come with no surprises. Therefore, systems designed in an era of a vastly different Internet topology may need to be re-evaluated again on the live Internet to see whether their claims are still valid.

4.8 Discussion

4.8.1 Reproducibility and Continuous Measurements

All experiments conducted for this set of work can be replicated using the same public infrastructure we utilized. Distributed traceroutes can be ran using RIPE Atlas [RIPE NCC], other sources such as NLNOG Ring [120, 58] and PlanetLab [32, 134] can also be used. BGP measurements can be conducted by partnering with the PEERING testbed [121], which is available for use by operators and academic researchers via an application process. The experiments in this work were conducted in the first half of 2018, but the measurement infrastructure and framework is open source [4] and can be deployed to conducted continuous measurements from the same or similar vantage points.

4.8.2 Experimental Limitations

Our experiments have several limitations. First, some ASes will filter poisoned advertisements, and it can be difficult to fully understand what is driving the behavior without having insider information from the ISP's policies. Second, when we see poisoned advertisements not pass through certain ASes, we are inherently unable to know whether the lack of propagation is due to filtering or if the router is invisible from our perspective on the control plane. We cannot distinguish a case where both our provider and their provider is filtering poisoned routes vs. our provider filtering and then our infrastructure not seeing the new route. This results in our measurements of filtering being an *upper bound* on the amount

[4] https://github.com/VolSec/active-bgp-measurement

of filtering occurring. Fortunately, continuous measurements would help address several of these limitations.

From a geographic perspective, we did not attempt to measure the differences in our re-routing feasibility experiments from Section 4.5 geographically. Recent work has found geolocation of routers and AS paths from public and commercial databases to be unreliable [49]; therefore, we focus on the topological differences in poisoning feasibility. Finally, the instability of RIPE Atlas as our distributed traceroute source can be limiting. Traceroute probes from RIPE Atlas suffer from infrequent instability due to the small profile of the probes and the load on the entire measurement network. To adjust for this behavior, we only use Atlas probes that remain stable for the entire experiment. We define stability here as Atlas probes that continue to respond to API requests and return successful responses when requesting AS-level network paths.

4.8.3 Strategy for Going Beyond Simulations

While we re-evaluate the Nyx [130] system from simulation actively on the live Internet earlier in Section 4.5.2, other security systems affected by poisoning such as Waterfall [98], RAD [122], and Feasible Re-Routing [149] can also be re-evaluated on the live Internet. While our work does not explicitly re-evaluate these systems, one can use the same model followed for Nyx. In particular, we define a general strategy for validating the effectiveness of security systems affected by poisoning. This strategy leverages active measurements *instead* of passive measurements or simulated evaluations.

1. First, determine the set of implicit or explicit assumptions and requirements for the system to work effectively in the presence of BGP poisoning.

2. Based on these requirements, design experiments leveraging the key components of our infrastructure: poisoning AS(es), AS(es) to poison/avoid, AS(es) to steer around the poisoned/avoided AS(es).

3. Conduct these experiments on available sources using our open-source platform. Publicly accessible sources include PEERING, RIPE ATLAS, and BGPStream. Private sources include the evaluator's own AS (e.g. in our work, our University) or partner ISPs. While PEERING, RIPE ATLAS, and BGPStream are publicly available and free, other sources are publicly available but are non-free, such as Amazon Web Services (AWS). These sources could provide distributions needed for certain evaluations (e.g. sources of censored content in the cloud).

4. Reference our analysis framework for Nyx to analyze the poisoning results for other security systems, including examining both the graph-theoretic implications on the Internet topology as well as the specific impacts to the system's effectiveness.

Based on these requirements, an evaluation of Routing Around Decoys [122] using poisoning to avoid Waterfall-like [98] adversaries may look like the following:

1. Consider this assumption: *BGP poisoning can be used by a censoring AS at the edge of the censoring region to route victim traffic around decoy routers.*

2. To evaluate this with active measurements, an experiment should measure the ability to steer victim traffic in a censoring region around sample decoy ASes. The poisoning AS

should be censor-controlled BGP routers, the poisoned ASes should be decoy routers, and the steered ASes would be Internet providers within the censored region where users are trying to navigate around the censoring using Waterfall [98].

3. Recent work from Levin [74] has shown that sources of traffic within censoring regions can be controlled from the outside of the region. This experiment could use these sources, or ISPs in the region as the steered ASes. Routers near the edge of the censoring region could be used as the poisoning ASes, similar to our use of the University AS. Finally, poisoned ASes should be the decoy routers found from active measurements from within the censored region.

4. The analysis for this experiment would reveal the percentage of cases censoring routers or near-censoring routers can route around decoys. This would demonstrate the validity of Routing Around Decoys [122] in practice if successful despite Waterfall [98] or game-theoretic [97] defenses; otherwise, it would highlight the futility of RAD in practice.

4.8.4 Recommendations for Re-Examining Other Security Work

There are many examples of other security systems from recent literature, while not focused on poisoning specifically, do make assumptions about BGP behavior and for the same policies as the inferred topology of the Internet to hold in practice. Notably, NetHide, which obfuscates traceroutes across the Internet [89], *tests across less than 1% of Internet ASes of a total 60,000+*, but instead only $\tilde{1}50$. Recent studies of defeating BGP hijacking of Bitcoin, including SABRE by Apostolaki *et al.* [16, 15], only test with the inferred CAIDA topology, which we show in this paper *does not match reality when it comes to poisoning.*

Less recent work such as RAPTOR [141], a look at routing attacks on Tor, claims that 24 prefixes are the only prefixes that cannot be defeated by a Tor adversary using RAPTOR, yet we showed earlier that large swaths of the Internet will respect a 25, allowing Tor to be de-anonymized in some cases. Finally, a study recently published at IEEE S&P 2020 by Tran *et al.* [148] focuses on advanced Bitcoin re-routing attacks. However, Tran makes direct claims that actively measuring the BGP policies they require for their attack *is difficult* and do not do so, though we find in this work that it is in fact possible to actively measure BGP policies.

We recommend all of these systems, like our re-evaluation of Nyx, to be reproduced on the live Internet before being deployed to protect actual users. Offensive tools should also be tested ethically before being integrated into the threat models of network practitioners. Much like a modern military would not conduct live fire testing of weapons, nor should academic researchers targeting the live Internet not attempt to ethically measure their tool to show whether the attack would stand up to real-world execution. Some systems from past and recent literature test their hypotheses relying on Internet routing behavior beyond simulation. An example offensive tool which tests using real-world BGP advertisements is SICO, which can launch interception attacks with BGP communities. SICO leverages PEERING [23], like this paper. Blink, again by Apostolaki *et al.*, which establishes fast connectivity using the data-plane [57], was tested on the live Internet. SCION [163] and Named Data Networking [162], both proposed "future Internet architectures", are actively deployed on the live Internet.

4.9 Related Measurements

Here, we cover related Internet measurement research. LIFEGUARD from Katz-Bassett *et al.* [69, 70] and Anwar *et al.*'s Interdomain Policy exploration [13] use algorithm similar to our return path steering methodology. They addressed steering return traffic around link failures between Amazon EC2 servers distributed among data centers. However, our algorithm explores greater depths in its breadth-first search of all possible paths from a single remote AS, rather than aggregating paths available from many poisoning ASes. While not directly related to our steering algorithm, work on BGP communities can influence inbound paths similarly to poisoning. Communities in the wild have been studied by Streibelt *et al.* [139].

Chapter 5

Applying Practical Constraints on Re-Routing to Nyx

5.1 Introduction

The previous chapter explored the feasibility of BGP poisoning on the live Internet, in particular the presence of ASes which filter poisoned advertisements, leading to a number of practical constraints on propagating poisoned paths. That study's findings hold key insights for evaluating Nyx's performance in simulation with *real-world routing behavior*. Specifically, these insights affect the simulator's limits on the maximum path length of ASes, the number of times a deployer can re-route a critical AS onto distinct alternative paths, and the effects of ASes that filter poisoned advertisements employed by Nyx.

Thus, this chapter examines the impact of each of these constraints on the Nyx system by adapting the Chaos BGP simulator described earlier in Chapter 3. We first cover the impact on routing success, then disturbance mitigation and path lining, and finally we cover the impact on performance success. We find that Nyx performs slightly worse under these practical routing constraints, including less success finding uncongested paths due to having fewer ASes available to poison when re-routing. Despite weaker routing success in some cases, we find that the presence of ASes that filter poisoned announcements has little effect on the Nyx system.

In the rest of this chapter, we first describe the adaptations to Nyx's system design in order to handle practical routing constraints. We then explore the impact of these constraints on routing success, disturbance, and performance success of Nyx.

5.2 Adapting Nyx System Design

5.2.1 Practically Reducing Disturbance

In the original configuration of Nyx from Section 3.2, path lining requires poisoning all neighbors of the alternative path - potentially thousands of ASes. Tran *et al.* [149] measure how many poisons that re-routing systems like Nyx would employ, finding that up 9,000 ASes are poisoned on average. By measuring the extent of filtering of long paths in practice, we found earlier in Chapter 4 that paths of length greater than 255 are filtered and dropped on the Internet. We also found that 99% of the Internet's ASes will forward paths of lengths of less than 255. To address their finding, we develop a *greedy* path lining algorithm extended from Tran's greedy path lining that limits the number of poisons used, given that 9,000 ASes must be reduced to 255. We account for up to 10 hops of the path outside of poisons, thus we must reduce the path to 245 poisoned ASes plus 10 normal ASes. We discuss later in Section 5.3 how this greedy algorithm affects disturbance caused by BGP poisoning.

Our greedy algorithm builds the *ideal* path line set, PL_{ideal} from all neighbors of the alternative path. We build the path line set by first choosing a poison limit, n. Then, the algorithm chooses the top n poisons from a ranking of the PL_{ideal} sorted by the relative size of the customer cone of each $AS_i \in PL_{ideal}$. n is chosen to be 245, the maximum number of poisoned ASes found able to be propagated from our BGP poisoning feasibility study in Chapter 4 on the live Internet. In our study, we found that any path with a maximum length of 245 or less propagates to 99% of ASes on the Internet when factoring in the advertising ASes' customer cones. Thus, when we evaluate the disturbance caused by Nyx with a limit on

path lining in Section 5.3, we choose 245 as the limit to respect practical filtering conditions as a result of BGP implementation details from ISP to ISP.

5.2.2 Practically Finding Performant Paths

By default, Nyx imposes a limit of 10 iterations when searching for uncongested paths. While we show this search limit is never reached earlier in Section 3.3, we also evaluate Nyx under the limits found by our work in measuring practical routing constraints on the live Internet in Chapter 4. We found that a poisoning AS can discover 2.5 unique paths on average across 1,000+ cases of re-routing on the live Internet. To that end, we adjust our simulations with limits of 3, 5, and 7 search iterations. We describe our results with these experiments in the next section. Since we found the average number of unique discoverable paths was roughly 3 over 1,800 active measurements, we choose 3 as the minimum limit. The core search algorithm does not change, and if Nyx cannot find a viable uncongested path, Nyx halts and chooses the best path discovered up to that point.

5.2.3 Handling Filtering in Practice

Nyx is also impacted by the presence of AS-level filtering discovered in Chapter 4. We evaluate Nyx when certain fractions of ASes filter poisoned advertisements. With this limitation, we expected to find weaker success in maneuvering the deployer's inbound traffic, yet we do not actually find this to be the case. Given this positive results, we do not include any new mitigations or changes to Nyx's design to improve performance when filtering does occur.

5.3 Evaluation

5.3.1 How Do Routing Constraints Impact Routing Success?

Impact of Maximum Path Length

The average path length on the Internet in October 2019 was 4.27 [14], while the maximum observed length was 13. When conducting BGP poisoning for Nyx, the deploying AS must prepend ASes to cause critical traffic to route around congestion. In the same October timeframe, the maximum observed prepended length was 46. However, Nyx often uses thousands of prepended ASes to mitigate disturbance via its path lining algorithm. We set out to study how Nyx is impacted when you limit the amount of poisons a single path can use. In our previous study and Kang *et al.*'s study [149], the maximum path length achievable on the Internet was roughly 245.

We now evaluate Nyx's ability to route onto alternative paths by limiting the maximum number of poisoned ASes to 245, since our study found at 245 ASes in length, 99% of the Internet still observed the path. Intuitively, routing success in Nyx should *not* be affected by the maximum path length restriction, since Nyx always starts its process by poisoning a *single* AS on the transit-link DDoS-impacted link. Unsurprisingly, we see in Figure 5.1 that for transit-link DDoS, and traditional DDoS, Nyx *is not* affected when attempting to route onto an alternative path. These results mirror the same graph without a poison limit, Figure 3.6a.

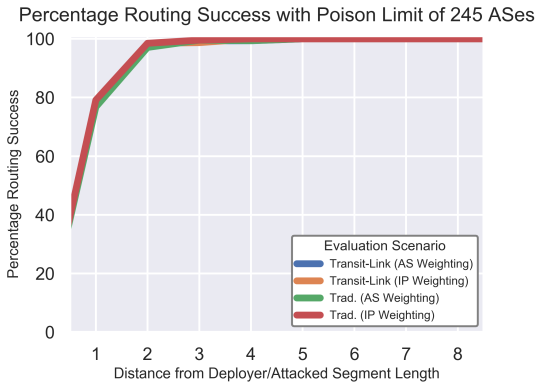


Figure 5.1: Routing Success when maximum path length allowed to propagate is 245 as found in practice

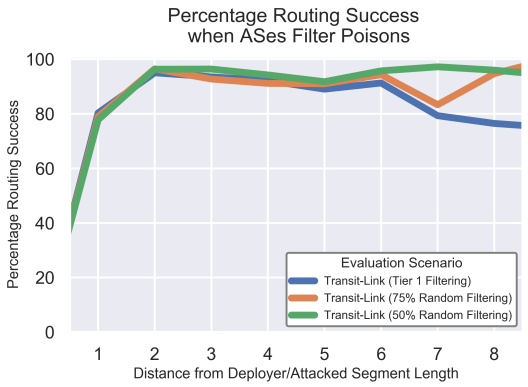


Figure 5.2: Routing Success when Real-World ASes filter poisoned paths

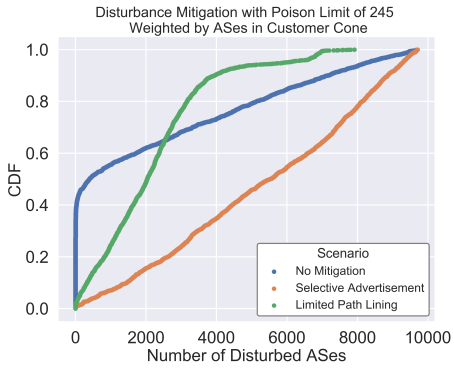


Figure 5.3: Disturbed ASes (Disturbance Mitigation) when greedy path lining by weighted AS customer cone under a limited poisoned path length of 245

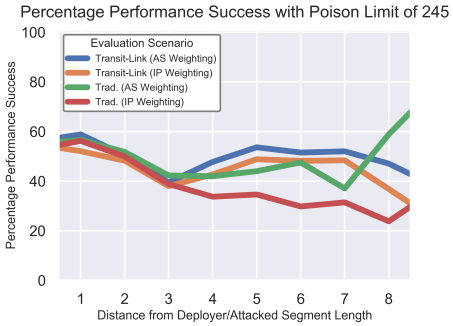


Figure 5.4: Strong Performance Success when path lining is restricted to 245 poisons (found in practice to be the maximum possible)

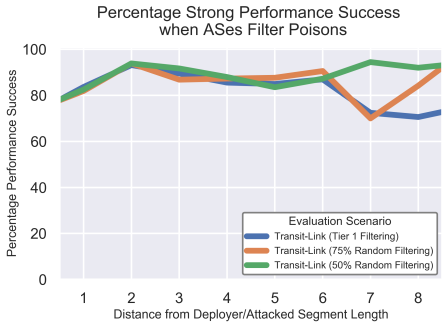


Figure 5.5: Strong Performance Success when Real-World ASes filter poisoned paths

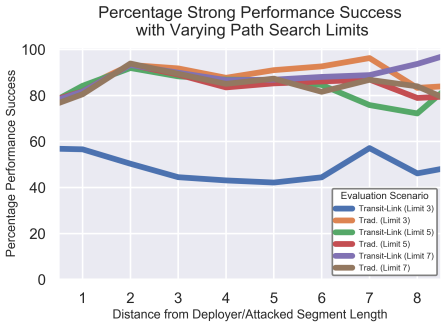


Figure 5.6: Strong Performance Success when greedy search heuristic is limited to real-world findings

Impact of ASes Filtering Poisoned Paths

Similar to AS path constraints on length, some ASes employ route filtering policies that drop poisoned paths. We found in our measurement study that larger ASes filter poisoned paths more often, while smaller ASes, based on customer cone size, filter less. While BGP poisoning is a compliant BGP behavior, some ASes seek to mitigate weird routing behavior and will filter paths needed by Nyx. To that end, in Figure 5.2, we highlight the effects of randomly sampled and Tier 1 (the largest) ASes filtering poisoned announcements like those used by Nyx.

We randomly sample 75% and 50% of all ASes in the core topology to filter out poisoned paths. These ASes will not respect these poisoned announcements, essentially causing them to stop propagation of poisoned announcements through themselves. For example, if the target link does not respect poisoning, then we cannot re-route critical ASes around it. Figure 5.2 reveals that despite over 75% of the ASes on the Internet filtering poisoned paths, when an attack is 2 to 6 hops away from the deployer, Nyx can still route around an transit-link DDoS attack **95% of the time**. Past 6 hops, the success drops to 80%. When all Tier 1 ASes filter, the success past 6 hops from the deployer drops to under 80% as well. Even when all Tier 1 ASes, which provide transit for the majority of the Internet, do not drop poisoned paths, Nyx can find a way around the core of the topology. Randomly sampled results reveal that even when 75% of randomly chosen ASes on the Internet drop the paths, the same is true. We revealed earlier that Tier 1 ASes do in fact filter the most, followed by Large ISPs, then Small ISPs, then stubs. However, after Large ISPs, almost all other ASes

filter poisons less than 20% of the time compared to upwards of 60% of the time for Tier 1 ASes.

5.3.2 How Do Routing Constraints Impact Disturbance from Nyx?

Impact of Maximum Path Length

While the routing success is not affected, disturbance mitigation (via path lining) is significantly affected by the maximum path length constraint. Disturbance mitigation refers to the path lining mechanic used by Nyx, described in Section 3.2, to isolate the poisoned announcements from spreading throughout the rest of the Internet. If isolation is not done, then malicious traffic can be re-routed onto the alternative path along with benign traffic. In unrestricted path lining, Nyx poisons up to 9,000 ASes, but here we limit it to 245 minus the AS path minus the poisoned target link.

Using the greedy path lining algorithm described earlier and modified based on Tran *et al.*'s [149] work using the same simulator, we show the new quantity of disturbed ASes in Figure 5.3. Notably, the number of disturbed ASes rises from 9 ASes on average to 3,000 80% of the time. Here we weight the ASes to optimally path line by the number of ASes in their customer cone. We also evaluated the disturbed ASes with a greedy path line weighted by the IPs in the customer cone of candidate path lined ASes, but found the same results. While Nyx is not finished, it does substantially impact the performance success of the system in practice as we will show in the next section. But first, we point out that future work should seek to optimize the greedy path lining mechanic even further. The algorithm could potentially choose path lined ASes by identifying which ASes are relevant based on

their customer, provider, and peering relationships to the ASes along the alternative path, which would free up further poisons for ASes with large customer cones vulnerable to high disturbance.

5.3.3 How Do Routing Constraints Impact Performance Success?

Impact of Maximum Path Length

The impact of a poison limit from routing behavior affects the performance success, or the ability to route onto uncongested paths. Figure 5.4 highlights the impact, revealing that for both transit-link DDoS and traditional DDoS, the success of the system with the simple greedy path lining algorithm presented earlier drops by roughly 25-35% on average. The original results for this evaluation show roughly 90% success for defeating transit-link DDoSs, while the poison limited system is successful against 55% of simulated attacks.

Recall that the greedy path lining algorithm adapted from Tran *et al.*'s work is still naive, and a more feasible approach should be developed in future work. For now, deploying ASes should evaluate their positioning in the topology on a case-by-case basis, using the simulator in this work, to see if their topological position in the control plane affects potential disturbance. For example, if a deploying ASes is relatively few hops from their critical ASes, and the path diversity in the region crosses few large ASes, then Nyx would be able to path line the largest ASes along their short paths, and enable Nyx to keep disturbance to a minimum. In that particular case, the performance success will likely mirror unrestricted path lining.

We add the caveat that Figure 5.4 shows the **strong** performance success at the *hardest* settings of congestion factor and bandwidth tolerance. Meaning, even when re-routing the benign traffic and potentially bot traffic from 3,000 additional ASes under practical routing limits, Nyx still finds a totally uncongested path for a single critical AS 55% of the time. The congested path in this case is 5 times oversubscribed, which for the core of the topology, could be hundreds of Gbps over the typical capacity. Furthermore, we limit the bandwidth tolerance to 1.1, which sets the available capacity of the deployer on the smaller side of the inference, a further disadvantage to the deployer. For weak performance success with a more favorable bandwidth tolerance and lesser congestion factor, Nyx will perform more favorably in real-world deployment even with significant disturbance.

Impact of ASes Filtering Poisoned Paths

When ASes filter poisoned announcements, the performance success drops no lower than 10% than the case without filtering ASes. Shown by Figure 5.5, Nyx deployers can effectively find totally uncongested paths while under attack from transit-link DDoS/transit-link DDoS over 95% of the time on average. These results match with the impact of filtering on routing success discussed earlier and presented in Figure 5.2. Despite filtering by 50% and 75% of all ASes, and by Tier 1 ASes, Nyx can maintain the same performance under practical routing conditions.

Impact of New Path Discovery Limits

We end our discussion of exploring the impact of practical routing constraints by focusing on the impact of discoverable alternative paths on the performance success of Nyx. In our

measurement study, we found ASes on average can deploy Nyx policies and re-route onto 2.5 average completely unique paths. In some cases, they found up to 10 paths could be discovered. The search algorithm used by Nyx will attempt to find new alternative paths until the performance goal is satisfied or a limit of 10 is reached. Given that the in practice constraint lies closer to 3 on average, we evaluate Nyx and reveal in Figure 5.6 the strong performance success of Nyx when search limits of 3, 5, and 7 are placed onto the system.

We find that Nyx can successfully route a critical AS onto a totally uncongested path over 80% of the time for transit-link DDoS DDoS, while the worst case of roughly 50% success comes when the deployer AS is targeted directly and a search limit of 3 available paths is chosen, matching the average realistic case from Chapter 4. This finding corroborates the analysis presented earlier in Section 3.3 in Figure 3.12, where we show for the original, unrestricted Nyx system that Nyx does not search for paths more than 1.5 times on average, except under traditional DDoS when the attack is centered at more than 5 hops away and the traffic is directed at the deployer. Again, these two figures, 3.12 and 5.6 align, revealing that ultimately Nyx does not suffer great performance hits from a realistic discoverable path limit on the searching algorithm presented in the system design section, Section 3.2.

Chapter 6

Leveraging Nyx to Mitigate DDoS Against U.S. Critical Infrastructure

6.1 Introduction

A combination of consumer growth and widespread industry initiatives have led to the modern energy grid becoming "smart" [45, 43, 55]. With this new intelligence, the world's energy infrastructure has inherently become more connected, both among components and to the Internet [86, 31, 55]. With this connectedness, operators can balance power production with demand, collect increasingly advanced metrics, and efficiently share produced power among utilities. Yet, this connectedness comes at the cost of heightened vulnerability to traditional cyber security threats that have yet to be mitigated.

Among those threats, Distributed Denial of Service (DDoS) is one of the most potent *and viable* for both amateur and advanced adversaries [68, 67, 3]. Botnets, which enable DDoS, are easily able to rent from anyone with an Internet connection [96], and the rise of Internet of Things (IoT) devices has led to botnets becoming further distributed [143, 12]. The botnets of today have the ability to target nearly any critical link on the modern Internet, especially when combined with novel advances in DDoS targeting mechanisms [87, 65, 140].

The combination of an Internet-connected critical energy infrastructure with the ability for anyone to execute multi-terabit DDoS potentially opens the door to scenarios not envisioned by the original creators of the smart grid: blackouts, brownouts, and other large-scale power failure scenarios induced by a class of attacks executable from anything with a web browser and model of where to direct the attack. While the only publicly-known attacks on power infrastructure have come from nation states [76, 34, 44] or other sophisticated groups, an attack executable on the grid by DDoS could be done at extremely low cost and

effort. These prior attacks not using DDoS have either required custom-built malware or credentials to operator systems, both of which come at a high cost.

Only recently has an attack against power generation been shown to combine modern botnets and an attack model with specific targeting information. MadIoT [132] (Manipulation of demand via IoT) leverages IoT device botnets, much like the botnets that can execute DDoS, to increase power consumption of devices in order to drive spikes in demand at generation facilities of power utilities. This can lead to widespread failure of the overall grid. Unfortunately, as we will describe in this chapter, MadIoT attacks via botnet control is not the only way to leverage botnets as a blackout-inducing tool.

In this third and final thrust, we characterize the vulnerability of U.S. power generation facilities to DDoS, including recent classes of DDoS known as Link Flooding Attacks (LFAs) covered earlier in Section 2.2.2. LFAs target Internet links outside the control of the network affected by the DDoS. If an electric utility is targeted by an LFA, they will have no chance to filter or drop the malicious attack traffic. We utilize Maestro [87] to increase the amount of bots that an adversary can target a utility with using BGP poisoning[1]. To understand what utilities are vulnerable, we first construct a novel utility-to-Internet-connectivity model that maps utilities in the U.S. to the Internet links they rely on. We build this model by combining public data on U.S. power plants with Internet scan data on exposed Supervisory Control and Data Acquisition (SCADA) devices used in generation facilities based on prior studies. With our model, we analyze the vulnerability of utilities to DDoS attacks, including LFAs. Through various methods, we find thousands of utilities with exposed devices used for control of grid operations, and filter this down after factoring in non-unique SCADA device

[1]Maestro was co-developed by the author after Nyx was developed, but led by a collaborator

ISPs to a total of 584 US-based utilities in our model. These utilities cover 35 states and 202 unique zip codes with a total population of 5,216,654 residents based on the 2010 U.S. census.

While DDoS on its own would negatively impact electric utilities, we specifically explore DDoS and its impact on the stability of power systems by cutting off SCADA communications used by Automatic Generation Control (AGC). AGC uses SCADA communications over Modbus and DNP3 between geographically separated generation sites (i.e. power plants) to regulate the generation of power in response to consumer demand. However, for utilities that do not have completely disconnected smart grids, AGC is conducted over public Internet links, thus vulnerable to DDoS. To connect the vulnerable and exposed utilities from our model and DDoS simulations, we use real-world Matlab Simulink AGC models to model power grid stability when AGC is cut off due to SCADA communication loss between remote generation sites. When consumer demand drops or increases, or a cyber-physical attack destroys a facility or power line, AGC, if disabled by DDoS, cannot bring the power system back to stable state due to the ongoing DDoS of the utilities Internet providers. In other words, a motivated adversary with control of a DDoS service and either the ability to start the DDoS when consumer demand spikes, or via a cyber-physical attack like MadIOT, can drive the stable state of generation control sites into dangerous territory. We find that when combining LFA-based DDoS with Maestro, an adversary can execute full-scale LFA DDoS against exposed utilities in our model with over 22 Petabits/s of DDoS with only 23% of the Mirai botnet flowing over the vulnerable links in our model. Without Maestro, adversaries can control up to 12% of Mirai and induce DDoS attacks of sizes in the hundreds of Terabits/s. When these bots cause packet loss for the utilities, AGC may cease to be effective, and our

models show that the adversary can quickly drive the stability of the targeted power grid into regions of danger based on North American Electric Reliability (NERC) standards [100] for extended periods of time. Notably, we find that a particular utility responsible for serving Washington D.C. can be targeted with our attacks with up to 85% of the Mirai botnet, or over 2.38m bots.

Finally, we explore how well control-plane defenses such as Nyx can alleviate the congestion caused by the DDoS shown to viable by our simulations. Since utilities do not run at high-profit margins due to government regulations (or they are publicly-owned non-profits or not-for-profit) [52], subscribing to expensive DDoS defense services or CDN-based solutions could be too costly as a mitigation strategy. Furthermore, CDN-based solutions do not fundamentally alleviate LFA-based DDoS, as shown in Chapter 3, instead turning the problem into a war of bandwidth. By relying on a system such as Nyx, utilities with power generation sites to protect can employ BGP poisoning to re-route their critical SCADA traffic around the congestion impairing the use of AGC to stabilize the grid under sudden load increase/decrease events. In all of our analyses, we show our findings in the context of the original Internet-connected utility model. With Nyx, our simulations reveal utilities can route around congestion on average 80% of the time and find totally uncongested paths nearly 60% of the time. We find that 11 states with utilities in our model can wield Nyx to completely mitigate DDoS against them that would have otherwise brought down AGC communications.

Note on Ethics:

In our examination of the vulnerability and exposure of power generation facilities to DDoS, we neither actively probed power grid devices nor sent live attack traffic towards them. We

rely on regular scans of devices by trusted online services and we model DDoS entirely in simulation using real-world constraints. We discuss our the ethical considerations of our experiments further in Section 6.4.3.

Our Contributions

- We built first-of-their-kind quantitative models of the Internet connectivity of U.S. utilities with power generation sites. See Section 6.3.
- We explore the vulnerability of the utilities in our model to DDoS, LFAs, and Maestro. See Section 6.5.
- We examine the impact of loss of SCADA traffic connectivity (from DDoS) to Automatic Generation Control (AGC), an industry-standard technique to regulate power generation in response to consumer demand. See Section 6.6.
- We examine the ability for Nyx to defend utility networks with BGP routers against DDoS. See Section 6.7.
- We conclude this thrust with a discussion in Section 6.8 about limitations of our presented attack and other defenses against this attack.

6.2 Background on Power Systems

In this section, we provide background on power systems, which is critical to understanding how DDoS can impact parts of large power systems. For background information on DDoS, please refer back to Section 2.2.

6.2.1 Power Generation and Frequency Control

The stability of the power grid depends on operators maintaining a continuous balance between the supply of power and existing demand for power from consumers. To keep the balance of supply and demand, operators use a variety of mechanisms to both predict ahead-of-the-day demand for power as well as real-time adaptation of power supply to meet demand. Operators use weather data, historical usage, current events, and readings from sensors in the power grid to regulate the amount of generators providing adequate power supply without overloading any power lines [133].

In the field of power systems, the rotation speed of synchronous generators in the grid corresponds to the *frequency* of the overall system. Synchronous turbine generators follow Isaac Newton's first law of motion. Their behavior is calculated by the inertia of the generators. When demand for power rises higher than the supply, the generator's rotor's rotation speeds decelerate. This leads to the kinetic energy in the rotors being released into the power system (i.e. lines, towards consumers, etc.) as a response to the increased demand. The release of energy then causes a drop in the frequency of the system. If the available supply is higher than the demand for power by consumers, this results the generators' rotors accelerating and leads to an increase in the frequency.

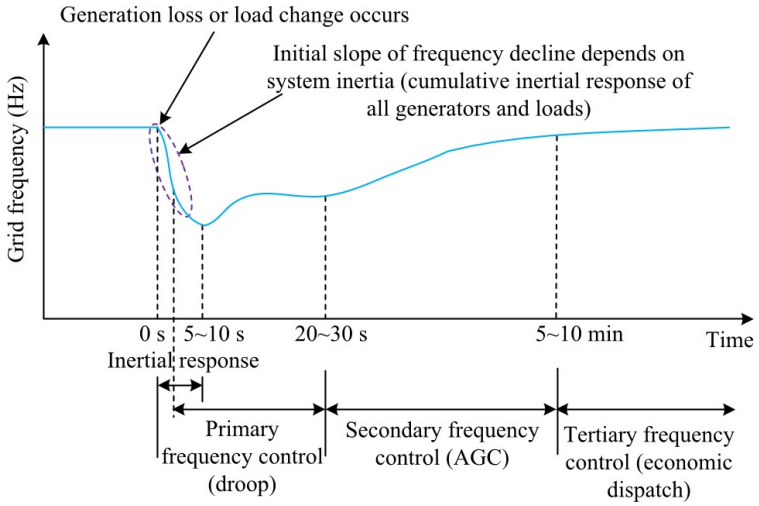


Figure 6.1: Grid Frequency in Response to Change in Load [38]

The decrease or increase in the system frequency in response to load changes cannot be sustained for a significant amount of time. If the frequency is lower than the normal values defined by NERC in the U.S. [100], the instability can damage the generators inside power plants. When the settling frequency changes drastically, relays (i.e. a switch) can be configured to cut off the generators from the power system. This directly leads to power loss to downstream consumers, but is necessary to keep the generators from breaking and enduring long-term damage. When demand spikes, within several seconds of the initial frequency decrease, the primary controller at the generation sites turn on and increase the rotor speed in the generator. This then increases the generator's output power and frequency of the system [42]. The inertia of the system determines the rate at which the system frequency returns to normal. Systems in practice with more generators have more inertia and are more resilient to sudden increases or decreases in demand or supply (i.e. consumers needing less power or loss in power being produced, respectively).

Table 6.1: NERC System Reliability Limits for Frequency of Eastern and Western U.S. Interconnects (units in Hz)

	Eastern Interconnect		**Western Interconnect**	
	Low	*High*	*Low*	*High*
Initial Trigger (FTL)	59.950	60.050	59.856	60.144
Emergency Trigger (FAL)	59.908	60.092	59.80	60.20

The governor response (also known as droop) [82] is the rate of increase in the generation of power when the primary controllers are activated. By leveraging the governor response, operators allow the system to automatically increase its generator output at a distinct rate until the overall generation of power is equal to the demand for power before the load increase, as long as none of the generators reach their demand limit. If this does occur, then the system may not stabilize. Each power system only has so much reserved capacity to increase supply of power in response to sudden demand increases.

However, even after the initial droop or governor response, the overall system may not settle back at a normal frequency. This led to the introduction of a *secondary controller*, also referred to as Automatic Generation Control (AGC). The secondary controller or AGC will start within seconds to minutes to restore the system's frequency back to the original value. AGC will modify the amount of power being generated by deploying available extra generation capacity via new generators and can regulate the demand between different, physically-separated generation sites (i.e. power plants) in the overall grid. AGC can use common Industrial Control System (ICS) protocols, including Modbus or DNP3, for transport of control data between remote sites. This communication relies on the Internet and ISPs when the utility is not large enough to run their own telecommunications network.

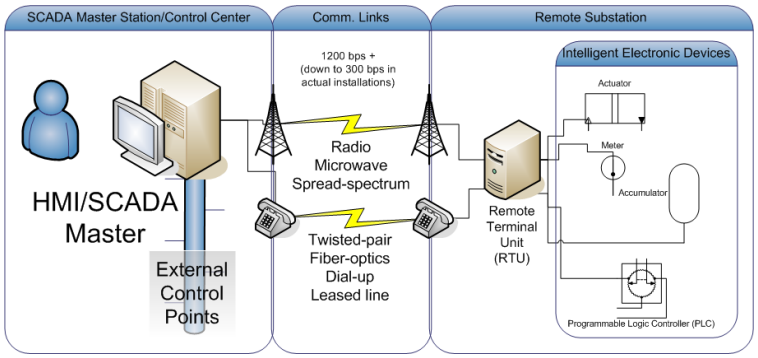


Figure 6.2: Overview of DNP3 Communication Use Case

If not disrupted, AGC can bring the system back to the normal range specified as safe by NERC. Figure 6.1 shows how the grid responds during a sudden increase in load or decrease in generation with the governor (primary) and AGC (secondary) responses.

In the U.S., the nominal (i.e safe) operating frequency of generators in a power system is 60 Hz. Frequency Trigger Limits (FTLs) are defined as thresholds to alarm when frequency falls out of the safe range, which keeps power on essentially. An FTL-low alarm comes on if the frequency is too low for 5 or more minutes, and the same is true for the FTL-high alarm. According to NERC, the FTL-low is 59.950 Hz and 59.856 Hz for the eastern and western interconnects, respectively. According to NERC, the overall system reliability deteriorates quickly after the this limit is reached and not corrected. The FAL (Frequency Emergency Trigger Limits) indicate limits at which the system becomes extremely unstable. The other FTL alarms and FAL alarms are shown in Table 6.1.

6.2.2 ICS Communications

Utilities rely on several networking protocols to control and monitor industrial control systems. Among the most common protocols in electric power utilities are Modbus [94, 142]

and DNP3 [37, 2]. Modbus was originally created in 1979 by Schneider Electric to control programmable logic controllers (PLCs). Since then, Modbus has evolved to allow communication over the Internet via either serial, Ethernet, or wireless communication. Modbus is used to connect power plants and devices within them to each other as well as to remote sites. Modbus is a simple protocol compared to other application layer protocols like HTTP, so to address the lack of features it was created with, the utility industry created DNP3 to fill the need for distributed communications between SCADA devices in critical infrastructure. DNP3 is especially used widely in electric and water utilities, and an example of how it may be used by operators or automated systems to manipulate meters, generation facilities, or other devices over the Internet or radio connections can be seen in Figure 6.2.

Both DNP3 and Modbus, along with many other SCADA protocols, were designed for use in serial-only, disconnected systems. Yet with the rise of the Internet, utilities connect these devices to the networks relying on ISPs for Internet access via extensions to the protocol allowing the use of Ethernet and TCP/IP. These SCADA protocols, like any other networking protocol using TCP/IP, use host ports to communicate and setup connections. In recent work, a multitude of SCADA devices have been found to be exposed on the Internet by looking for the ports which run the protocols [92]. Mirian *et al.* also find that Modbus and DNP3 are primarily used for electric power utilities [92], including those that include generation facilities running AGC (see prior section). In practice, DNP3 runs on port 20000 and Modbus on port 502, according to their standards. The existence of these devices indicates they are in use in real utilities, in a testbed for testing-only purposes, or are honeypot devices.

In this work, we rely heavily on Internet-wide IPv4 scans to map SCADA devices, specifically Modbus and DNP3, to utilities which own them using public power plant data. We cover this mapping, which forms our utility-to-Internet-connectivity model in the next section.

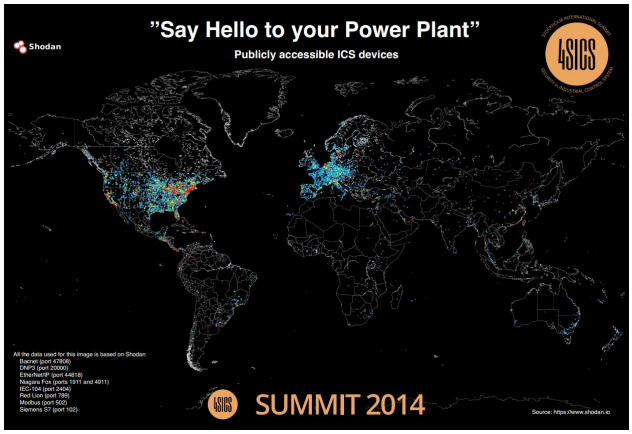


Figure 6.3: 2014 Study by Shodan of Power Plants on the Internet

6.3 Inferring US Power Generation's Reliance on the Public Internet

In this section, we discuss our construction of a model for mapping U.S. power generation sites (i.e. power plants) and the utilities that run them to exposed SCADA devices. These SCADA devices run either Modbus or DNP3, and as discussed in the prior section, can be used for controlling industrial processes at these plants, including DNP3.

6.3.1 Utility Internet Reliance

We described in the prior section how utilities, especially power providers, may rely on the public Internet for transiting SCADA traffic. Unfortunately, there are no comprehensive studies with ground truth on the extent to which utilities rely on the Internet, though many anecdotal examples exist. Peter "Mudge" Zatko, then a security researcher and later a BBN and DARPA employee, found in 1999 that up to 30 utilities whose plant control networks could be accessed remotely [90]. Later, in 2004, Gartner found that IP networks in utilities

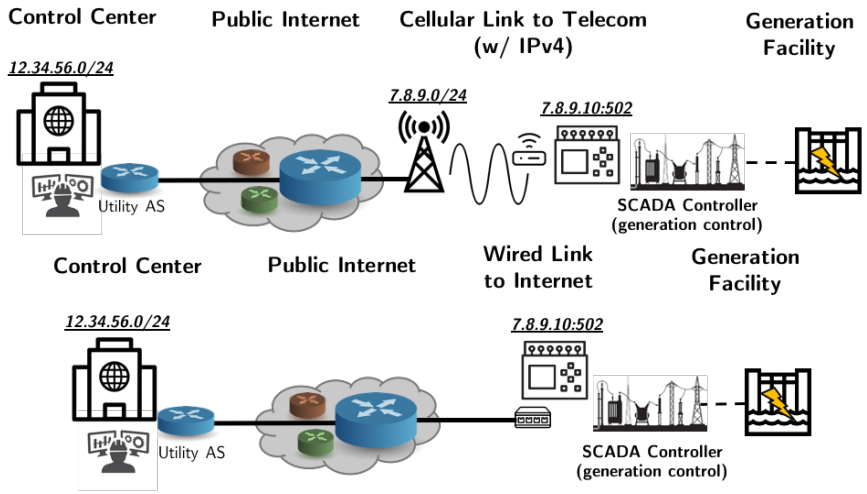


Figure 6.4: Example of Utility Dependence on the Internet via Cellular/Wireless or Direct (Wired) Connection

continued to be deployed and left systems accessible remotely [90]. In 2014, Shodan and firm InfraCritical found XXX plants accessible over the Internet via Internet-wide IPv4 scans of common SCADA protocols [127, 63]. A map of the devices can be seen in Figure 6.3. As recent as 2020, CyberX did a survey of over 1,800 utilities between 2018 and 2019 and found 56% of utilities exposed plant control devices remotely over common protocols like Modbus, DNP3, SSH and RDP [33]. Additionally, they found 27% of the utilities have "direct connections" to the Internet [33].

In figure 6.4, the utility headquarters (HQ) leverages the Internet to speak to downstream control devices in generation facilities. These connections can rely on either wireless/cellular *or* wired connections over fiber or copper cables. While some utilities, like the Southeast U.S.'s Tennessee Valley Authority (TVA) have their private network and act as their own ISP to facilitate communications between the operators in the control center the downstream generation sites [113], many utilities do not have the budget nor manpower to manage their own network. Thus, generation control techniques like AGC, which use SCADA

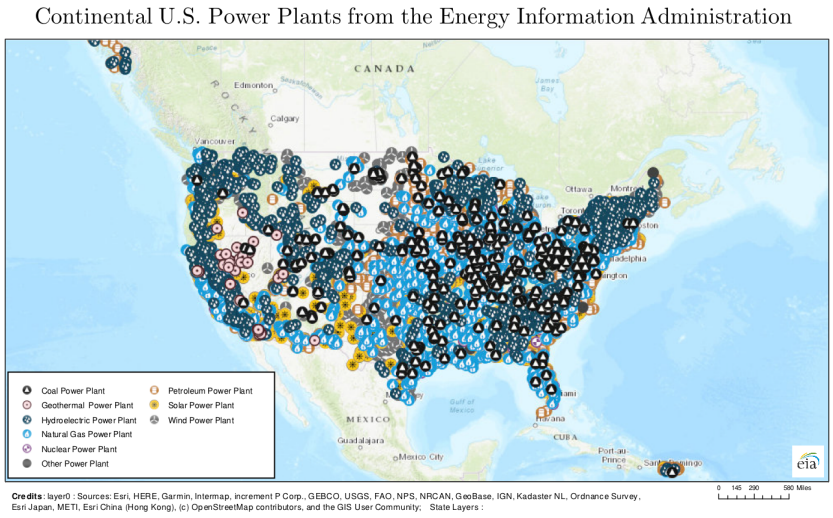


Figure 6.5: Map of U.S. Power Generation Facilities from the U.S. Energy Information Administration (eia.gov)

communications between the control center and generation sites to regulate generation during load events, must traverse the public Internet in order to be effective. In later sections, we reveal how DDoS attacks presented earlier in Chapter 3 can degrade connections between utility HQs and SCADA plant control devices, but first, we need a set of AS-to-AS links which utilities rely on for plant control. To do this, we built a state-of-the-art model to tie U.S. power utilities to potential SCADA devices they control via the Internet.

6.3.2 Model Construction

To build our model, we pull data on generation sites and exposed SCADA devices across the Internet from three distinct sources:

1. Public data on the location of power plants/generation sites in the U.S. and the utility they are associated with, including geographic coordinates for each plant. A map of

the plants we use in our study is shown in Figure 6.5. This data comes from the Energy Information Administration (EIA) [41].

2. Internet-wide IPv4 scan of Modbus (Port 502) and DNP3 (Port 20,000) devices from Shodan [85], including IPs with geographic coordinates obtained via IP geolocation techniques from October 2019. These IPs are also then used to pull out Autonomous System Numbers (ASNs) which indicate the Internet Service Provider providing Internet connectivity for that device.

3. Internet-wide IPv4 scan of Modbus and DNP3 devices from Censys [40], also including geographic information and from October 2019.

To build our model of Internet-connected utilities, we take every SCADA device combined between both Shodan and Censys data and every generation site and we attempt to iteratively pair SCADA devices with likely hosting generation sites using three distinct techniques:

1. Matching City/State: If the SCADA device's coordinates pertain to a unique city/state pair that forms an exact match on a generation site city/state pair, then we tie that SCADA device and that generation site together. We also incorporated fuzzy matches on the city/state pair to account for examples like "New York City" versus "new york city".

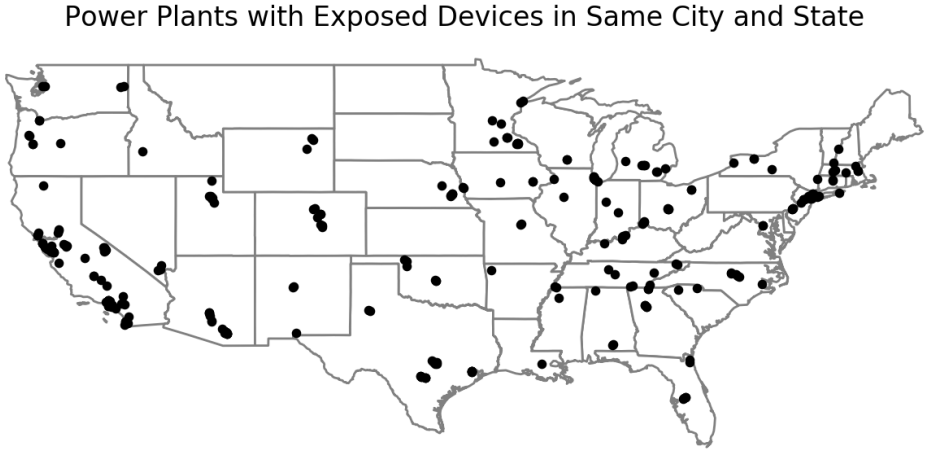
2. Within 40 km: While geolocating IP addresses is a known difficult problem [49, 22], we leverage GeoIP data since it is the best available to map devices to an approximate location. MaxMind, a widely used GeoIP service and one used by Shodan and Censys,

states that "68% of IP locations are accurate within the U.S. city level up to 50 km". Therefore, we pair the closest SCADA device within 40 km of the coordinates of each power plant from the EIA data. By doing this, we inherently leave out any devices above the threshold of 40 km.

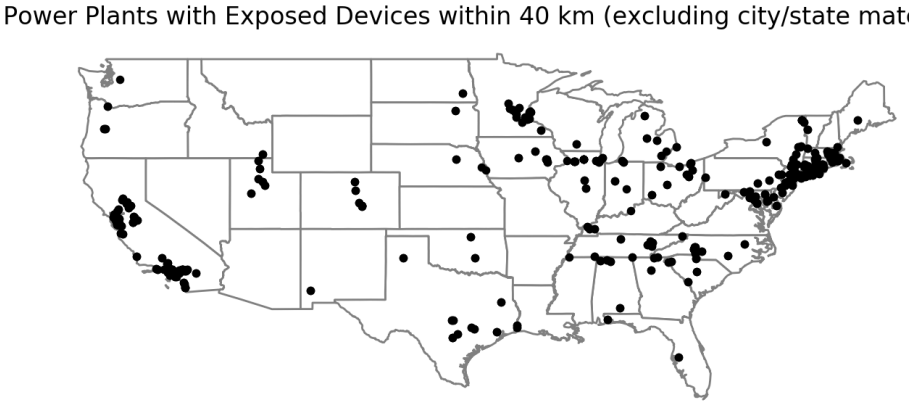
3. In the same manner as the prior technique, we also match on thresholds of 10 km and under as our final technique.

For each dataset, we end up with utility's generation sites that we call "reliant on the public Internet", in the sense that after mapping a SCADA device that is likely theirs to that utility's site, by being able to access the device with Internet scans, it is inherently connected to the Internet. Figure 6.6 shows these three datasets from a geographic perspective. We use the same three figures representing each dataset geographically throughout the rest of the paper. We found a total of 1,420, 2,912, and 405 connected utilities for the city/state, 40 km, and 10 km datasets, respectively for Shodan. For Censys, the numbers were 336, 2,796, and 383 for the same models. This does not yet remove non-unique ASes for the utility side, we do this in the next step.

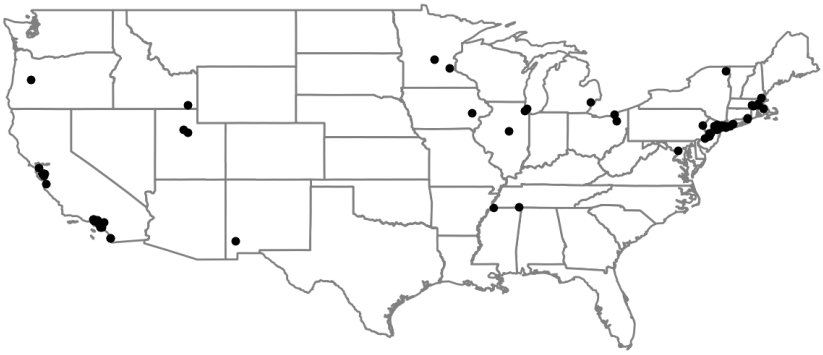
With the SCADA device ISPs mapped to utilities, we must then determine the associated Autonomous System Numbers (ASes) for each utility, where existent. To do this, we take the utility names from the EIA dataset [41] per mapped generation site and we search for either exact or fuzzy matches of each utility name in AS registration data from the U.S. routing registry ARIN [150]. The registration data includes the AS full name, short name, region, and other metadata. We specifically match on the full name field. After filtering out connected utilities from the aforementioned numbers, we end up with a total of 584 unique



(a) Matching City-State Model



(b) Within 40 km Model



(c) Within 10 km Model

Figure 6.6: Exposed devices mapped to corresponding power plants for 3 different models using both Shodan and Censys datasets from October 2019

connected utilities to work with. Because many utilities share the same ISPs for their SCADA devices, the number of identified utilities in the first step is significantly reduced when we go to build the paths on the Internet each utility may rely on. With the ASNs for the utilities and the ASes of the devices which we mapped to them, we can build these paths through the Internet which this utility uses to transit data from their control center to their generation sites. To get these paths, we use the same BGP simulator used by Nyx in Chapter 3. In total, we end up with 433 paths through the Internet which if attacked, would lead to significant degradation of the connected utility's SCADA communications.

We note that this model *does not explicitly* determine whether the mapped utilities actually rely on the Internet; however, the lack of any public model of this and any other relevant study means we must come up with a rough model of utility to SCADA device Internet paths in order to model the impact of DDoS. In the next section, we layout the assumptions made when building this model, and then follow with potential improvements to the model.

6.3.3 Assumptions

In building this model, we make several assumptions:

- We assume that a device being exposed from within a generation site means that utility relies on the Internet. This is not necessarily true, as the device may have been left exposed and not actually control any process within the plant.

- We assume that the utility makes power grid control decisions via a control-center located at their headquarters. We also assume their HQ is operated from the same AS we found when mapping the utilities name to an associated ASN.

6.3.4 Drawbacks to Potential Model Improvements

While our model could be improved, there are drawbacks to these improvements. These include:

- Exposed SCADA devices could be mapped to the city/county zone they reside in (e.g. commercial, industrial, residential). This would ensure exposed SCADA devices in zones that would not be tied to generation sites *are not* considered as possible targets. The issue with zoning codes is not their usefulness, it is the lack of public data on zoning codes by city and region. Large cities like Los Angelas and New York City have public data on zones; however, smaller cities do not have easily accessible data. The lack of a national, easily parsable database on zoning makes this improvement to the model extremely difficult to execute.

- Power plants and utilities may publish data on the area they serve, including zip codes and maps of this data. However, this data is not widely accessible for all utilities across the country. If the data was accessible, this could be used to both estimate the impact (in terms of number of people) of a DDoS event against a utility, as well as mapping exposed SCADA devices to utilities more accurately.

- Rather than relying on IPv4 scan data from services like Shodan and Censys, we could actively probe SCADA devices using the Modbus and DNP3 protocols to pull any

metadata that would indicate which utility a particular device belongs to. Modbus, for example, has a "Project Code" which in some cases may indicate the usage of the device. However, in our analysis of exposed data, no project codes for exposed devices tied a device to a particular U.S. utility. While we would like to actively probe devices, for ethical reasons we do not do this in order to avoid disrupting actual power plants processes.

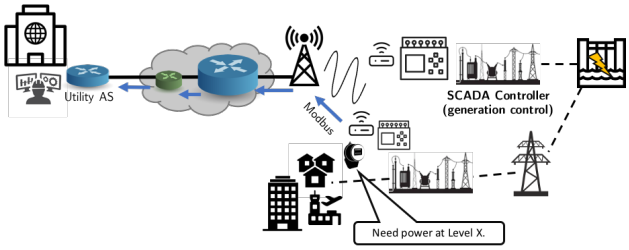
- Finally, situations arise in which clusters of power plants and SCADA devices make associating a SCADA device with a power plant and utility difficult, due to the density of both plants and devices in the area. In these cases, we cannot be sure which plants own which devices. An example of this is New York City, where our scan data reveals many devices, and there are also many power plants from the EIA.gov dataset. The only remedy to this situation would be more fine-grained GeoIP data on the exposed SCADA devices, which does not exist.

With our model of ASes, which if targeted with DDoS, would make U.S. utilities vulnerable to loss of SCADA communications, we can now examine the vulnerability of these links to DDoS. Before we present that evaluation, we first lay out our threat model and experimental ethical considerations in the next section.

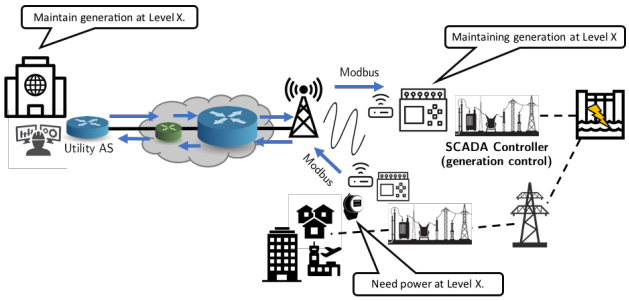
6.4 Threat Model

Our study focuses on a realistic threat model that could be realized in existing U.S. power and Internet infrastructure. Utilities in the U.S. (and elsewhere) rely on the Internet for generation site control, including increasing or lowering the amount of power being generated. By relying on the Internet, these utilities place their critical SCADA traffic at the mercy of well-known cyber attacks like DDoS. This DDoS can then be targeted at the parts of the Internet where the utility traffic bottlenecks. By overloading these Internet links, the adversary can cut off SCADA connectivity between utility control centers and downstream consumers (where sensors are for detecting power usage) and generation sites (i.e. power plants). In the absence of rapid, digital communication, a single large demand event on the grid causes the system to steer into dangerous system frequencies. Without AGC to bring the frequency back in check, due to the inability for the control center to receive SCADA communications, power loss, generator failure, and other adverse side-effects can take place.

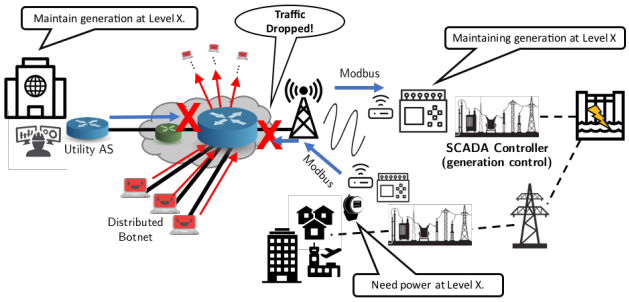
Figure 6.7 illustrates this threat model. In response to consumer demand for power, connected meters in Figure 6.7a signal to the utility control center (in the utility AS) that they need power generated at a certain level via Modbus. The utility control center in Figure 6.7b then sends a Modbus message to their generation sites to generate power at that level, which they then respond successfully and maintain power supply at that level. However, a botnet in Figure 6.7c then uses an LFA to disrupt the Modbus communications between the utility control center and the generation site, so the generation site continues to generate power at the previous level. In Figure 6.7d, a sudden demand increase or decrease



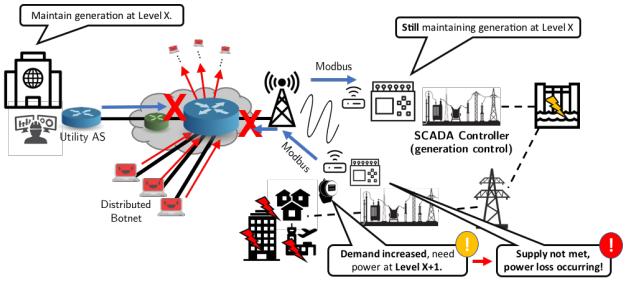
(a) Connected Smart meters signal to the control center they need power at a certain level



(b) The utility AS responds by sending control commands to generation sites to generate power at the requested level.



(c) DDoS impairs the communication between the source of SCADA commands and the generation sites and smart meters



(d) With a sudden increase in demand, power supply cannot be met and consumer impact (i.e. blackouts/brownouts) is seen because control of generation sites (via AGC) is impaired

Figure 6.7: Overview of Threat Model

may occur, either due to weather, public events (e.g. start of the workday), or a cyber-physical attack (e.g. an adversary disconnects a substation, causing load to decrease or increase). This demand event then causes the grid to report that more power is needed, but this communication will ultimately not reach the control center due to the DDoS of the Internet links the utility implicitly relies on. Due to the lack of communication back to the control center, AGC cannot enable a secondary response of either more or less power generation to bring the system frequency back to a stable point. When the supply is not met, the frequency moves into dangerous levels, as shown earlier in Section 6.2 and power loss at consumers begins to occur or other adverse side-effects, like generators failing or power lines overloading may happen.

6.4.1 Who is the Adversary and What Do They Control?

There have been several recent incidents where extremely capable nation-states or state-sponsored groups have infiltrated a company's power systems and wreaked havoc. Examples include Ukraine [76], Stuxnet [44], and the Triton malware [34]. However, to carry out the attack described above, one does not need to be a nation-state or anyone near that advanced of an adversary.

We consider the adversary to be at a minimum an entity with both 1) Internet access in order to rent a botnet, 2) the "range" of that botnet (i.e. what parts of the Internet the botnet can target), and 3) a model of what Internet AS-to-AS links the target utility relies on. To satisfy requirement 1), dozens of DDoS-renting services exist on the Internet [75, 48, 128]. Our prior research provides the data needed for requirement 2) [87]. Finally, we showed in

the prior section how an adversary can build a model for requirement 3) of what Internet links a utility may rely on. Nation-state adversaries or adversaries with deeper knowledge of a particular utilities ISPs can target utilities even more effectively without having to rely on public scan data and inferred Internet models. Non-well-resourced adversaries can increase their knowledge of specific utility Internet connectivity, and thus their likelihood to disrupt AGC via DDoS, by being willing to *actively* probe a utilities network with traffic scans or targeted Modbus/DNP3 packets to infer ISP connectivity.

The adversary *does not* need to necessarily be able to trigger grid demand events in order to reap the benefits of grid instability after executing DDoS. While the ability for an adversary to physically or digitally cause load increase/decrease events would pair extremely well with a targeted DDoS, a less-resourced adversary could pull public power demand data from EIA.gov or other sites and trigger their DDoS with a rentable botnet during predicted peak hours of power usage on either the rise of the peak (demand increase) or fall of the peak (demand decrease). The study of both sudden and gradual demand increases and decreases and predicting them is a well-studied topic going back decades [144, 145, 146, 61, 160]. Demand response, including the occurrence of sudden load increases and decreases, is also well-studied [9, 153, 10]. This research could be leveraged by a motivated adversary.

6.4.2 What Does the Adversary NOT Control?

We *do not* assume our adversary controls the following infrastructure, abilities, or information:

- Internet infrastructure, such as ASes, routers, their own botnet (though this would be useful), or ISPs in general.
- Nation-state resources.
- The ability to execute physical attacks or cyber-physical attacks against a utility's system, as they can rely on normal demand events. However, being able to trigger both an attack like MadIoT [132] *and* DDoS against the utilities links at the same time would be able to *force* the grid into an unstable state.
- Insider knowledge of the targeted utility.
- Knowledge of and ability to mitigate where botnets will congest Internet links along the way to the target, as LFAs can be primed to estimate the ability to takedown a target link with low-traffic flows until the attack is ready to fully execute [65, 87].
- Any knowledge not known by a typical DDoS/LFA adversary. See Section 2 for more details on LFAs.
- Knowledge of specific botnet distributions at the time of attack, as the botnet itself can be used to measure its ability to target a certain Internet link as shown by our prior work [87].

6.4.3 Experimental Ethics

In this study, we do not actively perform any DDoS attacks against actual utilities on the Internet. Furthermore, we do not actively probe the SCADA devices used to build our model in Section 6.3. Instead, we rely on Shodan and Censys to probe the devices and

then pull their data and any metadata they collected. When we model the effects of DDoS against AS-to-AS links that the utilities in our model were found to rely on, we leverage the BGP simulator from Chapters 3 and 5. During our analysis of the impact of failed AGC communications on an actual power grid, we *do not* test turning off AGC on an actual utility's network. Instead, we rely on public Matlab Simulink models of AGC in generation facilities.

Though our work would be improved by 1) actively probing SCADA devices in an attempt to extract data from them that indicates the utility they belong to, and 2) using a platform like PEERING [121] to test BGP poisoning against utility links on a small-scale, we leave this to utility operators to take the insights from our work as motivation to test the vulnerability of their own networks.

6.5 Evaluating the Vulnerability of Utilities to DDoS

In the next three sections, we present our evaluations of the viability of DDoS attacks against Internet links utilities rely on (this section), the impact of AGC failing in an example power system (Section 6.6), and finally, defending Internet-connected utilities with Nyx (Section 6.7). To conduct our evaluations, we rely on extensive simulations of both the Internet and its routing properties as well as AGC and frequency response in power systems. We follow each discussion of simulation methodology with evaluation results and analysis.

6.5.1 Simulation Methodology

As described in our threat model from the last section, DDoS, specifically link-flooding attacks, can be wielded against utilities relying on the public Internet. Link Flooding Attacks are covered in great detail in the background in Section 2.2.2 and in the presentation of Nyx throughout Chapter 3. To simulate LFAs, we rely on the Chaos BGP simulator, an open-source Internet routing simulator used for Nyx evaluation. Chaos allows inputting an AS-to-AS topology of the Internet, and then successive simulation of routing dynamics coupled with high-level traffic simulation. After feeding multiple botnet models into the simulator, we can calculate the amount of bots in each model that can normally flow over, or target, certain Internet links. In our evaluations, we rely on the same botnets from our work in prior chapters: Mirai, Conficker, and BlackEnergy.

With the baseline amount of botnet traffic that can flow over a targeted utility link, we can determine the percentage of the botnet that can drive traffic over the target, thus inducing DDoS. Like the work in Nyx, we *do not* make an attempt to say how much traffic

would overwhelm the link, since the bandwidth of links on the Internet is an unknown factor not published by ISPs. However, we *can* evaluate and present the amount of botnet traffic that normally flows over targets versus the amount that *could* flow over when you combine a typical Link Flooding Attack with an amplification mechanism.

Our prior work, though not covered in this dissertation, examines the use of BGP poisoning (i.e. the technique from Nyx) to route botnet traffic *onto* a target link. While Nyx drives congested benign traffic away from congested links due to DDoS, Maestro [87] collapses many botnet flows onto a single target link when possible. In this manner, Maestro can amplify a botnet that may normally be able to target a link with 10% of its available bots, all the way to 100% of its bots. In terms of a botnet like Mirai, that is going from 290,000 bots to 2.9 million bots. With each bot capable of sending flows in the range of 4 kb/s undetected as malicious traffic (i.e. an HTTP GET), this amount of amplification poses a serious threat to core Internet links.

To simulate Maestro, we leverage the same Chaos BGP simulator from Nyx with extensions to enable the evaluation of the Maestro concept by McDaniel *et al.* [87]. In the rest of this section, we present both the pre-Maestro vulnerability of utility links to DDoS along with the post-Maestro vulnerability of the same utilities. We also tie the results back to the geographic perspective of the utility-to-device connectivity model from Section 6.3.

6.5.2 Results and Analysis

Baseline Vulnerability of Utilities

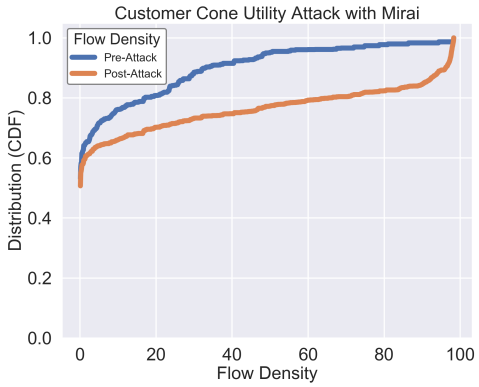
By taking the AS links from the utility connectivity model from Section 6.3, we have the targets for DDoS which an adversary must overwhelm in order to cause loss of AGC communications. In total, we have 584 total utilities with their ASes identified and the ISP of the SCADA devices controlling generation. Of these, we identified a total of 433 paths between utilities and the ISPs. The average AS length of these paths was 3.89, leading to 819 total links available to DDoS which qualify as an LFA (e.g. attacking the link would lead to the malicious traffic being unfilterable by the utility AS). First, we computed the vulnerability of these links to DDoS by examining what percentage of the Mirai, Conficker, and BlackEnergy botnets could normally flow over the links. This metric is referred to as *flow density*, and it is the percentage of the total botnet which can flow over a particular link.

In Figure 6.8, the blue lines reveal the baseline vulnerability of these 584 total utilities per botnet. The plots are CDFs, and represent the distribution of pre- and post-Maestro flow densities of each botnet over the utilities in our model. On average, the baseline vulnerabilities per botnet for the connected utilities, in terms of flow density, are 9.78%, 10.18%, and 9.96%. In terms of raw numbers of bots, given that our model of Mirai used in Nyx has roughly 2.8 million bots, Miria can drive an average of 273,840 bots over the connected utilities with no assistance, only with an off-the-shelf rented botnet. LFAs can be configured to send low-bandwidth flows from each bot to every other bot, such that all botnet flows flow over the target link. In this study, the target links from Figure 6.8 are the

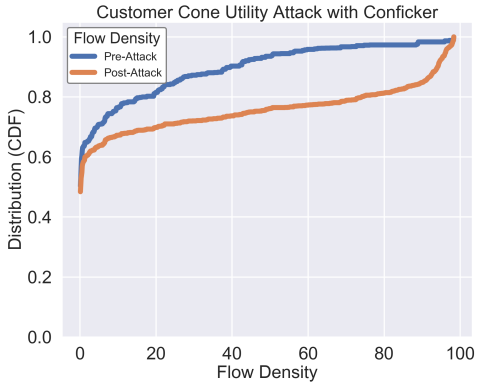
utilities. If each bot sets up an HTTP server on their host that responds to HTTP GETs with a substantially larger response than request, then we can make a simple calculation for the amount of traffic that these pre-Maestro flows can target these utilities with.

Starting with only the first HTTP GET, if we assume that the traffic amount sent by each bot is 4 Kb/s, then 273,840 bots from Mirai on average leads to the following calculation of flows: $(273,840\ bots)^2 = 74,529,000,000\ flows$. Therefore, the *maximum* pre-Maestro, baseline amount of traffic that can be pushed over the utility links is 37.26 terabytes/s. Assuming the HTTP servers are then setup to respond with 4x the original response size, then we end up with *1184 terabits/s* that can DDoS these utilities dependent links with only 9.78% of the total Mirai botnet. The Conficker botnet and BlackEnergy botnet are both smaller; therefore, the total flow possible at the baseline amount of flows would also be smaller. We note that 1,184 Tb/s can easily overwhelm any link on the Internet, especially the ones used by these utilities, given recent Mirai DDoS events against Dyn DNS and and the Internet links responsible for Liberia have taken down massive ISPs with only a few TB/s of flow [6, 3]. We carefully note that not any off-the-shelf botnet renting service may have the entire Mirai botnet available to launch traffic from, and that the 1,184 Tb/s is a *maximum bound* if we assume that 9.78% of the botnet when collected in 2017 was available to be leveraged against utility ASes.

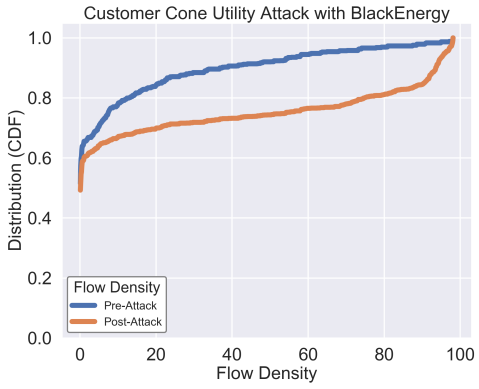
To conclude our discussion of the baseline vulnerability of these utilities, the maps in Figure 6.9 reveal the vulnerability by region of the country of the utilities in our dataset. Darker colors indicate a higher ability for the Mirai botnet to target the particular utility. Recall these maps combine both the Shodan and Censys data, and are for each variant of our model: matching city/state, within 40 km, and within 10 km, where the within n km



(a) Pre- vs. Post-Maestro Mirai Botnet Vulnerability of Power Plants



(b) Pre- vs. Post-Maestro Conficker Botnet Vulnerability of Power Plants



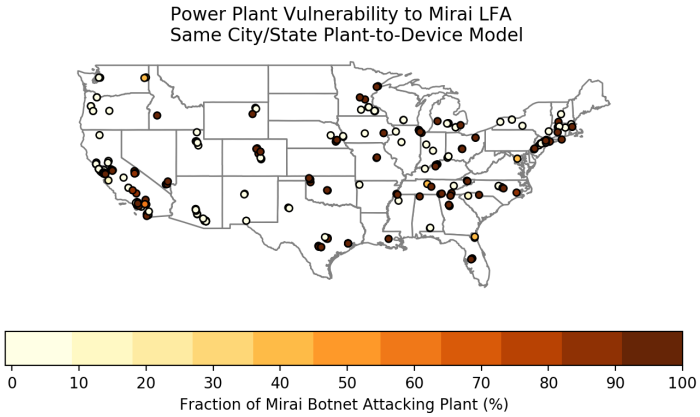
(c) Pre- vs. Post-Maestro BlackEnergy Botnet Vulnerability of Power Plants

Figure 6.8: Maestro Pre- vs. Post-Flow Density distribution across all datasets and Mirai, Conficker, and BlackEnergy botnets

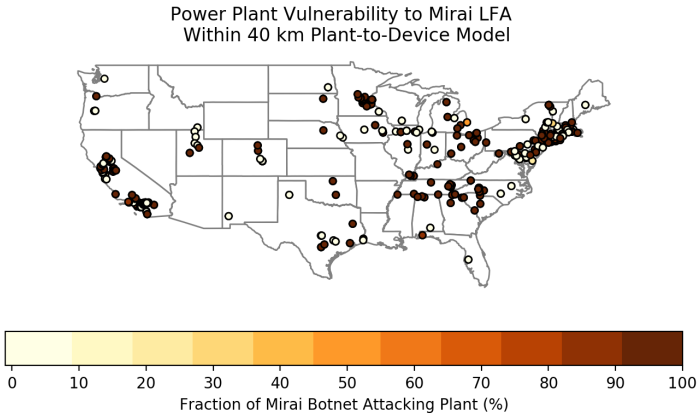
indicates that the SCADA device was within that range of the nearest generation site tied to a utility in our dataset. Across all of the models, we see that the eastern side of the U.S. is particularly vulnerable. The sheer size of the NYC region led to many utilities in our dataset lying there, and we see that the baseline vulnerability of that region is upwards of 70% of the Mirai botnet being able to target utility links there, which they critically depend on for serving AGC communications. Also note that on each of these models, the vulnerability is either extremely high or very little vulnerability in general, as shown by the heatmap colors either being very dark or very light, with little orange. This may be due to the types of ASes providing traffic for the utilities. For ISPs responsible for a utility's traffic that are positioned on the fringes of the Internet, the likelihood of a majority of any botnet being able to reach them is low. This was covered in our prior work on Maestro [87]. For the more affected utilities, they may be able to defend themselves better if the utility chose to use an ISP for their SCADA devices that relied on links on the Internet which do not traverse areas of high betweenness. By high betweenness, we mean links that many ASes on the Internet rely on for their normal "best paths" in the BGP decision process.

Amplified Vulnerability of Utilities

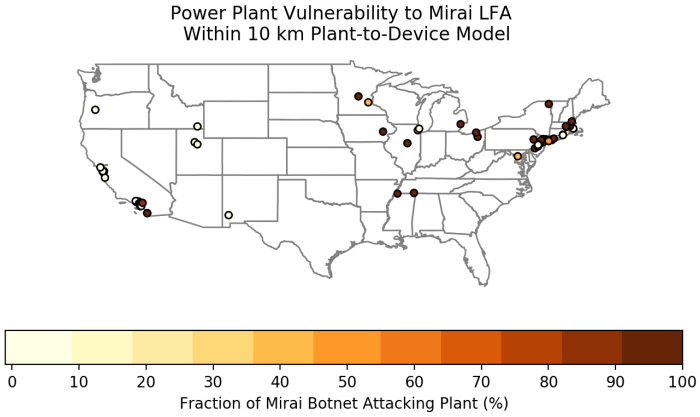
Now we present our findings on the vulnerability of connected utilities to an LFA *after factoring in Maestro*, a tool that leverages BGP poisoning to drive up the flow density of a botnet. Maestro uses BGP poisoning for this like Nyx uses BGP to steer benign traffic around targeted links. Rather than steering benign traffic around congestion, Maestro steers botnets *botnets onto target links* they were not on before or on less. Introducing Maestro though adds a critical piece to the threat model of our adversary. With Maestro, the adversary must



(a) Matching City-State Model



(b) Within 40 km Model



(c) Within 10 km Model

Figure 6.9: Baseline Mirai LFA Vulnerability of Power Plants per Plant-to-Device Model across both Shodan and Censys data

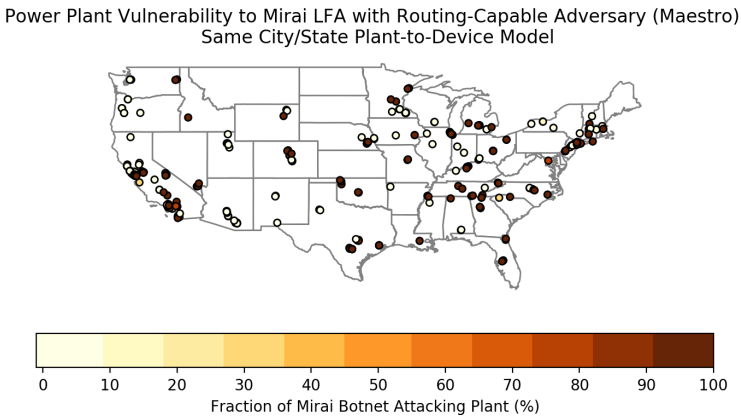
control a BGP router on the Internet, in order to execute BGP poisoning. There are multiple ways to do this. First, the adversary can be an AS willing to essentially DDoS itself in order to drive more botnet flows over a vulnerable utility link. Nation-states with authoritarian control of their country's internet infrastructure (e.g. China) can easily commandeer an AS that if using Maestro, could amplify an LFA over a particular target link in the U.S., thus inducing DDoS, cutting off AGC communications, and leading to power loss for consumers or failure of generators and other power systems equipment. The second option is to compromise an AS. An example of this would be an adversary willing to execute a spear phishing attack against an operator of a U.S. company with a BGP router, or the ability for an adversary to leverage red team toolkits like Metasploit to exploit a vulnerable BGP router from a company like Cisco or Juniper.

If we assume that an adversary could and would do one of these two options, then the connected utilities in our model face a significantly worse vulnerability to LFAs than without Maestro. Recall that the prior baseline vulnerabilities for each botnet against the 584 utilities in our dataset was 9.78%, 10.18%, and 9.96%. With Maestro in an unoptimized setting, meaning that we did no tuning of the BGP speaker selection in order to drive up the vulnerability, these same utilities are now made vulnerable to 23.36%, 23.97%, and 24.31% of Mirai, Conficker, and BlackEnergy. Figure 6.8 illustrates this with the increase shown by the movement from the blue line to the orange line for pre- to post-Maestro flow density. For the Mirai example earlier, this brings the unoptimized maximum-bound on traffic able to target these utilities with to 1,704 Tb/s, an increase in an additional 520 Tb/s from the baseline vulnerability. Given that nation-states like Russia have already successfully taken down other country's power grids with far more sophisticated attacks [76, 5], this new attack

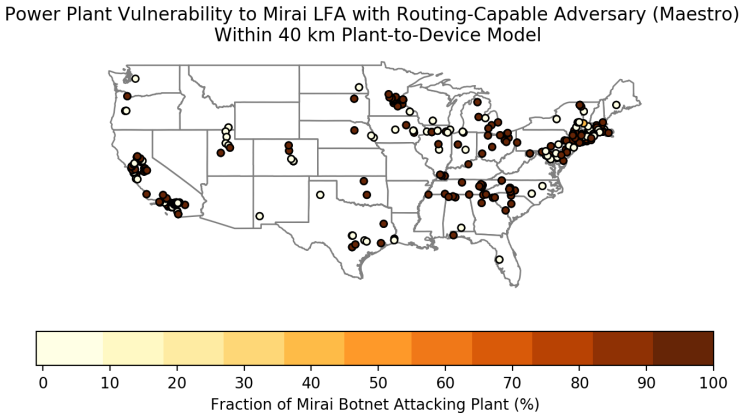
with DDoS against unconnected utilities would give a vast amount of power to a nation willing to commandeer an AS within its own country and ability to rent a botnet (or use its own botnet).

When we look at the geographical impact and additional coverage an attack could make against the U.S. in Figure 6.10, Maestro enables more targets in the southeastern and midwest to fall under the threat of this DDoS. This can be seen especially in the increase in darker markers from Figure 6.9a to Figure 6.10a. A notable example is a utility in Washington D.C. The pre-Maestro baseline vulnerability of this utility rises from only 40% of the botnet able to target that utility, to upwards of 85% of the botnet. For Mirai, that means a rise from roughly 5 Petabits/s of DDoS flow from 1.12m bots in Mirai to *over 22 petabits/s* maximum flow over the utility's links in D.C., the nation's capital. Again, this is the maximum bound, but even if only 10% of this total of the Mirai botnet flow density was available to the adversary, the resulting flow over this particular utility would still exceed 2.2 Pb/s for an LFA with full bot-to-bot level attack (i.e. as shown by Coremelt [140]). The largest ever known DDoS attack, which took down GitHub in February 2018, was 0.00059% the size of this example DDoS against the D.C. utility's links. While GitHub is a global service, capable of serving hundreds of thousands and into the millions of requests per second, utilities do not have near the capacity in their SCADA communication infrastructure.

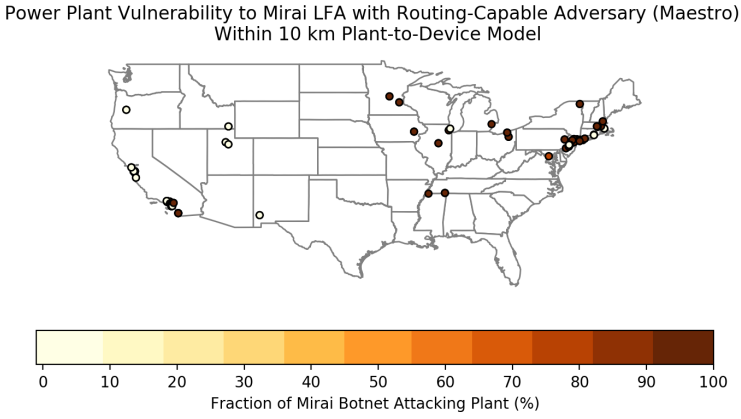
With this analysis and our simulations, we have presented the first study of an LFA DDoS against connected utilities in the U.S. Our findings demonstrate that these utilities, even without a motivated and well-resourced nation-state adversary, can reach levels easily capable of overwhelming any Internet link, yet alone links that utilities rely on. Worse, with



(a) Matching City-State Model



(b) Within 40 km Model



(c) Within 10 km Model

Figure 6.10: Post-Maestro Mirai LFA Vulnerability of Power Plants per Plant-to-Device Model across both Shodan and Censys data

a motivated nation-state capable of compromising a BGP router or commandeering one, the state of U.S. power utilities reliant on the Internet is dire.

6.6 Evaluating the Impact of Failed Automatic Generation Control (AGC) on Power System Stability

In this section, we present a set of simulations of sudden load increase and decrease events on the power grid when AGC is in play. When AGC is down due to DDoS, sudden load increases and decreases can drive the stability of the power system into unsafe territory. The prior section showed how utilities are vulnerable to DDoS. This section then assumes that DDoS is possible against utilities and that AGC SCADA communications would be impacted. With that assumption, we simulate with realistic models of a power system with AGC what would happen to the system stability, or settling frequency of the system, when AGC is impaired.

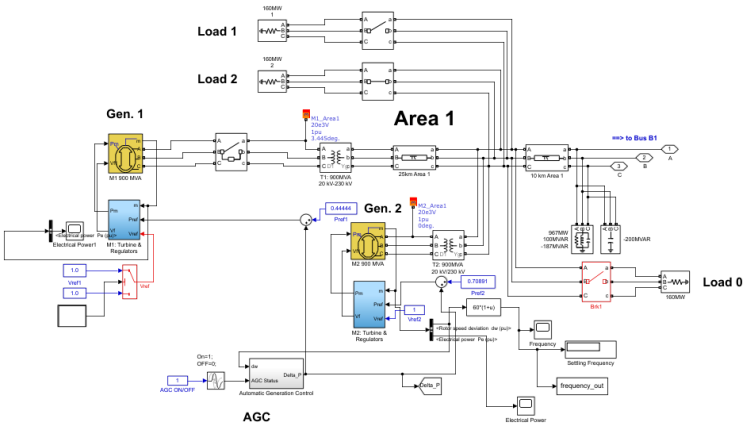
Prior work from Srikantha *et al.*, Sargolzae *et al.*, and Liu *et al.* [136, 119, 77] demonstrated that when AGC communications face packet loss or delay, the generation of power can be significantly affected, leading to blackouts or outright generator failure. These works used Matlab/Simulink, a set of software with the ability to simulate power systems and grid frequency response along with AGC. Soltan *et al.* [132] also used Matlab software to simulate power system response during sudden load increase and decrease events. Our simulations do not significantly differ from those of [136, 119, 77], yet we include them to show how power loss can occur when AGC is impacted and load events occur.

6.6.1 Simulation Methodology

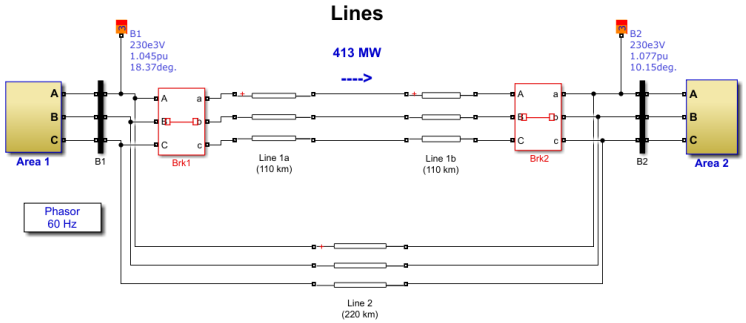
To simulate AGC response to sudden load increase and decrease events, we leverage a Matlab/Simulink model of multiple generators, power lines, consumer load, and AGC across

multiple sites [159]. This simulation setup has been used in prior work to model the effects of failed AGC [136, 119, 77]. Figure 6.11 reveals the three pieces of the simulated power system: Area 1, Lines (1 and 2), and Area 2. Area 1 has two synchronous generators, regulated by AGC, and 3 potential load sources. In reality, these loads could come from a variety of sources: neighborhoods, buildings, factories, or other utilities purchasing power from the utilities running these generators. Each generator can be at a physically separate site in the real world, and AGC would leverage SCADA protocols, particularly Modbus or DNP3, to regulate these generator's output during load increase/decrease events. In Figure 6.11b, two transmission lines carry power from Area 2 in Figure 6.11c where two additional generators live. In Area 2, we place a breaker which can cut off the power from Generator 4 to simulate the loss of that generator source. In the real-world, that loss could come from downed power lines, failure of the generation site itself, or a cyber attack. The three load sources in Area 1 from Figure 6.11a can be turned on or off to simulate sudden load increases or decreases.

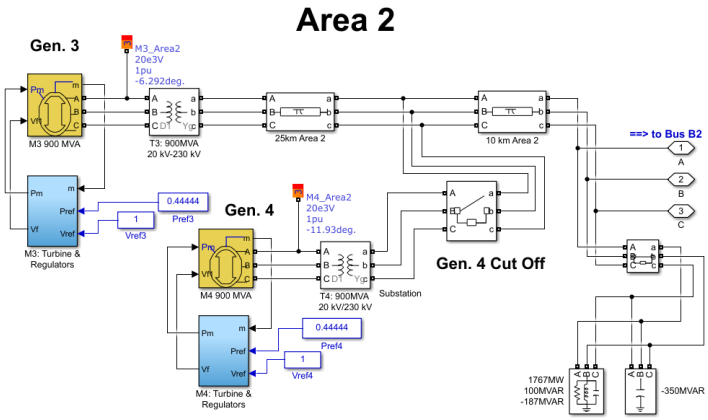
When the simulation is run, AGC is started at $t = 40s$, and disturbances made up of either a load increase or decrease are applied *prior* to AGC being enabled to see what effect AGC has on bringing the disturbance under control. Recall from Section 6.2 that the stable frequency in U.S. power systems is 60 Hz, and the NERC limits on safe frequencies are defined in Table 6.1. Next, we explore the effects on the steady-state frequency following the contingency in the power system from Figure 6.11 when AGC is functional and when AGC is disabled in the three following cases: sudden load increase, sudden load decrease, two consecutive sudden load increases.



(a) Area 1 of the Simulated Power System



(b) Power Transmission Lines between Area 1 and Area 2



(c) Area 2 of the Simulated Power System

Figure 6.11: Matlab/Simulink AGC Simulation Setup

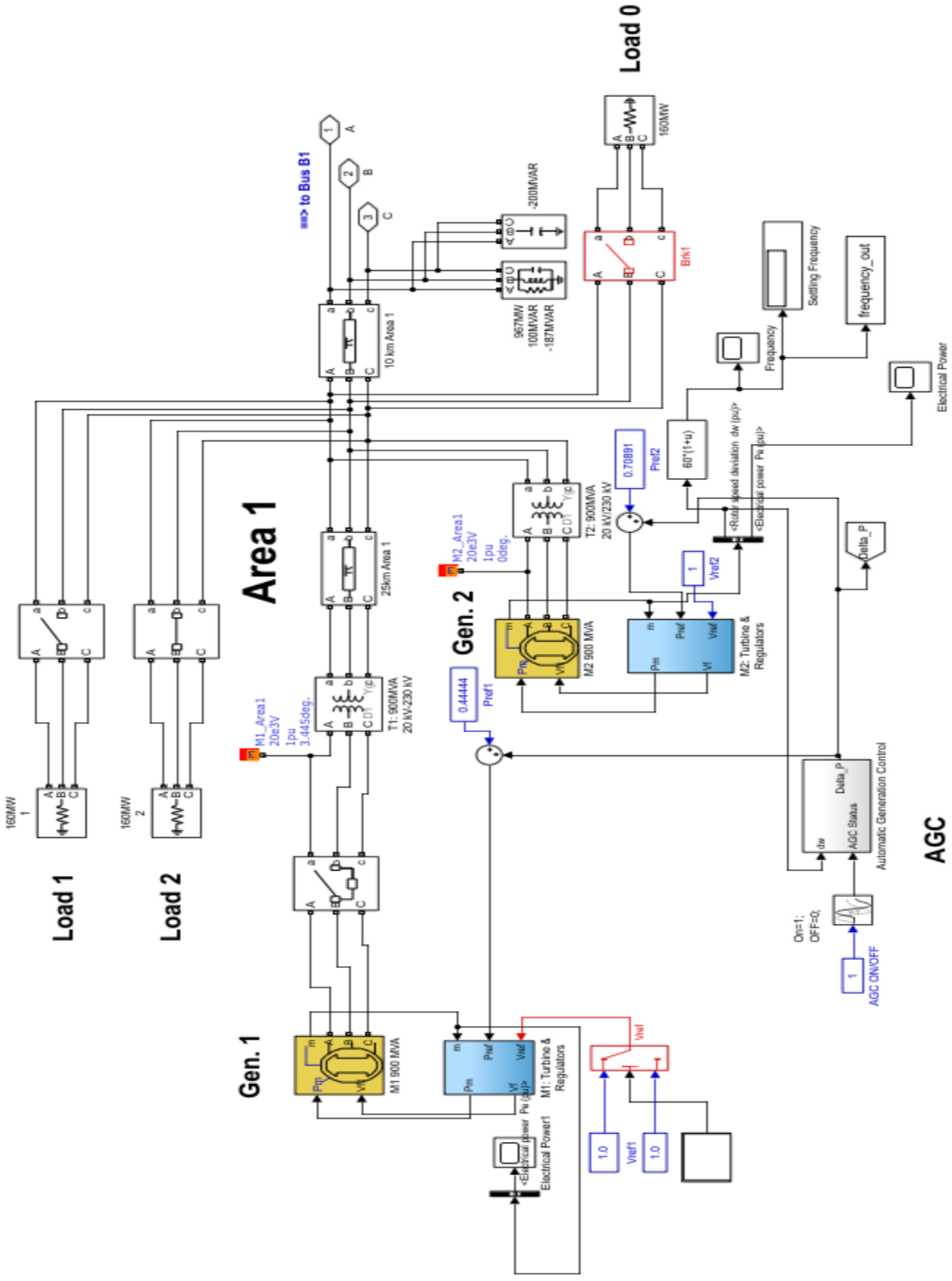


Figure 6.12: Larger View of Area 1 of the Simulated Power System

6.6.2 Results and Analysis

Figure 6.13 shows the impact on the power system from Figure 6.11 when AGC is both off (due to DDoS) and on (regular operation) under three cases: sudden increase in load (Figure 6.13a), sudden decrease in load (Figure 6.13b), and sudden increase followed by another sudden greater increase in load (Figure 6.13c. Each figure has the disturbance(s) marked with the orange lines, and the stable system frequency (60 Hz) marked with the green line. The NERC FAL emergency limits on the frequency of the system are marked with the dotted black lines. AGC is turned on at $t = 40s$ in each simulation, which is similar to the time after a real-world increase would have been initially regulated by the primary governor response (for background on the governor response, see Section 6.2 earlier).

Sudden Load Increase

In Figure 6.13a, we see that when the disturbance of a sudden load increase is triggered at the orange line at 40s, the frequency of the system drops to a minimum of 59 Hz. At this point, this is incredibly dangerous for the system, but the frequency quickly recovers with the primary governor response to nearly 59.9 Hz. At this point, if AGC were not hampered by DDoS, then the blue line shows this secondary response from AGC driving the settling frequency back to the nominal frequency of 60 Hz. However, with AGC disabled, the frequency stays outside of the NERC safety limits for the FAL-low (these FAL alarms are for the Eastern U.S.). At a settling frequency of under 59.9 Hz, the system *is unbalanced* and the sudden increase of load will not be met by the available generators. Without the ability for AGC to tell the other generators in Area 1 to increase their power production,

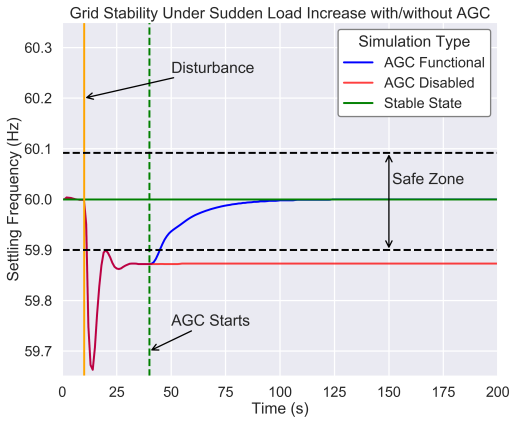
the demand will not be met, power loss will occur, and the system may face other unstable conditions. A study of the 2003 blackouts in the U.S. explored several negative effects on the grid when the system is unstable [11].

Sudden Load Decrease

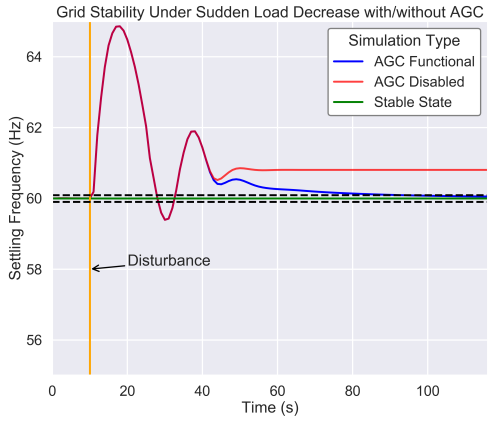
The next scenario occurs when we reset the system to its stable state and suddenly cut off a load source from the power system. In reality, this could occur at the beginning of the work day when residents leave their homes and the HVAC units switch to a different daytime temperature. Another case could be when a power line drawing power from the generators fails, and the load is no longer needed from the generators, yet power continues to be produced. In this case, AGC would start after the primary response and tell the generators with SCADA communications to slowly stop generating the same levels of power as before, and reduce the total amount generated. Figure 6.13b illustrates this. Again, we see when AGC is not enabled, the settling frequency remains now above the FAL-high alarm limit. In this case, rather than blackouts occurring, the generators and power lines would be at risk because more power than is needed is being generated and pushed through the system, without anywhere to go.

Successive Load Increase

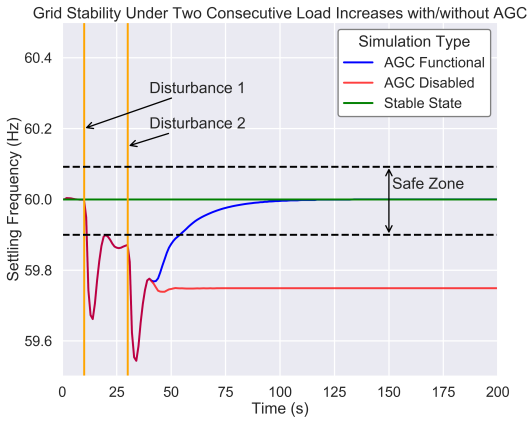
The final scenario shown in Figure 6.13c illustrates the impact of failed AGC due to DDoS when two load increases follow each other. A nation-state adversary could wield this type of attack to drive the settling frequency even further away from the stable state than a less-resourced adversary. By using an attack like MadIoT [132] to turn on many high-wattage



(a) Single Increase in Load



(b) Single Decrease in Load



(c) Successive Increases in Load

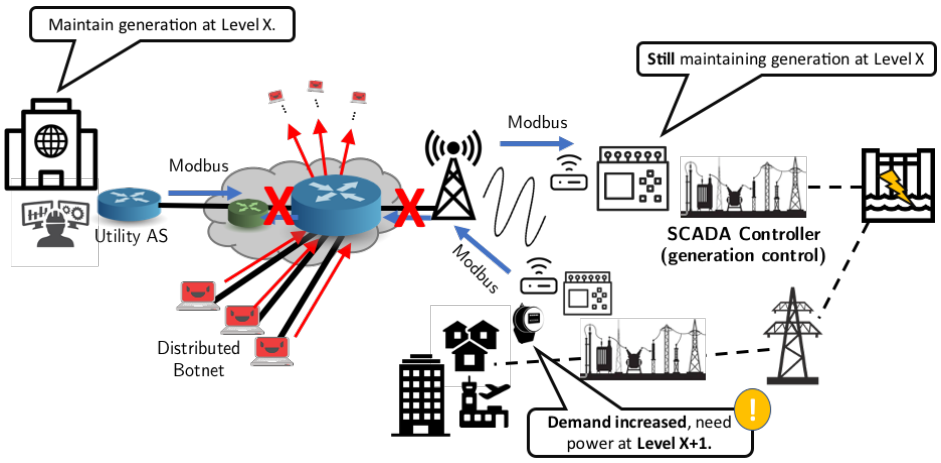
Figure 6.13: Impact of Failed AGC SCADA Communication on Power Grid Stability

IoT devices at once, then following this action with a DDoS attack against the utility, the grid could be driven into a frequency low enough to trigger long-term damage to the utility's infrastructure. The longer the frequency stays out of the NERC safe zone (i.e. within the FTL and FAL limits), the worse events could happen to the utility's infrastructure. Rather than only causing temporary blackouts lasting up to several hours, physical generators could be destroyed due to the inability for operators to automatically respond to the FAL alarms with AGC. In this case, operators would be relegated to using other means (e.g. phone calls) to call operators and adjust generation control to manually fill the role of AGC.

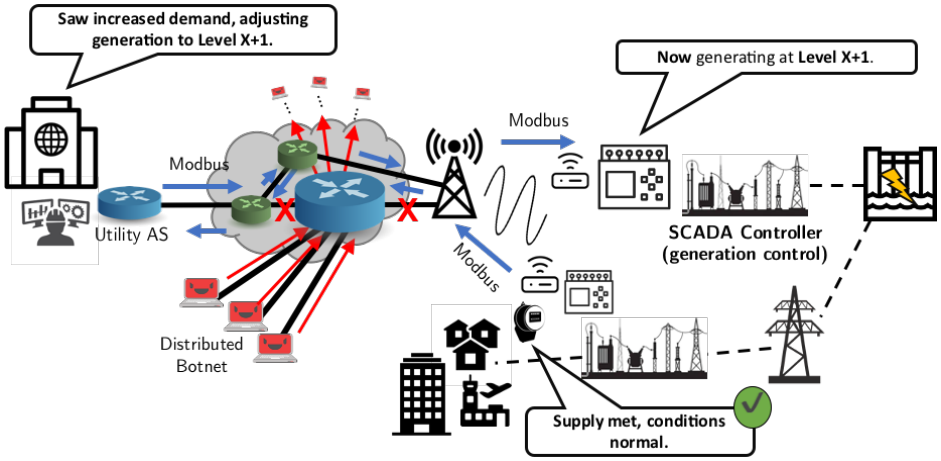
6.7 Evaluating the Ability to Defend Utilities with Nyx

Now, we employ Nyx from Chapter 3 as a mechanism to defend the utilities from our Model in Section 6.3 from the DDoS attacks presented earlier. Nyx, unlike other solutions, requires no coordination from ISPs and no costly subscriptions to DDoS-mitigation services. With Nyx, a utility that is also their own AS can leverage the control-plane of the Internet to route SCADA traffic used for AGC *around* the DDoS against the links they depend on. By doing this, the utility effectively makes the DDoS *irrelevant* to their network, with only scripts to adjust routing rules required to be deployed by the utility's network operators. These scripts mirror the same software infrastructure used for the experiments conducted in our measurement study from Chapter 4, and they can be deployed against software or hardware BGP routers. In this section, we lay out our analysis of Nyx as a utility's DDoS defense tool with extensive simulations using the Chaos BGP simulator [131].

At a high-level, Figure 6.14 shows Nyx used by a utility to route around congestion from DDoS. In Figure 6.14a, the utility relies on SCADA communications between the devices monitoring power consumption and controlling generation of power, and the utility's AS where the communications are received and appropriate actions taken. However, in this example, the Internet links the utility relies on are targeted by an LFA, such as Crossfire or Coremelt [65, 140]. We show how these utilities are vulnerable to these DDoS attacks in Section 6.5 earlier. Now, when the utility detects this DDoS is occurring by seeing that their SCADA traffic is being dropped, the utility can then deploy Nyx, shown by Section 6.14b. Here, the utility deploys BGP poisoned advertisements coordinated by Nyx to alter the current path the SCADA communication is taking from the SCADA device ISP(s).



(a) Utility impacted by DDoS



(b) Utility using Nyx to re-route SCADA traffic

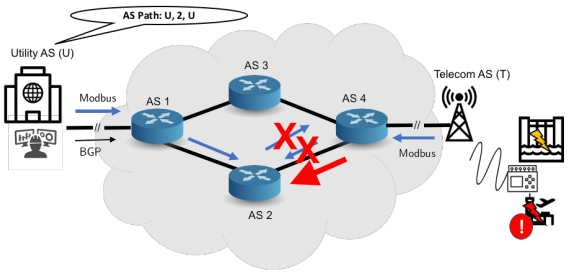
Figure 6.14: High-level view of routing around DDoS against utilities with Nyx

The utility will swap onto new paths for the SCADA traffic until congestion is sufficiently alleviated. In the case of Nyx, the system will rely on practical defaults discovered by our measurement study in Chapter 4 and evaluated in Chapter 5. The particular default limit for searching for new, unaffected paths is 3 in Nyx, though we also test Nyx finding uncongested paths up to a limit of 10 in the original experiments.

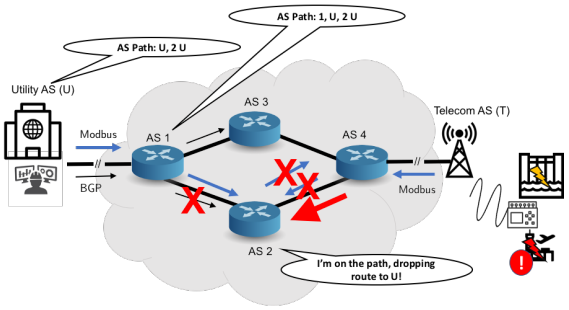
An affected utility can find uncongested paths using BGP poisoning, the core technique Nyx relies on. The process of using BGP poisoning to discover new paths and route around congestion for an affected utility is shown in Figure 6.15. Note that the utility leveraging Nyx would experience power loss, generator failure, or in the best case have to resort to manual frequency control if the DDoS continued to impact the power system if Nyx did not mitigate the congestion. In the prior section we explored the impact of failed AGC on a simulated, but realistic power grid. By using Nyx, the utility can rely on a conventional routing protocol like BGP to re-establish communication with the utility's metering infrastructure and generation sites, enabling AGC to correct the settling frequency and return that frequency to limits within NERC's guidelines for safe operation [100].

6.7.1 Simulation Methodology

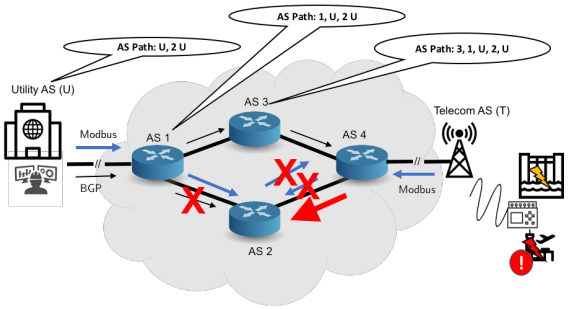
To evaluate Nyx in this use case, we rely on the same simulator from the original evaluation of Nyx. We utilize the practical routing constraints found from our measurement study and evaluated earlier in past chapters. Specifically, we limit the search capacity of Nyx to 3 alternate paths. We also limit the maximum number of poisons available to 245 poisons in total, and we enforce poison filtering. In the next section, we look at the effectiveness of



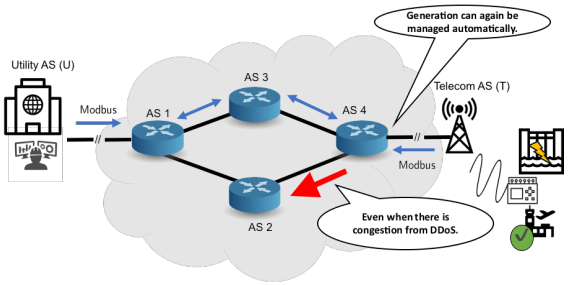
(a) First, utility advertises a poisoned path with the congested AS on the path



(b) Then, congested AS drops path to utility, and path continues to propagate down alternate path(s)



(c) Next, poisoned path reaches ISP for utility's SCADA devices



(d) Finally, the SCADA device ISP swaps to new route, since congested path is no longer available

Figure 6.15: Low-level view of utility using Nyx

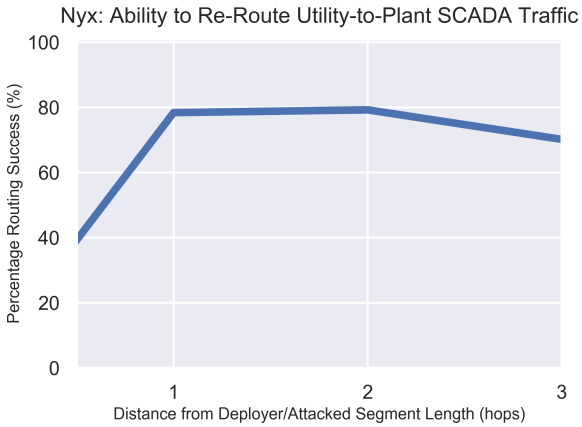
Nyx through three lenses: routing success, performance success, and performance success in the context of our utility-connectivity model as done in the section evaluating utility vulnerability to DDoS.

6.7.2 Results and Analysis

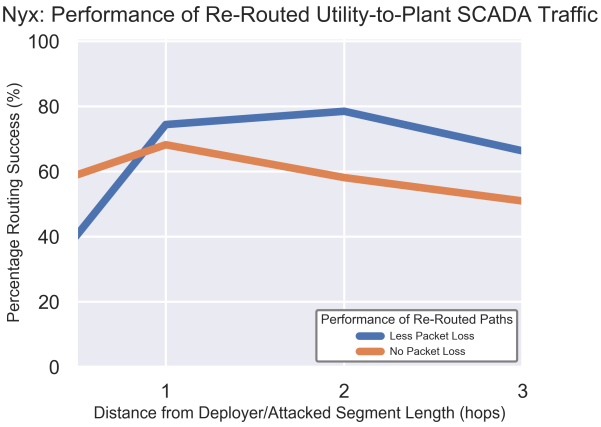
Figure 6.16 shows the distribution of routing success and performance success of Nyx when used to protect the vulnerable utilities in our model of utility-connectivity from Section 6.3. We evaluate Nyx's performance for this sample of utilities in the same manner as the random sampling of 10,000 deployer-critical AS pairs from Chapter 3. Recall that routing success is defined as Nyx successfully enabling the deployer (i.e. here, the targeted utility) to find an alternate route other than the currently utilized route, which is experiencing DDoS. Performance success is defined as not only finding an alternate route, but also finding an alternate route with either *less or no congestion*, where weak performance success is the former and strong performance success is the latter. In our simulations here, we target the Internet paths the 584 utilities rely on with an LFA DDoS, which cannot be filtered by the utility. We use a congestion factor of 5.0, meaning the DDoS over-subscribed the utility's reliant links by 5x more traffic than the link can handle. We also use a bandwidth tolerance of 1.1, which is the hardest scenario for the Nyx deployer. Finally, we use the Mirai botnet to target the links in the LFA.

Nyx Routing Success and Performance Success

In Figure 6.16a, we see that routing success hovers around 80% on average for the 584 sample utilities in our model when links between 1 and 3 hops out from the deployer are targeted.



(a) Routing Success for LFA

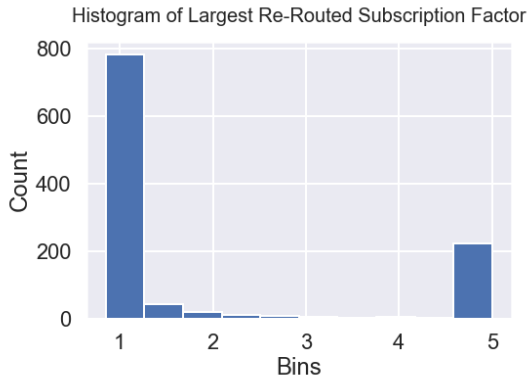


(b) Performance Success for LFA for CF of 5.0

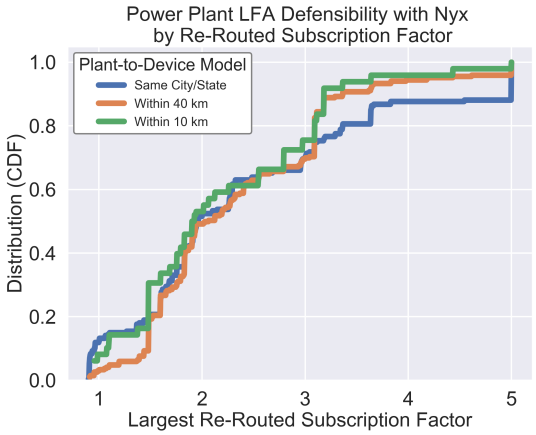
Figure 6.16: Routing Success and Performance Success of Nyx for defending against Miria LFAs deployed at targeted utilities

We cut off the graph after an attacked segment length of 3 because the sample size for the number of utility to SCADA device ISP paths of length greater than 3 is not statistically significant. Given the routing success of 80%, this means that the vast majority of the time, Nyx will allow a DDoS'd utility to find an alternate path that is not directly targeted by the LFA. While this does not completely solve the problem of congestion, this finding does mean that the utilities from our model are well-positioned to use BGP poisoning to re-route. As we showed in our measurement study from the previous chapter, this ability to re-route is not always guaranteed for certain Internet paths. In Figure 6.16b, we see that for the cases where Nyx can re-route the utility's affected AGC traffic, Nyx can find on average a path with less congestion than the original 5x over-subscribed link again nearly 80% of the time. For paths that Nyx finds that are completely uncongested, Nyx can assist a targeted utility around 60% of the time across all 584 utilities. This means that for 60% of utilities, Nyx can allow 60% of the targeted utilities in our dataset to find a totally uncongested path, thus alleviating the packet loss of AGC communications and potentially bringing the power system frequency back to a normal state.

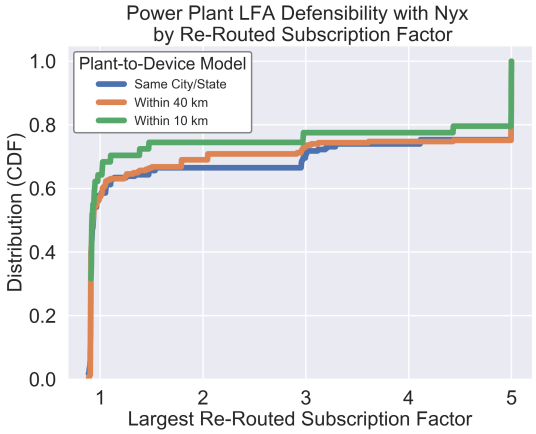
When we examine the post-Nyx largest subscription factors, we find a notable trend. Figure 6.17b highlights the mean largest re-routed subscription factor for all 584 utilities as a CDF for each model from our Shodan and Censys utility-connectivity dataset. From this perspective, Nyx appears to perform mediocre across all utilities, with the largest subscription factors along the post-re-routed path distributed evenly from no congestion to the original amount of congestion (i.e. 5x over-subscribed). However, the plots from Figure 6.16 seem to contradict this perspective. Fortunately, we can explain this by looking at the outliers in this dataset. Figure 6.17a displays a histogram of the largest re-routed



(a) Histogram of Largest Re-Routed Subscription Factors



(b) Mean Largest Re-Routed Subscription Factor



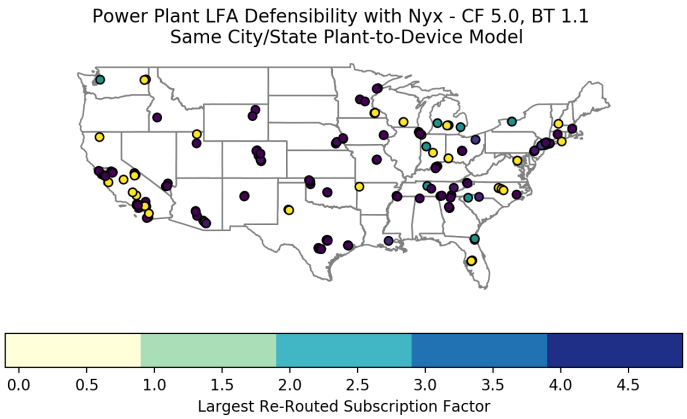
(c) Median Largest Re-Routed Subscription Factor

Figure 6.17: High-level overview of Nyx success across all utilities by Largest Re-Routed (Post-Nyx) Subscription Factor

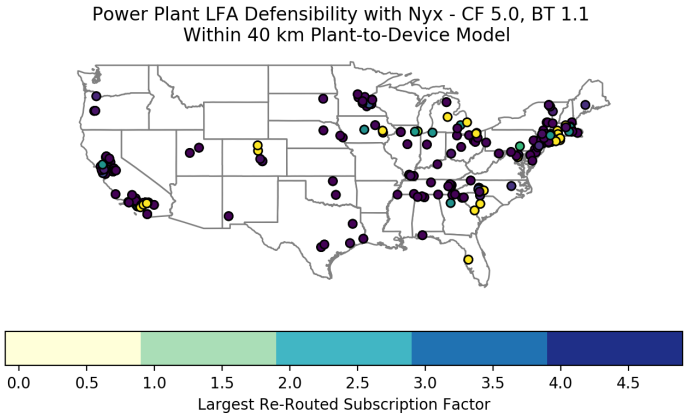
subscription factors between normally subscribed (1x) to original attack subscription (5x over-subscribed) on the target link. Clearly, most of the time, Nyx is able to successfully mitigate the attack across all links for all 584 utilities seen by the largest bin being the first. However, the non-negligible size of the largest bin illustrates why the *mean* largest re-routed subscription factor may be a misleading approach to examining the performance of Nyx. To correct this, we show in Figure 6.17c the same view as Figure 6.17b but with the *median* largest re-routed subscription factor. By using the median, we account for the outliers shown by the histogram. The perspective in Figure 6.17c mirrors the results from the CDF of performance success earlier. We see that around 60% of the links across the 584 utilities targeted by the LFA can be completely avoided with Nyx, regardless of the model of connectivity used. Since our results are robust against each different model of connectivity, we have confidence that the approach we take in Section 6.3 is at least consistent between the three different strategies to tying utilities to their respective ASes and SCADA device ISPs (i.e. ASes) they rely on for AGC communications.

Nyx Performance per Utility Connectivity Model

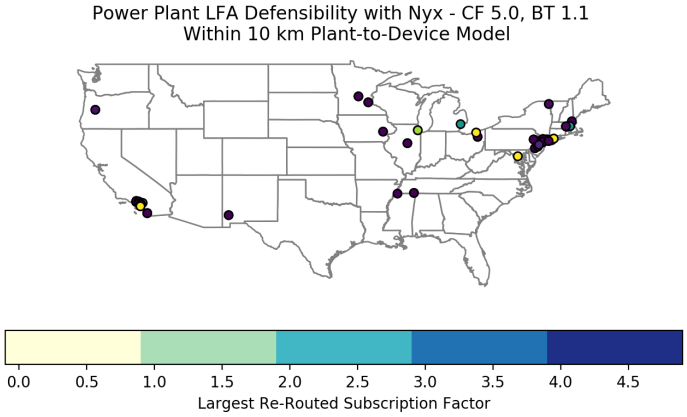
Figure 6.18 shows the geographic perspective of Nyx's performance across the utilities in the model. While these maps from the DDoS section earlier (Section 6.5) uses a heatmap to show the percentage of the botnet on each utility (i.e. flow density), this heat map shows the *median largest re-routed subscription factor* from Figure 6.17 above. While the results across all models follow the same trend numerically, the utilities by region in Figure 6.18 differ somewhat in how much Nyx can defend each region. Notably, the ability for Nyx to defend west coast utilities from the matching city-state model is strong, shown by Figure 6.18a,



(a) Matching City-State Model



(b) Within 40 km Model



(c) Within 10 km Model

Figure 6.18: Nyx ability to defend against Mirai LFAs per Plant-to-Device model across both Shodan and Censys data

$$\frac{|1.5 - 5.0|}{\left(\frac{(1.5+5.0)}{2}\right)} \times 100 = 107.69\%$$

Figure 6.19: Example of calculating Nyx vulnerability reduction

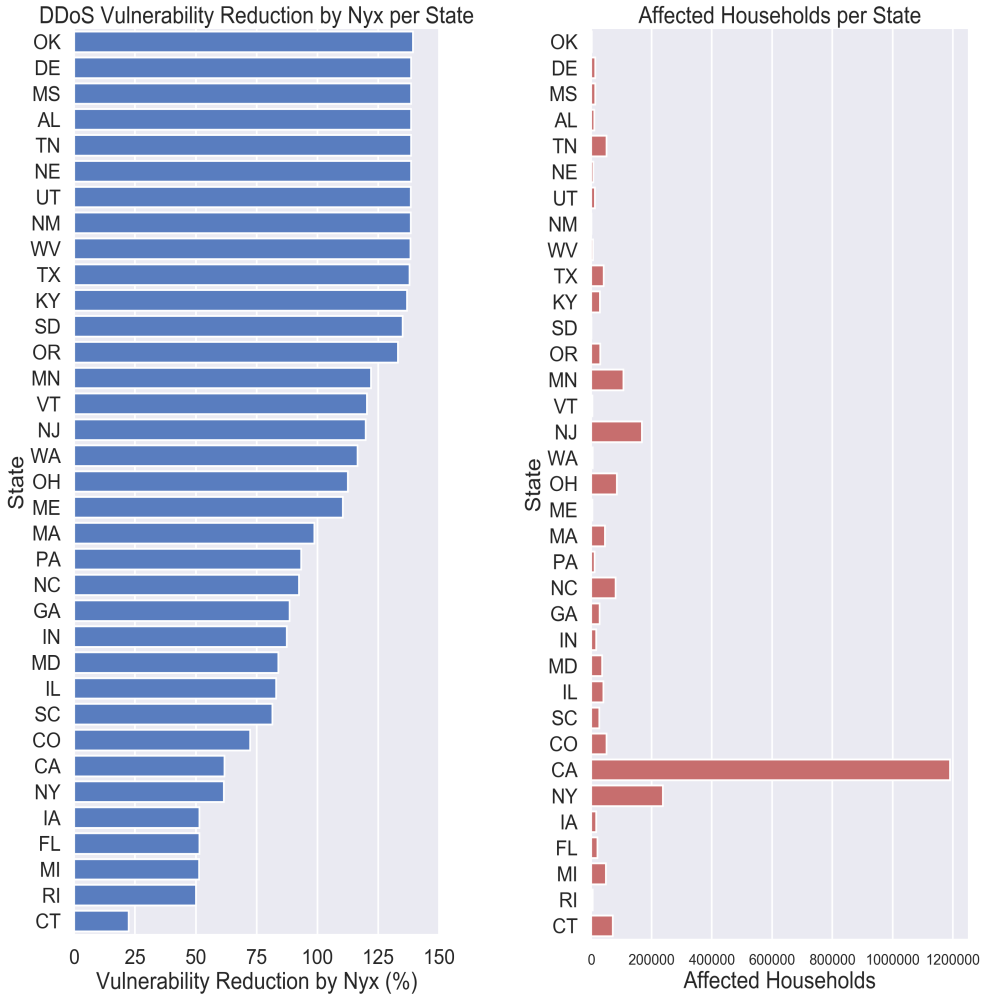
the west coast utilities are less defensible when relaxing the utility connectivity parameters for the "Within 40 km model" from Figure 6.18b. In the DDoS section, we pointed out how a particular Washington D.C. utility could be targeted with 80% of the Mirai botnet, effectively wiping out any chance of AGC being able to pass through the public Internet and regulate the power system settling frequency. With Nyx, though, we see that Nyx can alleviate *all* of the congestion caused by that DDoS. This is shown by the yellow tint on the same utility in D.C. for the matching city-state model in Figure 6.18a. Given the significance of per region vulnerability to residents in that region, we expand our analysis by breaking down the vulnerability reduction by Nyx per state and by affected occupied households per state for the "Within 40 km" model in Figure 6.20a and Figure 6.20b.

Vulnerability Reduction by State and Households

The "vulnerability reduction" is defined as the percentage difference between the original 5x over-subscribed targeted utility links and the largest subscription factor after Nyx attempts to mitigate the LFA. For example, if Utility A can be protected by Nyx with a median largest subscription factor of 1.5, then the vulnerability reduction is calculated as 107.69% by Equation 6.19. Figure 6.20a shows the vulnerability reduction per state in our "Within 40 km" model.

Since we have the zip codes of each utility in our utility-connectivity model, we pull from the most recent U.S. census from 2010 to gather the number of occupied households in each

zip code served by the utility. We then aggregate all zip codes per utility in our model by state. Figure 6.20b shows the same order of states (sorted by most vulnerability reduction to least) by occupied households. While not shown here, the population by state for the affected households follows the same trend. For example, Nyx can reduce the vulnerability of the 84 California utilities in our model by 61.7% for a total of 1.19 million affected households and 4.07 million residents. On the upper end, states like Oklahoma, Delaware, Mississippi, Alabama, Utah, Texas, Oregon, New Mexico, West Virginia, South Dakota, and Tennessee all have over 133% reduction in vulnerability. Any utility in a state with over 133% vulnerability reduction can use Nyx to protect themselves from LFA DDoS, since this is the threshold at which Nyx can make the DDoS irrelevant by re-routing the AGC traffic onto totally uncongested paths. Without Nyx, these utilities would either have to subscribe to a costly DDoS mitigation service that still may not protect them, or risk losing AGC communications due to DDoS-induced congestion and succumb to potentially devastating effects on the power service to residents.



(a) Percent Reduction in Vulnerability (Link Subscription Factor) of Utilities per State in Model

(b) Occupied Households per State in Model

Figure 6.20: LFA DDoS Vulnerability Reduction by Nyx by State and Occupied Households (according to 2010 U.S. Census)

6.8 Discussion

We briefly cover other defenses against the attacks presented in this paper, as well as real-world limitations of our attack on Internet-connected utilities.

6.8.1 Other Defenses

DDoS Defenses

Beyond Nyx, DDoS against utilities can be addressed in many of the same ways presented in Section 3.5 when describing alternatives to Nyx. Examples of such approaches include a redefined Internet architecture that focuses on bandwidth allocation and reservation rather than assuming enough capacity will be available. SCION and SIBRA are examples of this type of future Internet architecture, where the latter adds a DDoS defense component to the former new architecture [163, 19]. SDN filtering, network appliances, and CDN services also may help eliminate DDoS. However, these services may *never* protect a utility against a true LFA, as such an attack is completely outside the control of any defense that lives at the utility's border. Nyx, on the other hand, provides a low-cost, low-effort solution to routing around congestion caused by LFAs.

Disconnecting the Smart Grid

Another approach would be to completely disconnect the power grid from the public Internet. Massive utilities like TVA function as their own private ISP, and even provide Internet service for others [113]. However, as shown in Section 6.3, many utilities, especially smaller ones than giants such as TVA, do rely on the public Internet. Unfortunately, disconnecting the grid

from the Internet is not a small task given the current level of entanglement with "connected devices" the utility industry has reached. An effort by the U.S. Department of Energy known as "DarkNet" seeks to decouple the power grid from the public Internet by running grid communications on "dark" fiber not used by telecoms and other organizations with fiber along common utility poles [47]. However, this effort has been stalled in implementation and may never reach full deployment, since the DOE cannot yet mandate that every utility abide by new rules of connecting their infrastructure. While disconnecting the grid seems like a reasonable proposal, it would be the equivalent of asking an existing cellular network provider to upgrade all of its field equipment in the entire country to a version that runs with an entirely new set of software, hardware, and technical requirements.

6.8.2 Real-World Attack Limitations

The LFA attacks presented in Section 6.5 are limited by several real-world factors. First, any LFA executed against a utility reliant on the Internet is beholden to experiencing congestion *prior to reaching the utility's critical links*. If this occurs, the LFA may not induce a significant amount of packet loss on the AGC communications. Second, our utility connectivity model from Section 6.3 is only a model, and may not match up with the real-world connectivity of utilities. Finally, any adversary which launches a DDoS against an Internet-connected utility will likely not have direct visibility into whether the attack was successful *unless* the attack severely impairs the power system. A blackout due to a DDoS would certainly lead to news articles, but short of that outcome, the adversary would need to be in the area the utility provides power to in order to know whether an actual power

grid failure occurred. In future work, we plan to model the effects of a range of packet loss scenarios against AGC in order to provide the adversary with a sense of how much or how little congestion must be forced to cause a desired power system impact.

Similarly, any defensive approach using poisoning, such as Nyx, may be limited as well. We simulated Nyx's ability to defend utilities with the real-world limitations explored in Chapter 5. While it was possible to defend many of the utilities in this study from our model of connectivity, other utilities not in our model may suffer from a position in the Internet where Nyx may not be as effective. In particular, if a utility relies on links that traverse the core of the Internet, Nyx will be less effective. To combat this, one could combine Nyx with BGP communities to route around ASes in the core of the Internet where poisoning is filtered out. Another limitation of using Nyx to defend a utility from LFAs is the fact that any utility wishing to use Nyx must advertise their own BGP routes. Additionally, the utility must use the AS they control for the SCADA communications to be re-routed. A utility *cannot* re-route their own SCADA traffic around congestion if it is pushed through the Internet with another AS. In that case, the utility would be BGP hijacking, a malicious approach to re-routing traffic.

6.8.3 Additional Use Cases (and Non-Use Cases) for Nyx

While protecting critical infrastructure is one use case for Nyx, there are other use cases (and non-use cases) worth considering.

First, other use cases include:

- Defending against LFA DDoS in general. DDoS solutions available on the commercial market that are deployed as appliances at the end network cannot defeat LFAs, as this traffic cannot be filtered. A control-plane based defense like Nyx would be required to avoid such types of DDoS.

- When DDoS directly targeting an end network is occurring, Nyx can be used to help route around such DDoS. The previous chapter on Nyx covers this type of DDoS defense in the analysis of traditional (direct) DDoS defense. See Section 3.3 for more details.

- The core technique used by Nyx can also be used to conduct DDoS. This use case is explored in the Maestro system by McDaniel *et al.* [87] and used earlier in attacking U.S. utilities in Section 6.5.

- Nyx can be extremely useful for protecting networks which while they operate their own AS (and thus BGP), do not have the financial resources to purchase protection from a CDN or DDoS defense company..

Next, there are use cases where Nyx may not be applicable:

- When DDoS can be easily filtered, or detected by networks and dropped prior to reaching the targeted link(s), Nyx may not be useful.

- Nyx may need to be supplemented by BGP communities for use in parts of the Internet where BGP poisoning is not supported. Certain parts of the Internet are known to filter BGP poisoning, as discovered by our measurements in Chapter 4.

6.9 Related Work

The most similar work to this final thrust is split into two areas: attacks against power systems via traditional cyber or cyber-physical angles, and attempts to model the connectivity of the power grid to the Internet.

The first area concerns attacks. Soltan *et al.* presented MadIoT attacks [132], an attack against power systems by causing many IoT devices to suddenly turn on or compute and drive up power demand. Soltan followed up their work with a defensive approach to MadIoT attacks by leveraging economic dispatch principles in power system stability [133]. Acharaya *et al.* explore attacking power grids with high-demand power devices as well, including electric vehicle (EV) chargers [8]. Barreto *et al.* presented a variety of attacks against electricity markets, where power is sold and transferred, using game-theoretic techniques [18]. Most relevant to our work are three studies revealing that denial of service attacks, or congestion/delay added to communications in AGC, can severely compromise the stability of the power grid being attacked [136, 119, 77]. Our work connects the vulnerability of AGC systems to *local* failures in communications to the effects of attacks against *Internet* infrastructure that modern utilities increasingly rely on.

The second are concerns attempts and studies that try to model the connectivity of the modern power grid to the Internet. While we presented some of this work in Section 6.2, we note that there is no single conclusive model that answers the question: "How connected is the U.S. power grid to the Internet?" and then successively "How reliant is the U.S. power grid on the Internet?". This study answers the former, but cannot answer the latter without explicit cooperation with utilities. Unfortunately, utilities are not known to share

this information publicly, nor can this information, if disclosed privately, then be published in academic research. The closest work to this is a survey of power utilities done by Oak Ridge National Lab as part of a U.S. Department of Energy Cybersecurity for Energy Delivery Systems project [47, 108] as part of the DOE DarkNet project.

Chapter 7

Future Work and Concluding Remarks

7.1 Future Work

In this section we will consider future directions and challenges for this research arc.

7.1.1 Combatting Routing-Aware Adversaries

The Nyx adversarial model does not consider a global adversary that is routing-aware. This adversarial model becomes important when defenders want to protect their networks from an adversary controlling a significant amount of ASes. In general, combating a global adversary would require our system to detect upstream adverse network conditions and congestion without assistance, and utilize bandwidth and botnet models representative of routing-aware adversaries. It would also require Nyx to support not just multi-critical setups, but multi-deployer setups, as global adversaries may target more than one AS and we must know whether multiple Nyx deployers operating at the same time in the same topological region would still work effectively.

Furthermore, Nyx relies on searching for alternate paths until an uncongested or non-degraded path is found. If the deployer AS had the ability to detect degraded quality of service along upstream links, our system could make more informed decisions of where to migrate critical traffic. While the search heuristic in Nyx is simple, it is effective, but it could still be adapted and tuned to combat global adversaries. Finally, global adversaries may seek to adapt their link flooding attacks to proactively switch onto links that Nyx re-routes onto, much like the work of Tran *et al.* [149], Nyx could just as easily using strategies from the field of moving target defense to also shift onto links that the adversary cannot predict.

7.1.2 Challenges of Extending to Multiple Deployers

While we do not address the effects of multiple Nyx deployers on the overall system performance in Chapter 3, we briefly discuss several challenges yet to be addressed for a multi-critical, multi-deployer scenario. We define multi-deployer Nyx as multiple ASes actively executing Nyx re-routing policies in practice. They each execute these policies in order to steer inbound traffic from their own set of potentially overlapping critical ASes, all while under DDoS or other adverse network conditions.

First, when we introduce multiple deployers, per-link bandwidth availability becomes a larger problem. Consider a scenario where two deploying ASes with a shared set of critical ASes, try to shift their traffic off of the same targeted link. If the shared critical ASes all take the same set of alternative paths, then the new alternative path must hold up with its normal traffic, the residual attack traffic that has spread its way, and *two* sets of new ASes' inbound traffic (the critical ASes of each deployer). Now, consider not two deployer ASes, but 200. Expressed outside of BGP, this would be the equivalent of Google Maps directing hundreds of cars to take the narrow, two-lane side-road parallel to the congested 6-lane highway. The two-lane road is generally less traveled on average, therefore planning for that road did not necessarily include rush-hour traffic diverting its way.

Second, multiple deployers advertising poisoned paths may introduce additional load on the *control-plane* infrastructure of the Internet. While the prior bandwidth scenario focused on the data-plane, or TCP and UDP packets encapsulating HTTP or other application layer traffic, multiple deployers can introduce burden onto the propagation of updates and their receipt throughout the interconnected ASes. For September 2019, there were 328 combined

BGP announcements and withdrawls per hour on average, based on measurements from the APNIC (Asia Pacific Region Routing Registry) measurement service at bgp.potaroo.net. Normal rates of BGP updates would spike linearly with each new AS leveraging BGP poisoning via Nyx to reactively re-route around congestion. One might expect the control-plane of the Internet to be fairly stable, given that its failure would cascade into the failure of data-plane connectivity for the entire Internet; however, even the core of the Internet's control-plane is vulnerable to large spikes in traffic, both due to adversaries [124] and router bugs [sup].

Despite these challenges, there are unexplored strategies to mitigate such issues. Intelligent usage of available capacity via the potential coordination or signaling between deployers, or use of resource allocation strategies for BGP that typically are reserved for protocols like TCP both may alleviate these two primary challenges to ASes deploying Nyx simultaneously. A self-contained examination of these challenges merits a longer evaluation and analysis by future studies. While we do not cover this analysis here, our Chaos BGP simulator can be adapted to study multiple deployers.

7.1.3 Expanding BGP Measurements

In Chapter 4 we presented a re-evaluation of Nyx on the live Internet. Other security systems, including censorship circumvention approaches, could be re-evaluated using our measurement infrastructure on PEERING. These re-evaluations could be supported with continuous measurements of the properties we measured. These continuous measurements could be integrated into existing measurement platforms from CAIDA and RIPE NCC.

To expand our measurements further, one could partner with larger ISPs to see if our results hold up at an even larger, non-publicly available sample size. While our measurements are representative, more vantage points would increase the usefulness of the measurements to operators making decisions about whether or not to deploy Nyx and other uses of BGP poisoning.

7.1.4 Standardization

With the knowledge gained about the existing state of BGP poisoning on the live Internet in Chapter 4, BGP poisoning could potentially be merged into the BGP specification as a fundamental feature of the protocol, rather than as a side-effect of conventional protocol behavior. While standardized mechanisms like BGP communities and the BGP Multi-Exit Discriminator (MED) attribute exist, other ASes do not have to respect these configurations when routing their traffic. While BGP poisoning works now, as AS-path filtering becomes more pervasive, systems like Nyx may be harder to execute without combining poisoning with techniques like BGP communities.

7.2 Conclusions

In this dissertation we have investigated the thesis that rather than redefining the way the Internet works, networks can leverage conventional Internet routing protocol behavior to defeat advanced DDoS. We explored three aspects of this thesis, that it is *possible, practical, and useful*. First, we presented and rigorously evaluated Nyx, a state-of-the-art DDoS defense system that makes this thesis possible. Next, we conducted an Internet-scale measurement study of the core technique behind Nyx, BGP poisoning, which demonstrated the practicality of Nyx. Finally, we examined the usefulness of Nyx by applying Nyx as a defensive tool to a problem of national importance: U.S. electric utility susceptibility to DDoS. After first constructing a model of Internet-reliant utilities based on real-world data, we then showed these utilities are susceptible to both traditional and LFA DDoS. Finally, we conducted a range of simulations exploring both the power system impact and showed Nyx's ability to mitigate DDoS against the power grid via Nyx deployment at the utility network. We can draw two major conclusions from the work in this dissertation.

First, in its current form, Nyx is easily deployable at the border of ASes for use in DDoS defense. However, the balance between operator desires and security when configuring BGP policies (and Internet infrastructure in general) has skewed towards operators moving to consider techniques like BGP poisoning more of a problem than a solution. This is not only dangerous, but in the absence of widely supported alternatives (e.g. BGP communities), the BGP community risks alienating those who wish to use protocol behavior for positive goals in order to combat those who leverage the protocol behavior for malicious intentions. While there is no clear answer to the question of "to filter or not to filter BGP poisoning",

our work leads to an straightforward conclusion: techniques considered harmful to some in Internet routing may have use cases beneficial to those without the funding, time, or people to buy into other alternative techniques or technologies. Worse, other techniques, like BGP communities, the BGP Multi-Exit Discriminator, and entirely new Internet architectures are either not significantly deployed or they lack the versatility to make a positive impact on the availability of an ASes' infrastructure.

Second, a substantial body of academic work and industry investment has been poured into countering DDoS with either new approaches or applying vast amounts of bandwidth to the problem. Unfortunately, neither of these strategies has led to a decrease in the prevalence and devastation of DDoS. Our work demonstrates that operators, academics, and industry practitioners must not take for granted both the explicit features and implicit side-effects of the pre-existing Internet protocols and infrastructure. Protocols like BGP are already widely deployed, and we show that rather than deploying a new protocol, one can rely on the existing protocol to mitigate a problem previously thought to only be viably defeated with new approaches or vast amounts of bandwidth. This philosophy applies to much more than DDoS defense. Any field of science that relies on existing infrastructure to solve current problems can be introspective about what the field currently has available, implicitly or explicitly, to solve new problems without devising shiny new approaches with the latest technologies and techniques.

Bibliography

[sup] Reckless Driving on the Internet, url=https://dyn.com/blog/the-flap-heard-around-the-world/, author=Earl Zmijewski, journal=Oracle Dyn DNS Blog, year=2009. 89, 210

[2] (2012). IEEE Standard for Electric Power Systems Communications-Distributed Network Protocol (DNP3). *IEEE Std 1815-2012 (Revision of IEEE Std 1815-2010)*, pages 1–821. 152

[3] (2016). Dyn Analysis Summary Of Friday October 21 Attack. https://dyn.com/blog/dyn-analysis-summary-of-friday-october-21-attack. 2, 12, 45, 70, 143, 172

[4] (2016). Dyn Analysis Summary Of Friday October 21 Attack. https://nakedsecurity.sophos.com/2016/09/29/why-a-massive-ddos-attack-on-a-blogger-has-internet-experts-worried. 2, 12, 70

[5] (2016). Inside the Cunning, Unprecedented Hack of Ukraine's Power Grid. https://www.wired.com/2016/03/inside-cunning-unprecedented-hack-ukraines-power-grid. 176

[6] (2016). Mirai IoT botnet blamed for smashing Liberia off the internet. https://www.theregister.co.uk/2016/11/04/liberia_ddos. 2, 12, 16, 22, 45, 70, 172

[7] (2018). CAIDA AS Rank. http://as-rank.caida.org/. 99, 101

[8] Acharya, S., Dvorkin, Y., and Karri, R. (2020). Public Plug-in Electric Vehicles+ Grid Data: Is a New Cyberattack Vector Viable? *IEEE Transactions on Smart Grid*. 205

[9] Albadi, M. H. and El-Saadany, E. F. (2007). Demand response in electricity markets: An overview. In *2007 IEEE power engineering society general meeting*, pages 1–5. IEEE. 166

[10] Albadi, M. H. and El-Saadany, E. F. (2008). A summary of demand response in electricity markets. *Electric power systems research*, 78(11):1989–1996. 166

[11] Andersson, G., Donalek, P., Farmer, R., Hatziargyriou, N., Kamwa, I., Kundur, P., Martins, N., Paserba, J., Pourbeik, P., Sanchez-Gasca, J., et al. (2005). Causes of the 2003 major grid blackouts in North America and Europe, and recommended means to improve system dynamic performance. *IEEE transactions on Power Systems*, 20(4):1922–1928. 185

[12] Antonakakis, M., April, T., Bailey, M., Bernhard, M., Bursztein, E., Cochran, J., Durumeric, Z., Halderman, J. A., Invernizzi, L., Kallitsis, M., et al. (2017). Understanding the mirai botnet. In *26th {USENIX} Security Symposium ({USENIX} Security 17)*, pages 1093–1110. 143

[13] Anwar, R., Niaz, H., Choffnes, D. R., Cunha, Í. S., Gill, P., and Katz-Bassett, E. (2015). Investigating Interdomain Routing Policies in the Wild. *Internet Measurement Conference*. 91, 96, 109, 130

[14] APNIC (2018). AS65000 BGP Routing Table Analysis Report. https://bgp.potaroo.net/as2.0/bgp-active.html. 88, 91, 110, 115, 118, 135

[15] Apostolaki, M., Marti, G., Muller, J., and Vanbever, L. (2019). SABRE: Protecting Bitcoin against Routing Attacks. *Networking and Distributed Systems Symposium (NDSS)*. 128

[16] Apostolaki, M., Zohar, A., and Vanbever, L. (2017). Hijacking Bitcoin: Routing Attacks on Cryptocurrencies. In *IEEE Symposium on Security and Privacy (S&P)*. 128

[17] Argyraki, K. J. and Cheriton, D. R. (2005). Active Internet Traffic Filtering - Real-Time Response to Denial-of-Service Attacks. *USENIX Annual Technical Conference, General Track*. 70

[18] Barreto, C. and Koutsoukos, X. (2019). Attacks on Electricity Markets. In *2019 57th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pages 705–711. IEEE. 205

[19] Basescu, C., Reischuk, R. M., Szalachowski, P., Perrig, A., Zhang, Y., Hsiao, H.-C., Kubota, A., and Urakawa, J. (2016). SIBRA: Scalable Internet Bandwidth Reservation Architecture. In *Network and Distributed System Security Symposium (NDSS)*. 17, 24, 26, 33, 201

[20] Bedi, H. S., Roy, S., and Shiva, S. (2011). Game theory-based defense mechanisms against DDoS attacks on TCP/TCP-friendly flows. *... in Cyber Security (CICS)*. 71

[21] Belenky, A. and Ansari, N. (2007). On deterministic packet marking. *Computer Networks*. 16, 70

[22] Belson, D. (2018). Finding Yourself: The Challenges of Accurate IP Geolocation. https://support.maxmind.com/geoip-faq/geoip2-and-geoip-legacy-databases/how-accurate-are-your-geoip2-and-geoip-legacy-databases/. 157

[23] Birge-Lee, H., Wang, L., Rexford, J., and Mittal, P. (2019). SICO: Surgical Interception Attacks by Manipulating BGP Communities. In *ACM Conference on Computer and Communications Security (CCS)*. 33, 129

[24] Bocovich, C. and Goldberg, I. (2016). Slitheen - Perfectly Imitated Decoy Routing through Traffic Replacement. *ACM Conference on Computer and Communications Security*. 74, 101

[25] Bocovich, C. and Goldberg, I. (2018). Secure Asymmetry and Deployability for Decoy Routing Systems. *Proceedings on Privacy Enhancing Technologies Symposium (PETS)*. 74, 81, 112, 120, 122

[26] Bush, R., Maennel, O., Roughan, M., and Uhlig, S. (2009). Internet Optometry - Assessing The Broken Glasses in Internet Reachability. *Internet Measurement Conference*, page 242. 106, 118, 122

[27] Chandra, R., Traina, P., and Li, T. (1996). BGP communities attribute. Technical report, RFC 1997, August. 33

[28] Chou, J. C. Y., Lin, B., Sen, S., and Spatscheck, O. (2009). Proactive Surge Protection: A Defense Mechanism for Bandwidth-Based Attacks. *IEEE/ACM Transactions on Networking*, 17(6):1711–1723. 70

[29] Chung, Taejoong and Aben, Emile and Bruijnzeels, Tim and Chandrasekaran, Balakrishnan and Choffnes, David and Levin, Dave and Maggs, Bruce M and Mislove, Alan and Rijswijk-Deij, Roland van and Rula, John and others (2019). RPKI is Coming of Age: A Longitudinal Study of RPKI Deployment and Invalid Route Origins. In *ACM IMC*. 10

[30] Cloudflare (2020). Famous DDoS Attacks | The Largest DDoS Attacks Of All Time. https://www.cloudflare.com/learning/ddos/famous-ddos-attacks. 2

[31] Collier, S. E. (2016). The emerging enernet: Convergence of the smart grid with the internet of things. *IEEE Industry Applications Magazine*, 23(2):12–16. 143

[32] Consortium, P. (2018). PlanetLab. https://www.planet-lab.org/. 125

[33] CyberX (2020). 2020 GLOBAL IOT/ICS RISK REPORT. *https://cyberx-labs.com/resources/risk-report-2020/*. 155

[34] Di Pinto, A., Dragoni, Y., and Carcano, A. (2018). TRITON: The first ICS cyber attack on safety instrument systems. In *Proc. Black Hat USA*, pages 1–26. 143, 165

[35] Dittrich, D. and Kenneally, E. (2012). The Menlo Report: Ethical Principles Guiding Information and Communication Technology Research. Technical report, U.S. Department of Homeland Security. 87

[36] Dixon, C., Anderson, T. E., and Krishnamurthy, A. (2008). Phalanx - Withstanding Multimillion-Node Botnets. *NSDI*. 70

[37] DNP.org (2020). DNP3 User Group. *https://www.dnp.org/*. 152

[38] Dreidy, M., Mokhlis, H., and Mekhilef, S. (2017). Inertia response and frequency control techniques for renewable energy sources: A review. *Renewable and Sustainable Energy Reviews*, 69:144 – 155. xix, 149

[39] Durand, J., Pepelnjak, I., and Doering, G. (2015). BGP Operations and Security. RFC 7454. 8, 110, 121

[40] Durumeric, Z., Adrian, D., Mirian, A., Bailey, M., and Halderman, J. A. (2015). A Search Engine Backed by Internet-Wide Scanning. In *22nd ACM Conference on Computer and Communications Security*. 157

[41] EIA (2020). U.S. Energy Mapping System. *https://www.eia.gov/state/maps.php*. 157, 158

[42] ENTSOE (2004). Continental Europe Operation Handbook. https://www.entsoe.eu/publications/system-operations-reports/#continental-europe-operation-handbook. 149

[43] EPRI (2020). Smart Grid Resource Center. *https://smartgrid.epri.com/*. 143

[44] Falliere, N., Murchu, L. O., and Chien, E. (2011). W32. stuxnet dossier. *White paper, Symantec Corp., Security Response*, 5(6):29. 143, 165

[45] Fang, X., Misra, S., Xue, G., and Yang, D. (2011). Smart grid–The new and improved power grid: A survey. *IEEE communications surveys & tutorials*, 14(4):944–980. 143

[46] Fayaz, S. K., Tobioka, Y., Sekar, V., and Bailey, M. (2015). Bohatei: Flexible and Elastic DDoS Defense. In *Usenix Security*, pages 817–832. 16

[47] Fuhr, P. (2018). DarkNet – Architecture Oak Ridge National Laboratory (ORNL). https://www.energy.gov/sites/prod/files/2018/12/f58/ORNL. 202, 206

[48] Gallagher, S. (2016). How one rent-a-botnet army of cameras, DVRs caused Internet chaos. *Ars Technica*, 25. 165

[49] Gharaibeh, M., Shah, A., Huffaker, B., Zhang, H., Ensafi, R., and Papadopoulos, C. (2017). A look at router geolocation in public and commercial databases. In *Proceedings of the 2017 Internet Measurement Conference*, pages 463–469. 126, 157

[50] Gilad, Y., Cohen, A., Herzberg, A., Schapira, M., and Shulman, H. (2017). Are We There Yet? On RPKI's Deployment and Security. In *NDSS*. 10

[51] Gill, P., Schapira, M., and Goldberg, S. (2011). Let the market drive deployment: A strategy for transitioning to BGP security. In *ACM SIGCOMM Computer Communication Review*. 42, 43

[52] Girouard, C. (2015). How Do Electric Utilities Make Money? https://blog.aee.net/how-do-electric-utilities-make-money. 146

[53] Goldberg, Sharon (2014). Why is it taking so long to secure internet routing? *CACM*. 10

[54] Gosain, D., Agarwal, A., Chakravarty, S., and Acharya, H. B. (2017). The Devils in The Details: Placing Decoy Routers in the Internet. *arXiv.org*. 81, 101

[55] Gungor, V. C., Sahin, D., Kocak, T., Ergut, S., Buccella, C., Cecati, C., and Hancke, G. P. (2011). Smart grid technologies: Communication technologies and standards. *IEEE transactions on Industrial informatics*, 7(4):529–539. 143

[56] Hawkinson, J. and T., B. (1996). RFC 1930: Guidelines for Creation, Selection, and Registration of an Autonomous System (AS). 8, 73

[57] Holterbach, T., Molero, E. C., Apostolaki, M., Dainotti, A., Vissicchio, S., and Vanbever, L. (2019). Blink: Fast Connectivity Recovery Entirely in the Data Plane. In *USENIX Networked Systems Design and Implementation (NSDI 19)*. 129

[58] Holterbach, T., Pelsser, C., and Bush, R. (2014). On the Suitability of Two Large-Scale Internet Measurement Platforms. *RIPE 69*. 125

[59] Houmansadr, A., Nguyen, G., and Caesar, M. (2011). Cirripede: Circumvention Infrastructure Using Router Redirection with Plausible Deniability. *ACM Conference on Computer and Communications Security (CCS)*. 74

[60] Houmansadr, A., Wong, E. L., and Shmatikov, V. (2014). No Direction Home - The True Cost of Routing Around Decoys. *NDSS*. 74, 80

[61] Hyndman, R. J. and Fan, S. (2009). Density forecasting for long-term peak electricity demand. *IEEE Transactions on Power Systems*, 25(2):1142–1153. 166

[IANA] IANA. IANA Autonomous System (AS) Numbers. https://www.iana.org/assignments/as-numbers/as-numbers.xhtml. 43, 44

[63] Infracritical (2014). Project SHINE Findings Report (1-Oct-2014). *https://www.slideshare.net/BobRadvanovsky/project-shine-findings-report-dated-1oct2014*. 155

[64] Ioannidis, J. and Bellovin, S. M. (2002). Implementing Pushback - Router-Based Defense Against DDoS Attacks. *NDSS*. 70

[65] Kang, M. S., Lee, S. B., and Gligor, V. D. (2013). The Crossfire Attack. *IEEE S&P*. 12, 16, 19, 22, 46, 143, 167, 188

[66] Karlin, J., Ellard, D., Jackson, A. W., Jones, C. E., Lauer, G., Mankins, D., and Strayer, W. T. (2011). Decoy Routing - Toward Unblockable Internet Communication. *FOCI*. 74

[67] Kaspersky (2017). Kaspersky Labs Q1 2017 DDoS Report. https://securelist.com/ddos-attacks-in-q1-2017/78285. 11, 143

[68] Kaspersky (2018). DDoS attacks in Q2 2018. https://securelist.com/ddos-report-in-q2-2018/86537/. 2, 143

[69] Katz-Bassett, E., Madhyastha, H. V., Adhikari, V. K., Scott, C., Sherry, J., van Wesep, P., Anderson, T. E., and Krishnamurthy, A. (2010). Reverse Traceroute. *NSDI*. 130

[70] Katz-Bassett, E., Scott, C., Choffnes, D. R., Cunha, Í., Valancius, V., Feamster, N., Madhyastha, H. V., Anderson, T. E., and Krishnamurthy, A. (2012). LIFEGUARD - Practical Repair of Persistent Route Failures. *SIGCOMM*, page 395. 74, 81, 104, 106, 112, 115, 123, 130

[71] Krenc, T. and Feldmann, A. (2016). BGP Prefix Delegations - A Deep Dive. *Internet Measurement Conference (IMC)*. 121

[72] Labovitz, C., Iekel-Johnson, S., McPherson, D., Oberheide, J., and Jahanian, F. (2011). Internet inter-domain traffic. *ACM SIGCOMM Computer Communication Review*. 42, 43

[73] Lepinski, Matt and Sriram, Kotikalapudi (2013). RFC 8205 - BGPSEC protocol specification. *IETF*. 10

[74] Levin, D. (2019). Automatically Learning How to Evade Censorship. *USENIX ScAINet*. 128

[75] Li, Z., Liao, Q., and Striegel, A. (2009). Botnet economics: uncertainty matters. In *Managing information risk and the economics of security*, pages 245–267. Springer. 165

[76] Liang, G., Weller, S. R., Zhao, J., Luo, F., and Dong, Z. Y. (2016). The 2015 ukraine blackout: Implications for false data injection attacks. *IEEE Transactions on Power Systems*, 32(4):3317–3318. 143, 165, 176

[77] Liu, S., Liu, X. P., and El Saddik, A. (2013). Denial-of-service (DoS) attacks on load frequency control in smart grids. In *2013 IEEE PES Innovative Smart Grid Technologies Conference (ISGT)*, pages 1–6. IEEE. 180, 181, 205

[78] Liu, X, Yang, X, and Lu, Y (2008). StopIt: Mitigating DoS flooding attacks from multi-million botnets. 70

[79] Luckie, M., Huffaker, B., Dhamdhere, A., Giotsas, V., et al. (2013). AS relationships, customer cones, and validation. In *Proceedings of the 2013 conference on Internet measurement conference*, pages 243–256. ACM. 99

[80] M. Lepinski and S. Kent (2012). An Infrastructure to Support Secure Internet Routing. https://tools.ietf.org/html/rfc6480. 10

[81] Ma, M. (2005). Tabu marking scheme for ip traceback. *Parallel and Distributed Processing Symposium*. 16, 70

[82] Machowski, J., Bialek, J., Bumby, J. R., and Bumby, J. (1997). *Power system dynamics and stability*. John Wiley & Sons. 150

[83] Mahajan, R., Bellovin, S. M., Floyd, S., Ioannidis, J., Paxson, V., and Shenker, S. (2002). Controlling high bandwidth aggregates in the network. *Computer Communication Review*. 70

[84] MANRS (2019). MANRS Initiative. https://www.manrs.org/. 114

[85] Matherly, J. (2020). The search engine for the Internet of Things. *https://shodan.io*. 157

[86] McDaniel, P. and McLaughlin, S. (2009). Security and privacy challenges in the smart grid. *IEEE Security & Privacy*, 7(3):75–77. 143

[87] McDaniel, T., Smith, J. M., and Schuchard, M. (2019). The Maestro Attack: Orchestrating Malicious Flows with BGP. 143, 144, 165, 167, 170, 174, 204

[88] Media, S. (2017). Russia Blamed for DDoS Attacks on Baltic Power Grid. https://www.scmagazine.com/russia-blamed-for-ddos-attacks-on-baltic-power-grid/article/661360/. 2

[89] Meier, R., Tsankov, P., Lenders, V., Vanbever, L., and Vechev, M. (2018). NetHide: Secure and Practical Network Topology Obfuscation. In *USENIX Security*. 128

[90] Mills, E. (2020). Just how vulnerable is the electrical grid? *https://www.cnet.com/news/just-how-vulnerable-is-the-electrical-grid/*. 154, 155

[91] Minaei, M., Moreno-Sanchez, P., and Kate, A. (2018). R3C3 - Cryptographically secure Censorship Resistant Rendezvous using Cryptocurrencies. *IACR Cryptology ePrint Archive*. 81

[92] Mirian, A., Ma, Z., Adrian, D., Tischer, M., Chuenchujit, T., Yardley, T., Berthier, R., Mason, J., Durumeric, Z., Halderman, J. A., et al. (2016). An internet-wide view of ics devices. In *2016 14th Annual Conference on Privacy, Security and Trust (PST)*, pages 96–103. IEEE. 152

[93] Mirkovic, J and Reiher, P (2004). A taxonomy of DDoS attack and DDoS defense mechanisms. *ACM SIGCOMM Computer Communication ...*. 70

[94] Modbus.org (2020). The Modbus Organization. *http://modbus.org/*. 151

[95] Muthuprasanna, M. and Manimaran, G. (2008). Distributed Divide-and-Conquer Techniques for Effective DDoS Attack Defenses. *ICDCS*. 16, 70

[96] Nagpal, B., Sharma, P., Chauhan, N., and Panesar, A. (2015). DDoS tools: Classification, analysis and comparison. In *2015 2nd International Conference on Computing for Sustainable Global Development (INDIACom)*, pages 342–346. IEEE. 143

[97] Nasr, M. and Houmansadr, A. (2016). GAME OF DECOYS - Optimal Decoy Routing Through Game Theory. *ACM Conference on Computer and Communications Security (CCS)*. 81, 101, 112, 120, 122, 128

[98] Nasr, M., Zolfaghari, H., and Houmansadr, A. (2017). The Waterfall of Liberty. In *ACM Conference on Computer and Communications Security (CCS)*. 80, 100, 104, 112, 115, 120, 122, 123, 126, 127, 128

[99] Nazario, J. (2007). Blackenergy DDoS Bot Analysis. *Arbor Networks*. 11, 45

[100] NERC (2020). Leading Indicator: Frequency Response (ACE). https://www.nerc.com/pa/RAPA/PA/Pages/FrequencyResponse.aspx. 146, 149, 190

[101] Netlab360 (2017). Mirai Scanner. http://data.netlab.360.com/mirai-scanner/. 43, 44

[102] Noction (2018). BGP Route Dampening: obsolete or still used in the industry? https://www.noction.com/blog/bgp-dampening. 9

[103] Nordström, Ola and Dovrolis, Constantinos (2004). Beware of BGP attacks. *ACM SIGCOMM*. 10

[104] North-American-Network-Operator-Group (2019a). BGP person from Bell Canada/AS577. *https://mailman.nanog.org/pipermail/nanog/2019-June/101487.html*. 75

[105] North-American-Network-Operator-Group (2019b). BGP Prefix Filter List. *https://mailman.nanog.org/pipermail/nanog/2019-May/101260.html*. 75

[106] North-American-Network-Operator-Group (2019c). Someone is Using My AS Number. *https://mailman.nanog.org/pipermail/nanog/2019-June/101407.html*. 75

[107] Oliveira, R., Willinger, W., and Zhang, B. (2008). Quantifying the completeness of the observed internet AS level structure. *cs.ucla.edu*. 9, 99, 106, 111, 113

[108] ORNL (2017). An Assessment of Electric Utility Dependence on the Public Internet. 206

[109] Orsini, C., King, A., Giordano, D., Giotsas, V., and Dainotti, A. (2016). BGPStream - A Software Framework for Live and Historical BGP Data Analysis. *Internet Measurement Conference*. 85, 110, 121

[110] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. (2011). Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12:2825–2830. 44

[111] PeeringDB (2017). PeeringDB: The Interconnection Database. https://www.peeringdb.com. 42, 43, 44

[112] Pelsser, C., Bush, R., Patel, K., Mohapatra, P., and Maennel, O. (2014). Making Route Flap Damping Usable. RFC 7196. 9, 32, 85, 86

[113] Reicher, M. (2017). TVA announces $300 million fiber network expansion. *https://www.tennessean.com/story/news/2017/05/11/tva-announces-30-million-fiber-network-expansion/318598001*. 155, 201

[114] Rekhter, Y., Hares, S., and Li, T. (2006). A Border Gateway Protocol 4 (BGP-4). RFC 4271. 8, 9, 31, 73, 85, 86, 109

[RIPE NCC] RIPE NCC. RIPE Atlas, url=https://atlas.ripe.net/, journal=RIPE Network Coordination Centre, year=2018, howpublished=https://atlas.ripe.net/. 76, 85, 125

[116] RIPE Routing Information Service (RIS) (2018). RIPE RIS BGP Data. https://www.ripe.net/analyse/internet-measurements/routing-information-service-ris. 76, 85

[117] RouteViews (2018). RouteViews Dataset. http://www.routeviews.org/. 43, 64, 76, 85

[118] Sandvine (2017). Global Internet Phenomena Report. https://www.sandvine.com/trends/global-internet-phenomena. 42, 43

[119] Sargolzaei, A., Yen, K., and Abdelghani, M. N. (2014). Delayed inputs attack on load frequency control in smart grid. In *ISGT 2014*, pages 1–5. IEEE. 180, 181, 205

[120] Scheitle, Q., Gasser, O., Rouhi, M., and Carle, G. (2017). Large-scale classification of IPv6-IPv4 siblings with variable clock skew. In *Network Traffic Measurement and Analysis Conference (TMA)*. IEEE. 125

[121] Schlinker, B., Zarifis, K., Cunha, I., Feamster, N., and Katz-Bassett, E. (2014). PEERING: An AS for Us. In *Proc. ACM HotNets*. 76, 83, 125, 168

[122] Schuchard, M., Geddes, J., Thompson, C., and Hopper, N. (2012). Routing Around Decoys. In *Proceedings of the 2012 ACM Conference on Computer and Communications Security*, CCS '12. 39, 74, 80, 100, 104, 108, 109, 112, 115, 120, 122, 123, 126, 127, 128

[123] Schuchard, M. and Hopper, N. (2016). E-Embargoes - Discouraging the Deployment of Traffic Manipulating Boxes With Economic Incentives. *arXiv Preprint*. 74, 95

[124] Schuchard, M., Mohaisen, A., Foo Kune, D., Hopper, N., Kim, Y., and Vasserman, E. Y. (2010). Losing Control of the Internet. In *ACM Conference on Computer and Communications Security (CCS)*. 210

[125] Senthilmahesh, P., Hemalatha, S., Rodrigues, P., and Shanthakumar, A. (2013). DDoS Attacks Defense System Using Information Metrics. 71

[126] Shin, S., Gu, G., Reddy, N., and Lee, C. P. (2012). A Large-Scale Empirical Study of Conficker. *IEEE Transactions on Information Forensics and Security*. 10

[127] Shodan (2014). Map of Industrial Control Systems on the Internet. *https://icsmap.shodan.io/*. 155

[128] Silva, S. S., Silva, R. M., Pinto, R. C., and Salles, R. M. (2013). Botnets: A survey. *Computer Networks*, 57(2):378–403. 165

[129] Siris, V. A. and Stavrakis, I. (2007). Provider-based deterministic packet marking against distributed DoS attacks. *J. Network and Computer Applications*. 16, 70

[130] Smith, J. M. and Schuchard, M. (2018). Routing Around Congestion: Defeating DDoS Attacks and Adverse Network Conditions via Reactive BGP Routing. *IEEE Symposium on Security and Privacy*. 74, 81, 95, 104, 108, 109, 111, 115, 120, 123, 126

[131] Smith, J. M. and Schuchard, M. (2019). Chaos BGP Simulator. https://github.com/VolSec/chaos. 95, 188

[132] Soltan, S., Mittal, P., and Poor, H. V. (2018). BlackIoT: IoT botnet of high wattage devices can disrupt the power grid. In *27th {USENIX} Security Symposium ({USENIX} Security 18)*, pages 15–32. 144, 167, 180, 185, 205

[133] Soltan, S., Mittal, P., and Poor, V. (2019). Protecting the Grid against MAD Attacks. *IEEE Transactions on Network Science and Engineering*. 148, 205

[134] Spring, N., Peterson, L., Bavier, A., and Pai, V. (2006). Using PlanetLab for network research: myths, realities, and best practices. *ACM SIGOPS Operating Systems Review*. 125

[135] Spyridopoulos, T., Karanikas, G., Tryfonas, T., and Oikonomou, G. (2013). A game theoretic defence framework against DoS/DDoS cyber attacks. *Computers & Security*. 71

[136] Srikantha, P. and Kundur, D. (2015). Denial of service attacks and mitigation for stability in cyber-enabled power grid. In *2015 IEEE Power & Energy Society Innovative Smart Grid Technologies Conference (ISGT)*, pages 1–5. IEEE. 180, 181, 205

[137] Sriram, K. and Montgomery, D. C. (2019). Resilient Interdomain Traffic Exchange: BGP Security and DDoS Mitigation. *NIST Report*. 10

[138] Statista (2019). Number of internet users worldwide from 2005 to 2018 (in millions). https://www.statista.com/statistics/273018/number-of-internet-users-worldwide/. 118

[139] Streibelt, F., Lichtblau, F., Beverly, R., Feldmann, A., Pelsser, C., Smaragdakis, G., and Bush, R. (2018). BGP Communities - Even more Worms in the Routing Can. *IMC*. 130

[140] Studer, A. and Perrig, A. (2009). The Coremelt Attack. *ESORICS*. 12, 16, 19, 22, 143, 177, 188

[141] Sun, Y., Edmundson, A., Vanbever, L., Li, O., Rexford, J., Chiang, M., and Mittal, P. (2015). RAPTOR: Routing Attacks on Privacy in Tor. In *USENIX Security*. 129

[142] Swales, A. et al. (1999). Open modbus/tcp specification. *Schneider Electric*, 29. 151

[143] Symantec (2016). Mirai: what you need to know about the botnet behind recent major DDoS attacks. https://tiny.utk.edu/orVeO. Accessed: 17 January 2017. 11, 16, 44, 143

[144] Taylor, J. W. (2003). Short-term electricity demand forecasting using double seasonal exponential smoothing. *Journal of the Operational Research Society*, 54(8):799–805. 166

[145] Taylor, J. W. and Buizza, R. (2003). Using weather ensemble predictions in electricity demand forecasting. *International Journal of Forecasting*, 19(1):57–70. 166

[146] Taylor, J. W., De Menezes, L. M., and McSharry, P. E. (2006). A comparison of univariate methods for forecasting electricity demand up to a day ahead. *International journal of forecasting*, 22(1):1–16. 166

[147] Thomas, M. and Mohaisen, A. (2014). Kindred Domains: Detecting and Clustering Botnet Domains Using DNS Traffic. In *Proceedings of the 23rd International Conference on World Wide Web*, WWW '14 Companion, pages 707–712, New York, NY, USA. ACM. 43, 45

[148] Tran, M., Choi, I., Moon, G. J., Vu, A. V., and Kang, M. S. (2020). A Stealthier Partitioning Attack against Bitcoin Peer-to-Peer Network. *IEEE Symposium on Security and Privacy (S&P)*. 129

[149] Tran, M., Kang, M. S., Hsiao, H.-C., Chiang, W.-H., Tung, S.-P., and Wang, Y.-S. (2019). On the Feasibility of Rerouting-based DDoS Defenses. In *Proceedings of IEEE Symposium on Security and Privacy (IEEE S&P)*. 74, 81, 95, 104, 105, 111, 113, 115, 123, 126, 133, 135, 138, 208

[150] UCSD CAIDA (2018). CAIDA AS Relationship Dataset. 24, 27, 39, 41, 43, 44, 64, 95, 109, 158

[151] Villamizar, C., Chandra, R., and Govindan, D. R. (1998). BGP Route Flap Damping. RFC 2439. 9, 85, 86

[152] Von Ahn, L., Blum, M., and Langford, J. (2004). Telling humans and computers apart automatically. *Communications of the ACM*, 47(2):56–60. 70

[153] Widén, J. and Wäckelgård, E. (2010). A high-resolution stochastic model of domestic activity patterns and electricity demand. *Applied energy*, 87(6):1880–1892. 166

[154] WorldBankGroup (2017). World Bank Global Indicators. http://data.worldbank.org/indicator. 42, 43

[155] Wustrow, E., Swanson, C., and Halderman, J. A. (2014). TapDance: End-to-Middle Anti-censorship without Flow Blocking. *USENIX Security*. 74

[156] Wustrow, E., Wolchok, S., and Goldberg, I. (2011). Telex: Anticensorship in the Network Infrastructure. *USENIX Security*. 74

[157] Xiang, Y. and Zhou, W. (2006). Protecting information infrastructure from ddos attacks by madf. *International Journal of High ...*. 16, 70

[158] Xiang, Y., Zhou, W., and Guo, M. (2009). Flexible Deterministic Packet Marking - An IP Traceback System to Find the Real Source of Attacks. *IEEE Trans. Parallel Distrib. Syst.* 16, 70

[159] Yadav, V. (2015). Automatic Generation Control by Conventional Synchronous Generators. *MATLAB Central File Exchange (https://www.mathworks.com/*

matlabcentral/fileexchange/53687-automatic-generation-control-by-conventional-synchronous-generators). 181

[160] Yao, A. W., Chi, S., and Chen, J. (2003). An improved grey-based approach for electricity demand forecasting. *Electric Power Systems Research*, 67(3):217–224. 166

[161] Yau, D. K. Y., Lui, J. C. S., Liang, F., and Yam, Y. (2005). Defending against distributed denial-of-service attacks with max-min fair server-centric router throttles. *IEEE/ACM Trans. Netw.* 70

[162] Zhang, L., Estrin, D., Burke, J., Jacobson, V., Thornton, J. D., Smetters, D. K., Zhang, B., Tsudik, G., Massey, D., Papadopoulos, C., et al. (2010). Named data networking (ndn) project. *Relatório Técnico NDN-0001, Xerox Palo Alto Research Center-PARC*, 157. 129

[163] Zhang, X., Hsiao, H.-C., Hasker, G., Chan, H., Perrig, A., and Andersen, D. G. (2011). SCION - Scalability, Control, and Isolation on Next-Generation Networks. *IEEE Symposium on Security and Privacy (S&P)*. 17, 26, 33, 129, 201

[164] Zhao, D., Traore, I., Sayed, B., Lu, W., and Saad, S. (2013). Botnet detection based on traffic behavior analysis and flow intervals. *Computers & ...*, 39:2–16. 71

[165] Zhou, W., Jia, W., Wen, S., Xiang, Y., and Zhou, W. (2014). Detection and defense of application-layer DDoS attacks in backbone web traffic. *Future Generation Computer ...*, 38:36–46. 71

Vita

As a Chancellor's Fellow at the University of Tennessee, Jared's PhD research focuses on improving the resilience of large-scale distributed systems, as well as work on cryptocurrencies, operating system security, and censorship circumvention. Jared's research at UT has been published in top venues such as the IEEE Symposium for Security and Privacy (Oakland), the Network and Distributed Systems Security Conference (NDSS), and USENIX Security.

Jared is also the Lead Scientist for AI in Cybersecurity and an R&D Staff member at Oak Ridge National Lab. Since 2015, Jared has been the Principal Investigator for several efforts at the intersection of security and data science. Specifically, he leads projects or plays a key role on teams focused on adversarial machine learning, automating digital forensics and incident response (DFIR), SCADA and ICS systems (cyber-physical) security, constructing "grand challenges" for executing commercial security product evaluations for the U.S. Navy, and building novel detection and remediation systems for advanced forms of malware. Jared's work at ORNL has appeared at ACM Conference on Computer and Communications Security (CCS), Annual Computer Security Applications Conference (ACSAC), Detection of Intrusions and Malware & Vulnerability Assessments (DIMVA), and leading digital forensics conferences such as DFWRS.

Prior to ORNL, Jared worked on the internal product security R&D group at Cisco Systems, the Advanced Security Initiatives Group (ASIG). At ASIG, Jared engaged in red team evaluations of external 3rd-party products as well as software engineering efforts for building security tools.

Jared holds an MS and BS in Computer Science from the University of Tennessee, Knoxville, where he founded and led the annual student-run hackathon, now in it's 4th year, VolHacks, and the student-run Cyber Security organization, HackUTK. Jared also currently co-organizes Knoxville, Tennessee's open data focused City-sponsored hackathon, KNXHX. Jared has taught security and data science topics with Treehouse, one of the largest online e-learning platforms (taught the OWASP Top 10 course among others), serves on the board of advisors for several Tennessee startups and non-profits, and gives regular conference talks across the US and Canada on security, the python programming language, and data science.